Global Constraint Catalog
Volume II
Time-Series Constraints

Ekaterina Arafailova, Nicolas Beldiceanu, Rémi Douence
Mines Nantes (CNRS/INRIA), Nantes, France

Mats Carlsson
SICS, Kista, Sweden

Pierre Flener, María Andreina Francisco Rodríguez, Justin Pearson
Uppsala University, Uppsala, Sweden

Helmut Simonis
Insight Centre for Data Analytics, University College Cork, Ireland

Abstract: First this report presents a restricted set of finite transducers used to synthesise structural time-series constraints described by means of a multi-layered function composition scheme. Second it provides the corresponding synthesised catalogue of structural time-series constraints where each constraint is explicitly described in terms of automata with accumulators.

Keywords: constraint programming, global constraint, finite transducer, automaton with accumulators, reversible automaton, glue matrix, sharp bounds, meta-data, ontology, sequential pattern mining, minimum description length.

September 24, 2016

1Corresponding author, Email: Nicolas.Beldiceanu@mines-nantes.fr

This work is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License. To view a copy of this license, visit http://creativecommons.org/licenses/by-nc-sa/4.0/ or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.
# Contents

<table>
<thead>
<tr>
<th>Preface</th>
<th>i</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Describing a global constraint</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Basic data types</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Compound data types</td>
<td>2</td>
</tr>
<tr>
<td>1.3 Restrictions</td>
<td>3</td>
</tr>
<tr>
<td>1.4 Declaring a global constraint</td>
<td>8</td>
</tr>
<tr>
<td>1.5 Keywords</td>
<td>8</td>
</tr>
<tr>
<td>2 Patterns, Seed Transducers, Glue matrices and Decoration Tables</td>
<td>9</td>
</tr>
<tr>
<td>2.1 Patterns, their Seed Transducers and their Parametrised Glue Matrices</td>
<td>13</td>
</tr>
<tr>
<td>2.1.1 BUMP_ON_DECREASING_SEQUENCE</td>
<td>15</td>
</tr>
<tr>
<td>2.1.2 DECREASING</td>
<td>19</td>
</tr>
<tr>
<td>2.1.3 DECREASING_SEQUENCE</td>
<td>21</td>
</tr>
<tr>
<td>2.1.4 DECREASING_TERRACE</td>
<td>24</td>
</tr>
<tr>
<td>2.1.5 DIP_ON_INCREASING_SEQUENCE</td>
<td>27</td>
</tr>
<tr>
<td>2.1.6 GORGE</td>
<td>31</td>
</tr>
<tr>
<td>2.1.7 INCREASING</td>
<td>35</td>
</tr>
<tr>
<td>2.1.8 INCREASING_SEQUENCE</td>
<td>37</td>
</tr>
<tr>
<td>2.1.9 INCREASING_TERRACE</td>
<td>40</td>
</tr>
<tr>
<td>2.1.10 INFLEXION</td>
<td>43</td>
</tr>
<tr>
<td>2.1.11 PEAK</td>
<td>45</td>
</tr>
<tr>
<td>2.1.12 PLAIN</td>
<td>48</td>
</tr>
<tr>
<td>2.1.13 PLATEAU</td>
<td>51</td>
</tr>
<tr>
<td>2.1.14 PROPER_PLAIN</td>
<td>54</td>
</tr>
<tr>
<td>2.1.15 PROPER_PLATEAU</td>
<td>57</td>
</tr>
<tr>
<td>2.1.16 STEADY</td>
<td>60</td>
</tr>
<tr>
<td>2.1.17 STEADY_SEQUENCE</td>
<td>62</td>
</tr>
<tr>
<td>2.1.18 STRICTLY_DECREASING_SEQUENCE</td>
<td>64</td>
</tr>
<tr>
<td>2.1.19 STRICTLY_INCREASING_SEQUENCE</td>
<td>67</td>
</tr>
<tr>
<td>2.1.20 SUMMIT</td>
<td>70</td>
</tr>
<tr>
<td>2.1.21 VALLEY</td>
<td>74</td>
</tr>
<tr>
<td>2.1.22 ZIGZAG</td>
<td>77</td>
</tr>
<tr>
<td>2.2 Decoration Tables</td>
<td>83</td>
</tr>
</tbody>
</table>
# Contents

3 Global Constraint Catalogue

3.1 ALL_EQUAL_HEIGHT_DECREASING_TERRACE ........................................ 126
3.2 ALL_EQUAL_HEIGHT_INCREMENTING_TERRACE ...................................... 130
3.3 ALL_EQUAL_HEIGHT_PLAIN ........................................................... 134
3.4 ALL_EQUAL_HEIGHT_PLATEAU ......................................................... 138
3.5 ALL_EQUAL_HEIGHT_PROPER_PLAIN .................................................. 142
3.6 ALL_EQUAL_HEIGHT_PROPER_PLATEAU ............................................. 146
3.7 ALL_EQUAL_HEIGHT_STEADY ........................................................ 150
3.8 ALL_EQUAL_HEIGHT_STEADY_SEQUENCE ......................................... 154
3.9 ALL_EQUAL_MAX_BUMP_ON_DECREASING_SEQUENCE ............................. 158
3.10 ALL_EQUAL_MAX_DECREASING ....................................................... 162
3.11 ALL_EQUAL_MAX_DECREASING_SEQUENCE ....................................... 166
3.12 ALL_EQUAL_MAX_DIP_ON_INCREMENTING_SEQUENCE ............................ 170
3.13 ALL_EQUAL_MAX_INCREMENTING ...................................................... 174
3.14 ALL_EQUAL_MAX_INCREMENTING_SEQUENCE ..................................... 178
3.15 ALL_EQUAL_MAX_INFLEXION .......................................................... 182
3.16 ALL_EQUAL_MAX_PEAK ................................................................. 184
3.17 ALL_EQUAL_MAX.Strictly_DECREASINGSEQUENCE ............................... 188
3.18 ALL_EQUAL_MAX.Strictly_INCREMENTINGSEQUENCE ............................ 192
3.19 ALL_EQUAL_MAX_SUMMIT ............................................................. 196
3.20 ALL_EQUAL_MAX_ZIGZAG ............................................................... 200
3.21 ALL_EQUAL_MIN_BUMP_ON_DECREASING_SEQUENCE ............................ 204
3.22 ALL_EQUAL_MIN_DECREASING ........................................................ 208
3.23 ALL_EQUAL_MIN_DECREASING_SEQUENCE ....................................... 212
3.24 ALL_EQUAL_MIN_DIP_ON_INCREMENTING_SEQUENCE ........................... 216
3.25 ALL_EQUAL_MIN_GORGE ............................................................... 220
3.26 ALL_EQUAL_MIN_INCREMENTING ...................................................... 222
3.27 ALL_EQUAL_MIN_INCREMENTING_SEQUENCE ..................................... 226
3.28 ALL_EQUAL_MIN_INFLEXION .......................................................... 230
3.29 ALL_EQUAL_MIN.Strictly_DECREASINGSEQUENCE ............................... 234
3.30 ALL_EQUAL_MIN.Strictly_INCREMENTINGSEQUENCE ............................ 238
3.31 ALL_EQUAL_MIN_VALLEY ............................................................... 242
3.32 ALL_EQUAL_MIN_ZIGZAG ............................................................... 246
3.33 ALL_EQUAL_RANGE_DECREASING .................................................. 250
3.34 ALL_EQUAL_RANGE_DECREASING_SEQUENCE ................................... 254
3.35 ALL_EQUAL_RANGE_INCREMENTING ............................................... 258
3.36 ALL_EQUAL_RANGE_INCREMENTING_SEQUENCE ................................... 262
3.37 ALL_EQUAL_RANGE.Strictly_DECREASINGSEQUENCE ........................... 266
3.38 ALL_EQUAL_RANGE.Strictly_INCREMENTINGSEQUENCE ........................ 270
3.39 ALL_EQUAL_SURF_BUMP_ON_DECREASING_SEQUENCE ......................... 274
3.40 ALL_EQUAL_SURF_DECREASING ...................................................... 278
3.41 ALL_EQUAL_SURF_DECREASING_SEQUENCE ...................................... 282
3.42 ALL_EQUAL_SURF_DECREASING_TERRACE ....................................... 286
3.43 ALL_EQUAL_SURF_DIP_ON_INCREMENTING_SEQUENCE .......................... 290
3.44 ALL_EQUAL_SURF_GORGE ............................................................. 294
3.45 ALL_EQUAL_SURF_INCREMENTING .................................................... 296
<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.92</td>
<td>DECREASING_MAX_INFLEXION</td>
<td>482</td>
</tr>
<tr>
<td>3.93</td>
<td>DECREASING_MAX_PEAK</td>
<td>486</td>
</tr>
<tr>
<td>3.94</td>
<td>DECREASING_MAX_STRICTLY_DECREASING_SEQUENCE</td>
<td>490</td>
</tr>
<tr>
<td>3.95</td>
<td>DECREASING_MAX_STRICTLY_INCREASING_SEQUENCE</td>
<td>494</td>
</tr>
<tr>
<td>3.96</td>
<td>DECREASING_MAX_SUMMIT</td>
<td>498</td>
</tr>
<tr>
<td>3.97</td>
<td>DECREASING_MAX_ZIGZAG</td>
<td>502</td>
</tr>
<tr>
<td>3.98</td>
<td>DECREASING_MIN_BUMP_ON_DECREASING_SEQUENCE</td>
<td>506</td>
</tr>
<tr>
<td>3.99</td>
<td>DECREASING_MIN_DECREASING</td>
<td>510</td>
</tr>
<tr>
<td>3.100</td>
<td>DECREASING_MIN_DECREASING_SEQUENCE</td>
<td>514</td>
</tr>
<tr>
<td>3.101</td>
<td>DECREASING_MIN_DIP_ON_INCREASING_SEQUENCE</td>
<td>518</td>
</tr>
<tr>
<td>3.102</td>
<td>DECREASING_MIN_GORGE</td>
<td>522</td>
</tr>
<tr>
<td>3.103</td>
<td>DECREASING_MIN_INCREASING</td>
<td>526</td>
</tr>
<tr>
<td>3.104</td>
<td>DECREASING_MIN_INCREASING_SEQUENCE</td>
<td>530</td>
</tr>
<tr>
<td>3.105</td>
<td>DECREASING_MIN_INFLEXION</td>
<td>534</td>
</tr>
<tr>
<td>3.106</td>
<td>DECREASING_MIN_STRICTLY_DECREASING_SEQUENCE</td>
<td>538</td>
</tr>
<tr>
<td>3.107</td>
<td>DECREASING_MIN_STRICTLY_INCREASING_SEQUENCE</td>
<td>542</td>
</tr>
<tr>
<td>3.108</td>
<td>DECREASING_MIN_VALLEY</td>
<td>546</td>
</tr>
<tr>
<td>3.109</td>
<td>DECREASING_MIN_ZIGZAG</td>
<td>550</td>
</tr>
<tr>
<td>3.110</td>
<td>DECREASING_RANGE_DECREASING</td>
<td>554</td>
</tr>
<tr>
<td>3.111</td>
<td>DECREASING_RANGE_DECREASING_SEQUENCE</td>
<td>558</td>
</tr>
<tr>
<td>3.112</td>
<td>DECREASING_RANGE_INCREASING</td>
<td>562</td>
</tr>
<tr>
<td>3.113</td>
<td>DECREASING_RANGE_INCREASING_SEQUENCE</td>
<td>566</td>
</tr>
<tr>
<td>3.114</td>
<td>DECREASING_RANGE_STRICTLY_DECREASING_SEQUENCE</td>
<td>570</td>
</tr>
<tr>
<td>3.115</td>
<td>DECREASING_RANGE_STRICTLY_INCREASING_SEQUENCE</td>
<td>574</td>
</tr>
<tr>
<td>3.116</td>
<td>DECREASING_SURF_BUMP_ON_DECREASING_SEQUENCE</td>
<td>578</td>
</tr>
<tr>
<td>3.117</td>
<td>DECREASING_SURF_DECREASING</td>
<td>582</td>
</tr>
<tr>
<td>3.118</td>
<td>DECREASING_SURF_DECREASING_SEQUENCE</td>
<td>586</td>
</tr>
<tr>
<td>3.119</td>
<td>DECREASING_SURF_DECERRING_TERRACE</td>
<td>590</td>
</tr>
<tr>
<td>3.120</td>
<td>DECREASING_SURF_DIP_ON_INCREASING_SEQUENCE</td>
<td>594</td>
</tr>
<tr>
<td>3.121</td>
<td>DECREASING_SURF_GORGE</td>
<td>598</td>
</tr>
<tr>
<td>3.122</td>
<td>DECREASING_SURF_INCREASING</td>
<td>602</td>
</tr>
<tr>
<td>3.123</td>
<td>DECREASING_SURF_INCREASING_SEQUENCE</td>
<td>604</td>
</tr>
<tr>
<td>3.124</td>
<td>DECREASING_SURF_INCREASING_TERRACE</td>
<td>608</td>
</tr>
<tr>
<td>3.125</td>
<td>DECREASING_SURF_INFLEXION</td>
<td>612</td>
</tr>
<tr>
<td>3.126</td>
<td>DECREASING_SURF_PEAK</td>
<td>616</td>
</tr>
<tr>
<td>3.127</td>
<td>DECREASING_SURF_PLAIN</td>
<td>620</td>
</tr>
<tr>
<td>3.128</td>
<td>DECREASING_SURF_PLATEAU</td>
<td>624</td>
</tr>
<tr>
<td>3.129</td>
<td>DECREASING_SURF_PROPERPLAIN</td>
<td>628</td>
</tr>
<tr>
<td>3.130</td>
<td>DECREASING_SURF_PROPER_PLATEAU</td>
<td>632</td>
</tr>
<tr>
<td>3.131</td>
<td>DECREASING_SURF_STEADY</td>
<td>636</td>
</tr>
<tr>
<td>3.132</td>
<td>DECREASING_SURF_STEADY_SEQUENCE</td>
<td>640</td>
</tr>
<tr>
<td>3.133</td>
<td>DECREASING_SURF_STRICTLY_DECREASING_SEQUENCE</td>
<td>644</td>
</tr>
<tr>
<td>3.134</td>
<td>DECREASING_SURF_STRICTLY_INCREASING_SEQUENCE</td>
<td>648</td>
</tr>
<tr>
<td>3.135</td>
<td>DECREASING_SURF_SUMMIT</td>
<td>652</td>
</tr>
<tr>
<td>3.136</td>
<td>DECREASING_SURF_VALLEY</td>
<td>656</td>
</tr>
<tr>
<td>3.137</td>
<td>DECREASING_SURF_ZIGZAG</td>
<td>660</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
<td></td>
</tr>
<tr>
<td>---------------------------------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>3.184 INCREASING_MIN_STRICTLY_INCREASING_SEQUENCE</td>
<td>846</td>
<td></td>
</tr>
<tr>
<td>3.185 INCREASING_MIN_VALLEY</td>
<td>850</td>
<td></td>
</tr>
<tr>
<td>3.186 INCREASING_MIN_ZIGZAG</td>
<td>854</td>
<td></td>
</tr>
<tr>
<td>3.187 INCREASING_RANGE_DECREASING</td>
<td>858</td>
<td></td>
</tr>
<tr>
<td>3.188 INCREASING_RANGE_DECREASING_SEQUENCE</td>
<td>862</td>
<td></td>
</tr>
<tr>
<td>3.189 INCREASING_RANGE_INCREASING</td>
<td>866</td>
<td></td>
</tr>
<tr>
<td>3.190 INCREASING_RANGE_INCREASING_SEQUENCE</td>
<td>870</td>
<td></td>
</tr>
<tr>
<td>3.191 INCREASING_RANGE_STRICTLY_DECREASING_SEQUENCE</td>
<td>874</td>
<td></td>
</tr>
<tr>
<td>3.192 INCREASING_RANGE_STRICTLY_INCREASING_SEQUENCE</td>
<td>878</td>
<td></td>
</tr>
<tr>
<td>3.193 INCREASING_SURF_BUMP_ON_DECREASING_SEQUENCE</td>
<td>882</td>
<td></td>
</tr>
<tr>
<td>3.194 INCREASING_SURF_DECREASING</td>
<td>886</td>
<td></td>
</tr>
<tr>
<td>3.195 INCREASING_SURF_DECREASING_SEQUENCE</td>
<td>888</td>
<td></td>
</tr>
<tr>
<td>3.196 INCREASING_SURF_DECREASING_TERRACE</td>
<td>892</td>
<td></td>
</tr>
<tr>
<td>3.197 INCREASING_SURF_DIP_ON_INCREASING_SEQUENCE</td>
<td>896</td>
<td></td>
</tr>
<tr>
<td>3.198 INCREASING_SURF_GORGE</td>
<td>900</td>
<td></td>
</tr>
<tr>
<td>3.199 INCREASING_SURF_INCREASING</td>
<td>904</td>
<td></td>
</tr>
<tr>
<td>3.200 INCREASING_SURF_INCREASING_SEQUENCE</td>
<td>908</td>
<td></td>
</tr>
<tr>
<td>3.201 INCREASING_SURF_INCREASING_TERRACE</td>
<td>912</td>
<td></td>
</tr>
<tr>
<td>3.202 INCREASING_SURF_INFLEXION</td>
<td>916</td>
<td></td>
</tr>
<tr>
<td>3.203 INCREASING_SURF_PEAK</td>
<td>920</td>
<td></td>
</tr>
<tr>
<td>3.204 INCREASING_SURF_PLAIN</td>
<td>924</td>
<td></td>
</tr>
<tr>
<td>3.205 INCREASING_SURF_PLATEAU</td>
<td>928</td>
<td></td>
</tr>
<tr>
<td>3.206 INCREASING_SURF_PROPER_PLAIN</td>
<td>932</td>
<td></td>
</tr>
<tr>
<td>3.207 INCREASING_SURF_PROPER_PLATEAU</td>
<td>936</td>
<td></td>
</tr>
<tr>
<td>3.208 INCREASING_SURF_STEADY</td>
<td>940</td>
<td></td>
</tr>
<tr>
<td>3.209 INCREASING_SURF_STEADY_SEQUENCE</td>
<td>944</td>
<td></td>
</tr>
<tr>
<td>3.210 INCREASING_SURF_STRICTLY_DECREASING_SEQUENCE</td>
<td>948</td>
<td></td>
</tr>
<tr>
<td>3.211 INCREASING_SURF_STRICTLY_INCREASING_SEQUENCE</td>
<td>952</td>
<td></td>
</tr>
<tr>
<td>3.212 INCREASING_SURF_SUMMIT</td>
<td>956</td>
<td></td>
</tr>
<tr>
<td>3.213 INCREASING_SURF_VALLEY</td>
<td>960</td>
<td></td>
</tr>
<tr>
<td>3.214 INCREASING_SURF_ZIGZAG</td>
<td>964</td>
<td></td>
</tr>
<tr>
<td>3.215 INCREASING_WIDTH_DECREASING_SEQUENCE</td>
<td>968</td>
<td></td>
</tr>
<tr>
<td>3.216 INCREASING_WIDTH_DECREASING_TERRACE</td>
<td>972</td>
<td></td>
</tr>
<tr>
<td>3.217 INCREASING_WIDTH_GORGE</td>
<td>976</td>
<td></td>
</tr>
<tr>
<td>3.218 INCREASING_WIDTH_INCREASING_SEQUENCE</td>
<td>980</td>
<td></td>
</tr>
<tr>
<td>3.219 INCREASING_WIDTH_INCREASING_TERRACE</td>
<td>984</td>
<td></td>
</tr>
<tr>
<td>3.220 INCREASING_WIDTH_INFLEXION</td>
<td>988</td>
<td></td>
</tr>
<tr>
<td>3.221 INCREASING_WIDTH_PEAK</td>
<td>992</td>
<td></td>
</tr>
<tr>
<td>3.222 INCREASING_WIDTH_PLAIN</td>
<td>996</td>
<td></td>
</tr>
<tr>
<td>3.223 INCREASING_WIDTH_PLATEAU</td>
<td>1000</td>
<td></td>
</tr>
<tr>
<td>3.224 INCREASING_WIDTH_PROPER_PLAIN</td>
<td>1004</td>
<td></td>
</tr>
<tr>
<td>3.225 INCREASING_WIDTH_PROPER_PLATEAU</td>
<td>1008</td>
<td></td>
</tr>
<tr>
<td>3.226 INCREASING_WIDTH_STEADY_SEQUENCE</td>
<td>1012</td>
<td></td>
</tr>
<tr>
<td>3.227 INCREASING_WIDTH_STRICTLY_DECREASING_SEQUENCE</td>
<td>1014</td>
<td></td>
</tr>
<tr>
<td>3.228 INCREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE</td>
<td>1018</td>
<td></td>
</tr>
<tr>
<td>3.229 INCREASING_WIDTH_SUMMIT</td>
<td>1022</td>
<td></td>
</tr>
<tr>
<td>Section</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>-----------</td>
<td>--------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>3.276</td>
<td>MAX_SURF_INCREASING</td>
<td>1214</td>
</tr>
<tr>
<td>3.277</td>
<td>MAX_SURF_INCREASING_SEQUENCE</td>
<td>1218</td>
</tr>
<tr>
<td>3.278</td>
<td>MAX_SURF_INCREASING_TERRACE</td>
<td>1222</td>
</tr>
<tr>
<td>3.279</td>
<td>MAX_SURF_INFLEXION</td>
<td>1226</td>
</tr>
<tr>
<td>3.280</td>
<td>MAX_SURF_PEAK</td>
<td>1230</td>
</tr>
<tr>
<td>3.281</td>
<td>MAX_SURF_PLAIN</td>
<td>1234</td>
</tr>
<tr>
<td>3.282</td>
<td>MAX_SURF_PLATEAU</td>
<td>1238</td>
</tr>
<tr>
<td>3.283</td>
<td>MAX_SURF_PROPER_PLAIN</td>
<td>1242</td>
</tr>
<tr>
<td>3.284</td>
<td>MAX_SURF_PROPER_PLATEAU</td>
<td>1246</td>
</tr>
<tr>
<td>3.285</td>
<td>MAX_SURF_STEADY</td>
<td>1250</td>
</tr>
<tr>
<td>3.286</td>
<td>MAX_SURF_STEADY_SEQUENCE</td>
<td>1254</td>
</tr>
<tr>
<td>3.287</td>
<td>MAX_SURF STRICTLY_DECREASING_SEQUENCE</td>
<td>1258</td>
</tr>
<tr>
<td>3.288</td>
<td>MAX_SURF STRICTLY_INCREASING_SEQUENCE</td>
<td>1262</td>
</tr>
<tr>
<td>3.289</td>
<td>MAX_SURF_SUMMIT</td>
<td>1266</td>
</tr>
<tr>
<td>3.290</td>
<td>MAX_SURF_VALLEY</td>
<td>1270</td>
</tr>
<tr>
<td>3.291</td>
<td>MAX_SURF_ZIGZAG</td>
<td>1274</td>
</tr>
<tr>
<td>3.292</td>
<td>MAX_WIDTH_DECREASING_SEQUENCE</td>
<td>1280</td>
</tr>
<tr>
<td>3.293</td>
<td>MAX_WIDTH_DECREASING_TERRACE</td>
<td>1284</td>
</tr>
<tr>
<td>3.294</td>
<td>MAX_WIDTH GORGE</td>
<td>1288</td>
</tr>
<tr>
<td>3.295</td>
<td>MAX_WIDTH_INCREASING_SEQUENCE</td>
<td>1294</td>
</tr>
<tr>
<td>3.296</td>
<td>MAX_WIDTH_INCREASING_TERRACE</td>
<td>1298</td>
</tr>
<tr>
<td>3.297</td>
<td>MAX_WIDTH_INFLEXION</td>
<td>1302</td>
</tr>
<tr>
<td>3.298</td>
<td>MAX_WIDTH_PEAK</td>
<td>1306</td>
</tr>
<tr>
<td>3.299</td>
<td>MAX_WIDTH_PLAIN</td>
<td>1310</td>
</tr>
<tr>
<td>3.300</td>
<td>MAX_WIDTH_PLATEAU</td>
<td>1314</td>
</tr>
<tr>
<td>3.301</td>
<td>MAX_WIDTH_PROPER_PLAIN</td>
<td>1318</td>
</tr>
<tr>
<td>3.302</td>
<td>MAX_WIDTH_PROPER_PLATEAU</td>
<td>1322</td>
</tr>
<tr>
<td>3.303</td>
<td>MAX_WIDTH_STEADY_SEQUENCE</td>
<td>1326</td>
</tr>
<tr>
<td>3.304</td>
<td>MAX_WIDTH STRICTLY_DECREASING_SEQUENCE</td>
<td>1330</td>
</tr>
<tr>
<td>3.305</td>
<td>MAX_WIDTH STRICTLY_INCREASING_SEQUENCE</td>
<td>1334</td>
</tr>
<tr>
<td>3.306</td>
<td>MAX_WIDTH_SUMMIT</td>
<td>1338</td>
</tr>
<tr>
<td>3.307</td>
<td>MAX_WIDTH VALLEY</td>
<td>1344</td>
</tr>
<tr>
<td>3.308</td>
<td>MAX_WIDTH ZIGZAG</td>
<td>1348</td>
</tr>
<tr>
<td>3.309</td>
<td>MIN_HEIGHT_DECREASING_TERRACE</td>
<td>1354</td>
</tr>
<tr>
<td>3.310</td>
<td>MIN_HEIGHT_INCREASING_TERRACE</td>
<td>1358</td>
</tr>
<tr>
<td>3.311</td>
<td>MIN_HEIGHT PLAIN</td>
<td>1362</td>
</tr>
<tr>
<td>3.312</td>
<td>MIN_HEIGHT PLATEAU</td>
<td>1366</td>
</tr>
<tr>
<td>3.313</td>
<td>MIN_HEIGHT_PROPER_PLAIN</td>
<td>1370</td>
</tr>
<tr>
<td>3.314</td>
<td>MIN_HEIGHT_PROPER_PLATEAU</td>
<td>1374</td>
</tr>
<tr>
<td>3.315</td>
<td>MIN_HEIGHT STEADY</td>
<td>1378</td>
</tr>
<tr>
<td>3.316</td>
<td>MIN_HEIGHT_STEADY_SEQUENCE</td>
<td>1382</td>
</tr>
<tr>
<td>3.317</td>
<td>MIN_MAX_BUMP_ON_DECREASING_SEQUENCE</td>
<td>1386</td>
</tr>
<tr>
<td>3.318</td>
<td>MIN_MAX_DECREASING</td>
<td>1390</td>
</tr>
<tr>
<td>3.319</td>
<td>MIN_MAX_DECREASING_SEQUENCE</td>
<td>1394</td>
</tr>
<tr>
<td>3.320</td>
<td>MIN_MAX_DIP_ON_INCREASING_SEQUENCE</td>
<td>1398</td>
</tr>
<tr>
<td>3.321</td>
<td>MIN_MAX_INCREASING</td>
<td>1402</td>
</tr>
</tbody>
</table>
3.460 POS_MAX_SURF_VALLEY .................................................. 1968
3.461 POS_MAX_SURF_ZIGZAG .................................................. 1972
3.462 POS_MAX_WIDTH_DECREASING_SEQUENCE ......................... 1976
3.463 POS_MAX_WIDTH_DECREASING_TERRACE ............................... 1980
3.464 POS_MAX_WIDTH_GORGE ................................................. 1984
3.465 POS_MAX_WIDTH_INCREASING_SEQUENCE ......................... 1988
3.466 POS_MAX_WIDTH_INCREASING_TERRACE ............................... 1992
3.467 POS_MAX_WIDTH_INFLEXION ............................................. 1996
3.468 POS_MAX_WIDTH_PEAK ................................................... 2000
3.469 POS_MAX_WIDTH_PLAIN .................................................. 2004
3.470 POS_MAX_WIDTH_PLATEAU .............................................. 2008
3.471 POS_MAX_WIDTH_PROPER_PLAIN ...................................... 2012
3.472 POS_MAX_WIDTH_PROPER_PLATEAU .................................... 2016
3.473 POS_MAX_WIDTH_STEADY_SEQUENCE ................................ 2020
3.474 POS_MAX_WIDTH_STRICTLY_DECREASING_SEQUENCE ............... 2024
3.475 POS_MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE ............... 2028
3.476 POS_MAX_WIDTH_SUMMIT ............................................... 2032
3.477 POS_MAX_WIDTH_VALLEY ............................................... 2036
3.478 POS_MAX_WIDTH_ZIGZAG .............................................. 2040
3.479 POS_MIN_HEIGHT_DECREASING_TERRACE ............................. 2044
3.480 POS_MIN_HEIGHT_INCREASING_TERRACE ............................. 2048
3.481 POS_MIN_HEIGHT_PLAIN ............................................... 2052
3.482 POS_MIN_HEIGHT_PLATEAU ............................................ 2056
3.483 POS_MIN_HEIGHT_PROPER_PLAIN ..................................... 2060
3.484 POS_MIN_HEIGHT_PROPER_PLATEAU .................................. 2064
3.485 POS_MIN_HEIGHT_STEADY .............................................. 2068
3.486 POS_MIN_HEIGHT_STEADY_SEQUENCE ................................. 2072
3.487 POS_MIN_MAX_BUMP_ON_DECREASING_SEQUENCE ................. 2076
3.488 POS_MIN_MAX_DECREASING ........................................... 2080
3.489 POS_MIN_MAX_DECREASING_SEQUENCE ............................... 2084
3.490 POS_MIN_MAX_DIP_ON_INCREASING_SEQUENCE ..................... 2088
3.491 POS_MIN_MAX_INCREASING ............................................ 2092
3.492 POS_MIN_MAX_INCREASING_SEQUENCE ................................ 2096
3.493 POS_MIN_MAX_INFLEXION .............................................. 2100
3.494 POS_MIN_MAX_PEAK .................................................... 2104
3.495 POS_MIN_MAX_STRICTLY_DECREASING_SEQUENCE .................. 2108
3.496 POS_MIN_MAX_STRICTLY_INCREASING_SEQUENCE ................. 2112
3.497 POS_MIN_MAX_SUMMIT ............................................... 2116
3.498 POS_MIN_MAX_ZIGZAG ................................................. 2120
3.499 POS_MIN_MIN_BUMP_ON_DECREASING_SEQUENCE ................. 2124
3.500 POS_MIN_MIN_DECREASING ............................................ 2128
3.501 POS_MIN_MIN_DECREASING_SEQUENCE ................................ 2132
3.502 POS_MIN_MIN_DIP_ON_INCREASING_SEQUENCE .................... 2136
3.503 POS_MIN_MIN_GORGE .................................................. 2140
3.504 POS_MIN_MIN_INCREASING ............................................ 2144
3.505 POS_MIN_MIN_INCREASING_SEQUENCE ............................... 2148
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.506</td>
<td>POS_MIN_MIN_INFLEXION</td>
<td>2152</td>
</tr>
<tr>
<td>3.507</td>
<td>POS_MIN_MIN_STRICTLY_DECREASING_SEQUENCE</td>
<td>2156</td>
</tr>
<tr>
<td>3.508</td>
<td>POS_MIN_MIN_STRICTLY_INCREASING_SEQUENCE</td>
<td>2160</td>
</tr>
<tr>
<td>3.509</td>
<td>POS_MIN_MIN_VALLEY</td>
<td>2164</td>
</tr>
<tr>
<td>3.510</td>
<td>POS_MIN_MIN_ZIGZAG</td>
<td>2168</td>
</tr>
<tr>
<td>3.511</td>
<td>POS_MIN_SURF_BUMP_ON_DECREASING_SEQUENCE</td>
<td>2172</td>
</tr>
<tr>
<td>3.512</td>
<td>POS_MIN_SURF_DECREASING</td>
<td>2176</td>
</tr>
<tr>
<td>3.513</td>
<td>POS_MIN_SURF_DECREASING_SEQUENCE</td>
<td>2180</td>
</tr>
<tr>
<td>3.514</td>
<td>POS_MIN_SURF_DECREASING_TERRACE</td>
<td>2184</td>
</tr>
<tr>
<td>3.515</td>
<td>POS_MIN_SURF_DIP_ON_INCREASING_SEQUENCE</td>
<td>2188</td>
</tr>
<tr>
<td>3.516</td>
<td>POS_MIN_SURF_GORGE</td>
<td>2192</td>
</tr>
<tr>
<td>3.517</td>
<td>POS_MIN_SURF_INCREASING</td>
<td>2196</td>
</tr>
<tr>
<td>3.518</td>
<td>POS_MIN_SURF_INCREASING_SEQUENCE</td>
<td>2200</td>
</tr>
<tr>
<td>3.519</td>
<td>POS_MIN_SURF_INCREASING_TERRACE</td>
<td>2204</td>
</tr>
<tr>
<td>3.520</td>
<td>POS_MIN_SURF_INFLEXION</td>
<td>2208</td>
</tr>
<tr>
<td>3.521</td>
<td>POS_MIN_SURF_PEAK</td>
<td>2212</td>
</tr>
<tr>
<td>3.522</td>
<td>POS_MIN_SURF_PLAIN</td>
<td>2216</td>
</tr>
<tr>
<td>3.523</td>
<td>POS_MIN_SURF_PLATEAU</td>
<td>2220</td>
</tr>
<tr>
<td>3.524</td>
<td>POS_MIN_SURF_PROPER_PLAIN</td>
<td>2224</td>
</tr>
<tr>
<td>3.525</td>
<td>POS_MIN_SURF_PROPER_PLATEAU</td>
<td>2228</td>
</tr>
<tr>
<td>3.526</td>
<td>POS_MIN_SURF_STEADY</td>
<td>2232</td>
</tr>
<tr>
<td>3.527</td>
<td>POS_MIN_SURF_STEADY_SEQUENCE</td>
<td>2236</td>
</tr>
<tr>
<td>3.528</td>
<td>POS_MIN_SURF_STRICTLY_DECREASING_SEQUENCE</td>
<td>2240</td>
</tr>
<tr>
<td>3.529</td>
<td>POS_MIN_SURF_STRICTLY_INCREASING_SEQUENCE</td>
<td>2244</td>
</tr>
<tr>
<td>3.530</td>
<td>POS_MIN_SURF_SUMMIT</td>
<td>2248</td>
</tr>
<tr>
<td>3.531</td>
<td>POS_MIN_SURF_VALLEY</td>
<td>2252</td>
</tr>
<tr>
<td>3.532</td>
<td>POS_MIN_SURF_ZIGZAG</td>
<td>2256</td>
</tr>
<tr>
<td>3.533</td>
<td>POS_MIN_WIDTH_DECREASING_SEQUENCE</td>
<td>2260</td>
</tr>
<tr>
<td>3.534</td>
<td>POS_MIN_WIDTH_DECREASING_TERRACE</td>
<td>2264</td>
</tr>
<tr>
<td>3.535</td>
<td>POS_MIN_WIDTH_GORGE</td>
<td>2268</td>
</tr>
<tr>
<td>3.536</td>
<td>POS_MIN_WIDTH_INCREASING_SEQUENCE</td>
<td>2272</td>
</tr>
<tr>
<td>3.537</td>
<td>POS_MIN_WIDTH_INCREASING_TERRACE</td>
<td>2276</td>
</tr>
<tr>
<td>3.538</td>
<td>POS_MIN_WIDTH_INFLEXION</td>
<td>2280</td>
</tr>
<tr>
<td>3.539</td>
<td>POS_MIN_WIDTH_PEAK</td>
<td>2284</td>
</tr>
<tr>
<td>3.540</td>
<td>POS_MIN_WIDTH_PLAIN</td>
<td>2288</td>
</tr>
<tr>
<td>3.541</td>
<td>POS_MIN_WIDTH_PLATEAU</td>
<td>2292</td>
</tr>
<tr>
<td>3.542</td>
<td>POS_MIN_WIDTH_PROPER_PLAIN</td>
<td>2296</td>
</tr>
<tr>
<td>3.543</td>
<td>POS_MIN_WIDTH_PROPER_PLATEAU</td>
<td>2300</td>
</tr>
<tr>
<td>3.544</td>
<td>POS_MIN_WIDTH_STEADY_SEQUENCE</td>
<td>2304</td>
</tr>
<tr>
<td>3.545</td>
<td>POS_MIN_WIDTH_STRICTLY_DECREASING_SEQUENCE</td>
<td>2308</td>
</tr>
<tr>
<td>3.546</td>
<td>POS_MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE</td>
<td>2312</td>
</tr>
<tr>
<td>3.547</td>
<td>POS_MIN_WIDTH_SUMMIT</td>
<td>2316</td>
</tr>
<tr>
<td>3.548</td>
<td>POS_MIN_WIDTH_VALLEY</td>
<td>2320</td>
</tr>
<tr>
<td>3.549</td>
<td>POS_MIN_WIDTH_ZIGZAG</td>
<td>2324</td>
</tr>
<tr>
<td>3.550</td>
<td>SUM_HEIGHT_DECREASING_TERRACE</td>
<td>2328</td>
</tr>
<tr>
<td>3.551</td>
<td>SUM_HEIGHT_INCREASING_TERRACE</td>
<td>2334</td>
</tr>
</tbody>
</table>
CONTENTS

3.598 SUM_SURF_PEAK ........................................ 2528
3.599 SUM_SURF_PLAIN ......................................... 2532
3.600 SUM_SURF_PLATEAU ..................................... 2536
3.601 SUM_SURF_PROPER_PLAIN .............................. 2540
3.602 SUM_SURF_PROPER_PLATEAU ............................ 2544
3.603 SUM_SURF_STEADY ...................................... 2548
3.604 SUM_SURF_STEADY_SEQUENCE ......................... 2552
3.605 SUM_SURF_STRICTLY_DECREASING_SEQUENCE ...... 2556
3.606 SUM_SURF_STRICTLY_INCREASING_SEQUENCE ....... 2560
3.607 SUM_SURF_SUMMIT ...................................... 2564
3.608 SUM_SURF_VALLEY ....................................... 2570
3.609 SUM_SURF_ZIGZAG ....................................... 2574
3.610 SUM_WIDTH_DECREASING_SEQUENCE ................ 2580
3.611 SUM_WIDTH_DECREASING_TERRACE ................... 2584
3.612 SUM_WIDTH_GORGE ...................................... 2588
3.613 SUM_WIDTH_INCREASING_SEQUENCE ................. 2594
3.614 SUM_WIDTH_INCREASING_TERRACE .................... 2598
3.615 SUM_WIDTH_INFLEXION .................................. 2602
3.616 SUM_WIDTH_PEAK ....................................... 2606
3.617 SUM_WIDTH_PLAIN ....................................... 2610
3.618 SUM_WIDTH_PLATEAU ..................................... 2614
3.619 SUM_WIDTH_PROPER_PLAIN ............................ 2618
3.620 SUM_WIDTH_PROPER_PLATEAU .......................... 2622
3.621 SUM_WIDTH_STEADY_SEQUENCE ....................... 2626
3.622 SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE ..... 2630
3.623 SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE ...... 2634
3.624 SUM_WIDTH_SUMMIT ...................................... 2638
3.625 SUM_WIDTH_VALLEY ...................................... 2644
3.626 SUM_WIDTH_ZIGZAG ..................................... 2648

A Electronic Constraint Catalogue .......................... 2655

Bibliography .................................................. 2657

Index .................................................................. 2659
Preface

“Efficiency can only be attained through generality.”

This second volume of the Global Constraint Catalogue [6] is devoted to time-series constraints. Within the context of Constraint Programming, time-series constraints go back to the work of Goldin and Kanellakis [10]. This volume contains 626 constraints, which are explicitly described in terms of automata with accumulators [3]. Checkers and propagators for all these constraints were synthesised [4] from 22 transducers [15, 8, 12, 14] described in the second chapter of this report.

As in the first volume, the global constraints described in this second volume are not only accessible to humans, who can read the catalogue when searching for some information. It is also available to machines, which can read and interpret it. This is why there also exists an electronic version of this catalogue where one can get, for all time-series constraints, a complete description in terms of meta-data used in the first volume. In fact, unlike the first volume, all the meta-data of the electronic version as well as all text and figures of this second volume were automatically generated. While this second volume is by no means supposed to contain all possible time-series constraints, it contributes in the context of time-series constraints to the systematic reconstruction of the Global Constraint Catalogue that we have previously advocated [7]. This reconstruction is based on the following methodology:

• First reuse, adapt or come up with abstractions, which allow to concisely represent structures and properties of time series as abstract combinatorial objects. In our context these abstractions essentially correspond to:

   1. Transducers where letters of the output alphabet are interpreted as semantic letters indicating how to recognise pattern occurrences.

   2. Transducers glue matrices expressing the relationship between the prefix, the suffix and the full sequence passed to a transducer.

   3. Properties associated to regular expressions corresponding to fragments of the input language of our transducers.

• Second, create from these abstract combinatorial objects a data base of concrete combinatorial objects.
• Third, synthesise concrete code for various technologies, languages, tasks from this data base of concrete combinatorial objects. In this context, correctness and efficiency [11] of the synthesised code are essentially side product of:

  – The correctness of the formulae of our data base which is itself based on the wellformedness of our abstractions.
  – The generality behind our abstract combinatorial objects.

The time-series catalogue is done in the following way:

• All time-series constraints are now defined in a *compositional way* from a few basic constituents, i.e., patterns, features, aggregators, and predicates, which completely define the meaning of a constraint, where patterns are defined using regular expressions.

• Constraint names are now constructed in a systematic way as the *concatenation* of pattern name, feature name, and aggregation or predicate name.

• Given a pattern $p$, checkers and constraints are now *systematically synthesised* from a transducer that, given an input sequence over the input alphabet $\{<,=,>\}$, compares two adjacent values of a time-series and determines an output sequence over a output semantic alphabet describing how to recognise the occurrences of $p$.

• For each time-series constraint associated with a pattern $p$, the generation of an automaton with accumulators is completely driven by the transducer associated with pattern $p$ as well as by *decoration tables* describing for each semantic letter of the output alphabet of the transducers how to generate accumulator updates. Code optimisation is ensured by using decoration tables that depend on properties of the pattern, of the feature, and of the aggregator associated with the time-series constraint.

• Lower and upper bounds of characteristics of time-series that appear in the restriction slot of a time-series constraint are synthesised from a few *parameterised formulae* that only depend on a restricted set of characteristics of the regular expression associated with the pattern.

• Parametrised glue matrices are provided for each transducer that corresponds to *reversible time-series constraints*. A concrete glue matrix is given for each reversible time-series constraint.

• Linear invariants are systematically obtained by applying the Farkas Lemma [9] to the automata with accumulators that were synthesised. They consist of *linear constraints typically linking consecutive accumulator values*, e.g., see the legend of the second automaton of the $\text{MAX\_MAX\_PEAK}$, $\text{MAX\_RANGE\_DECREASING}$, $\text{MAX\_RANGE\_INCREASING}$, $\text{MAX\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE}$, $\text{MAX\_WIDTH\_STRICTLY\_INCREASING\_SEQUENCE}$, $\text{MIN\_MAX\_PEAK}$, $\text{MIN\_WIDTH\_PLAIN}$, $\text{MIN\_WIDTH\_PLATEAU}$, $\text{NB\_BUMP\_ON\_DECREASING\_SEQUENCE}$, $\text{NB\_DIP\_ON\_INCREASING\_SEQUENCE}$, $\text{NB\_GORGE}$, $\text{NB\_PEAK}$, $\text{NB\_SUMMIT}$, $\text{NB\_VALLEY}$,
and \texttt{NB	extunderscore ZIGZAG} constraints, which are generated even with non-linear accumulator updates. Missing linear invariants will be completed later on.

- Last but not least, time-series constraints were used for generating time-series verifying a conjunction of constraints both in the context of Constraint Programming \cite{2} and in the context of Linear Programming. In this last case a linear reformulation is given in \cite{1}.

- In the context of sequential pattern mining, time-series constraint checkers can be used to identify and extract patterns from fixed sequences. While the time-series catalogue may need to be extended in order to capture more patterns, having a possibly large set of fixed time-series constraints is a natural safeguard to prevent overfitting when dealing with few sequences, at a price of not finding patterns that are not covered by the catalogue.

- Finally, both SICStus and MiniZinc code are synthesised. The later allows using time series constraints on many plate forms such as Choco, Gecode, ORtools, Cplex or Gurobi and is available from the \textit{Electronic Constraint Catalogue} in Appendix A of this document.

The catalogue contains the following types of figures, which were all synthesised by computer programs producing \textit{TikZ} \cite{16} code:

- Figures representing the logo of each pattern.

- Figures for visualising the transducer associated with each pattern. The following scheme was consistently used over all transducers:
  - An arc labelled by the output symbol \texttt{out}, \texttt{maybe}, \texttt{found}, or \texttt{in} is respectively coloured in red, orange, blue, and violet.
  - If all input transitions of a state have the same colour, then the state uses this colour, otherwise it is coloured in black.
  - The name of a state is preceded by all input symbols of the transitions that enter this state. We use special characters for representing combinations of input symbols, e.g. $\leq$ for $<$ and $\texttt{=}$. 

- Figures giving time series that respectively achieve lower and upper bounds of the \texttt{VALUE} argument of a time-series constraint.

- Figures illustrating each constraint on a relevant time series.

- Figures providing the synthesised automaton with accumulators associated with each constraint. Such automata use the same graphical scheme as the transducers from which they were generated.

In order to see PDF annotations and attached files, you are advised to use Adobe \textit{Reader} with the pdf version of the catalogue.\footnote{Since we are using the \LaTeX{} packages \texttt{pdftex} and \texttt{attachfile} and since most PDF viewers do not support PDF annotations or attachments. If you do not see on your screen a small}
yellow bullet at the beginning of this paragraph, you are using a PDF viewer that does not fully support PDF annotations.

To get started the reader should first read the introduction of the overview chapter 2, which presents the notions of pattern, seed transducer and glue matrix used throughout this document, as well as the conventions used for drawing time-series.

The initial work on synthesising automata with accumulators from transducers was done by N. Beldiceanu, R. Douence, and H. Simonis. The generation of the catalogue was done by H. Simonis except, for the examples and figures, which were handled by N. Beldiceanu. The code generation part dealing with constraint checkers was rewritten by M. Carlsson for getting optimised checkers that do not use any constraint. The creation of dedicated decoration tables that depend on properties of patterns, features, and aggregators was done by E. Arafailova under the supervision of N. Beldiceanu and R. Douence [2]. The bounds were done by E. Arafailova under the supervision of N. Beldiceanu; the generation of the corresponding code was done by H. Simonis. The generation of linear invariants [13, 2] and the parametrised glue matrices [5, 1] were done by M. A. Francisco Rodríguez under the supervision of P. Flener and J. Pearson.

The authors have benefit from the following support:

- E. Arafailova and R. Douence were supported by the European H2020 FETPROACT-2014 project “GRACeFUL” (project number 640954).
- From 2014 to 2016 N. Beldiceanu was partially supported by the Gaspard Monge Program for Optimization and Operations Research (PGMO), as well as by the European H2020 FETPROACT-2014 project “GRACeFUL”.
- M. Carlsson was supported by SICS.
- P. Flener, J. Pearson and M. A. Francisco Rodríguez were supported by grants 2011-6133 and 2012-4908 of the Swedish Research Council (VR).
- During 2014 H. Simonis was partially supported by a senior researcher chair by the Région Pays de la Loire and by the EU FET grant ICON (project number 284715). From 2014 to 2016 H. Simonis was also supported by Science Foundation Ireland under Grant Number SFI/10/IN.1/I3032. The INSIGHT Centre for Data Analytics is supported by Science Foundation Ireland under Grant Number SFI/12/RC/2289.

Readers may send their suggestions via email to the corresponding author with catalogue as subject.

Cork, Ireland, Nantes, France, Uppsala, Sweden, September 2016
— EA, NB, RD, MC, PF, MAFR, JP, HS
1

Describing a global constraint

This chapter, taken from the first volume of the global constraint catalogue [6], recalls how to describe a global constraint and its arguments.

Since global constraints have to receive their arguments in some form, no matter whether we use the graph-based or automaton-based description, we start by describing the abstract data types that we use in order to specify the arguments of a global constraint. These abstract data types are not related to any specific programming language like Caml, C, C++, Java, or Prolog. If one wants to focus on a specific language, then one has to map these abstract data types to the data types that are available within the considered language. Second we describe all the restrictions that one can impose on the arguments of a global constraint. Finally, we show how to use these ingredients in order to declare the arguments of a global constraint.

1.1 Basic data types

We use the following basic data types:

- **atom** corresponds to an atom. Predefined atoms are MININT and MAXINT, which respectively correspond to the smallest integer and largest integer.

- **int** corresponds to an integer value.

- **dvar** corresponds to a domain variable. A domain variable $V$ is a variable that will be assigned an integer value taken from an initial finite set of integer values
1. DESCRIBING A GLOBAL CONSTRAINT

denoted by \( \text{dom}(V) \). \( \underline{V} \) and \( \overline{V} \) respectively denote the minimum and maximum values of \( \text{dom}(V) \).

- **fdvar** corresponds to a *possibly unbounded domain variable*. A *possibly unbounded domain variable* is a variable that will be assigned an integer value from an initial finite set of integer values denoted by \( \text{dom}(V) \) or from \( ] - \infty, +\infty[ \).

- **sint** corresponds to a *finite set of integer values*.

- **svar** corresponds to a *set variable*. A *set variable* \( V \) is a variable that will be assigned to a *finite set* of integer values. Its *lower bound* \( \underline{V} \) denotes the set of integer values that for sure belong to \( V \), while its *upper bound* \( \overline{V} \) denotes the set of integer values that may belong to \( V \). Let \( \text{dom}(V) = \{v_1, \ldots, v_n, v_{n+1}, \ldots, v_m\} \) be a shortcut for combining the lower and upper bounds of \( V \) in a single notation:
  - **Bold** values designate those values that only belong to \( \underline{V} \).
  - **Italic** values indicate those values that belong only to \( \overline{V} \).

- **mint** corresponds to a *multiset of integer values*.

- **mvar** corresponds to a *multiset variable*. A *multiset variable* is a variable that will be assigned to a *multiset of integer values*.

- **real** corresponds to a *real number*.

- **rvar** corresponds to a *real number variable*. A *real number variable* is a variable that will be assigned a *real number* taken from an initial finite set of intervals. A real number is usually represented by an interval of two floating-point numbers.

1.2 Compound data types

We use the following *compound data types*:

- **list(\( T \))** corresponds to a list of elements of type \( T \), where \( T \) is a basic or compound data type.

- **collection(\( A_1, A_2, \ldots, A_n \))** corresponds to a collection of ordered items, where each item consists of \( n > 0 \) attributes \( A_1, A_2, \ldots, A_n \). Each attribute is an expression of the form \( a - T \), where \( a \) is the *name* of the attribute and \( T \) the *type* of the attribute (a basic or compound data type). All names of the attributes of a given collection should be distinct and different from the keyword *key*, which corresponds to an implicit\(^1\) attribute. The value of the *key* attribute is the position of an item within the collection. The first item of a collection is associated with position 1.

\(^1\)This attribute is not explicitly defined.
1.3. RESTRICTIONS

The following notation is used for instantiated arguments:

- A list of elements \( e_1, e_2, \ldots, e_n \) is denoted by \([e_1, e_2, \ldots, e_n]\).
- A finite set of integers \( i_1, i_2, \ldots, i_n \) is denoted by \(\{i_1, i_2, \ldots, i_n\}\).
- A collection of \( n \) items, each item having \( m \) attributes, is denoted by \(\langle a_{11}, a_{12}, \ldots, a_{mn}\rangle\). Each item is separated from the previous item by a comma. If the items of the collection involve a single attribute \( a_{11} \), then \(\langle v_{11}, v_{21}, \ldots, v_{1n}\rangle\) can be used as a shortcut for \(\langle a_{11}, a_{12}, \ldots, a_{1n}\rangle\).
- The \(i^{th}\) item of a collection \( c \) is denoted by \(c[i]\).
- The value of the attribute \( a \) of the \(i^{th}\) item of a collection \( c \) is denoted by \(c[i].a\). Note that, within an arithmetic expression, we can use the shortcut \(c[i]a\) when the collection \( c \) involves a single attribute.
- The number of items of a collection \( c \) is denoted by \(|c|\).

1.3 Restrictions

When defining the arguments of a global constraint, it is often the case that one needs to express additional conditions that refine the type declarations of its arguments. For this purpose we provide restrictions that allow the specification of these additional conditions. Currently the list of restrictions is:

- **in_list(Arg,ListAtoms)**
  - Arg is an argument of type atom,
  - ListAtoms is a non-empty list of distinct atoms.

This restriction enforces Arg to be one of the atoms specified in ListAtoms.

- **in_list(Arg,Attr,ListIntOrAtom)**
  - Arg is an argument of type collection,
  - Attr is an attribute of type int or atom of Arg,
  - If Attr is an attribute of type int, then ListIntOrAtom is a non-empty list of distinct integers; if Attr is an attribute of type atom, then ListIntOrAtom is a non-empty list of distinct atoms.

This restriction enforces, for all items of Arg, the attribute Attr to take its value within ListIntOrAtom.

- **in_attr(Arg1,Attr1,Arg2,Attr2)**
  - Arg1 is an argument of type collection,
1. Describing a Global Constraint

- Attr1 is an attribute of type dvar or int of Arg1,
- Arg2 is an argument of type collection,
- Attr2 is an attribute of type int of Arg2.

Let $S_2$ denote the set of values assigned to the Attr2 attributes of the items of Arg2. This restriction enforces the following condition: for all items of Arg1, the attribute Attr1 takes its value in the set $S_2$.

- **distinct(Arg, Attrs)**
  - Arg is an argument of type collection,
  - Attrs is an attribute of type int or dvar, or a (possibly empty) list of distinct attributes of type int or dvar of Arg.

For all pairs of distinct items of Arg this restriction enforces that there be at least one attribute specified by Attrs with two distinct values. If Attrs is the empty list, then all items of Arg should be distinct.

- **increasing_seq(Arg, Attrs)**
  - Arg is an argument of type collection,
  - Attrs is an attribute of type int or a list of distinct attributes of type int of Arg.

Let $n$ and $m$ respectively denote the number of items of Arg, and the number of attributes of Attrs. For item $i$ of Arg let $t_i$ denote the tuple of values $\langle v_{i,1}, v_{i,2}, \ldots, v_{i,m}\rangle$ where $v_{i,j}$ is the value of attribute $j$ of Attrs of item $i$ of Arg. The restriction enforces a strict lexicographical ordering on the tuples $t_1, t_2, \ldots, t_n$.

- **non_increasing_size(Arg, Attr)**
  - Arg is an argument of type collection,
  - Attr is an attribute of type collection of Arg.

This restriction enforces for each pair of consecutive items $\text{Arg}[i], \text{Arg}[i+1]$ that the number of items of $\text{Arg}[i].\text{Attr}$ is greater than or equal to the number of items of the collection $\text{Arg}[i+1].\text{Attr}$.

- **required(Arg, Attrs)**
  - Arg is an argument of type collection,
  - Attrs is an attribute or a list of distinct attributes of Arg.

This restriction enforces that all attributes denoted by Attrs be explicitly used within all items of Arg.

This restriction is usually systematically used for every attribute of a collection. It is not used when some attributes may be implicitly defined according to other
attributes. In this context, we use the require at least restriction, which we now introduce.

- **require at least**(_Atleast_, _Arg_, _Attrs_)
  
  - _Atleast_ is a positive integer,
  - _Arg_ is an argument of type collection,
  - _Attrs_ is a non-empty list of distinct attributes of _Arg_. The length of this list should be strictly greater than _Atleast_.

This restriction enforces that at least _Atleast_ attributes of the list _Attrs_ be explicitly used within all items of _Arg_.

- **same size**(_Arg_, _Attr_)
  
  - _Arg_ is an argument of type collection,
  - _Attr_ is an attribute of type collection of _Arg_.

This restriction enforces that all collections of _Attr_ have the same number of items.

- **Term_1 Comparison Term_2**
  
  - _Term_1 is a term. A term is an expression that can be evaluated to one or possibly several integer values.
  - Comparison is one of the comparison operators \(\leq, \geq, <, >, =, \neq\).
  - _Term_2 is a term.

Let \(v_{1,1}, v_{1,2}, \ldots, v_{1,n_1}\) and \(v_{2,1}, v_{2,2}, \ldots, v_{2,n_2}\) be the values respectively associated with _Term_1 and _Term_2. The restriction _Term_1 Comparison _Term_2 enforces \(v_{1,i} \text{ Comparison } v_{2,j}\) to hold for every \(i \in [1, n_1]\) and every \(j \in [1, n_2]\).

A term is one of the following expressions:

- \(e\), where _e_ is an integer. The corresponding value is _e_.
- \(|c|\), where _c_ is an argument of type collection. The value of \(|c|\) is the number of items of the collection _c_.
- **first(c.a)**: _first(c.a)_ denotes the value assigned to the attribute _a_ of the first item of the collection _c_. It is equal to 0 if the collection is empty.
- **last(c.a)**: _last(c.a)_ denotes the value assigned to the attribute _a_ of the last item of the collection _c_. It is equal to 0 if the collection is empty.
- **sum(c.a)**: _sum(c.a)_ denotes the sum of the values assigned to the attribute _a_ of the collection _c_; it is equal to 0 if the collection is empty.
- **sum(ℓ)**: _sum(ℓ)_ , where _ℓ_ is a list of collection attributes, each of the form _c_i.a_i_ (with _i_ \(\in [1, n]\)), is the sum of the values assigned to the attributes _a_i_ of the collections _c_i_ (with _i_ \(\in [1, n]\)).
1. DESCRIBING A GLOBAL CONSTRAINT

- **range(c.a)**: \text{range}(c.a) denotes the difference between the maximum value and minimum value plus one of the values assigned to the attribute \(a\) of \(c\); it is equal to 0 if the collection is empty.

- **range(\ell)**: \text{range}(\ell), where \(\ell\) is a list of collection attributes, each of the form \(c_i.a_i\) (with \(i \in [1, n]\)), is the difference between the maximum value and minimum value plus one of the values assigned to the attributes \(a_i\) of the \(c_i\) (with \(i \in [1, n]\)).

- **minval(c.a)**: \text{minval}(c.a) denotes the minimum of the values assigned to the attribute \(a\) of the collection denoted by \(c\); it is equal to 0 if the collection is empty.

- **minval(\ell)**: \text{minval}(\ell), where \(\ell\) is a list of collection attributes, each of the form \(c_i.a_i\) (with \(i \in [1, n]\)), is the minimum of the values assigned to the attributes \(a_i\) of the \(c_i\) (with \(i \in [1, n]\)).

- **maxval(c.a)**: \text{maxval}(c.a) denotes the maximum of the values assigned to the attribute \(a\) of \(c\); it is equal to 0 if the collection is empty.

- **maxval(\ell)**: \text{maxval}(\ell), where \(\ell\) is a list of collection attributes, each of the form \(c_i.a_i\) (with \(i \in [1, n]\)), is the maximum of the values assigned to the attributes \(a_i\) of the \(c_i\) (with \(i \in [1, n]\)).

- **nval(c.a)**: \text{nval}(c.a) denotes the number of distinct values over the values assigned to the attribute \(a\) of \(c\); it is equal to 0 if the collection is empty.

- **nval(\ell)**: \text{nval}(\ell), where \(\ell\) is a list of collection attributes, each of the form \(c_i.a_i\) (with \(i \in [1, n]\)), is the number of distinct values over the values assigned to the attributes \(a_i\) of the \(c_i\) (with \(i \in [1, n]\)).

- **prod(c.a)**: \text{prod}(c.a) denotes the product of the values assigned to the attribute \(a\) of \(c\); it is equal to 1 if the collection is empty.

- **prod(\ell)**: \text{prod}(\ell), where \(\ell\) is a list of collection attributes, each of the form \(c_i.a_i\) (with \(i \in [1, n]\)), is the product of the values assigned to the attributes \(a_i\) of the \(c_i\) (with \(i \in [1, n]\)).

- **pfdc(c.a)**: \text{pfdc}(c.a), where \text{pfdc} is a pure functional dependency constraint of the form \text{pfdc}(v, col) that computes a value \(v\) from a collection of variables \(col\), and where \(c.a\) is a collection with attribute \(a\) denotes the \text{pfdc} of the values assigned to the attribute \(a\) of \(c\); it is equal to 0 if the collection \(c\) is empty.

- **pfdc(\ell)**: \text{pfdc}(\ell), where \text{pfdc} is a pure functional dependency constraint of the form \text{pfdc}(v, col) that computes a value \(v\) from a collection of variables \(col\), and where \(\ell\) is a list of collection attributes, each of the form \(c_i.a_i\) (with \(i \in [1, n]\)), is the \text{pfdc} of the values assigned to the attributes \(a_i\) of the \(c_i\) (with \(i \in [1, n]\)).
1.3. RESTRICTIONS

- \( t \), where \( t \) is an argument of type \( \text{int} \). The value of \( t \) is the value of the corresponding argument.

- \( v \), where \( v \) is an argument of type \( \text{dvar} \). The value of \( v \) is the value assigned to variable \( v \). Note that restrictions are defined on the ground instance of a global constraint.

- \( s \), where \( s \) is an argument of type \( \text{sint} \) or \( \text{svar} \). The values denoted by \( s \) are all the values of the corresponding set.

- \( c.a \), where \( c \) is an argument of type \( \text{collection} \) and \( a \) an attribute of \( c \) of type \( \text{int} \) or \( \text{dvar} \). The values denoted by \( c.a \) are all the values corresponding to attribute \( a \) for the items of \( c \). When \( c.a \) designates a domain variable we consider the value assigned to that variable.

- \( c.a \), where \( c \) is an argument of type \( \text{collection} \) and \( a \) an attribute of \( c \) of type \( \text{sint} \) or \( \text{svar} \). The values denoted by \( c.a \) are all the values belonging to the sets corresponding to attribute \( a \) for the items of \( c \). When \( c.a \) designates a set variable we consider the values that belong to that set.

- \( t_1 \text{ op } t_2 \), where \( t_1 \) and \( t_2 \) are terms and \( \text{op} \) one of the operators \( +, -, \ast, / \) or \( \text{mod} \).\(^2\) Let \( \mathcal{V}_1 \) and \( \mathcal{V}_2 \) denote the sets of values respectively associated with the terms \( t_1 \) and \( t_2 \). The set of values associated with \( t_1 \text{ op } t_2 \) is \( \mathcal{V}_{12} = \{ v : v = v_1 \text{ op } v_2, v_1 \in \mathcal{V}_1, v_2 \in \mathcal{V}_2 \} \).

- \(|t|\), where \( t \) is a term. Let \( \mathcal{V} \) denote the set of values associated with the term \( t \). The set of values associated with \(|t|\) is \( \mathcal{V} = \{ v : v = |val|, val \in \mathcal{V} \} \).

- \( \min_{v \in [\ell,u]}(t) \), where \( v \) is a variable that occurs in the term \( t \), and \( \ell, u \) are two terms. The value of \( \min_{v \in [\ell,u]}(t) \) is the smallest value among all possible values of the term \( t \) where \( v \) is assigned a value in the interval \( [\ell,u] \).

- \( \max_{v \in [\ell,u]}(t) \), where \( v \) is a variable that occurs in the term \( t \), and \( \ell, u \) are two terms. The value of \( \max_{v \in [\ell,u]}(t) \) is the largest value among all possible values of the term \( t \) where \( v \) is assigned a value in the interval \( [\ell,u] \).

- Using the \( \lor \) connector we can express a disjunction between two restrictions.

- Using the \( \Rightarrow \) connector we can express an implication between a restriction (or a conjunction of restrictions) and a restriction.

- We can also use a constraint \( C \) of the catalogue for expressing a restriction as long as that constraint is not defined according to the constraint under consideration. The constraint \( C \) should have a graph-based or automaton-based description so that its meaning is explicitly defined.

Within the restriction slot it is possible to define local variables to designate intermediate expressions. This is done by using the \textbf{where} statement after the set of restrictions.

\(^2\) floor division, that is a division in which the result is rounded to the nearest integer that is smaller or equal.
1.4 Declaring a global constraint

Declaring a global constraint consists of providing the following information:

- A term constraint \((A_1, A_2, \ldots, A_n)\), where constraint corresponds to the name of the global constraint and \(A_1, A_2, \ldots, A_n\) to its arguments.

- A possibly empty list of type declarations, where each declaration has the form type:type_declaration; type is the name of the new type we define and type_declaration is a basic data type, a compound data type, or a previously defined type.

- An argument declaration \(A_1:T_1, A_2:T_2, \ldots, A_n:T_n\) giving for each argument \(A_1, A_2, \ldots, A_n\) of the global constraint constraint its type. Each type is a basic data type, a compound data type, or a type that was declared in the list of type declarations.

- A possibly empty list of restrictions, where each restriction is one of the restrictions described in Section 1.3.

1.5 Keywords

This section lists the keywords used in the time-series catalogue.

- added: Denotes the fact that, even if the same constant is added to all variables of the VARIABLES collection, the corresponding constraint still holds.

- functional dependency: A constraint that allows for representing a functional dependency between possibly several domain variables and a single domain variable. A sequence of variables \(X_1, X_2, \ldots, X_n\) is said to functionally determine another variable \(Y\) if and only if each potential tuple of values of \(X_1, X_2, \ldots, X_n\) is associated with exactly one potential value of \(Y\) (i.e., \(Y\) is a function of \(X_1, X_2, \ldots, X_n\)).

- reversed: Denotes the fact that, even if the variables of the VARIABLES collection are reversed, the corresponding constraint still holds.

- reverse of a constraint: A constraint which has a reverse constraint, where the reverse is defined in the following way. Consider two constraints \(c(col, r_1, \ldots, r_n)\) and \(c'(col, r_1, \ldots, r_n)\) for which, in both cases, the argument \(col\) is a collection of items that functionally determines all the other arguments \(r_1, \ldots, r_n\).

The constraint \(c'\) is the reverse constraint of constraint \(c\) if, for any collection of items \(col\), we have the equivalence \(c(col, r_1, \ldots, r_n) \Leftrightarrow c'(col^{rev}, r_1, \ldots, r_n)\), where \(col^{rev}\) denotes the collection \(col\) where the items of the collection are reversed. When constraints \(c\) and \(c'\) are identical we say that constraint \(c\) is its own reverse.
Patterns, Seed Transducers, Glue matrices and Decoration Tables

Contents

2.1 Patterns, their Seed Transducers and their Parametrised Glue Matrices ........................................ 13
  2.1.1 BUMP_ON_DECREASING_SEQUENCE ............................... 15
  2.1.2 DECREASING ...................................................... 19
  2.1.3 DECREASING_SEQUENCE ......................................... 21
  2.1.4 DECREASING_TERRACE ........................................ 24
  2.1.5 DIP_ON_INCREASING_SEQUENCE ............................... 27
  2.1.6 GORGE ............................................................. 31
  2.1.7 INCREASING ...................................................... 35
  2.1.8 INCREASING_SEQUENCE ......................................... 37
  2.1.9 INCREASING_TERRACE ......................................... 40
  2.1.10 INFLEXION ...................................................... 43
  2.1.11 PEAK .............................................................. 45
  2.1.12 PLAIN ............................................................. 48
  2.1.13 PLATEAU .......................................................... 51
  2.1.14 PROPER_PLAIN .................................................. 54
  2.1.15 PROPER_PLATEAU .............................................. 57
  2.1.16 STEADY ........................................................... 60
  2.1.17 STEADY_SEQUENCE ............................................ 62
  2.1.18 STRICTLY_DECREASING_SEQUENCE ......................... 64
  2.1.19 STRICTLY_INCREASING_SEQUENCE ......................... 67
  2.1.20 SUMMIT .......................................................... 70
  2.1.21 VALLEY .......................................................... 74
Patterns focus on the topological aspect of subsequences of a time series. They are defined by two components:

- First a regular expression over the alphabet \( \{<,=,>\} \).
- Second two non-negative integers \( b \) and \( a \) that are intended to respectively delete a prefix and a suffix of the pattern that should be discarded for computing a characteristic of an occurrence of pattern.

Given a pattern and a time series \( x_0, x_1, \ldots, x_{n-1} \) of integer constants, called the input values and forming the input sequence, a single integer is computed as follows:

I. Compare each pair of adjacent input values in order to build a sequence \( s_0, s_1, \ldots, s_{n-2} \) of signature values over the alphabet \( \{<,=,>\} \), as follows:
   \[
   (x_i < x_{i+1} \iff s_i = '<') \land (x_i = x_{i+1} \iff s_i = '=') \land (x_i > x_{i+1} \iff s_i = '>').
   \]
   The signature values form the signature sequence.

II. Within the signature sequence, find all maximal words matching the regular expression associated with the pattern.

III. For each found pattern occurrence discard its prefix and suffix of length \( b \) and \( a \) to obtain an integer sequence \( e \) for which we compute an integer feature value, so that we obtain a feature sequence. The features we currently consider are one, width, surface, min, max, and range, and correspond respectively to the value 1, to the number of elements of \( e \), to \( \sum_{i \in e} x_i \), to \( \min_{i \in e} x_i \), to \( \max_{i \in e} x_i \), and to \( \max_{i \in e} x_i - \min_{i \in e} x_i \).

IV. Aggregate the values of the feature sequence into a single integer value. The aggregators we currently consider are taking the sum (\( \text{Sum} \)), taking the minimum (\( \text{Min} \)), and taking the maximum (\( \text{Max} \)). The feature one only makes sense with the \( \text{Sum} \) aggregator.

**Definition 1** (s-occurrence, i-occurrence, e-occurrence). Given an input sequence \( x_0, x_1, \ldots, x_{n-1} \), its signature sequence \( S = s_0, s_1, \ldots, s_{n-2} \), a pattern \( \langle r, b, a \rangle \), and a non-empty signature subsequence \( s_i, s_{i+1}, \ldots, s_j \), with \( 0 \leq i \leq j \leq n - 2 \), forming a maximal word that matches \( r \), the s-occurrence \( \langle i..j \rangle \) is the index sequence \( i, i+1, \ldots, j \); the i-occurrence \( \langle (i+b)..j \rangle \) is the index sequence \( i+b, \ldots, j \); and the e-occurrence \( \langle [(i+b)..(j+1-a)] \rangle \) is the index sequence \( i+b, \ldots, j+1-a \).

Figures representing time-series use the following convention for denoting i-occurrences, s-occurrences and e-occurrences:

- Positions belonging to an i-occurrence are represented by a red circle ⚫.
- Positions belonging to a s-occurrence, but not to an i-occurrence, are represented by a circle ◦.
• Positions of an occurrence of pattern corresponding to an e-occurrence have a background coloured in yellow.

When an i-occurrence and an e-occurrence of two consecutive occurrences of a pattern coincide a red circle • is used.

The name of a constraint is defined by the concatenation of the aggregator function, the feature name and the pattern. For each pattern a so-called seed transducer recognises all pattern occurrences.

**Definition 2** (seed transducer). A seed transducer is a transducer for which all states are accepting with input alphabet \{<, =, >\}, output alphabet, also-called semantic alphabet, \{out, maybe_b, out, found, maybe_b, in, out, found\}.

**Definition 3** (t-occurrence). Given an input sequence \(s\) over the input alphabet \{<, =, >\} the t-occurrence of \(s\) wrt a given seed transducer corresponds to the indices of the semantic letters of a maximum word within the generated output sequence that matches ‘maybe_b, found_e | maybe_b, found(maybe, in)\’.

**Definition 4** (wellformedness). A seed transducer is well formed if (1) all output sequences it produces are recognised by the finite deterministic automaton depicted by Figure 2.1, and if (2) the t-occurrence for any input sequence \(s\) coincides with the i-occurrence of the pattern for \(s\).

Wellformedness of seed transducers guarantees that the automata with accumulators obtained by applying the decoration tables for replacing semantic letters of the output alphabet by accumulator updates compute the expected result [4].

![Figure 2.1: Automaton describing the output language of a well-formed transducer; states \(\notin, ?, \in\) respectively mean that (1) we are outside an occurrence of pattern, (2) we are potentially inside an occurrence of pattern, (3) we are inside an occurrence of pattern that is not yet finished.](image)

Following the presentation in [5, 1], we introduce the notion of glue matrix. A glue matrix for a reversible time-series constraint \(c\) specified by a seed transducer \(\sigma\), an aggregator \(g\) and a feature \(f\) is a matrix indexed by the states of \(\sigma\) and \(\sigma^{rev}\) and parametrised by \(g\) and \(f\), where \(\sigma^{rev}\) is a seed transducer of the reverse of constraint.
Consider the automaton $A_{\sigma}$ (respectively $A_{\sigma}^{rev}$) with accumulators $R$, $C$ and $D$ obtained by applying the decoration table 2.35 to the seed transducer $\sigma$ (respectively $\sigma^{rev}$). Let $A_{\sigma}$ reach state $\overrightarrow{Q}$ and accumulator values $\langle \overrightarrow{C}, \overrightarrow{D}, \overrightarrow{R} \rangle$ on a prefix of a word $w$. Similarly let $A_{\sigma}^{rev}$ reach state $\overleftarrow{Q}$ and accumulator values $\langle \overleftarrow{C}, \overleftarrow{D}, \overleftarrow{R} \rangle$ on the reverse of the corresponding suffix of $w$. The value returned by $A_{\sigma}$ on the entire word $w$ is $\phi_{g}(\overrightarrow{R}, \overrightarrow{R}, \Gamma)$, where $\Gamma$ is the entry of the glue matrix corresponding to row $\overrightarrow{Q}$ and column $\overleftarrow{Q}$. That entry contains an expression parametrised by $g$ and $f$ involving a subset of the accumulators $\overrightarrow{C}, \overrightarrow{D}, \overleftarrow{C}, \overleftarrow{D}$. 
2.1 Patterns, their Seed Transducers and their Parametrised Glue Matrices

Table 2.1: (Left) Features: identity, minimum, and maximum values; the operators \( \phi_f \) and \( \delta_f^i \) recursively define the feature value \( v_u \) of a time series \( x_\ell, \ldots, x_u \) by \( v_\ell = \phi_f(\text{id}_f, \delta_f^\ell) \) and \( v_i = \phi_f(v_{i-1}, \delta_f^i) \) for \( i > \ell \), where \( \delta_f^i \) is the contribution of \( x_i \) to \( v_u \); \( n \) stands for the length of the time-series. (Right) Aggregators: operators and identity values relative a feature \( f \).
### Table 2.2: Default value for a predicate $\theta$ used in the decoration table 2.36

<table>
<thead>
<tr>
<th>Predicate $\theta$</th>
<th>$\text{default}_\theta$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\leq$</td>
<td>$-\infty$</td>
</tr>
<tr>
<td>$=$</td>
<td>$x$ (free variable)</td>
</tr>
<tr>
<td>$\geq$</td>
<td>$+\infty$</td>
</tr>
</tbody>
</table>

Table 2.2: Default value for a predicate $\theta$ used in the decoration table 2.36
2.1. PATTERNS, THEIR SEED TRANSDUCERS AND THEIR PARAMETRISED GLUE MATRICES

2.1.1 BUMP_ON_DECREASING_SEQUENCE

Given a sequence $S$ over the alphabet $\{<, =, >\}$, an occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence of $S$ which matches the regular expression $>>><>$. Part (A) and parts (B,C) of Figure 2.2 respectively depict the seed transducer associated with the BUMP_ON_DECREASING_SEQUENCE pattern as well as one example of its execution on a time series.
2. PATTERNS, SEED TRANSDUCERS, GLUE MATRICES AND DECORATION TABLES

Figure 2.2: (A) Seed transducer for the BUMP_ON_DECREASING_SEQUENCE pattern: $>>\textless\textless\textgreater$ with $b = 2$ and $a = 1$; (B) Illustrating the execution of the seed transducer on a time series (within the output o, o, m, f are shortcut for out, out, maybe, found); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.
2.1. PATTERNS, THEIR SEED TRANSDUCERS AND THEIR PARAMETRISED GLUE MATRICES

\[ \begin{align*}
C & \leftarrow \text{default}_{gf} \\
D & \leftarrow \text{id}_f \\
R & \leftarrow \text{default}_{gf}
\end{align*} \]

\[ \phi_g(R, C) \]

\[ \begin{align*}
\leq & \Rightarrow \\
> & \Rightarrow \\
> & \Rightarrow
\end{align*} \]

Figure 2.3: Parametrised automaton for any functional time-series constraints of the BUMP_ON_DECREASING_SEQUENCE pattern obtained by applying the decoration table 2.35 to the corresponding transducer.
Figure 2.4: Parametrised automaton for any predicate time-series constraints of the \textsc{bump_on_decreasing_sequence} pattern obtained by applying the decoration table 2.36 to the corresponding transducer, where $\theta$ is the predicate (i.e., $\leq$, $=\geq$) used for comparing two consecutive feature values.
2.1. PATTERNS, THEIR SEED TRANSDUCERS AND THEIR PARAMETRISED GLUE MATRICES

2.1.2 DECREASING

Given a sequence $S$ over the alphabet $\{<,=,>\}$, an occurrence of the pattern DECREASING is the subsequence of $S$ which matches the regular expression $>$. Part (A) and parts (B,C) of Figure 2.5 respectively depict the seed transducer associated with the DECREASING pattern as well as one example of its execution on a time series.

![Seed transducer for the DECREASING pattern](image)

**Figure 2.5:** (A) Seed transducer for the DECREASING pattern: $>$ with $b = 0$ and $a = 0$; (B) Illustrating the execution of the seed transducer on a time series (within the output $o$, $f$ are shortcut for out, found); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.

\[
\phi_g(C, \overline{C})
\]

Table 2.3: Parametrised glue matrix for any $g \_ f$ DECREASING constraint
Figure 2.6: Parametrised automaton for any functional time-series constraints of the DECREASING pattern obtained by applying the decoration table 2.35 to the corresponding transducer

Figure 2.7: Parametrised automaton for any predicate time-series constraints of the DECREASING pattern obtained by applying the decoration table 2.36 to the corresponding transducer, where $\theta$ is the predicate (i.e., $\leq$, $=\$, $\geq$) used for comparing two consecutive feature values
2.1. PATTERNS, THEIR SEED TRANSDUCERS AND THEIR PARAMETRISED GLUE MATRICES

2.1.3 DECREASING_SEQUENCE

Given a sequence $S$ over the alphabet $\{<, =, >\}$, an occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence of $S$ which matches the regular expression $>(> | =)^* > | >$. Part (A) and parts (B,C) of Figure 2.8 respectively depict the seed transducer associated with the DECREASING_SEQUENCE pattern as well as one example of its execution on a time series.

Figure 2.8: (A) Seed transducer for the DECREASING_SEQUENCE pattern: $>(> | =)^* > | >$ with $b = 0$ and $a = 0$; (B) Illustrating the execution of the seed transducer on a time series (within the output o, f m, o are shortcut for out, found maybe, out); (C) Illustrating the relation between the s-occurrence, the i-occurrence and the e-occurrence.
Figure 2.9: Parametrised automaton for any functional time-series constraints of the DECREASING_SEQUENCE pattern obtained by applying the decoration table 2.35 to the corresponding transducer.

Figure 2.10: Parametrised automaton for any predicate time-series constraints of the DECREASING_SEQUENCE pattern obtained by applying the decoration table 2.36 to the corresponding transducer, where $\theta$ is the predicate (i.e., $\leq$, $=$, $\geq$) used for comparing two consecutive feature values.
### 2.1. Patterns, Their Seed Transducers and Their Parametrised Glue Matrices

#### Table 2.4: Parametrised glue matrix for any \( g \_f \_DECREASING\_SEQUENCE \) constraint

<table>
<thead>
<tr>
<th></th>
<th>( s )</th>
<th>( t )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>( \phi_g(\vec{C}, \vec{C}) )</td>
<td>( \phi_g(\vec{C}, \vec{C}) )</td>
</tr>
<tr>
<td>( t )</td>
<td>( \phi_g(\vec{C}, \vec{C}) )</td>
<td>( \phi_f(\vec{C}, \vec{C}, \vec{D}, \vec{D}, \delta_f) )</td>
</tr>
</tbody>
</table>
2. PATTERNS, SEED TRANSDUCERS, GLUE MATRICES AND DECORATION TABLES

2.1.4 DECREASING_TERRACE

Given a sequence $S$ over the alphabet $\{<, =, >\}$, an occurrence of the pattern DECREASING_TERRACE is the maximal subsequence of $S$ which matches the regular expression $>\Rightarrow>$. Part (A) and parts (B,C) of Figure 2.11 respectively depict the seed transducer associated with the DECREASING_TERRACE pattern as well as one example of its execution on a time series.

Figure 2.11: (A) Seed transducer for the DECREASING_TERRACE pattern: $>\Rightarrow>$ with $b = 1$ and $a = 1$; (B) Illustrating the execution of the seed transducer on a time series (within the output $o$, $m_0$, $o_1$, $f_c$ are shortcut for $\text{out}$, $\text{maybe}_0$, $\text{out}_r$, $\text{found}_e$); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.
2.1. PATTERNS, THEIR SEED TRANSDUCERS AND THEIR PARAMETRISED GLUE MATRICES

\[
\begin{align*}
C & \leftarrow \text{default}_{gf} \\
D & \leftarrow \text{id}_{f} \\
R & \leftarrow \text{default}_{gf} \\
D & \leftarrow \text{id}_{f} \\
F & \leftarrow \text{default}_{0} \\
R & \leftarrow 1 \\
C & \leftarrow \phi_{f}(D, \delta_{f}) \\
D & \leftarrow \phi_{f}(D, \delta_{f}) \\
R & \leftarrow R \land \theta(F, \phi_{f}(D, \delta_{f}))
\end{align*}
\]

Figure 2.12: Parametrised automaton for any functional time-series constraints of the DECREASING_TERRACE pattern obtained by applying the decoration table 2.35 to the corresponding transducer

\[
\begin{align*}
C & \leftarrow \text{default}_{gf} \\
D & \leftarrow \text{id}_{f} \\
R & \leftarrow \text{default}_{gf} \\
D & \leftarrow \text{id}_{f} \\
F & \leftarrow \text{default}_{0} \\
R & \leftarrow 1 \\
C & \leftarrow \phi_{f}(D, \delta_{f}) \\
D & \leftarrow \phi_{f}(D, \delta_{f}) \\
R & \leftarrow R \land \theta(F, \phi_{f}(D, \delta_{f}))
\end{align*}
\]

Figure 2.13: Parametrised automaton for any predicate time-series constraints of the DECREASING_TERRACE pattern obtained by applying the decoration table 2.36 to the corresponding transducer, where \( \theta \) is the predicate (i.e., \( \leq, =, \geq \)) used for comparing two consecutive feature values.
Table 2.5: Parametrised glue matrix for any $g_f$ _DECREASING_TERRACE_ constraint

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\phi_g(\vec{C}, \vec{C})$</td>
<td>$\phi_g(\vec{C}, \vec{C})$</td>
<td>$\phi_g(\vec{C}, \vec{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\phi_g(\vec{C}, \vec{C})$</td>
<td>$\phi_g(\vec{C}, \vec{C})$</td>
<td>$\phi_f(\vec{D}, \vec{D}, \delta')$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\phi_g(\vec{C}, \vec{C})$</td>
<td>$\phi_f(\vec{D}, \vec{D}, \delta')$</td>
<td>$\phi_f(\vec{D}, \vec{D}, \delta')$</td>
</tr>
</tbody>
</table>
2.1.5 DIP_ON_INCREASING_SEQUENCE

Given a sequence $S$ over the alphabet $\{<,=,>\}$, an occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence of $S$ which matches the regular expression $<><><$. Part (A) and parts (B,C) of Figure 2.14 respectively depict the seed transducer associated with the DIP_ON_INCREASING_SEQUENCE pattern as well as one example of its execution on a time series.
2. PATTERNS, SEED TRANSDUCERS, GLUE MATRICES AND DECORATION TABLES

Figure 2.14: (A) Seed transducer for the DIP_ON_INCREASING_SEQUENCE pattern: <<><<< with \( b = 2 \) and \( a = 1 \); (B) Illustrating the execution of the seed transducer on a time series (within the output \( o, \alpha_i, m_b, f_e \) are shortcut for \( \text{out}, \text{out}_t, \text{maybe}_b, \text{found}_e \)); (C) Illustrating the relation between the \( s \)-occurrence, the \( i \)-occurrence and the \( e \)-occurrence.
Figure 2.15: Parametrised automaton for any functional time-series constraints of the DIP_ON_INCREASING_SEQUENCE pattern obtained by applying the decoration table 2.35 to the corresponding transducer.
Figure 2.16: Parametrised automaton for any predicate time-series constraints of the DIP_ON_INCREASING_SEQUENCE pattern obtained by applying the decoration table 2.36 to the corresponding transducer, where \( \theta \) is the predicate (i.e., \( \leq \), \( = \), \( \geq \)) used for comparing two consecutive feature values.
2.1.6 GORGE

Given a sequence $S$ over the alphabet $\{<, =, >\}$, an occurrence of the pattern GORGE is the maximal subsequence of $S$ which matches the regular expression $( (> (>= >)^* >) ( < (<= <)^* <))$. Part (A) and parts (B,C) of Figure 2.17 respectively depict the seed transducer associated with the GORGE pattern as well as one example of its execution on a time series.

Figure 2.17: (A) Seed transducer for the GORGE pattern: $( > (>(= >)^*>) ( < (<(= <)^* <))$ with $b = 1$ and $a = 1$; (B) Illustrating the execution of the seed transducer on a time series (within the output o, f, m, o, m, o, o are shortcut for out, found, maybeb, outb, maybeb, outc); (C) Illustrating the relation between the s-occurrence, the i-occurrence and the e-occurrence.
Figure 2.18: Parametrised automaton for any functional time-series constraints of the GORGE pattern obtained by applying the decoration table 2.35 to the corresponding transducer (transition $u \to r$ has the same accumulator update as transition $r \to u$)
2.1. PATTERNS, THEIR SEED TRANSDUCERS AND THEIR PARAMETRISED GLUE MATRICES

Figure 2.19: Parametrisied automaton for any predicate time-series constraints of the GORGE pattern obtained by applying the decoration table 2.36 to the corresponding transducer, where $\theta$ is the predicate (i.e., $\leq$, $=$, $\geq$) used for comparing two consecutive feature values; transition $u \to r$ has the same accumulator update as transition $r \to u$. 

\[
\begin{align*}
C &\leftarrow \text{default}_\theta \\
D &\leftarrow \text{id}_f \\
F &\leftarrow \text{default}_\theta \\
R &\leftarrow 1
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow \phi_f(D, \delta_f) \} &\quad \leq \quad \{ C \leftarrow \phi_f(D, \delta_f) \} \\
\{ D \leftarrow \phi_f(D, \delta_f) \} &\quad > \quad \{ D \leftarrow \phi_f(D, \delta_f) \} \\
\{ D \leftarrow \phi_f(D, \delta_f) \} &\quad < \quad \{ D \leftarrow \phi_f(D, \delta_f) \} \\
\{ D \leftarrow \phi_f(D, \delta_f) \} &\quad \leq \quad \{ D \leftarrow \phi_f(D, \delta_f) \} \\
\{ D \leftarrow \phi_f(D, \delta_f) \} &\quad = \quad \{ D \leftarrow \phi_f(D, \delta_f) \}
\end{align*}
\]
<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
<th>u</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\phi_g(\overline{c}, \overline{c})$</td>
<td>$\phi_g(\overline{c}, \overline{c})$</td>
<td>$\phi_g(\overline{c}, \overline{c})$</td>
<td>$\phi_g(\overline{c}, \overline{c})$</td>
</tr>
<tr>
<td>r</td>
<td>$\phi_g(\overline{c}, \overline{c})$</td>
<td>$\phi_f(\overline{d}, \overline{d}, \delta_j)$</td>
<td>$\phi_f(\overline{c}, \overline{d}, \overline{d}, \delta_j)$</td>
<td>$\phi_g(\overline{c}, \overline{c})$</td>
</tr>
<tr>
<td>t</td>
<td>$\phi_g(\overline{c}, \overline{c})$</td>
<td>$\phi_f(\overline{c}, \overline{d}, \overline{d}, \delta_j)$</td>
<td>$\phi_g(\overline{c}, \overline{c})$</td>
<td>$\phi_f(\overline{c}, \overline{d}, \overline{d}, \delta_j)$</td>
</tr>
<tr>
<td>u</td>
<td>$\phi_g(\overline{c}, \overline{c})$</td>
<td>$\phi_g(\overline{c}, \overline{c})$</td>
<td>$\phi_f(\overline{c}, \overline{d}, \overline{d}, \delta_j)$</td>
<td>$\phi_g(\overline{c}, \overline{c})$</td>
</tr>
</tbody>
</table>

Table 2.6: Parametrised glue matrix for any $g_{f\_GORGE}$ constraint
2.1. PATTERNS, THEIR SEED TRANSDUCERS AND THEIR PARAMETRISED GLUE MATRICES

2.1.7 INCREASING

Given a sequence $S$ over the alphabet $\{<, =, >\}$, an occurrence of the pattern INCREASING is the subsequence of $S$ which matches the regular expression $<$. Part (A) and parts (B,C) of Figure 2.20 respectively depict the seed transducer associated with the INCREASING pattern as well as one example of its execution on a time series.

Figure 2.20: (A) Seed transducer for the INCREASING pattern: $<$ with $b = 0$ and $a = 0$; (B) Illustrating the execution of the seed transducer on a time series (within the output $o$, $f$ are shortcut for $out$, $found_c$); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.

Table 2.7: Parametrised glue matrix for any $g \_ f \_ \text{INCREASING}$ constraint
36.2. PATTERNS, SEED TRANSDUCERS, GLUE MATRICES AND DECORATION TABLES

Figure 2.21: Parametrised automaton for any functional time-series constraints of the INCREASING pattern obtained by applying the decoration table 2.35 to the corresponding transducer.

Figure 2.22: Parametrised automaton for any predicate time-series constraints of the INCREASING pattern obtained by applying the decoration table 2.36 to the corresponding transducer, where $\theta$ is the predicate (i.e., $\leq$, $=$, $\geq$) used for comparing two consecutive feature values.
2.1. PATTERNS, THEIR SEED TRANSDUCERS AND THEIR PARAMETRISED GLUE MATRICES

2.1.8 INCREASING_SEQUENCE

Given a sequence $S$ over the alphabet $\{<,\leq,>,\geq\}$, an occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence of $S$ which matches the regular expression $< (\leq | =)^* < | <$. Part (A) and parts (B,C) of Figure 2.23 respectively depict the seed transducer associated with the INCREASING_SEQUENCE pattern as well as one example of its execution on a time series.

Figure 2.23: (A) Seed transducer for the INCREASING_SEQUENCE pattern: $< (\leq | =)^* < | <$ with $b = 0$ and $a = 0$; (B) Illustrating the execution of the seed transducer on a time series (within the output $o, f, m_a, o_a$ are shortcut for out, found, maybe, out); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.
38. PATTERNS, SEED TRANSUDERS, GLUE MATRICES AND DECORATION TABLES

Figure 2.24: Parametrised automaton for any functional time-series constraints of the `INCREASING_SEQUENCE` pattern obtained by applying the decoration table 2.35 to the corresponding transducer.

Figure 2.25: Parametrised automaton for any predicate time-series constraints of the `INCREASINGSEQUENCE` pattern obtained by applying the decoration table 2.36 to the corresponding transducer, where \( \theta \) is the predicate (i.e., \( \leq, =, \geq \)) used for comparing two consecutive feature values.
2.1. PATTERNS, THEIR SEED TRANSDUCERS AND THEIR PARAMETRISED GLUE MATRICES

Table 2.8: Parametrised glue matrix for any $g \_ f \_ \text{INCREASING\_SEQUENCE}$ constraint

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\phi_g(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\phi_g(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\phi_g(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\phi_f(\overrightarrow{C}, \overrightarrow{C}, \overrightarrow{B}, \overrightarrow{D}, \delta_f)$</td>
</tr>
</tbody>
</table>
2. PATTERNS, SEED TRANSDUCERS, GLUE MATRICES AND DECORATION TABLES

2.1.9 INCREASING_TERRACE

Given a sequence $S$ over the alphabet $\{<, =, >\}$, an occurrence of the pattern INCREASING_TERRACE is the maximal subsequence of $S$ which matches the regular expression $< = + <$. Part (A) and parts (B,C) of Figure 2.26 respectively depict the seed transducer associated with the INCREASING_TERRACE pattern as well as one example of its execution on a time series.

Figure 2.26: (A) Seed transducer for the INCREASING_TERRACE pattern: $< = + <$ with $b = 1$ and $a = 1$; (B) Illustrating the execution of the seed transducer on a time series (within the output $o$, $m_b$, $o_\epsilon$, $f_\epsilon$ are shortcut for $out$, $maybe_b$, $out_\epsilon$, $found_\epsilon$); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.
2.1. PATTERNS, THEIR SEED TRANSDUCERS AND THEIR PARAMETRISED GLUE MATRICES

Figure 2.27: Parametrised automaton for any functional time-series constraints of the Increasing Terrace pattern obtained by applying the decoration table 2.35 to the corresponding transducer.

Figure 2.28: Parametrised automaton for any predicate time-series constraints of the Increasing Terrace pattern obtained by applying the decoration table 2.36 to the corresponding transducer, where \( \theta \) is the predicate (i.e., \( \leq, =, \geq \)) used for comparing two consecutive feature values.
### Table 2.9: Parametrised glue matrix for any $g_f$ _INCREASING_TERRACE_ constraint

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\phi_g(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\phi_g(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\phi_g(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\phi_g(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\phi_g(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\phi_f(\overrightarrow{D}, \overrightarrow{D}, \delta_i)$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\phi_g(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\phi_f(\overrightarrow{D}, \overrightarrow{D}, \delta_j)$</td>
<td>$\phi_f(\overrightarrow{D}, \overrightarrow{D}, \delta_j)$</td>
</tr>
</tbody>
</table>
2.1. PATTERNS, THEIR SEED TRANSDUCERS AND THEIR PARAMETRISED GLUE MATRICES

2.1.10 INFLEXION

Given a sequence $S$ over the alphabet $\{<,=,>\}$, an occurrence of the pattern INFLEXION is the maximal subsequence of $S$ which matches the regular expression $<(<|)=)*|>|>(>|=)*<$. Part (A) and parts (B,C) of Figure 2.29 respectively depict the seed transducer associated with the INFLEXION pattern as well as one example of its execution on a time series.

Figure 2.29: (A) Seed transducer for the INFLEXION pattern: $<(<|)=)*>|>(>|=)*< \text{with } b = 1 \text{ and } a = 1$; (B) Illustrating the execution of the seed transducer on a time series (within the output $o$, $m_b$, $f_e$ are shortcut for out, maybe$_b$, found$_e$); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.
Figure 2.30: Parametrised automaton for any functional time-series constraints of the *INFLEXION* pattern obtained by applying the decoration table 2.35 to the corresponding transducer (transition $r \to t$ has the same accumulators updates as transition $t \to r$).

Figure 2.31: Parametrised automaton for any predicate time-series constraints of the *INFLEXION* pattern obtained by applying the decoration table 2.36 to the corresponding transducer, where $\theta$ is the predicate (i.e., $\leq$, $\geq$, $\approx$) used for comparing two consecutive feature values; transition $r \to t$ has the same accumulators updates as transition $t \to r$. 
2.1. PATTERNS, THEIR SEED TRANSDUCERS AND THEIR PARAMETRISED GLUE MATRICES

2.1.11 PEAK

Given a sequence $S$ over the alphabet $\{<, =, >\}$, an occurrence of the pattern PEAK is the maximal subsequence of $S$ which matches the regular expression $<(=|<)^* (>|=)^*>$. Part (A) and parts (B,C) of Figure 2.32 respectively depict the seed transducer associated with the PEAK pattern as well as one example of its execution on a time series.

Figure 2.32: (A) Seed transducer for the PEAK pattern: $<(=|<)^* (>|=)^*>$ with $b = 1$ and $a = 1$; (B) Illustrating the execution of the seed transducer on a time series (within the output $o$, $m_b$, $f$, $m_a$, $o_a$ are shortcut for $out$, $maybe_b$, $found$, $maybe_a$, $out_a$); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.
2. PATTERNS, SEED TRANSDUCERS, GLUE MATRICES AND DECORATION TABLES

Figure 2.33: Parametrised automaton for any functional time-series constraints of the PEAK pattern obtained by applying the decoration table 2.35 to the corresponding transducer.

Figure 2.34: Parametrised automaton for any predicate time-series constraints of the PEAK pattern obtained by applying the decoration table 2.36 to the corresponding transducer, where \( \theta \) is the predicate (i.e., \( \leq, =, \geq \)) used for comparing two consecutive feature values.
### 2.1. PATTERNS, THEIR SEED TRANSDUCERS AND THEIR PARAMETRISED GLUE MATRICES

**Table 2.10: Parametrised glue matrix for any $g_f$ PEAK constraint**

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\phi_g(\vec{C},\vec{C})$</td>
<td>$\phi_g(\vec{C},\vec{C})$</td>
<td>$\phi_g(\vec{C},\vec{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\phi_g(\vec{C},\vec{C})$</td>
<td>$\phi_f(\vec{D},\vec{D},\delta_j)$</td>
<td>$\phi_f(\vec{C},\vec{D},\delta_j)$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\phi_g(\vec{C},\vec{C})$</td>
<td>$\phi_f(\vec{C},\vec{D},\vec{D},\delta_j)$</td>
<td>$\phi_g(\vec{C},\vec{C})$</td>
</tr>
</tbody>
</table>
2.1.12 PLAIN

Given a sequence $S$ over the alphabet $\{<, =, >\}$, an occurrence of the pattern PLAIN is the maximal subsequence of $S$ which matches the regular expression $> = ^* <$. Part (A) and parts (B, C) of Figure 2.35 respectively depict the seed transducer associated with the PLAIN pattern as well as one example of its execution on a time series.

Figure 2.35: (A) Seed transducer for the PLAIN pattern: $> = ^* <$ with $b = 1$ and $a = 1$; (B) Illustrating the execution of the seed transducer on a time series (within the output $o, f, m_b, o$, are shortcut for out, found, maybe, out); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.
2.1. PATTERNS, THEIR SEED TRANSDUCERS AND THEIR PARAMETRISED GLUE MATRICES

$$\leq_s = \begin{cases} C \leftarrow \text{default} \\ D \leftarrow \text{id} \\ R \leftarrow \text{default} \end{cases}$$

$$\leq_r = \begin{cases} \phi_s[R,C] \\ D \leftarrow \phi_f(D,\delta_f) \\ R \leftarrow \phi_f(R,\delta_f) \end{cases}$$

$$\leq_t = \begin{cases} \phi_s[R,C] \\ D \leftarrow \phi_f(D,\delta_f) \\ R \leftarrow \phi_f(R,\delta_f) \end{cases}$$

$$\leq_p = \begin{cases} \phi_s[R,C] \\ D \leftarrow \phi_f(D,\delta_f) \\ R \leftarrow \phi_f(R,\delta_f) \end{cases}$$

$$\leq_q = \begin{cases} \phi_s[R,C] \\ D \leftarrow \phi_f(D,\delta_f) \\ R \leftarrow \phi_f(R,\delta_f) \end{cases}$$

$$\leq_h = \begin{cases} \phi_s[R,C] \\ D \leftarrow \phi_f(D,\delta_f) \\ R \leftarrow \phi_f(R,\delta_f) \end{cases}$$

$$\leq_i = \begin{cases} \phi_s[R,C] \\ D \leftarrow \phi_f(D,\delta_f) \\ R \leftarrow \phi_f(R,\delta_f) \end{cases}$$

$$\leq_j = \begin{cases} \phi_s[R,C] \\ D \leftarrow \phi_f(D,\delta_f) \\ R \leftarrow \phi_f(R,\delta_f) \end{cases}$$

$$\leq_k = \begin{cases} \phi_s[R,C] \\ D \leftarrow \phi_f(D,\delta_f) \\ R \leftarrow \phi_f(R,\delta_f) \end{cases}$$

$$\leq_l = \begin{cases} \phi_s[R,C] \\ D \leftarrow \phi_f(D,\delta_f) \\ R \leftarrow \phi_f(R,\delta_f) \end{cases}$$

$$\leq_m = \begin{cases} \phi_s[R,C] \\ D \leftarrow \phi_f(D,\delta_f) \\ R \leftarrow \phi_f(R,\delta_f) \end{cases}$$

$$\leq_n = \begin{cases} \phi_s[R,C] \\ D \leftarrow \phi_f(D,\delta_f) \\ R \leftarrow \phi_f(R,\delta_f) \end{cases}$$

$$\leq_o = \begin{cases} \phi_s[R,C] \\ D \leftarrow \phi_f(D,\delta_f) \\ R \leftarrow \phi_f(R,\delta_f) \end{cases}$$

$$\leq_p = \begin{cases} \phi_s[R,C] \\ D \leftarrow \phi_f(D,\delta_f) \\ R \leftarrow \phi_f(R,\delta_f) \end{cases}$$

$$\leq_q = \begin{cases} \phi_s[R,C] \\ D \leftarrow \phi_f(D,\delta_f) \\ R \leftarrow \phi_f(R,\delta_f) \end{cases}$$

$$\leq_r = \begin{cases} \phi_s[R,C] \\ D \leftarrow \phi_f(D,\delta_f) \\ R \leftarrow \phi_f(R,\delta_f) \end{cases}$$

$$\leq_s = \begin{cases} \phi_s[R,C] \\ D \leftarrow \phi_f(D,\delta_f) \\ R \leftarrow \phi_f(R,\delta_f) \end{cases}$$

$$\leq_t = \begin{cases} \phi_s[R,C] \\ D \leftarrow \phi_f(D,\delta_f) \\ R \leftarrow \phi_f(R,\delta_f) \end{cases}$$

$$\leq_u = \begin{cases} \phi_s[R,C] \\ D \leftarrow \phi_f(D,\delta_f) \\ R \leftarrow \phi_f(R,\delta_f) \end{cases}$$

$$\leq_v = \begin{cases} \phi_s[R,C] \\ D \leftarrow \phi_f(D,\delta_f) \\ R \leftarrow \phi_f(R,\delta_f) \end{cases}$$

$$\leq_w = \begin{cases} \phi_s[R,C] \\ D \leftarrow \phi_f(D,\delta_f) \\ R \leftarrow \phi_f(R,\delta_f) \end{cases}$$

$$\leq_x = \begin{cases} \phi_s[R,C] \\ D \leftarrow \phi_f(D,\delta_f) \\ R \leftarrow \phi_f(R,\delta_f) \end{cases}$$

$$\leq_y = \begin{cases} \phi_s[R,C] \\ D \leftarrow \phi_f(D,\delta_f) \\ R \leftarrow \phi_f(R,\delta_f) \end{cases}$$

$$\leq_z = \begin{cases} \phi_s[R,C] \\ D \leftarrow \phi_f(D,\delta_f) \\ R \leftarrow \phi_f(R,\delta_f) \end{cases}$$

Figure 2.36: Parametrised automaton for any functional time-series constraints of the PLAIN pattern obtained by applying the decoration table 2.35 to the corresponding transducer.

Figure 2.37: Parametrised automaton for any predicate time-series constraints of the PLAIN pattern obtained by applying the decoration table 2.36 to the corresponding transducer, where $\theta$ is the predicate (i.e., $\leq$, $=$, $\geq$) used for comparing two consecutive feature values.
### Table 2.11: Parametrised glue matrix for any $g_{f, \text{PLAIN}}$ constraint

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\phi_g(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\phi_g(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\phi_g(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\phi_g(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\phi_f(\overrightarrow{D}, \overrightarrow{D}, \delta_j)$</td>
<td>$\phi_f(\overrightarrow{D}, \overrightarrow{D}, \delta_j)$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\phi_g(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\phi_f(\overrightarrow{D}, \overrightarrow{D}, \delta_j)$</td>
<td>$\phi_f(\overrightarrow{D}, \overrightarrow{D}, \delta_j)$</td>
</tr>
</tbody>
</table>
2.1. PATTERNS, THEIR SEED TRANSDUCERS AND THEIR PARAMETRISED GLUE MATRICES

2.1.13 PLATEAU

Given a sequence $S$ over the alphabet $\{<,=,>\}$, an occurrence of the pattern PLATEAU is the maximal subsequence of $S$ which matches the regular expression $<=^*>$. Part (A) and parts (B,C) of Figure 2.38 respectively depict the seed transducer associated with the PLATEAU pattern as well as one example of its execution on a time series.

Figure 2.38: (A) Seed transducer for the PLATEAU pattern: $<=^*$ with $b = 1$ and $a = 1$; (B) Illustrating the execution of the seed transducer on a time series (within the output $o$, $f_e$, $m_b$, $o_i$ are shortcut for out, found_e, maybe_b, out_i); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.
Figure 2.39: Parametrised automaton for any functional time-series constraints of the PLATEAU pattern obtained by applying the decoration table 2.35 to the corresponding transducer.

Figure 2.40: Parametrised automaton for any predicate time-series constraints of the PLATEAU pattern obtained by applying the decoration table 2.36 to the corresponding transducer, where $\theta$ is the predicate (i.e., $\leq$, $=\$, $\geq$) used for comparing two consecutive feature values.
### 2.1. Patterns, Their Seed Transducers and Their Parametrised Glue Matrices

Table 2.12: Parametrised glue matrix for any $g_f$ _PLATEAU_ constraint

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\phi_g(\vec{C}, \vec{C})$</td>
<td>$\phi_g(\vec{C}, \vec{C})$</td>
<td>$\phi_g(\vec{C}, \vec{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\phi_g(\vec{C}, \vec{C})$</td>
<td>$\phi_f(\vec{D}, \vec{D}, \delta_f)$</td>
<td>$\phi_f(\vec{D}, \vec{D}, \delta_f)$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\phi_g(\vec{C}, \vec{C})$</td>
<td>$\phi_f(\vec{D}, \vec{D}, \delta_f)$</td>
<td>$\phi_f(\vec{D}, \vec{D}, \delta_f)$</td>
</tr>
</tbody>
</table>
PATTERNS, SEED TRANSDUCERS, GLUE MATRICES AND DECORATION TABLES

2.1.14 PROPER\_PLAIN

Given a sequence $S$ over the alphabet $\{<,=,>\}$, an occurrence of the pattern PROPER\_PLAIN is the maximal subsequence of $S$ which matches the regular expression $>\Rightarrow<$. Part (A) and parts (B,C) of Figure 2.41 respectively depict the seed transducer associated with the PROPER\_PLAIN pattern as well as one example of its execution on a time series.

Figure 2.41: (A) Seed transducer for the PROPER\_PLAIN pattern: $>\Rightarrow<$ with $b = 1$ and $a = 1$; (B) Illustrating the execution of the seed transducer on a time series (within the output $o$, $m_b$, $f_e$ are shortcut for $\text{out}$, $\text{maybe}_b$, $\text{found}_e$); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.
2.1. PATTERNS, THEIR SEED TRANSDUCERS AND THEIR PARAMETRISED GLUE MATRICES

Figure 2.42: Parametrised automaton for any functional time-series constraints of the PROPER/plain pattern obtained by applying the decoration table 2.35 to the corresponding transducer.

Figure 2.43: Parametrised automaton for any predicate time-series constraints of the PROPER/plain pattern obtained by applying the decoration table 2.36 to the corresponding transducer, where \( \theta \) is the predicate (i.e., \( \leq, =, \geq \)) used for comparing two consecutive feature values.
Table 2.13: Parametrised glue matrix for any $g_f$.PROPER.PLAIN constraint

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\phi_g(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\phi_g(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\phi_g(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\phi_g(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\phi_g(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\phi_f(\overrightarrow{D}, \overrightarrow{D}, \delta_j^i)$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\phi_g(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\phi_f(\overrightarrow{D}, \overrightarrow{D}, \delta_j^i)$</td>
<td>$\phi_f(\overrightarrow{D}, \overrightarrow{D}, \delta_j^i)$</td>
</tr>
</tbody>
</table>
2.1. PATTERNS, THEIR SEED TRANSDUCERS AND THEIR PARAMETRISED GLUE MATRICES

2.1.15 PROPER PLATEAU

Given a sequence $S$ over the alphabet $\{<, =, >\}$, an occurrence of the pattern \textsc{ProperPlateau} is the maximal subsequence of $S$ which matches the regular expression $< =^+ >$. Part (A) and parts (B,C) of Figure 2.44 respectively depict the seed transducer associated with the \textsc{ProperPlateau} pattern as well as one example of its execution on a time series.

Figure 2.44: (A) Seed transducer for the \textsc{ProperPlateau} pattern: $< =^+ >$ with $b = 1$ and $a = 1$; (B) Illustrating the execution of the seed transducer on a time series (within the output $o$, $m_b$, $f_e$ are shortcut for $\text{out}$, $\text{maybe}_b$, $\text{found}_e$); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.
Figure 2.45: Parametrised automaton for any functional time-series constraints of the PROPER_PLATEAU pattern obtained by applying the decoration table 2.35 to the corresponding transducer

Figure 2.46: Parametrised automaton for any predicate time-series constraints of the PROPER_PLATEAU pattern obtained by applying the decoration table 2.36 to the corresponding transducer, where \( \theta \) is the predicate (i.e., \( \leq \), \( = \), \( \geq \)) used for comparing two consecutive feature values
### Table 2.14: Parametrised glue matrix for any \( g_f \) _PROPER_PLATEAU_ constraint

<table>
<thead>
<tr>
<th></th>
<th>( s )</th>
<th>( r )</th>
<th>( t )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
</tr>
<tr>
<td>( r )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_f(\overrightarrow{D}, \overrightarrow{D}, \delta_f) )</td>
</tr>
<tr>
<td>( t )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_f(\overrightarrow{D}, \overrightarrow{D}, \delta_f) )</td>
<td>( \phi_f(\overrightarrow{D}, \overrightarrow{D}, \delta_f) )</td>
</tr>
</tbody>
</table>
2.1.16 STEADY

Given a sequence $S$ over the alphabet \{<, =, >\}, an occurrence of the pattern $STEADY$ is the subsequence of $S$ which matches the regular expression $=$. Part (A) and parts (B,C) of Figure 2.47 respectively depict the seed transducer associated with the $STEADY$ pattern as well as one example of its execution on a time series.

![Seed transducer for the STEADY pattern](image)

Figure 2.47: (A) Seed transducer for the $STEADY$ pattern: $=$ with $b = 0$ and $a = 0$; (B) Illustrating the execution of the seed transducer on a time series (within the output $f, o$ are shortcut for $\text{found}e$, $\text{out}$); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.

![Parametrised glue matrix](image)

Table 2.15: Parametrised glue matrix for any $g, f$ $STEADY$ constraint
2.1. PATTERNS, THEIR SEED TRANSDUCERS AND THEIR PARAMETRISED GLUE MATRICES

Figure 2.48: Parametrised automaton for any functional time-series constraints of the STEADY pattern obtained by applying the decoration table 2.35 to the corresponding transducer

Figure 2.49: Parametrised automaton for any predicate time-series constraints of the STEADY pattern obtained by applying the decoration table 2.36 to the corresponding transducer, where \( \theta \) is the predicate (i.e., \( \leq, =, \geq \)) used for comparing two consecutive feature values
2.1.17 STEADYSEQUENCE

Given a sequence $S$ over the alphabet $\{<,=,>\}$, an occurrence of the pattern STEADYSEQUENCE is the maximal subsequence of $S$ which matches the regular expression $=^+$. Part (A) and parts (B,C) of Figure 2.50 respectively depict the seed transducer associated with the STEADYSEQUENCE pattern as well as one example of its execution on a time series.

```
(A)

\[ \geq s \Rightarrow \text{out} \]
\[ \geq : \text{out} \Rightarrow \text{found} \]
\[ = r \Rightarrow \text{in} \]

(B)

output states
input
\[ s \rightarrow s, f, s, r, \]...
\[ > \rightarrow <, <, <, >, \]...

(C)

Table 2.16: Parametrised glue matrix for any $g\_f$ STEADYSEQUENCE constraint

\[
\begin{array}{c|cc}
\text{input} & s & r \\
\text{output} & \phi_g(\overrightarrow{C}, \overrightarrow{C}) & \phi_f(\overrightarrow{C}, \overrightarrow{C}) \\
\end{array}
\]
2.1. PATTERNS, THEIR SEED TRANSDUCERS AND THEIR PARAMETRISED GLUE MATRICES

Figure 2.51: Parametrised automaton for any functional time-series constraints of the steady_sequence pattern obtained by applying the decoration table 2.35 to the corresponding transducer.

Figure 2.52: Parametrised automaton for any predicate time-series constraints of the steady_sequence pattern obtained by applying the decoration table 2.36 to the corresponding transducer, where $\theta$ is the predicate (i.e., $\leq$, $=$, $\geq$) used for comparing two consecutive feature values.
2. PATTERNS, SEED TRANSDUCERS, GLUE MATRICES AND DECORATION TABLES

2.1.18 STRICTLY DECREASING_SEQUENCE

Given a sequence $S$ over the alphabet $\{<, =, >\}$, an occurrence of the pattern STRICTLY DECREASING_SEQUENCE is the maximal subsequence of $S$ which matches the regular expression $>^+$. Part (A) and parts (B,C) of Figure 2.53 respectively depict the seed transducer associated with the STRICTLY DECREASING_SEQUENCE pattern as well as one example of its execution on a time series.

Figure 2.53: (A) Seed transducer for the STRICTLY DECREASING_SEQUENCE pattern: $>^+$ with $b = 0$ and $a = 0$; (B) Illustrating the execution of the seed transducer on a time series (within the output o, f, o_a are shortcut for out, found, out_a); (C) Illustrating the relation between the s-occurrence, the i-occurrence and the e-occurrence.
2.1. PATTERNS, THEIR SEED TRANSDUCERS AND THEIR PARAMETRISED GLUE MATRICES

Figure 2.54: Parametrised automaton for any functional time-series constraints of the STRICTLY DECREASING SEQUENCE pattern obtained by applying the decoration table 2.35 to the corresponding transducer.

Figure 2.55: Parametrised automaton for any predicate time-series constraints of the STRICTLY DECREASING SEQUENCE pattern obtained by applying the decoration table 2.36 to the corresponding transducer, where \( \theta \) is the predicate (i.e., \( \leq, =, \geq \)) used for comparing two consecutive feature values.
Table 2.17: Parametrised glue matrix for any $g_f$ STRICTLY DECREASING SEQUENCE constraint

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\phi_g(\vec{C}, \vec{C})$</td>
<td>$\phi_g(\vec{C}, \vec{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\phi_g(\vec{C}, \vec{C})$</td>
<td>$\phi_f(\vec{C}, \vec{C}, \vec{D}, \vec{D}, \delta_f)$</td>
</tr>
</tbody>
</table>
2.1. PATTERNS, THEIR SEED TRANSDUCERS AND THEIR PARAMETRISED GLUE MATRICES

2.1.19 STRICTLY_INCREASING_SEQUENCE

Given a sequence $S$ over the alphabet $\{<,=,>\}$, an occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal subsequence of $S$ which matches the regular expression $<^+$. Part (A) and parts (B,C) of Figure 2.56 respectively depict the seed transducer associated with the STRICTLY_INCREASING_SEQUENCE pattern as well as one example of its execution on a time series.

$$\begin{align*}
(V_7, V_8, V_9) &\quad \text{with } (i, j) = (7..10) \\
\text{\textit{s-occurrence}} &\quad [(i+b)..(j+1-a)] \\
\text{\textit{i-occurrence}} &\quad [(i+0)..(j+1-0)] \\
\text{\textit{e-occurrence}} &\quad [(7+0)..(10+1-0)] 
\end{align*}$$

Figure 2.56: (A) Seed transducer for the STRICTLY_INCREASING_SEQUENCE pattern: $<^+$ with $b = 0$ and $a = 0$; (B) Illustrating the execution of the seed transducer on a time series (within the output $o$, $f$, $o_a$ are shortcut for $\text{out}$, $\text{found}$, $\text{out}_a$); (C) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.
Figure 2.57: Parametrised automaton for any functional time-series constraints of the \textsc{strictly increasing sequence} pattern obtained by applying the decoration table 2.35 to the corresponding transducer.

Figure 2.58: Parametrised automaton for any predicate time-series constraints of the \textsc{strictly increasing sequence} pattern obtained by applying the decoration table 2.36 to the corresponding transducer, where $\theta$ is the predicate (i.e., $\leq$, $=$, $\geq$) used for comparing two consecutive feature values.
2.1. PATTERNS, THEIR SEED TRANSDUCERS AND THEIR PARAMETRISED GLUE MATRICES

Table 2.18: Parametrised glue matrix for any $g_{-f, \text{STRICTLY, INCREASING, SEQUENCE}}$ constraint

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\phi_y(\overline{C}, \overline{C})$</td>
<td>$\phi_y(\overline{C}, \overline{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\phi_y(\overline{C}, \overline{C})$</td>
<td>$\phi_f(\overline{C}, \overline{C}, \overline{D}, \overline{D}, \delta_i)$</td>
</tr>
</tbody>
</table>
2. PATTERNS, SEED TRANSDUCERS, GLUE MATRICES AND DECORATION TABLES

2.1.20 SUMMIT

Given a sequence \( S \) over the alphabet \( \{<, =, >\} \), an occurrence of the pattern SUMMIT is the \textit{maximal} subsequence of \( S \) which matches the regular expression \( < (\langle = \mid < \rangle^* <) \rangle \mid (\langle > \mid > \rangle^* >) \). Part (A) and parts (B,C) of Figure 2.59 respectively depict the seed transducer associated with the SUMMIT pattern as well as one example of its execution on a time series.

Figure 2.59: (A) Seed transducer for the SUMMIT pattern: \( < (\langle = \mid < \rangle^* <) \rangle \mid (\langle > \mid > \rangle^* >) \) with \( b = 1 \) and \( a = 1 \); (B) Illustrating the execution of the seed transducer on a time series (within the output \( o, f, m_a, o_a, m_b, o_b \), \( o \) is shortcut for \( \text{out, found, maybe}_a, \text{out}_a, \text{maybe}_b, \text{out}_b \)); (C) Illustrating the relation between the \( s \)-occurrence, the \( i \)-occurrence and the \( e \)-occurrence.
Figure 2.60: Parametrised automaton for any functional time-series constraints of the SUMMIT pattern obtained by applying the decoration table 2.35 to the corresponding transducer (transition \( u \rightarrow r \) has the same accumulator update as transition \( r \rightarrow u \))
Figure 2.61: Parametrised automaton for any predicate time-series constraints of the SUMMIT pattern obtained by applying the decoration table 2.36 to the corresponding transducer, where $\theta$ is the predicate (i.e., $\leq$, $=$, $\geq$) used for comparing two consecutive feature values; transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$. 

\[
\begin{align*}
C &\leftarrow \text{default}_\theta \\
D &\leftarrow \text{id}_f \\
F &\leftarrow \text{default}_\theta \\
R &\leftarrow 1
\end{align*}
\]
Table 2.19: Parametrised glue matrix for any $g_{\_f \_SUMMIT}$ constraint
2. PATTERNS, SEED TRANSDUCERS, GLUE MATRICES AND DECORATION TABLES

2.1.21 VALLEY

Given a sequence $S$ over the alphabet $\{<,=,>\}$, an occurrence of the pattern VALLEY is the maximal subsequence of $S$ which matches the regular expression $> (|=>)^* (<|=)^* <$. Part (A) and parts (B,C) Figure 2.62 respectively depict the seed transducer associated with the VALLEY pattern as well as one example of its execution on a time series.

Figure 2.62: (A) Seed transducer for the VALLEY pattern: $> (|=>)^* (<|=)^* <$ with $b = 1$ and $a = 1$; (B) Illustrating the execution of the seed transducer on a time series (within the output $o$, $m_b$, $f$, $m_a$, $o_a$ are shortcut for out, maybe$_b$, found, maybe$_a$, out$_a$); (C) Illustrating the relation between the s-occurrence, the i-occurrence and the e-occurrence.
2.1. PATTERNS, THEIR SEED TRANSDUCERS AND THEIR PARAMETRISED GLUE MATRICES

Figure 2.63: Parametrised automaton for any functional time-series constraints of the VALLEY pattern obtained by applying the decoration table 2.35 to the corresponding transducer

Figure 2.64: Parametrised automaton for any predicate time-series constraints of the VALLEY pattern obtained by applying the decoration table 2.36 to the corresponding transducer, where $\theta$ is the predicate (i.e., $\leq$, $=\geq$) used for comparing two consecutive feature values
Table 2.20: Parametrised glue matrix for any \( g_f \) \_VALLEY constraint

<table>
<thead>
<tr>
<th></th>
<th>( s )</th>
<th>( r )</th>
<th>( t )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
</tr>
<tr>
<td>( r )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_f(\overrightarrow{D}, \overrightarrow{D}, \delta_j) )</td>
<td>( \phi_f(\overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}, \delta_j) )</td>
</tr>
<tr>
<td>( t )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_f(\overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}, \delta_j) )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
</tr>
</tbody>
</table>
2.1. PATTERNS, THEIR SEED TRANSDUCERS AND THEIR PARAMETRISED GLUE MATRICES

2.1.22 ZIGZAG

Given a sequence $S$ over the alphabet $\{<, =, >\}$, an occurrence of the pattern ZIGZAG is the maximal subsequence of $S$ which matches the regular expression $(<>)^+ (<|<>)(<>)^+(>|<>)+ (<|<>)(>).$ Figures 2.65 and 2.66 respectively depict the seed transducer associated with the ZIGZAG pattern as well as one example of its execution on a time series.
Figure 2.65: Seed transducer for the ZIGZAG pattern: $(<>)^+ (< | <> ) | (<>)^+ (> | >>)$ with $b = 1$ and $a = 1$; missing transitions from $a$, $d$ to $s$ are labelled by $= : \text{out}$, missing transitions from $b$, $e$ to $s$ are labelled by $= : \text{out}_r$, and missing transitions from $c$, $f$ to $s$ are labelled by $= : \text{out}_a$. 

\[< : \text{out} \]
\[> : \text{out} \]
\[< : \text{out}_a \]
\[> : \text{out}_a \]
\[< : \text{in} \]
\[> : \text{in} \]
2.1. PATTERNS, THEIR SEED TRANSUDCERS AND THEIR PARAMETRISED GLUE MATRICES

Figure 2.66: (A) Illustrating the execution of the seed transducer associated with the ZIGZAG pattern on a time series (within the output $\text{o, m}_{\text{e}}, \text{f}, \text{o}_{\text{s}}$ are shortcut for $\text{out, maybe}_{\text{e}}, \text{found, out}$); (B) Illustrating the relation between the $s$-occurrence, the $i$-occurrence and the $e$-occurrence.
Table 2.21: Parametrised glue matrix for any \( g \) / ZIGZAG constraint

<table>
<thead>
<tr>
<th></th>
<th>( s )</th>
<th>( a )</th>
<th>( b )</th>
<th>( c )</th>
<th>( d )</th>
<th>( e )</th>
<th>( f )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td></td>
</tr>
<tr>
<td>( a )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_f(\overrightarrow{C}, \overrightarrow{D}, \delta_j) )</td>
<td>( \phi_b(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_f(\overrightarrow{D}, \overrightarrow{D}, \delta_j) )</td>
<td>( \phi_b(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td></td>
</tr>
<tr>
<td>( b )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_f(\overrightarrow{D}, \overrightarrow{D}, \delta_j) )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_f(\overrightarrow{D}, \overrightarrow{D}, \delta_j) )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td></td>
</tr>
<tr>
<td>( c )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_f(\overrightarrow{C}, \overrightarrow{D}, \delta_j) )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_f(\overrightarrow{C}, \overrightarrow{D}, \delta_j) )</td>
<td>( \phi_b(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td></td>
</tr>
<tr>
<td>( d )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_f(\overrightarrow{C}, \overrightarrow{D}, \delta_j) )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_f(\overrightarrow{C}, \overrightarrow{D}, \delta_j) )</td>
<td>( \phi_b(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td></td>
</tr>
<tr>
<td>( e )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_f(\overrightarrow{D}, \overrightarrow{D}, \delta_j) )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_f(\overrightarrow{D}, \overrightarrow{D}, \delta_j) )</td>
<td>( \phi_b(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td></td>
</tr>
<tr>
<td>( f )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_f(\overrightarrow{C}, \overrightarrow{D}, \delta_j) )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \phi_f(\overrightarrow{C}, \overrightarrow{D}, \delta_j) )</td>
<td>( \phi_g(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td></td>
</tr>
</tbody>
</table>
2.1. PATTERNS, THEIR SEED TRANSDUCERS AND THEIR PARAMETRISED GLUE MATRICES

Figure 2.67: Parametrised automaton for any functional time-series constraints of the ZIGZAG pattern obtained by applying the decoration table 2.35 to the corresponding transducer; (1) missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=$; (2) on transitions from $b, c, e, f$ to $s$ the accumulator $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the accumulator $R$ is updated wrt $C$ and the accumulator $C$ is reset to its initial value.
Figure 2.68: Parametrised automaton for any predicate time-series constraints of the ZIGZAG pattern obtained by applying the decoration table 2.36 to the corresponding transducer, where \( \theta \) is the predicate (i.e., \( \leq \), \( = \), \( \geq \)) used for comparing two consecutive feature values; (1) missing transitions from \( a, b, c, d, e, f \) to \( s \) are labelled by \( = \); (2) on transitions from \( b, c, e, f \) to \( s \) the accumulator \( D \) is reset to its initial value; (3) on transitions from \( c, f \) to \( s \) the accumulator \( F \) is reset to \( C \), and the accumulator \( R \) is updated wrt \( C \) and \( F \).
2.2 Decoration Tables

This section provides the different decoration tables used for synthesising the automata with accumulators of this catalogue. While no automaton of this catalogue relies on tables 2.32 and 2.34, these two decoration tables may be used for generating distance constraints between consecutive occurrences of a pattern.
Table 2.22: Table for Type: cDRFoundEOut

<table>
<thead>
<tr>
<th>Initialization</th>
<th>out</th>
<th>out</th>
<th>maybe</th>
<th>maybe</th>
<th>found</th>
<th>found</th>
<th>in</th>
<th>found</th>
</tr>
</thead>
<tbody>
<tr>
<td>After Update of $C$</td>
<td>$C \leftarrow \text{default}_g$</td>
<td>$C \leftarrow \text{default}_g$</td>
<td>$C \leftarrow \phi_g(D, \delta^1)$</td>
<td>$C \leftarrow \phi_g(D, \delta^1)$</td>
<td>$C \leftarrow \phi_g(C, \phi_f(D, \delta^1))$</td>
<td>$C \leftarrow \phi_g(C, \phi_f(D, \delta^1))$</td>
<td>$C \leftarrow \phi_g(C, \phi_f(D, \delta^1))$</td>
<td>$C \leftarrow \phi_g(C, \phi_f(D, \delta^1))$</td>
</tr>
<tr>
<td>Update of $D$</td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$D \leftarrow \phi_f(D, \delta^1)$</td>
<td>$D \leftarrow \phi_f(D, \delta^1)$</td>
<td>$D \leftarrow \phi_f(D, \delta^1)$</td>
<td>$D \leftarrow \phi_f(D, \delta^1)$</td>
<td>$D \leftarrow \phi_f(D, \delta^1)$</td>
<td>$D \leftarrow \phi_f(D, \delta^1)$</td>
<td>$D \leftarrow \phi_f(D, \delta^1)$</td>
</tr>
<tr>
<td>Update of $R$</td>
<td>$R \leftarrow \phi_g(R, C)$</td>
<td>$R \leftarrow \phi_g(R, C)$</td>
<td>$R \leftarrow \phi_g(R, C)$</td>
<td>$R \leftarrow \phi_g(R, C)$</td>
<td>$R \leftarrow \phi_g(R, C)$</td>
<td>$R \leftarrow \phi_g(R, C)$</td>
<td>$R \leftarrow \phi_g(R, C)$</td>
<td>$R \leftarrow \phi_g(R, C)$</td>
</tr>
</tbody>
</table>

842. PATTERNS, SEED TRANSUDERS, GLUE MATRICES AND DECORATION TABLES
### Table 2.23: Table for Type: cDRFoundEOutNoMA

<table>
<thead>
<tr>
<th>Semantic Letter</th>
<th>After Update of $C$</th>
<th>Update of $D$</th>
<th>Update of $R$</th>
</tr>
</thead>
<tbody>
<tr>
<td>out</td>
<td>$C \leftarrow$ default$_{of}$</td>
<td>$D \leftarrow 1d_f$</td>
<td>$R \leftarrow$ default$_{of}$</td>
</tr>
<tr>
<td>out$_a$</td>
<td>$C \leftarrow$ default$_{of}$</td>
<td>$D \leftarrow 1d_f$</td>
<td>$R \leftarrow$ $\phi_g(R, C)$</td>
</tr>
<tr>
<td>maybe$_b$</td>
<td>$D \leftarrow \phi_f(D, \delta'_i)$</td>
<td>$D \leftarrow \phi_f(D, \delta'^{+1}_i)$</td>
<td></td>
</tr>
<tr>
<td>maybe$_a$</td>
<td>0</td>
<td>$D \leftarrow \phi_f(D, \delta'_i)$</td>
<td>$D \leftarrow \phi_f(D, \delta'^{+1}_i)$</td>
</tr>
<tr>
<td>maybe$_a$</td>
<td>1</td>
<td>$D \leftarrow \phi_f(D, \delta'_i)$</td>
<td>$D \leftarrow \phi_f(D, \delta'^{+1}_i)$</td>
</tr>
<tr>
<td>found</td>
<td>0</td>
<td>$C \leftarrow \phi_f(\phi_f(D, \delta'_i), \delta'^{+1}_i)$</td>
<td>$D \leftarrow 1d_f$</td>
</tr>
<tr>
<td>found$_a$</td>
<td>1</td>
<td>$C \leftarrow \phi_f(D, \delta'_i)$</td>
<td>$D \leftarrow 1d_f$</td>
</tr>
<tr>
<td>in</td>
<td>0</td>
<td>$C \leftarrow \phi_f(C, \delta'^{+1}_i)$</td>
<td>$D \leftarrow 1d_f$</td>
</tr>
<tr>
<td>found$_a$</td>
<td>1</td>
<td>$C \leftarrow \phi_f(C, \delta'_i)$</td>
<td>$D \leftarrow 1d_f$</td>
</tr>
<tr>
<td>found$_a$</td>
<td>0</td>
<td>$D \leftarrow 1d_f$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(\phi_f(D, \delta'_i), \delta'^{+1}_i))$</td>
</tr>
<tr>
<td>found$_a$</td>
<td>1</td>
<td>$D \leftarrow 1d_f$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta'_i))$</td>
</tr>
</tbody>
</table>
Table 2.24: Table for Type: cRFoundEOutA

<table>
<thead>
<tr>
<th>Initialization</th>
<th>$C \leftarrow \text{default}_g$</th>
<th>$R \leftarrow \text{default}_g$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\phi_g(R, C)$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Semantic Letter</th>
<th>After Update of $C$</th>
<th>Update of $R$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>out</strong></td>
<td>$C \leftarrow \text{default}_g$</td>
<td>$R \leftarrow \phi_g(R, C)$</td>
</tr>
<tr>
<td><strong>out</strong></td>
<td>$C \leftarrow \text{default}_g$</td>
<td>$R \leftarrow \phi_g(R, C)$</td>
</tr>
<tr>
<td><strong>maybe</strong></td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>0</td>
<td>$C \leftarrow \phi_f(\delta^i_j, \delta^{i+1}_j)$</td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>1</td>
<td>$C \leftarrow \delta^i_j$</td>
</tr>
<tr>
<td><strong>in</strong></td>
<td>0</td>
<td>$C \leftarrow \phi_f(C, \delta^{i+1}_j)$</td>
</tr>
<tr>
<td><strong>in</strong></td>
<td>1</td>
<td>$C \leftarrow \phi_f(C, \delta^i_j)$</td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>0</td>
<td>$C \leftarrow \text{default}_g$</td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>1</td>
<td>$C \leftarrow \text{default}_g$</td>
</tr>
</tbody>
</table>

| **return**      | $C \leftarrow \text{default}_g$ | $R \leftarrow \phi_g(R, \delta^i_j)$ |
| **return**      | $C \leftarrow \text{default}_g$ | $R \leftarrow \phi_g(R, \delta^i_j)$ |
2.2. DECORATION TABLES

Table 2.25: Table for Type: dRFoundFoundEInA

<table>
<thead>
<tr>
<th>Semantic Letter</th>
<th>Update of $D$</th>
<th>Update of $R$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Initialization</strong></td>
<td>$D \leftarrow \text{id}_D$</td>
<td>$R \leftarrow \text{default}_R$</td>
</tr>
<tr>
<td><strong>Return</strong></td>
<td>$R$</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>After</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>out</strong></td>
<td>$D \leftarrow \text{id}_D$</td>
<td></td>
</tr>
<tr>
<td><strong>out</strong></td>
<td>$D \leftarrow \text{id}_D$</td>
<td></td>
</tr>
<tr>
<td><strong>maybe</strong></td>
<td>$D \leftarrow \phi_f(D, \delta^i_f)$</td>
<td></td>
</tr>
<tr>
<td><strong>maybe</strong></td>
<td>$D \leftarrow \phi_f(D, \delta^i_f)$</td>
<td></td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>$D \leftarrow \text{id}_D$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \phi_f(D, \delta^i_f), \delta^i_f + 1))$</td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>$D \leftarrow \text{id}_D$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta^i_f))$</td>
</tr>
<tr>
<td><strong>in</strong></td>
<td>$D \leftarrow \text{id}_D$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta^i_f) + 1))$</td>
</tr>
<tr>
<td><strong>in</strong></td>
<td>$D \leftarrow \text{id}_D$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta^i_f))$</td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>$D \leftarrow \text{id}_D$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \phi_f(D, \delta^i_f), \delta^i_f + 1))$</td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>$D \leftarrow \text{id}_D$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta^i_f))$</td>
</tr>
</tbody>
</table>
Table 2.26: Table for Type: dRFoundFoundEInB

<table>
<thead>
<tr>
<th>Initialization</th>
<th>$D \leftarrow \text{id}_f$</th>
<th>$R \leftarrow \text{default}_g$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>return</strong></td>
<td>$R$</td>
<td></td>
</tr>
<tr>
<td>Semantic Letter</td>
<td>After Update of $D$</td>
<td>Update of $R$</td>
</tr>
<tr>
<td>out</td>
<td>$D \leftarrow \text{id}_f$</td>
<td></td>
</tr>
<tr>
<td>out$_r$</td>
<td>$D \leftarrow \text{id}_f$</td>
<td></td>
</tr>
<tr>
<td>maybe$_b$</td>
<td>$D \leftarrow \phi_j(D, \delta_f^i)$</td>
<td></td>
</tr>
<tr>
<td>maybe$_a$</td>
<td>$0 \ D \leftarrow \phi_j(D, \delta_f^i+1)$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$1 \ D \leftarrow \phi_j(D, \delta_f^i)$</td>
<td></td>
</tr>
<tr>
<td>found$_0$</td>
<td>$D \leftarrow \phi_j(D, \delta_f^i, \delta_f^{i+1})$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta_f^i, \delta_f^{i+1}))$</td>
</tr>
<tr>
<td>found$_1$</td>
<td>$D \leftarrow \phi_j(D, \delta_f^i)$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta_f^i))$</td>
</tr>
<tr>
<td>in$_0$</td>
<td>$D \leftarrow \phi_j(D, \delta_f^i)$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta_f^i))$</td>
</tr>
<tr>
<td>in$_1$</td>
<td>$D \leftarrow \phi_j(D, \delta_f^i)$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta_f^i))$</td>
</tr>
<tr>
<td>found$_a_0$</td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta_f^i, \delta_f^{i+1}))$</td>
</tr>
<tr>
<td>found$_a_1$</td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta_f^i))$</td>
</tr>
</tbody>
</table>
### Table 2.27: Table for Type: dRfoundFoundElInMonMB

<table>
<thead>
<tr>
<th>Initialization</th>
<th>$D \leftarrow \text{id}_f$</th>
<th>$R \leftarrow \text{default}_g$</th>
</tr>
</thead>
<tbody>
<tr>
<td>return</td>
<td>$R$</td>
<td></td>
</tr>
</tbody>
</table>

#### Semantic Letter Decoration

<table>
<thead>
<tr>
<th>Update of $D$</th>
<th>Update of $R$</th>
</tr>
</thead>
<tbody>
<tr>
<td>out</td>
<td></td>
</tr>
<tr>
<td>out$_r$</td>
<td></td>
</tr>
<tr>
<td>out$_a$</td>
<td></td>
</tr>
<tr>
<td>maybe$_b$</td>
<td>$D \leftarrow \delta^i_j$</td>
</tr>
<tr>
<td>maybe$_u$</td>
<td></td>
</tr>
<tr>
<td>found$_u$</td>
<td></td>
</tr>
<tr>
<td>found$_a$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta^i_j))$</td>
</tr>
</tbody>
</table>
Table 2.28: Table for Type: dRFoundFoundEnNoMA

\[
\begin{array}{lll}
\text{Initialization} & D \leftarrow \text{id}_f & R \leftarrow \text{default}_f \\
\text{return} & R\ & \\
\text{Semantic Letter} & & \\
\text{Decoration} & & \\
\text{Update of } D & & \\
\text{Update of } R & & \\
\hline
\text{out} & D \leftarrow \text{id}_f & \\
\text{out}_r & & \\
\text{maybe}_b & D \leftarrow \phi_f(D, \delta_f^i) & \\
\text{maybe}_a & 0 & D \leftarrow \phi_f(D, \delta_f^{i+1}) & \\
\text{maybe}_a & 1 & D \leftarrow \phi_f(D, \delta_f^i) & \\
\text{found} & 0 & D \leftarrow \text{id}_f & \\
\text{found} & 1 & D \leftarrow \text{id}_f & R \leftarrow \phi_u(R, \phi_f(D, \delta_f^i), \delta_f^{i+1}) & \\
\text{in} & 0 & D \leftarrow \text{id}_f & R \leftarrow \phi_u(R, \phi_f(D, \delta_f^i)) & \\
\text{in} & 1 & D \leftarrow \text{id}_f & R \leftarrow \phi_u(R, \delta_f^{i+1}) & \\
\text{found}_a & 0 & D \leftarrow \text{id}_f & R \leftarrow \phi_u(R, \phi_f(D, \delta_f^i), \delta_f^{i+1}) & \\
\text{found}_a & 1 & D \leftarrow \text{id}_f & R \leftarrow \phi_u(R, \phi_f(D, \delta_f^i)) & \\
\end{array}
\]
2.2. DECORATION TABLES

Table 2.29: Table for Type: dRFoundFoundInNoMBA

<table>
<thead>
<tr>
<th>Semantic Letter</th>
<th>After Update of $D$</th>
<th>Decoration Update of $R$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Initialization</strong></td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \text{default}_f$</td>
</tr>
<tr>
<td><strong>return</strong></td>
<td>$R$</td>
<td></td>
</tr>
<tr>
<td><strong>out</strong></td>
<td>$D \leftarrow \text{id}_f$</td>
<td></td>
</tr>
<tr>
<td><strong>out_o</strong></td>
<td>$D \leftarrow \text{id}_f$</td>
<td></td>
</tr>
<tr>
<td><strong>maybe_a</strong></td>
<td>$D \leftarrow \phi_f(D, \delta_f^{i+1})$</td>
<td></td>
</tr>
<tr>
<td><strong>maybe_o</strong></td>
<td>$D \leftarrow \phi_f(D, \delta_f^i)$</td>
<td></td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(\delta_f^i, \delta_f^{i+1}))$</td>
</tr>
<tr>
<td><strong>in</strong></td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \phi_g(R, \delta_f^i)$</td>
</tr>
<tr>
<td><strong>in</strong></td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta_f^{i+1}))$</td>
</tr>
<tr>
<td><strong>found_a</strong></td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(\delta_f^i, \delta_f^{i+1}))$</td>
</tr>
<tr>
<td><strong>found_a</strong></td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow \phi_g(R, \delta_f^i)$</td>
</tr>
</tbody>
</table>
### Table 2.30: Table for Type: dRFoundFoundEInNoMBB

<table>
<thead>
<tr>
<th>Semantic Letter</th>
<th>After Update of $D$</th>
<th>Update of $R$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Initialization</strong></td>
<td>$D \leftarrow id_f$</td>
<td>$R \leftarrow default_g$</td>
</tr>
<tr>
<td><strong>Return</strong></td>
<td>$R \leftarrow default_g$</td>
<td>$R \leftarrow default_g$</td>
</tr>
<tr>
<td><strong>out</strong></td>
<td>$D \leftarrow id_f$</td>
<td>$D \leftarrow id_f$</td>
</tr>
<tr>
<td><strong>out_a</strong></td>
<td>$D \leftarrow id_f$</td>
<td>$D \leftarrow id_f$</td>
</tr>
<tr>
<td><strong>maybe_a</strong></td>
<td>$D \leftarrow \phi_f(D, \delta^{i+1}_f)$</td>
<td>$D \leftarrow \phi_f(D, \delta^{i+1}_f)$</td>
</tr>
<tr>
<td><strong>maybe_a</strong></td>
<td>$D \leftarrow \phi_f(D, \delta^{i+1}_f)$</td>
<td>$D \leftarrow \phi_f(D, \delta^{i+1}_f)$</td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>$D \leftarrow \phi_f(D, \delta^{i+1}_f)$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta^{i+1}_f))$</td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>$D \leftarrow \phi_f(D, \delta^{i+1}_f)$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta^{i+1}_f))$</td>
</tr>
<tr>
<td><strong>in</strong></td>
<td>$D \leftarrow \phi_f(D, \delta^{i+1}_f)$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta^{i+1}_f))$</td>
</tr>
<tr>
<td><strong>in</strong></td>
<td>$D \leftarrow \phi_f(D, \delta^{i+1}_f)$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta^{i+1}_f))$</td>
</tr>
<tr>
<td><strong>found_a</strong></td>
<td>$D \leftarrow id_f$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta^{i+1}_f))$</td>
</tr>
<tr>
<td><strong>found_a</strong></td>
<td>$D \leftarrow id_f$</td>
<td>$R \leftarrow \phi_g(R, \phi_f(D, \delta^{i+1}_f))$</td>
</tr>
</tbody>
</table>
Table 2.31: Table for Type: footprint

<table>
<thead>
<tr>
<th>Semantic Letter</th>
<th>Guard</th>
<th>Update of C</th>
</tr>
</thead>
<tbody>
<tr>
<td>initialization</td>
<td>$p_n = 0$</td>
<td>$C \leftarrow 0$</td>
</tr>
<tr>
<td>return</td>
<td></td>
<td></td>
</tr>
<tr>
<td>out</td>
<td>$p_i = 0$</td>
<td></td>
</tr>
<tr>
<td>out$_r$</td>
<td>$p_i = 0$</td>
<td></td>
</tr>
<tr>
<td>out$_a$</td>
<td>$p_i = 0$</td>
<td></td>
</tr>
<tr>
<td>maybe$_b$</td>
<td>$p_i = p_i + 1$</td>
<td></td>
</tr>
<tr>
<td>maybe$_a$</td>
<td>$p_i = p_i + 1$</td>
<td></td>
</tr>
<tr>
<td>found$_a$</td>
<td>$p_i = C + 1$</td>
<td>$C \leftarrow C + 1$</td>
</tr>
<tr>
<td>found</td>
<td>$p_i = C + 1$</td>
<td>$C \leftarrow C + 1$</td>
</tr>
<tr>
<td>in</td>
<td>$p_i = C$</td>
<td></td>
</tr>
</tbody>
</table>
Table 2.32: Table for Type: footprint_distance

<table>
<thead>
<tr>
<th>Semantic Letter</th>
<th>Guard</th>
<th>Update of $C$</th>
<th>Update of $D$</th>
<th>Update of $F$</th>
<th>Update of $M$</th>
</tr>
</thead>
<tbody>
<tr>
<td>out</td>
<td></td>
<td>$D \leftarrow D + 1$</td>
<td>$M \leftarrow 0$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>out$_v$</td>
<td></td>
<td>$D \leftarrow D + M$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>out$_b$</td>
<td></td>
<td>$D \leftarrow D + 1$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>maybe$_b$</td>
<td></td>
<td>$D \leftarrow D + 1$</td>
<td></td>
<td></td>
<td>$M \leftarrow M + 1$</td>
</tr>
<tr>
<td>found$_a$</td>
<td>$F = 1$</td>
<td>$C \leftarrow \phi_b(C, D)$</td>
<td>$D \leftarrow 0$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>found$_b$</td>
<td>$F = 0$</td>
<td>$D \leftarrow 0$</td>
<td></td>
<td>$F \leftarrow 1$</td>
<td></td>
</tr>
<tr>
<td>found</td>
<td>$F = 1$</td>
<td>$C \leftarrow \phi_b(C, D)$</td>
<td>$D \leftarrow 0$</td>
<td></td>
<td>$F \leftarrow 1$</td>
</tr>
<tr>
<td>found</td>
<td>$F = 0$</td>
<td>$D \leftarrow 0$</td>
<td></td>
<td>$F \leftarrow 1$</td>
<td></td>
</tr>
<tr>
<td>in</td>
<td></td>
<td></td>
<td>$D \leftarrow 0$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 2.33: Table for Type: found

<table>
<thead>
<tr>
<th>Semantic Letter</th>
<th>Guard</th>
<th>Update of C</th>
<th>Update of D</th>
<th>Update of R</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>initialization</strong></td>
<td></td>
<td>C ← default_D</td>
<td>D ← id_D</td>
<td>R ← default_D</td>
</tr>
<tr>
<td></td>
<td>f_0 = 0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>C &lt; g R ⇒ ct_n = 1, at_n = 0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>return</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>C = R, R = default_D ⇒ ct_n = 0, at_n = 0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>C = R, R ≠ default_D ⇒ ct_n = 1, at_n = 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R &lt; g C ⇒ ct_n = 0, at_n = 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>return</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>φ &lt; g (R, C)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| **found**       |       |             |             |             |
|                 | 0     |             |             |             |
|                 | r < g r, f_i = 0, ct_i = 0, at_i = ct_i |             |             |             |
|                 | f_i = 0, ct_i+1 = ct_i, at_i+1 = at_i |             |             |             |
|                 | C ← default_D |             |             |             |
|                 | d, δ_f, δ_f^(i+1) |             |             |             |

| **found_a**     |       |             |             |             |
|                 | 0     |             |             |             |
|                 | r < g φ_f(d, δ_f) < g r, ct_i = f_i, at_i+1 = ct_i |             |             |             |
|                 | φ_f(d, δ_f) < g r, ct_i = f_i, at_i = ct_i, at_i+1 = ct_i |             |             |             |
|                 | f_i = 0, ct_i+1 = ct_i, at_i+1 = at_i |             |             |             |
|                 | C ← φ_f(C, φ_f(D, δ_f^(i+1))) |             |             |             |
|                 | D ← id_D |             |             |             |
|                 | d, δ_f, δ_f^(i+1) |             |             |             |

| **found_d**     |       |             |             |             |
|                 | 0     |             |             |             |
|                 | r < g φ_f(d, δ_f) < g r, ct_i = f_i, at_i = ct_i, at_i+1 = ct_i |             |             |             |
|                 | φ_f(d, δ_f) < g r, ct_i = f_i, at_i+1 = ct_i |             |             |             |
|                 | f_i = 0, ct_i+1 = ct_i, at_i+1 = at_i |             |             |             |
|                 | C ← φ_f(C, φ_f(D, δ_f^(i+1))) |             |             |             |
|                 | D ← id_D |             |             |             |
|                 | d, δ_f, δ_f^(i+1) |             |             |             |

| **found**       |       |             |             |             |
|                 | 1     |             |             |             |
|                 | r < g φ_f(d, δ_f) < g r, ct_i = f_i, at_i = ct_i, at_i+1 = ct_i |             |             |             |
|                 | φ_f(d, δ_f) < g r, ct_i = f_i, at_i = ct_i, at_i+1 = ct_i |             |             |             |
|                 | f_i = 0, ct_i+1 = ct_i, at_i+1 = at_i |             |             |             |
|                 | C ← φ_f(C, φ_f(D, δ_f^(i+1))) |             |             |             |
|                 | D ← id_D |             |             |             |
|                 | d, δ_f, δ_f^(i+1) |             |             |             |
Table 2.34: Table for Type: found\_distance

<table>
<thead>
<tr>
<th>Initialization</th>
<th>Guard</th>
<th>Update of C</th>
<th>Update of F</th>
</tr>
</thead>
<tbody>
<tr>
<td>return</td>
<td>C ← id_y</td>
<td>F ← 0</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Semantic Letter</th>
<th>Guard</th>
<th>Update of C</th>
<th>Update of F</th>
</tr>
</thead>
<tbody>
<tr>
<td>out</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>out_r</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>out_s</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>maybe_a</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>found_a</td>
<td>F &gt; 0</td>
<td>C ← φ_y(C, i − F)</td>
<td>F ← i</td>
</tr>
<tr>
<td>found_u</td>
<td>F = 0</td>
<td>C ← φ_y(C, i − F)</td>
<td>F ← i</td>
</tr>
<tr>
<td>found</td>
<td>F &gt; 0</td>
<td>C ← φ_y(C, i − F)</td>
<td>F ← i</td>
</tr>
<tr>
<td>found</td>
<td>F = 0</td>
<td>C ← φ_y(C, i − F)</td>
<td>F ← i</td>
</tr>
<tr>
<td>in</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## 2.2. DECORATION TABLES

<table>
<thead>
<tr>
<th>Initialization</th>
<th>( C \leftarrow \text{default}_f )</th>
<th>( D \leftarrow \text{id}_f )</th>
<th>( R \leftarrow \text{default}_f )</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Return</strong></td>
<td>( \phi_g(R,C) )</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Semantic Letter</strong></td>
<td><strong>Decoration</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>out</strong></td>
<td>( C \leftarrow \text{default}_1 )</td>
<td>( D \leftarrow \text{id}_f )</td>
<td>( R \leftarrow \phi_g(R,C) )</td>
</tr>
<tr>
<td><strong>out</strong></td>
<td></td>
<td>( D \leftarrow \text{id}_f )</td>
<td></td>
</tr>
<tr>
<td><strong>maybe</strong></td>
<td>( C \leftarrow \text{default}_1 )</td>
<td>( D \leftarrow \phi_1(D,\delta_f^1) )</td>
<td></td>
</tr>
<tr>
<td><strong>maybe</strong></td>
<td>( 0 )</td>
<td>( D \leftarrow \phi_1(D,\delta_f^{1+1}) )</td>
<td></td>
</tr>
<tr>
<td><strong>maybe</strong></td>
<td>( 1 )</td>
<td>( D \leftarrow \phi_1(D,\delta_f^1) )</td>
<td></td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>( 0 )</td>
<td>( C \leftarrow \phi_1(D,\delta_f^1) )</td>
<td>( D \leftarrow \text{id}_f )</td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>( 1 )</td>
<td>( C \leftarrow \phi_1(D,\delta_f^{1+1}) )</td>
<td>( D \leftarrow \text{id}_f )</td>
</tr>
<tr>
<td><strong>in</strong></td>
<td>( 0 )</td>
<td>( C \leftarrow \phi_1(C,\phi_1(D,\delta_f^{1+1})) )</td>
<td>( D \leftarrow \text{id}_f )</td>
</tr>
<tr>
<td><strong>in</strong></td>
<td>( 1 )</td>
<td>( C \leftarrow \phi_1(C,\phi_1(D,\delta_f^1)) )</td>
<td>( D \leftarrow \text{id}_f )</td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>( 0 )</td>
<td>( D \leftarrow \text{id}_f )</td>
<td>( R \leftarrow \phi_g(R,\phi_1(D,\delta_f^{1+1})) )</td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>( 1 )</td>
<td>( D \leftarrow \text{id}_f )</td>
<td>( R \leftarrow \phi_g(R,\phi_1(D,\delta_f^1)) )</td>
</tr>
</tbody>
</table>

*Table 2.35: Table for Type: function*
<table>
<thead>
<tr>
<th>Initialization</th>
<th>Semantic Letter</th>
<th>Glue Matrices and Decoration Tables</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C \leftarrow \text{default}_0$</td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$F \leftarrow \text{default}_0$</td>
</tr>
<tr>
<td>$R \leftarrow \theta(F, C)$</td>
<td>$R \leftarrow \theta(F, C)$</td>
<td></td>
</tr>
<tr>
<td><strong>Table 2.36: Table for Type: predicate</strong></td>
<td><strong>return</strong></td>
<td><strong>R</strong></td>
</tr>
<tr>
<td><strong>out</strong></td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$F \leftarrow C$</td>
</tr>
<tr>
<td><strong>out</strong></td>
<td>$D \leftarrow \text{id}_f$</td>
<td>$R \leftarrow R \land \theta(F, C)$</td>
</tr>
<tr>
<td><strong>maybe</strong></td>
<td>$D \leftarrow \phi_F(D, \delta_f')$</td>
<td>$F \leftarrow C$</td>
</tr>
<tr>
<td><strong>maybe</strong></td>
<td>$D \leftarrow \phi_F(D, \delta_f')$</td>
<td>$R \leftarrow R \land \theta(F, C)$</td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>$C \leftarrow \phi_F(D, \delta_f', \delta_f'^{+1})$</td>
<td>$D \leftarrow \text{id}_f$</td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>$C \leftarrow \phi_F(D, \delta_f')$</td>
<td>$D \leftarrow \text{id}_f$</td>
</tr>
<tr>
<td><strong>in</strong></td>
<td>$C \leftarrow \phi_F(D, \delta_f')$</td>
<td>$D \leftarrow \text{id}_f$</td>
</tr>
<tr>
<td><strong>in</strong></td>
<td>$C \leftarrow \phi_F(D, \delta_f')$</td>
<td>$D \leftarrow \text{id}_f$</td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>$C \leftarrow \phi_F(D, \delta_f', \delta_f'^{+1})$</td>
<td>$D \leftarrow \text{id}_f$</td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>$C \leftarrow \phi_F(D, \delta_f')$</td>
<td>$D \leftarrow \text{id}_f$</td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>$C \leftarrow \phi_F(D, \delta_f')$</td>
<td>$D \leftarrow \text{id}_f$</td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>$C \leftarrow \phi_F(D, \delta_f')$</td>
<td>$D \leftarrow \text{id}_f$</td>
</tr>
</tbody>
</table>

---

98.2 PATTERNS, SEED TRANSDUCERS, GLUE MATRICES AND DECORATION TABLES
Table 2.37: Table for Type: rFoundFoundE

<table>
<thead>
<tr>
<th>initialization</th>
<th>return</th>
<th>( R \leftarrow \text{default}_g / R )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Semantic Letter</td>
<td>Decoration</td>
<td>Update of ( R )</td>
</tr>
<tr>
<td>out</td>
<td>out,</td>
<td>out,</td>
</tr>
<tr>
<td>maybe,</td>
<td>maybe,</td>
<td>maybe,</td>
</tr>
<tr>
<td>found</td>
<td>found</td>
<td>( R \leftarrow \phi_g(R, \delta_i) )</td>
</tr>
<tr>
<td>in</td>
<td>found,</td>
<td>( R \leftarrow \phi_g(R, \delta'_i) )</td>
</tr>
</tbody>
</table>
### Table 2.38: Table for Type: rFoundFoundEin

<table>
<thead>
<tr>
<th>Initialization</th>
<th>$R \leftarrow \text{default}_{\text{if}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>return</td>
<td>$R$</td>
</tr>
<tr>
<td>Semantic Letter</td>
<td>Decoration</td>
</tr>
<tr>
<td>out</td>
<td>Update of $R$</td>
</tr>
<tr>
<td>out$_r$</td>
<td></td>
</tr>
<tr>
<td>out$_a$</td>
<td></td>
</tr>
<tr>
<td>maybe$_b$</td>
<td></td>
</tr>
<tr>
<td>found$_b$</td>
<td></td>
</tr>
<tr>
<td>found$_a$</td>
<td></td>
</tr>
<tr>
<td>found$_e$</td>
<td></td>
</tr>
</tbody>
</table>

$$
\text{out} \quad R \leftarrow \phi_g(R, \phi_f(\delta_i, \delta_{i+1}))
$$

$$
\text{out}_r \quad R \leftarrow \phi_g(R, \delta_{i+1})
$$

$$
\text{out}_a \quad R \leftarrow \phi_g(R, \delta_{i+1})
$$

$$
\text{maybe}_b \quad R \leftarrow \phi_g(R, \delta_{i+1})
$$

$$
\text{found}_b \quad R \leftarrow \phi_g(R, \phi_f(\delta_i, \delta_{i+1}))
$$

$$
\text{found}_a \quad R \leftarrow \phi_g(R, \phi_f(\delta_i, \delta_{i+1}))
$$
Table 2.39: Table for Type: rFoundInSWZigzag

<table>
<thead>
<tr>
<th>Initialization</th>
<th>$R \leftarrow \text{default}_{gf}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>return</td>
<td>$R$</td>
</tr>
<tr>
<td>Semantic Letter</td>
<td>Decoration</td>
</tr>
<tr>
<td>out</td>
<td>$R \leftarrow \phi_g(R, \delta^i_j, \delta^{i+1}_j)$</td>
</tr>
<tr>
<td>out$_r$</td>
<td>$R \leftarrow \phi_g(R, \delta^i_j)$</td>
</tr>
<tr>
<td>out$_a$</td>
<td>$R \leftarrow \phi_g(R, \delta^i_j)$</td>
</tr>
<tr>
<td>maybe$_b$</td>
<td>$R \leftarrow \phi_g(R, \delta^i_j)$</td>
</tr>
<tr>
<td>maybe$_a$</td>
<td>$R \leftarrow \phi_g(R, \delta^i_j)$</td>
</tr>
<tr>
<td>found</td>
<td>$R \leftarrow \phi_g(R, \delta^i_j)$</td>
</tr>
<tr>
<td>found$_a$</td>
<td>$R \leftarrow \phi_g(R, \delta^i_j)$</td>
</tr>
</tbody>
</table>
Table 2.40: Table for Type: rangeCHRFoundEOutA

<table>
<thead>
<tr>
<th>Initialization return</th>
<th>Semantic Letter</th>
<th>Decor</th>
<th>Update of C</th>
<th>Update of H</th>
<th>Update of R</th>
</tr>
</thead>
<tbody>
<tr>
<td>( C \leftarrow \text{default}_{\phi_f} )</td>
<td>out_r</td>
<td>( H \leftarrow \delta_{f}^{i-1} )</td>
<td>( H \leftarrow \delta_{f}^{i+1} )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( H \leftarrow 1 \text{VAR} )</td>
<td>out_a</td>
<td>( C \leftarrow \text{default}_{\phi_f} )</td>
<td>( H \leftarrow \delta_{f}^{i+1} )</td>
<td>( R \leftarrow \phi_f(R, C) )</td>
<td></td>
</tr>
<tr>
<td>( R \leftarrow \text{default}_{g_f} )</td>
<td>maybe_a</td>
<td>( C \leftarrow H - \delta_{f}^{i+1} )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( H \leftarrow \delta_{f}^{i+1} )</td>
<td>found</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( R \leftarrow \phi_f(R, H - \delta_{f}^{i+1}) )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

maybe_b | in | \( C \leftarrow H - \delta_{f}^{i+1} \) |

| maybe_a | in | \( C \leftarrow H - \delta_{f}^{i+1} \) |

found_a | in | \( H \leftarrow \delta_{f}^{i+1} \) | \( R \leftarrow \phi_f(R, H - \delta_{f}^{i+1}) \) |
### 2.2. DECORATION TABLES

Table 2.41: Table for Type: rangeCHRFoundEOutB

<table>
<thead>
<tr>
<th>Initialization</th>
<th>Return</th>
<th>Semantic Letter</th>
<th>Decoration</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C \leftarrow \text{default}_{gf}$</td>
<td>$H \leftarrow \text{VAR}_i$</td>
<td>$R \leftarrow \text{default}_{gf}$</td>
<td>$\phi_g(R, C)$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Action</th>
<th>Update of $C$</th>
<th>Update of $H$</th>
<th>Update of $R$</th>
</tr>
</thead>
<tbody>
<tr>
<td>out $\sigma$</td>
<td>$H \leftarrow \delta_f^{i+1}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>out $\alpha$</td>
<td>$C \leftarrow \text{default}_{gf}$</td>
<td>$H \leftarrow \delta_f^{i+1}$</td>
<td>$R \leftarrow \phi_g(R, C)$</td>
</tr>
<tr>
<td>maybe $\beta$</td>
<td>$C \leftarrow \text{default}_{gf}$</td>
<td>$H \leftarrow \delta_f^{i+1}$</td>
<td></td>
</tr>
<tr>
<td>maybe $\alpha$</td>
<td>$C \leftarrow \text{default}_{gf}$</td>
<td>$H \leftarrow \delta_f^{i+1}$</td>
<td></td>
</tr>
<tr>
<td>found $\sigma$</td>
<td>$C \leftarrow \delta_f^{i+1} - H$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>in $\alpha$</td>
<td>$C \leftarrow \delta_f^{i+1} - H$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>found $\alpha$</td>
<td></td>
<td>$H \leftarrow \delta_f^{i+1}$</td>
<td>$R \leftarrow \phi_g(R, \delta_f^{i+1} - H)$</td>
</tr>
</tbody>
</table>
Table 2.42: Table for Type: rangeHRFoundFoundEInA

<table>
<thead>
<tr>
<th>Initialization</th>
<th>$H \leftarrow \text{VAR}_1$</th>
<th>$R \leftarrow \text{default}_f$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Semantic Letter</td>
<td></td>
<td></td>
</tr>
<tr>
<td>out</td>
<td>$H \leftarrow \delta_j^{i+1}$</td>
<td></td>
</tr>
<tr>
<td>out_r</td>
<td>$H \leftarrow \delta_j^{i+1}$</td>
<td></td>
</tr>
<tr>
<td>out_a</td>
<td>$H \leftarrow \delta_j^{i+1}$</td>
<td></td>
</tr>
<tr>
<td>maybe_b</td>
<td></td>
<td></td>
</tr>
<tr>
<td>maybe_a</td>
<td></td>
<td></td>
</tr>
<tr>
<td>found</td>
<td></td>
<td>$R \leftarrow \phi_b(R, H - \delta_j^{i+1})$</td>
</tr>
<tr>
<td>in</td>
<td></td>
<td>$R \leftarrow \phi_b(R, H - \delta_j^{i+1})$</td>
</tr>
<tr>
<td>found_e</td>
<td>$H \leftarrow \delta_j^{i+1}$</td>
<td>$R \leftarrow \phi_e(R, H - \delta_j^{i+1})$</td>
</tr>
</tbody>
</table>
### 2.2. DECORATION TABLES

Table 2.43: Table for Type: rangeHRFoundFoundEInB

<table>
<thead>
<tr>
<th>Initialization</th>
<th>$H \leftarrow \forall k_1$</th>
<th>$R \leftarrow \text{default}_{gf}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return</td>
<td>$R$</td>
<td></td>
</tr>
<tr>
<td>Semantic Letter</td>
<td>Update of $H$</td>
<td>Update of $R$</td>
</tr>
<tr>
<td>out</td>
<td>$H \leftarrow \delta_i^{i+1}$</td>
<td></td>
</tr>
<tr>
<td>out$_r$</td>
<td>$H \leftarrow \delta_i^{i+1}$</td>
<td></td>
</tr>
<tr>
<td>out$_a$</td>
<td>$H \leftarrow \delta_i^{i+1}$</td>
<td></td>
</tr>
<tr>
<td>maybe$_b$</td>
<td>$H \leftarrow \delta_i^{i+1}$</td>
<td></td>
</tr>
<tr>
<td>maybe$_a$</td>
<td>$R \leftarrow \phi_g(R, \delta_i^{i+1} - H)$</td>
<td></td>
</tr>
<tr>
<td>found</td>
<td>$R \leftarrow \phi_g(R, \delta_i^{i+1} - H)$</td>
<td></td>
</tr>
<tr>
<td>in</td>
<td>$R \leftarrow \phi_g(R, \delta_i^{i+1} - H)$</td>
<td></td>
</tr>
<tr>
<td>found$_a$</td>
<td>$H \leftarrow \delta_i^{i+1}$</td>
<td>$R \leftarrow \phi_g(R, \delta_i^{i+1} - H)$</td>
</tr>
</tbody>
</table>
Table 2.44: Table for Type: rangeRFoundFoundEInA

<table>
<thead>
<tr>
<th>Initialization</th>
<th>$R \leftarrow \text{default}_{gf}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>return</td>
<td>$R$</td>
</tr>
<tr>
<td>Semantic Letter</td>
<td></td>
</tr>
<tr>
<td>Update of $R$</td>
<td></td>
</tr>
<tr>
<td><strong>out</strong></td>
<td>$R \leftarrow \phi_g(R, \max(0, \delta_j^k - \delta_j^{k+1}))$</td>
</tr>
<tr>
<td><strong>out$_r$</strong></td>
<td>$R \leftarrow \phi_g(R, \max(0, \delta_j^k - \delta_j^{k+1}))$</td>
</tr>
<tr>
<td><strong>out$_a$</strong></td>
<td></td>
</tr>
<tr>
<td><strong>maybe$_b$</strong></td>
<td></td>
</tr>
<tr>
<td><strong>found</strong></td>
<td>$R \leftarrow \phi_g(R, \max(0, \delta_j^k - \delta_j^{k+1}))$</td>
</tr>
<tr>
<td><strong>in</strong></td>
<td>$R \leftarrow \phi_g(R, \max(0, \delta_j^k - \delta_j^{k+1}))$</td>
</tr>
<tr>
<td><strong>found$_a$</strong></td>
<td>$R \leftarrow \phi_g(R, \max(0, \delta_j^k - \delta_j^{k+1}))$</td>
</tr>
</tbody>
</table>
Table 2.45: Table for Type: rangeR\text{Found}\text{FoundE}\text{InB}

<table>
<thead>
<tr>
<th>Initialization</th>
<th>$R \leftarrow \text{default}_{\theta f}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>return</td>
<td>$R$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Semantic Letter</th>
<th>Update of $R$</th>
</tr>
</thead>
<tbody>
<tr>
<td>out</td>
<td></td>
</tr>
<tr>
<td>out$_r$</td>
<td></td>
</tr>
<tr>
<td>out$_a$</td>
<td></td>
</tr>
<tr>
<td>maybe$_a$</td>
<td></td>
</tr>
<tr>
<td>found</td>
<td>$R \leftarrow \phi_{g}(R, \max(0, \delta_{i}^{j+1} - \delta_{j}^{j}))$</td>
</tr>
<tr>
<td>in</td>
<td>$R \leftarrow \phi_{g}(R, \max(0, \delta_{i}^{j+1} - \delta_{j}^{j}))$</td>
</tr>
<tr>
<td>found$_a$</td>
<td>$R \leftarrow \phi_{g}(R, \max(0, \delta_{i}^{j+1} - \delta_{j}^{j}))$</td>
</tr>
</tbody>
</table>
Table 2.46: Table for Type: range_function

<table>
<thead>
<tr>
<th>Initialization</th>
<th>Return</th>
<th>Decoration</th>
</tr>
</thead>
<tbody>
<tr>
<td>( C \leftarrow \text{default}_f )</td>
<td>( H \leftarrow \text{VAR}_1 )</td>
<td>( R \leftarrow \text{default}_f )</td>
</tr>
<tr>
<td><strong>Semantic Letter</strong></td>
<td><strong>Update of ( C )</strong></td>
<td><strong>Update of ( H )</strong></td>
</tr>
<tr>
<td>out</td>
<td>( H \leftarrow \delta_{i+1}^f )</td>
<td>( H \leftarrow \delta_{i+1}^f )</td>
</tr>
<tr>
<td>out(_{\text{r}})</td>
<td>( C \leftarrow \text{default}_f )</td>
<td>( R \leftarrow \phi_g(R,C) )</td>
</tr>
<tr>
<td>out(_{\text{u}})</td>
<td>( H \leftarrow \delta_{i+1}^f )</td>
<td>( R \leftarrow \phi_g(R,C) )</td>
</tr>
<tr>
<td>maybe(_{\text{b}})</td>
<td>( H \leftarrow \delta_{i+1}^f )</td>
<td>( R \leftarrow \phi_g(R,C) )</td>
</tr>
<tr>
<td>found(_{\text{u}})</td>
<td>( H \leftarrow \delta_{i+1}^f )</td>
<td>( R \leftarrow \phi_g(R,</td>
</tr>
<tr>
<td>found</td>
<td>( C \leftarrow</td>
<td>H - \delta_{i+1}^f</td>
</tr>
<tr>
<td>in</td>
<td>( C \leftarrow</td>
<td>H - \delta_{i+1}^f</td>
</tr>
</tbody>
</table>
### Table 2.47: Table for Type: range\_predicate

<table>
<thead>
<tr>
<th></th>
<th>Update of $C$</th>
<th>Update of $F$</th>
<th>Update of $H$</th>
<th>Update of $R$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Initialization</strong></td>
<td>$C \leftarrow \text{default}_0$</td>
<td>$F \leftarrow \text{default}_0$</td>
<td>$H \leftarrow \text{VAR}_1$</td>
<td>$R \leftarrow 1$</td>
</tr>
<tr>
<td><strong>Semantic Letter</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>out</td>
<td>$H \leftarrow \delta_j^{i+1}$</td>
<td>$H \leftarrow \delta_j^{i+1}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>out$_a$</td>
<td>$F \leftarrow</td>
<td>H - \delta_j^{i+1}</td>
<td>$</td>
<td>$H \leftarrow \delta_j^{i+1}$</td>
</tr>
<tr>
<td>maybe$_a$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>found$_a$</td>
<td>$C \leftarrow</td>
<td>H - \delta_j^{i+1}</td>
<td>$</td>
<td>$F \leftarrow</td>
</tr>
<tr>
<td>found</td>
<td>$C \leftarrow</td>
<td>H - \delta_j^{i+1}</td>
<td>$</td>
<td></td>
</tr>
<tr>
<td>in</td>
<td>$C \leftarrow</td>
<td>H - \delta_j^{i+1}</td>
<td>$</td>
<td></td>
</tr>
</tbody>
</table>
110 2. PATTERNS, SEED TRANSDUCERS, GLUE MATRICES AND DECORATION TABLES
3

Global Constraint Catalogue

“Mon problème, avec les classements, c’est qu’ils ne durent pas ; à peine ai-je fini de mettre de l’ordre que cet ordre est déjà caduc.”
– Georges Perec, *Penser/Classer*

```
<table>
<thead>
<tr>
<th>Contents</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1  ALL_EQUAL_HEIGHT_DECREASING_TERRACE</td>
<td>126</td>
</tr>
<tr>
<td>3.2  ALL_EQUAL_HEIGHT_INCREASING_TERRACE</td>
<td>130</td>
</tr>
<tr>
<td>3.3  ALL_EQUAL_HEIGHT_PLAIN</td>
<td>134</td>
</tr>
<tr>
<td>3.4  ALL_EQUAL_HEIGHT_PLATEAU</td>
<td>138</td>
</tr>
<tr>
<td>3.5  ALL_EQUAL_HEIGHT_PROPER_PLAIN</td>
<td>142</td>
</tr>
<tr>
<td>3.6  ALL_EQUAL_HEIGHT_PROPER_PLATEAU</td>
<td>146</td>
</tr>
<tr>
<td>3.7  ALL_EQUAL_HEIGHT_STEADY</td>
<td>150</td>
</tr>
<tr>
<td>3.8  ALL_EQUAL_HEIGHT_STEADY_SEQUENCE</td>
<td>154</td>
</tr>
<tr>
<td>3.9  ALL_EQUAL_MAX_BUMP_ON_DECREASING_SEQUENCE</td>
<td>158</td>
</tr>
<tr>
<td>3.10 ALL_EQUAL_MAX_DECREASING</td>
<td>162</td>
</tr>
<tr>
<td>3.11 ALL_EQUAL_MAX_DECREASING_SEQUENCE</td>
<td>166</td>
</tr>
<tr>
<td>3.12 ALL_EQUAL_MAX_DIP_ON_INCREASING_SEQUENCE</td>
<td>170</td>
</tr>
<tr>
<td>3.13 ALL_EQUAL_MAX_INCREASING</td>
<td>174</td>
</tr>
<tr>
<td>3.14 ALL_EQUAL_MAX_INCREASING_SEQUENCE</td>
<td>178</td>
</tr>
<tr>
<td>3.15 ALL_EQUAL_MAX_INFLEXION</td>
<td>182</td>
</tr>
<tr>
<td>3.16 ALL_EQUAL_MAX_PEAK</td>
<td>184</td>
</tr>
<tr>
<td>3.17 ALL_EQUAL_MAX_STRICTLY_DECREASING_SEQUENCE</td>
<td>188</td>
</tr>
<tr>
<td>3.18 ALL_EQUAL_MAX_STRICTLY_INCREASING_SEQUENCE</td>
<td>192</td>
</tr>
<tr>
<td>3.19 ALL_EQUAL_MAX_SUMMIT</td>
<td>196</td>
</tr>
<tr>
<td>3.20 ALL_EQUAL_MAX_ZIGZAG</td>
<td>200</td>
</tr>
<tr>
<td>3.21 ALL_EQUAL_MIN_BUMP_ON_DECREASING_SEQUENCE</td>
<td>204</td>
</tr>
<tr>
<td>3.22 ALL_EQUAL_MIN_DECREASING</td>
<td>208</td>
</tr>
<tr>
<td>3.23 ALL_EQUAL_MIN_DECREASING_SEQUENCE</td>
<td>212</td>
</tr>
</tbody>
</table>
```
3. GLOBAL CONSTRAINT CATALOGUE

3.24 ALL_EQUAL_MIN_DIP_ON_INCREASING_SEQUENCE ........ 216
3.25 ALL_EQUAL_MIN_GORGE ................................ 220
3.26 ALL_EQUAL_MIN_INCREASING .............................. 222
3.27 ALL_EQUAL_MIN_INCREASING_SEQUENCE ................. 226
3.28 ALL_EQUAL_MIN_INFLEXION ................................. 230
3.29 ALL_EQUAL_MIN_STRICTLY_DECREASING_SEQUENCE ...... 234
3.30 ALL_EQUAL_MIN_STRICTLY_INCREASING_SEQUENCE ....... 238
3.31 ALL_EQUAL_MIN_VALLEY .................................... 242
3.32 ALL_EQUAL_MIN_ZIGZAG .................................... 246
3.33 ALL_EQUAL_RANGE_DECREASING ......................... 250
3.34 ALL_EQUAL_RANGE_DECREASING_SEQUENCE .............. 254
3.35 ALL_EQUAL_RANGE_INCREASING ......................... 258
3.36 ALL_EQUAL_RANGE_INCREASING_SEQUENCE .............. 262
3.37 ALL_EQUAL_RANGE_STRICTLY_DECREASING_SEQUENCE ... 266
3.38 ALL_EQUAL_RANGE_STRICTLY_INCREASING_SEQUENCE .... 270
3.39 ALL_EQUAL_SURF_BUMP_ON_DECREASING_SEQUENCE .... 274
3.40 ALL_EQUAL_SURF_DECREASING ......................... 278
3.41 ALL_EQUAL_SURF_DECREASING_SEQUENCE .............. 282
3.42 ALL_EQUAL_SURF_DECREASING_TERRACE .................. 286
3.43 ALL_EQUAL_SURF_DIP_ON_INCREASING_SEQUENCE ...... 290
3.44 ALL_EQUAL_SURF_GORGE .................................... 294
3.45 ALL_EQUAL_SURF_INCREASING ......................... 298
3.46 ALL_EQUAL_SURF_INCREASING_SEQUENCE .............. 300
3.47 ALL_EQUAL_SURF_INCREASING_TERRACE ................. 304
3.48 ALL_EQUAL_SURF_INFLEXION ................................. 308
3.49 ALL_EQUAL_SURF_PEAK ...................................... 312
3.50 ALL_EQUAL_SURF_PLAIN ..................................... 316
3.51 ALL_EQUAL_SURF_PLATEAU ................................. 320
3.52 ALL_EQUAL_SURF_PROPER_PLAIN ......................... 324
3.53 ALL_EQUAL_SURF_PROPER_PLATEAU ....................... 328
3.54 ALL_EQUAL_SURF_STEADY .................................... 332
3.55 ALL_EQUAL_SURF_STEADY_SEQUENCE ...................... 336
3.56 ALL_EQUAL_SURF_STRICTLY_DECREASING_SEQUENCE .... 340
3.57 ALL_EQUAL_SURF_STRICTLY_INCREASING_SEQUENCE ..... 344
3.58 ALL_EQUAL_SURF_SUMMIT .................................... 348
3.59 ALL_EQUAL_SURF_VALLEY .................................... 352
3.60 ALL_EQUAL_SURF_ZIGZAG .................................... 356
3.61 ALL_EQUAL_WIDTH_DECREASING_SEQUENCE ............. 360
3.62 ALL_EQUAL_WIDTH_DECREASING_TERRACE ................. 364
3.63 ALL_EQUAL_WIDTH_GORGE .................................... 368
3.64 ALL_EQUAL_WIDTH_INCREASING_SEQUENCE .............. 372
3.65 ALL_EQUAL_WIDTH_INCREASING_TERRACE ................. 376
3.66 ALL_EQUAL_WIDTH_INFLEXION ............................... 380
<table>
<thead>
<tr>
<th>ID</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.67</td>
<td>ALL_EQUAL_WIDTH_PEAK</td>
<td>382</td>
</tr>
<tr>
<td>3.68</td>
<td>ALL_EQUAL_WIDTH_PLAIN</td>
<td>386</td>
</tr>
<tr>
<td>3.69</td>
<td>ALL_EQUAL_WIDTH_PLATEAU</td>
<td>390</td>
</tr>
<tr>
<td>3.70</td>
<td>ALL_EQUAL_WIDTH_PROPER_PLAIN</td>
<td>394</td>
</tr>
<tr>
<td>3.71</td>
<td>ALL_EQUAL_WIDTH_PROPER_PLATEAU</td>
<td>398</td>
</tr>
<tr>
<td>3.72</td>
<td>ALL_EQUAL_WIDTH_STEADY_SEQUENCE</td>
<td>402</td>
</tr>
<tr>
<td>3.73</td>
<td>ALL_EQUAL_WIDTH.Strictly_DECREASING_SEQUENCE</td>
<td>406</td>
</tr>
<tr>
<td>3.74</td>
<td>ALL_EQUAL_WIDTH.Strictly_INCREASING_SEQUENCE</td>
<td>410</td>
</tr>
<tr>
<td>3.75</td>
<td>ALL_EQUAL_WIDTH_SUMMIT</td>
<td>414</td>
</tr>
<tr>
<td>3.76</td>
<td>ALL_EQUAL_WIDTH_VALLEY</td>
<td>418</td>
</tr>
<tr>
<td>3.77</td>
<td>ALL_EQUAL_WIDTH_ZIGZAG</td>
<td>422</td>
</tr>
<tr>
<td>3.78</td>
<td>DECREASING_HEIGHT_DECREASING_TERRACE</td>
<td>426</td>
</tr>
<tr>
<td>3.79</td>
<td>DECREASING_HEIGHT_INCREASING_TERRACE</td>
<td>430</td>
</tr>
<tr>
<td>3.80</td>
<td>DECREASING_HEIGHT_PLAIN</td>
<td>434</td>
</tr>
<tr>
<td>3.81</td>
<td>DECREASING_HEIGHT_PLATEAU</td>
<td>438</td>
</tr>
<tr>
<td>3.82</td>
<td>DECREASING_HEIGHT_PROPER_PLAIN</td>
<td>442</td>
</tr>
<tr>
<td>3.83</td>
<td>DECREASING_HEIGHT_PROPER_PLATEAU</td>
<td>446</td>
</tr>
<tr>
<td>3.84</td>
<td>DECREASING_HEIGHT_STEADY</td>
<td>450</td>
</tr>
<tr>
<td>3.85</td>
<td>DECREASING_HEIGHT_STEADY_SEQUENCE</td>
<td>454</td>
</tr>
<tr>
<td>3.86</td>
<td>DECREASING_MAX_BUMP_ON_DECREASING_SEQUENCE</td>
<td>458</td>
</tr>
<tr>
<td>3.87</td>
<td>DECREASING_MAX_DECREASING</td>
<td>462</td>
</tr>
<tr>
<td>3.88</td>
<td>DECREASING_MAX_DECREASING_SEQUENCE</td>
<td>466</td>
</tr>
<tr>
<td>3.89</td>
<td>DECREASING_MAX_DIP_ON_INCREASING_SEQUENCE</td>
<td>470</td>
</tr>
<tr>
<td>3.90</td>
<td>DECREASING_MAX_INCREASING</td>
<td>474</td>
</tr>
<tr>
<td>3.91</td>
<td>DECREASING_MAX_INCREASING_SEQUENCE</td>
<td>478</td>
</tr>
<tr>
<td>3.92</td>
<td>DECREASING_MAX_INFLEXION</td>
<td>482</td>
</tr>
<tr>
<td>3.93</td>
<td>DECREASING_MAX_PEAK</td>
<td>486</td>
</tr>
<tr>
<td>3.94</td>
<td>DECREASING_MAX.Strictly_DECREASING_SEQUENCE</td>
<td>490</td>
</tr>
<tr>
<td>3.95</td>
<td>DECREASING_MAX.Strictly_INCREASING_SEQUENCE</td>
<td>494</td>
</tr>
<tr>
<td>3.96</td>
<td>DECREASING_MAX_SUMMIT</td>
<td>498</td>
</tr>
<tr>
<td>3.97</td>
<td>DECREASING_MAX_ZIGZAG</td>
<td>502</td>
</tr>
<tr>
<td>3.98</td>
<td>DECREASING_MIN_BUMP_ON_DECREASING_SEQUENCE</td>
<td>506</td>
</tr>
<tr>
<td>3.99</td>
<td>DECREASING_MIN_DECREASING</td>
<td>510</td>
</tr>
<tr>
<td>3.100</td>
<td>DECREASING_MIN_DECREASING_SEQUENCE</td>
<td>514</td>
</tr>
<tr>
<td>3.101</td>
<td>DECREASING_MIN_DIP_ON_INCREASING_SEQUENCE</td>
<td>518</td>
</tr>
<tr>
<td>3.102</td>
<td>DECREASING_MIN_GORGE</td>
<td>522</td>
</tr>
<tr>
<td>3.103</td>
<td>DECREASING_MIN_INCREASING</td>
<td>526</td>
</tr>
<tr>
<td>3.104</td>
<td>DECREASING_MIN_INCREASING_SEQUENCE</td>
<td>530</td>
</tr>
<tr>
<td>3.105</td>
<td>DECREASING_MIN_INFLEXION</td>
<td>534</td>
</tr>
<tr>
<td>3.106</td>
<td>DECREASING_MIN.Strictly_DECREASING_SEQUENCE</td>
<td>538</td>
</tr>
<tr>
<td>3.107</td>
<td>DECREASING_MIN.Strictly_INCREASING_SEQUENCE</td>
<td>542</td>
</tr>
<tr>
<td>3.108</td>
<td>DECREASING_MIN_VALLEY</td>
<td>546</td>
</tr>
<tr>
<td>3.109</td>
<td>DECREASING_MIN_ZIGZAG</td>
<td>550</td>
</tr>
</tbody>
</table>
### 3. GLOBAL CONSTRAINT CATALOGUE

<table>
<thead>
<tr>
<th>Constraint Name</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECREASING_RANGE_DECENDING</td>
<td>554</td>
</tr>
<tr>
<td>DECREASING_RANGE_DECENDING_SEQUENCE</td>
<td>558</td>
</tr>
<tr>
<td>DECREASING_RANGE_INCREASING</td>
<td>562</td>
</tr>
<tr>
<td>DECREASING_RANGE_INCREASING_SEQUENCE</td>
<td>566</td>
</tr>
<tr>
<td>DECREASING_RANGE.Strictly_Decreasing_Sequence</td>
<td>570</td>
</tr>
<tr>
<td>DECREASING_RANGE.Strictly_Increasing_Sequence</td>
<td>574</td>
</tr>
<tr>
<td>DECREASING_SURF_BUMP_ON_DECREASING_SEQUENCE</td>
<td>578</td>
</tr>
<tr>
<td>DECREASING_SURF_DECERING</td>
<td>582</td>
</tr>
<tr>
<td>DECREASING_SURF_DECENDING_SEQUENCE</td>
<td>586</td>
</tr>
<tr>
<td>DECREASING_SURF_DECENDING_TERRACE</td>
<td>590</td>
</tr>
<tr>
<td>DECREASING_SURF_DIP_ON_INCREASING_SEQUENCE</td>
<td>594</td>
</tr>
<tr>
<td>DECREASING_SURF_GORGE</td>
<td>598</td>
</tr>
<tr>
<td>DECREASING_SURF_INCREASING</td>
<td>602</td>
</tr>
<tr>
<td>DECREASING_SURF_INCREASING_SEQUENCE</td>
<td>604</td>
</tr>
<tr>
<td>DECREASING_SURF_INCREASING_TERRACE</td>
<td>608</td>
</tr>
<tr>
<td>DECREASING_SURF_INflexION</td>
<td>612</td>
</tr>
<tr>
<td>DECREASING_SURF_PEAK</td>
<td>616</td>
</tr>
<tr>
<td>DECREASING_SURF.PLAIN</td>
<td>620</td>
</tr>
<tr>
<td>DECREASING_SURF_PLATEAU</td>
<td>624</td>
</tr>
<tr>
<td>DECREASING_SURF_PROPER.PLAIN</td>
<td>628</td>
</tr>
<tr>
<td>DECREASING_SURF_PROPER_PLATEAU</td>
<td>632</td>
</tr>
<tr>
<td>DECREASING_SURF_STEADY</td>
<td>636</td>
</tr>
<tr>
<td>DECREASING_SURF_STEADY.SEQUENCE</td>
<td>640</td>
</tr>
<tr>
<td>DECREASING_SURF.Strictly_DECENDING_Sequence</td>
<td>644</td>
</tr>
<tr>
<td>DECREASING_SURF.Strictly_Increasing_Sequence</td>
<td>648</td>
</tr>
<tr>
<td>DECREASING_SURF_Summit</td>
<td>652</td>
</tr>
<tr>
<td>DECREASING_SURF_Valley</td>
<td>656</td>
</tr>
<tr>
<td>DECREASING_SURF_ZIGZAG</td>
<td>660</td>
</tr>
<tr>
<td>DECREASING_WIDTH_DECENDING_SEQUENCE</td>
<td>664</td>
</tr>
<tr>
<td>DECREASING_WIDTH_DECENDING_TERRACE</td>
<td>668</td>
</tr>
<tr>
<td>DECREASING_WIDTH.GORGE</td>
<td>672</td>
</tr>
<tr>
<td>DECREASING_WIDTH_INCREASING_SEQUENCE</td>
<td>676</td>
</tr>
<tr>
<td>DECREASING_WIDTH_INCREASING_TERRACE</td>
<td>680</td>
</tr>
<tr>
<td>DECREASING_WIDTH_INflexION</td>
<td>684</td>
</tr>
<tr>
<td>DECREASING_WIDTH_PEAK</td>
<td>688</td>
</tr>
<tr>
<td>DECREASING_WIDTH.PLAIN</td>
<td>692</td>
</tr>
<tr>
<td>DECREASING_WIDTH_PLATEAU</td>
<td>696</td>
</tr>
<tr>
<td>DECREASING_WIDTH_PROPER.PLAIN</td>
<td>700</td>
</tr>
<tr>
<td>DECREASING_WIDTH_PROPER_PLATEAU</td>
<td>704</td>
</tr>
<tr>
<td>DECREASING_WIDTH_STEADY.SEQUENCE</td>
<td>708</td>
</tr>
<tr>
<td>DECREASING_WIDTH.Strictly_DECENDING_Sequence</td>
<td>710</td>
</tr>
<tr>
<td>DECREASING_WIDTH.Strictly_Increasing_Sequence</td>
<td>714</td>
</tr>
<tr>
<td>DECREASING_WIDTH.Summit</td>
<td>718</td>
</tr>
<tr>
<td>Section</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>3.153</td>
<td>DECREASING_WIDTH_VALLEY</td>
</tr>
<tr>
<td>3.154</td>
<td>DECREASING_WIDTH_ZIGZAG</td>
</tr>
<tr>
<td>3.155</td>
<td>INCREASING_HEIGHT_DECREASING_TERRACE</td>
</tr>
<tr>
<td>3.156</td>
<td>INCREASING_HEIGHT_INCREASING_TERRACE</td>
</tr>
<tr>
<td>3.157</td>
<td>INCREASING_HEIGHT_PLAIN</td>
</tr>
<tr>
<td>3.158</td>
<td>INCREASING_HEIGHT_PLATEAU</td>
</tr>
<tr>
<td>3.159</td>
<td>INCREASING_HEIGHT_PROPER_PLAIN</td>
</tr>
<tr>
<td>3.160</td>
<td>INCREASING_HEIGHT_PROPER_PLATEAU</td>
</tr>
<tr>
<td>3.161</td>
<td>INCREASING_HEIGHT_STEADY</td>
</tr>
<tr>
<td>3.162</td>
<td>INCREASING_HEIGHT_STEADY_SEQUENCE</td>
</tr>
<tr>
<td>3.163</td>
<td>INCREASING_MAX_BUMP_ON_DECREASING_SEQUENCE</td>
</tr>
<tr>
<td>3.164</td>
<td>INCREASING_MAX_DECREASING</td>
</tr>
<tr>
<td>3.165</td>
<td>INCREASING_MAX_DECREASING_SEQUENCE</td>
</tr>
<tr>
<td>3.166</td>
<td>INCREASING_MAX_DIP_ON_INCREASING_SEQUENCE</td>
</tr>
<tr>
<td>3.167</td>
<td>INCREASING_MAX_INCREASING</td>
</tr>
<tr>
<td>3.168</td>
<td>INCREASING_MAX_INCREASING_SEQUENCE</td>
</tr>
<tr>
<td>3.169</td>
<td>INCREASING_MAX_INFLEXION</td>
</tr>
<tr>
<td>3.170</td>
<td>INCREASING_MAX_PEAK</td>
</tr>
<tr>
<td>3.171</td>
<td>INCREASING_MAX.Strictly_DECREASING_SEQUENCE</td>
</tr>
<tr>
<td>3.172</td>
<td>INCREASING_MAX.Strictly_INCREASING_SEQUENCE</td>
</tr>
<tr>
<td>3.173</td>
<td>INCREASING_MAX_SUMMIT</td>
</tr>
<tr>
<td>3.174</td>
<td>INCREASING_MAX_ZIGZAG</td>
</tr>
<tr>
<td>3.175</td>
<td>INCREASING_MIN_BUMP_ON_DECREASING_SEQUENCE</td>
</tr>
<tr>
<td>3.176</td>
<td>INCREASING_MIN_DECREASING</td>
</tr>
<tr>
<td>3.177</td>
<td>INCREASING_MIN_DECREASING_SEQUENCE</td>
</tr>
<tr>
<td>3.178</td>
<td>INCREASING_MIN_DIP_ON_INCREASING_SEQUENCE</td>
</tr>
<tr>
<td>3.179</td>
<td>INCREASING_MIN_GORGE</td>
</tr>
<tr>
<td>3.180</td>
<td>INCREASING_MIN_INCREASING</td>
</tr>
<tr>
<td>3.181</td>
<td>INCREASING_MIN_INCREASING_SEQUENCE</td>
</tr>
<tr>
<td>3.182</td>
<td>INCREASING_MIN_INFLEXION</td>
</tr>
<tr>
<td>3.183</td>
<td>INCREASING_MIN.Strictly_DECREASING_SEQUENCE</td>
</tr>
<tr>
<td>3.184</td>
<td>INCREASING_MIN.Strictly_INCREASING_SEQUENCE</td>
</tr>
<tr>
<td>3.185</td>
<td>INCREASING_MIN_VALLEY</td>
</tr>
<tr>
<td>3.186</td>
<td>INCREASING_MIN_ZIGZAG</td>
</tr>
<tr>
<td>3.187</td>
<td>INCREASING_RANGE_DECREASING</td>
</tr>
<tr>
<td>3.188</td>
<td>INCREASING_RANGE_DECREASING_SEQUENCE</td>
</tr>
<tr>
<td>3.189</td>
<td>INCREASING_RANGE_INCREASING</td>
</tr>
<tr>
<td>3.190</td>
<td>INCREASING_RANGE_INCREASING_SEQUENCE</td>
</tr>
<tr>
<td>3.191</td>
<td>INCREASING_RANGE.Strictly_DECREASING_SEQUENCE</td>
</tr>
<tr>
<td>3.192</td>
<td>INCREASING_RANGE.Strictly_INCREASING_SEQUENCE</td>
</tr>
<tr>
<td>3.193</td>
<td>INCREASING_SURF_BUMP_ON_DECREASING_SEQUENCE</td>
</tr>
<tr>
<td>3.194</td>
<td>INCREASING_SURF_DECREASING</td>
</tr>
<tr>
<td>3.195</td>
<td>INCREASING_SURF_DECREASING_SEQUENCE</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>3.196</td>
<td>INCREASING_SURF_DECREASING_TERRACE</td>
</tr>
<tr>
<td>3.197</td>
<td>INCREASING_SURF_DIP_ON_INCREASING_SEQUENCE</td>
</tr>
<tr>
<td>3.198</td>
<td>INCREASING_SURF_GORGE</td>
</tr>
<tr>
<td>3.199</td>
<td>INCREASING_SURF_INCREASING</td>
</tr>
<tr>
<td>3.200</td>
<td>INCREASING_SURF_INCREASING_SEQUENCE</td>
</tr>
<tr>
<td>3.201</td>
<td>INCREASING_SURF_INCREASING_TERRACE</td>
</tr>
<tr>
<td>3.202</td>
<td>INCREASING_SURF_INFLEXION</td>
</tr>
<tr>
<td>3.203</td>
<td>INCREASING_SURF_PEAK</td>
</tr>
<tr>
<td>3.204</td>
<td>INCREASING_SURF_PLAIN</td>
</tr>
<tr>
<td>3.205</td>
<td>INCREASING_SURF_PLATEAU</td>
</tr>
<tr>
<td>3.206</td>
<td>INCREASING_SURF_PROPER_PLAIN</td>
</tr>
<tr>
<td>3.207</td>
<td>INCREASING_SURF_PROPER_PLATEAU</td>
</tr>
<tr>
<td>3.208</td>
<td>INCREASING_SURF_STEADY</td>
</tr>
<tr>
<td>3.209</td>
<td>INCREASING_SURF_STEADY_SEQUENCE</td>
</tr>
<tr>
<td>3.210</td>
<td>INCREASING_SURF_STRICTLY_DECREASING_SEQUENCE</td>
</tr>
<tr>
<td>3.211</td>
<td>INCREASING_SURF_STRICTLY_INCREASING_SEQUENCE</td>
</tr>
<tr>
<td>3.212</td>
<td>INCREASING_SURF_SUMMIT</td>
</tr>
<tr>
<td>3.213</td>
<td>INCREASING_SURF_VALLEY</td>
</tr>
<tr>
<td>3.214</td>
<td>INCREASING_SURF_ZIGZAG</td>
</tr>
<tr>
<td>3.215</td>
<td>INCREASING_WIDTH_DECREASING_SEQUENCE</td>
</tr>
<tr>
<td>3.216</td>
<td>INCREASING_WIDTH_DECREASING_TERRACE</td>
</tr>
<tr>
<td>3.217</td>
<td>INCREASING_WIDTH_GORGE</td>
</tr>
<tr>
<td>3.218</td>
<td>INCREASING_WIDTH_INCREASING_SEQUENCE</td>
</tr>
<tr>
<td>3.219</td>
<td>INCREASING_WIDTH_INCREASING_TERRACE</td>
</tr>
<tr>
<td>3.220</td>
<td>INCREASING_WIDTH_INFLEXION</td>
</tr>
<tr>
<td>3.221</td>
<td>INCREASING_WIDTH_PEAK</td>
</tr>
<tr>
<td>3.222</td>
<td>INCREASING_WIDTH_PLAIN</td>
</tr>
<tr>
<td>3.223</td>
<td>INCREASING_WIDTH_PLATEAU</td>
</tr>
<tr>
<td>3.224</td>
<td>INCREASING_WIDTH_PROPER_PLAIN</td>
</tr>
<tr>
<td>3.225</td>
<td>INCREASING_WIDTH_PROPER_PLATEAU</td>
</tr>
<tr>
<td>3.226</td>
<td>INCREASING_WIDTH_STEADY_SEQUENCE</td>
</tr>
<tr>
<td>3.227</td>
<td>INCREASING_WIDTH_STRICTLY_DECREASING_SEQUENCE</td>
</tr>
<tr>
<td>3.228</td>
<td>INCREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE</td>
</tr>
<tr>
<td>3.229</td>
<td>INCREASING_WIDTH_SUMMIT</td>
</tr>
<tr>
<td>3.230</td>
<td>INCREASING_WIDTH_VALLEY</td>
</tr>
<tr>
<td>3.231</td>
<td>INCREASING_WIDTH_ZIGZAG</td>
</tr>
<tr>
<td>3.232</td>
<td>MAX_HEIGHT_DECREASING_TERRACE</td>
</tr>
<tr>
<td>3.233</td>
<td>MAX_HEIGHT_INCREASING_TERRACE</td>
</tr>
<tr>
<td>3.234</td>
<td>MAX_HEIGHT_PLAIN</td>
</tr>
<tr>
<td>3.235</td>
<td>MAX_HEIGHT_PLATEAU</td>
</tr>
<tr>
<td>3.236</td>
<td>MAX_HEIGHT_PROPER_PLAIN</td>
</tr>
<tr>
<td>3.237</td>
<td>MAX_HEIGHT_PROPER_PLATEAU</td>
</tr>
<tr>
<td>3.238</td>
<td>MAX_HEIGHT_STEADY</td>
</tr>
<tr>
<td>Page</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>3.239</td>
<td>MAX_HEIGHT_STEADY_SEQUENCE</td>
</tr>
<tr>
<td>3.240</td>
<td>MAX_MAX_BUMP_ON_DECREASING_SEQUENCE</td>
</tr>
<tr>
<td>3.241</td>
<td>MAX_MAX_DECR</td>
</tr>
<tr>
<td>3.242</td>
<td>MAX_MAX_DECoğluNG_SEQUENCE</td>
</tr>
<tr>
<td>3.243</td>
<td>MAX_MAX_DIP_ON_INCREASING_SEQUENCE</td>
</tr>
<tr>
<td>3.244</td>
<td>MAX_MAX_INC</td>
</tr>
<tr>
<td>3.245</td>
<td>MAX_MAX_INCR</td>
</tr>
<tr>
<td>3.246</td>
<td>MAX_MAX_INFLEXION</td>
</tr>
<tr>
<td>3.247</td>
<td>MAX_MAX_PEAK</td>
</tr>
<tr>
<td>3.248</td>
<td>MAX_MAX_STRICTLY_DEC</td>
</tr>
<tr>
<td>3.249</td>
<td>MAX_MAX_STRICTLY_INC</td>
</tr>
<tr>
<td>3.250</td>
<td>MAX_MAX_SUMMIT</td>
</tr>
<tr>
<td>3.251</td>
<td>MAX_MAX_ZIGZAG</td>
</tr>
<tr>
<td>3.252</td>
<td>MAX_MIN_BUMP_ON_DEC</td>
</tr>
<tr>
<td>3.253</td>
<td>MAX_MIN_DCR</td>
</tr>
<tr>
<td>3.254</td>
<td>MAX_MIN_DECR</td>
</tr>
<tr>
<td>3.255</td>
<td>MAX_MIN_INCR</td>
</tr>
<tr>
<td>3.256</td>
<td>MAX_MIN_DIP_ON_INC</td>
</tr>
<tr>
<td>3.257</td>
<td>MAX_MIN_GORGE</td>
</tr>
<tr>
<td>3.258</td>
<td>MAX_MIN_INCR</td>
</tr>
<tr>
<td>3.259</td>
<td>MAX_MIN_INFLEXION</td>
</tr>
<tr>
<td>3.260</td>
<td>MAX_MIN_STRICTLY_DEC</td>
</tr>
<tr>
<td>3.261</td>
<td>MAX_MIN_STRICTLY_INC</td>
</tr>
<tr>
<td>3.262</td>
<td>MAX_MIN_VALLEY</td>
</tr>
<tr>
<td>3.263</td>
<td>MAX_MIN_ZIGZAG</td>
</tr>
<tr>
<td>3.264</td>
<td>MAX_RANGE_DEC</td>
</tr>
<tr>
<td>3.265</td>
<td>MAX_RANGE_DECSEQUENCE</td>
</tr>
<tr>
<td>3.266</td>
<td>MAX_RANGE_INCR</td>
</tr>
<tr>
<td>3.267</td>
<td>MAX_RANGE_INCRSEQUENCE</td>
</tr>
<tr>
<td>3.268</td>
<td>MAX_RANGE_STRICTLY_DEC</td>
</tr>
<tr>
<td>3.269</td>
<td>MAX_RANGE_STRICTLY_INC</td>
</tr>
<tr>
<td>3.270</td>
<td>MAX_SURF_BUMP_ON_INCR</td>
</tr>
<tr>
<td>3.271</td>
<td>MAX_SURF_DEC</td>
</tr>
<tr>
<td>3.272</td>
<td>MAX_SURF_DECSEQUENCE</td>
</tr>
<tr>
<td>3.273</td>
<td>MAX_SURF_DECSEQUENCE_TERRACE</td>
</tr>
<tr>
<td>3.274</td>
<td>MAX_SURF_DIP_ON_INCRSEQUENCE</td>
</tr>
<tr>
<td>3.275</td>
<td>MAX_SURF_GORGE</td>
</tr>
<tr>
<td>3.276</td>
<td>MAX_SURF_INCR</td>
</tr>
<tr>
<td>3.277</td>
<td>MAX_SURF_INCRSEQUENCE</td>
</tr>
<tr>
<td>3.278</td>
<td>MAX_SURF_INCRSEQUENCE_TERRACE</td>
</tr>
<tr>
<td>3.279</td>
<td>MAX_SURF_INFLEXION</td>
</tr>
<tr>
<td>3.280</td>
<td>MAX_SURF_PEAK</td>
</tr>
<tr>
<td>3.281</td>
<td>MAX_SURF_PLAIN</td>
</tr>
</tbody>
</table>
## 3. GLOBAL CONSTRAINT CATALOGUE

<table>
<thead>
<tr>
<th>Constraint Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAX_SURF_PLATEAU</td>
<td>1238</td>
</tr>
<tr>
<td>MAX_SURF_PROPER_PLAIN</td>
<td>1242</td>
</tr>
<tr>
<td>MAX_SURF_PROPER_PLATEAU</td>
<td>1246</td>
</tr>
<tr>
<td>MAX_SURF_STEADY</td>
<td>1250</td>
</tr>
<tr>
<td>MAX_SURF_STEADY_SEQUENCE</td>
<td>1254</td>
</tr>
<tr>
<td>MAX_SURF_STRICTLY_DECREASING_SEQUENCE</td>
<td>1258</td>
</tr>
<tr>
<td>MAX_SURF_STRICTLY_INCREASING_SEQUENCE</td>
<td>1262</td>
</tr>
<tr>
<td>MAX_SURF_SUMMIT</td>
<td>1266</td>
</tr>
<tr>
<td>MAX_SURF_VALLEY</td>
<td>1270</td>
</tr>
<tr>
<td>MAX_SURF_ZIGZAG</td>
<td>1274</td>
</tr>
<tr>
<td>MAX_WIDTH_DECREASING_SEQUENCE</td>
<td>1280</td>
</tr>
<tr>
<td>MAX_WIDTH_DECREASING_TERRACE</td>
<td>1284</td>
</tr>
<tr>
<td>MAX_WIDTH_GORGE</td>
<td>1288</td>
</tr>
<tr>
<td>MAX_WIDTH_INCREASING_SEQUENCE</td>
<td>1292</td>
</tr>
<tr>
<td>MAX_WIDTH_INCREASING_TERRACE</td>
<td>1298</td>
</tr>
<tr>
<td>MAX_WIDTH_INFLEXION</td>
<td>1302</td>
</tr>
<tr>
<td>MAX_WIDTH_PEAK</td>
<td>1306</td>
</tr>
<tr>
<td>MAX_WIDTH_PLAIN</td>
<td>1310</td>
</tr>
<tr>
<td>MAX_WIDTH_PLATEAU</td>
<td>1314</td>
</tr>
<tr>
<td>MAX_WIDTH_PROPER_PLAIN</td>
<td>1318</td>
</tr>
<tr>
<td>MAX_WIDTH_PROPER_PLATEAU</td>
<td>1322</td>
</tr>
<tr>
<td>MAX_WIDTH_STEADY_SEQUENCE</td>
<td>1326</td>
</tr>
<tr>
<td>MAX_WIDTH_STRICTLY_DECREASING_SEQUENCE</td>
<td>1330</td>
</tr>
<tr>
<td>MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE</td>
<td>1334</td>
</tr>
<tr>
<td>MAX_WIDTH_SUMMIT</td>
<td>1338</td>
</tr>
<tr>
<td>MAX_WIDTH_VALLEY</td>
<td>1342</td>
</tr>
<tr>
<td>MAX_WIDTH_ZIGZAG</td>
<td>1348</td>
</tr>
<tr>
<td>MIN_HEIGHT_DECREASING_TERRACE</td>
<td>1352</td>
</tr>
<tr>
<td>MIN_HEIGHT_INCREASING_TERRACE</td>
<td>1358</td>
</tr>
<tr>
<td>MIN_HEIGHT_PLAIN</td>
<td>1362</td>
</tr>
<tr>
<td>MIN_HEIGHT_PLATEAU</td>
<td>1366</td>
</tr>
<tr>
<td>MIN_HEIGHT_PROPER_PLAIN</td>
<td>1370</td>
</tr>
<tr>
<td>MIN_HEIGHT_PROPER_PLATEAU</td>
<td>1374</td>
</tr>
<tr>
<td>MIN_HEIGHT_STEADY</td>
<td>1378</td>
</tr>
<tr>
<td>MIN_HEIGHT_STEADY_SEQUENCE</td>
<td>1382</td>
</tr>
<tr>
<td>MIN_MAX_BUMP_ON_DECREASING_SEQUENCE</td>
<td>1386</td>
</tr>
<tr>
<td>MIN_MAX_DECREASING</td>
<td>1390</td>
</tr>
<tr>
<td>MIN_MAX_DECREASING_SEQUENCE</td>
<td>1394</td>
</tr>
<tr>
<td>MIN_MAX_DIP_ON_INCREASING_SEQUENCE</td>
<td>1398</td>
</tr>
<tr>
<td>MIN_MAX_INCREASING</td>
<td>1402</td>
</tr>
<tr>
<td>MIN_MAX_INCREASING_SEQUENCE</td>
<td>1406</td>
</tr>
<tr>
<td>MIN_MAX_INFLEXION</td>
<td>1410</td>
</tr>
<tr>
<td>MIN_MAX_PEAK</td>
<td>1414</td>
</tr>
<tr>
<td>Constraint ID</td>
<td>Constraint Name</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------</td>
</tr>
<tr>
<td>3.368</td>
<td>MIN_SURF_ZIGZAG</td>
</tr>
<tr>
<td>3.369</td>
<td>MIN_WIDTH_DEC_INCREMENTAL_SEQUENCE</td>
</tr>
<tr>
<td>3.370</td>
<td>MIN_WIDTH_EXPIAL_INCREMENTAL_TERRACE</td>
</tr>
<tr>
<td>3.371</td>
<td>MIN_WIDTH_GORGE</td>
</tr>
<tr>
<td>3.372</td>
<td>MIN_WIDTH_EXPIAL_SEQUENCE</td>
</tr>
<tr>
<td>3.373</td>
<td>MIN_WIDTH_EXPIAL_TERRACE</td>
</tr>
<tr>
<td>3.374</td>
<td>MIN_WIDTH_INFLEXION</td>
</tr>
<tr>
<td>3.375</td>
<td>MIN_WIDTH_PEAK</td>
</tr>
<tr>
<td>3.376</td>
<td>MIN_WIDTH_PLAIN</td>
</tr>
<tr>
<td>3.377</td>
<td>MIN_WIDTH_PLATEAU</td>
</tr>
<tr>
<td>3.378</td>
<td>MIN_WIDTH_PROPER_PLAIN</td>
</tr>
<tr>
<td>3.379</td>
<td>MIN_WIDTH_PROPER_PLATEAU</td>
</tr>
<tr>
<td>3.380</td>
<td>MIN_WIDTH_STEADY_SEQUENCE</td>
</tr>
<tr>
<td>3.381</td>
<td>MIN_WIDTH_STRICTLY_EXPIAL_SEQUENCE</td>
</tr>
<tr>
<td>3.382</td>
<td>MIN_WIDTH_STRICTLY_EXPIAL_INCREMENTAL_SEQUENCE</td>
</tr>
<tr>
<td>3.383</td>
<td>MIN_WIDTH_SUMMIT</td>
</tr>
<tr>
<td>3.384</td>
<td>MIN_WIDTH_VALLEY</td>
</tr>
<tr>
<td>3.385</td>
<td>MIN_WIDTH_ZIGZAG</td>
</tr>
<tr>
<td>3.386</td>
<td>NB_BUMP_ON_EXPIAL_SEQUENCE</td>
</tr>
<tr>
<td>3.387</td>
<td>NB_DECENDING</td>
</tr>
<tr>
<td>3.388</td>
<td>NB_EXPIAL_SEQUENCE</td>
</tr>
<tr>
<td>3.389</td>
<td>NB_EXPIAL_TERRACE</td>
</tr>
<tr>
<td>3.390</td>
<td>NB_DIP_ON_EXPIAL_SEQUENCE</td>
</tr>
<tr>
<td>3.391</td>
<td>NB_GORGE</td>
</tr>
<tr>
<td>3.392</td>
<td>NB_INCRING</td>
</tr>
<tr>
<td>3.393</td>
<td>NB_INCRINGAL_SEQUENCE</td>
</tr>
<tr>
<td>3.394</td>
<td>NB_INCRINGAL_TERRACE</td>
</tr>
<tr>
<td>3.395</td>
<td>NB_INFLEXION</td>
</tr>
<tr>
<td>3.396</td>
<td>NB_PEAK</td>
</tr>
<tr>
<td>3.397</td>
<td>NB_PLAIN</td>
</tr>
<tr>
<td>3.398</td>
<td>NB_PLATEAU</td>
</tr>
<tr>
<td>3.399</td>
<td>NB_PROPER_PLAIN</td>
</tr>
<tr>
<td>3.400</td>
<td>NB_PROPER_PLATEAU</td>
</tr>
<tr>
<td>3.401</td>
<td>NB_STEADY</td>
</tr>
<tr>
<td>3.402</td>
<td>NB_STEADYAL_SEQUENCE</td>
</tr>
<tr>
<td>3.403</td>
<td>NB_STRICTLY_EXPIAL_SEQUENCE</td>
</tr>
<tr>
<td>3.404</td>
<td>NB_STRICTLY_EXPIAL_INCREMENTAL_SEQUENCE</td>
</tr>
<tr>
<td>3.405</td>
<td>NB_SUMMIT</td>
</tr>
<tr>
<td>3.406</td>
<td>NB_VALLEY</td>
</tr>
<tr>
<td>3.407</td>
<td>NB_ZIGZAG</td>
</tr>
<tr>
<td>3.408</td>
<td>POS_MAX_HEIGHT_EXPIAL_TERRACE</td>
</tr>
<tr>
<td>3.409</td>
<td>POS_MAX_HEIGHT_EXPIAL_INCREMENTAL_TERRACE</td>
</tr>
<tr>
<td>3.410</td>
<td>POS_MAX_HEIGHT_PLAIN</td>
</tr>
</tbody>
</table>
3.411 POS_MAX_HEIGHT_PLATEAU .......................... 1772
3.412 POS_MAX_HEIGHT_PROPERPLAIN .................. 1776
3.413 POS_MAX_HEIGHT_PROPER_PLATEAU ................. 1780
3.414 POS_MAX_HEIGHT_STEADY .......................... 1784
3.415 POS_MAX_HEIGHT_STEADY_SEQUENCE ............... 1788
3.416 POS_MAX_MAX_BUMP_ON_DECREASING_SEQUENCE ..... 1792
3.417 POS_MAX_MAX_DECREASING ......................... 1796
3.418 POS_MAX_MAX_DECREASING_SEQUENCE ............... 1800
3.419 POS_MAX_MAX_DIP_ON_INCREASING_SEQUENCE ....... 1804
3.420 POS_MAX_MAX_INCREASING ......................... 1808
3.421 POS_MAX_MAX_INCREASING_SEQUENCE ............... 1812
3.422 POS_MAX_MAX_INFLEXION .......................... 1816
3.423 POS_MAX_MAX_PEAK ............................... 1820
3.424 POS_MAX_MAX.StrictlyDecreasingSequence ....... 1824
3.425 POS_MAX_MAX.StrictlyIncreasingSequence ........ 1828
3.426 POS_MAX_MAX_SUMMIT .............................. 1832
3.427 POS_MAX_MAX_ZIGZAG ............................. 1836
3.428 POS_MAX_MIN_BUMP_ON_DECREASING_SEQUENCE ..... 1840
3.429 POS_MAX_MIN_DECREASING ......................... 1844
3.430 POS_MAX_MIN_DECREASING_SEQUENCE ............... 1848
3.431 POS_MAX_MIN_DIP_ON_INCREASINGSEQUENCE ....... 1852
3.432 POS_MAX_MIN_GORGE ................................ 1856
3.433 POS_MAX_MIN_INCREASING ......................... 1860
3.434 POS_MAX_MIN_INCREASING_SEQUENCE ............... 1864
3.435 POS_MAX_MIN_INFLEXION .......................... 1868
3.436 POS_MAX_MIN.StrictlyDecreasingSequence ....... 1872
3.437 POS_MAX_MIN.StrictlyIncreasingSequence ........ 1876
3.438 POS_MAX_MIN_VALLEY .............................. 1880
3.439 POS_MAX_MIN_ZIGZAG ............................. 1884
3.440 POS_MAX_SURF_BUMP_ON_DECREASING_SEQUENCE ..... 1888
3.441 POS_MAX_SURF_DECREASING ......................... 1892
3.442 POS_MAX_SURF_DECREASING_SEQUENCE ............... 1896
3.443 POS_MAX_SURF_DECREASING.TERRACE ............... 1900
3.444 POS_MAX_SURF_DIP_ON_INCREASINGSEQUENCE ....... 1904
3.445 POS_MAX_SURF_GORGE .............................. 1908
3.446 POS_MAX_SURF_INCREASING ......................... 1912
3.447 POS_MAX_SURF_INCREASING_SEQUENCE ............... 1916
3.448 POS_MAX_SURF_INCREASING.TERRACE ............... 1920
3.449 POS_MAX_SURF_INFLEXION .......................... 1924
3.450 POS_MAX_SURF_PEAK ............................... 1928
3.451 POS_MAX_SURF_PLAIN .............................. 1932
3.452 POS_MAX_SURF_PLATEAU ............................ 1936
3.453 POS_MAX_SURF_PROPER_PLAIN ....................... 1940
### 3.454 POS_MAX_SURF_PROPER_PLATEAU

1944

### 3.455 POS_MAX_SURF_STEADY

1948

### 3.456 POS_MAX_SURF_STEADY_SEQUENCE

1952

### 3.457 POS_MAX_SURF_STRICTLY_DECREASING_SEQUENCE

1956

### 3.458 POS_MAX_SURF_STRICTLY_INCREASING_SEQUENCE

1960

### 3.459 POS_MAX_SURF_SUMMIT

1964

### 3.460 POS_MAX_SURF_VALLEY

1968

### 3.461 POS_MAX_SURF_ZIGZAG

1972

### 3.462 POS_MAX_WIDTH_DECREASING_SEQUENCE

1976

### 3.463 POS_MAX_WIDTH_DECREASING_TERRACE

1980

### 3.464 POS_MAX_WIDTH_GORGE

1984

### 3.465 POS_MAX_WIDTH_INCREASING_SEQUENCE

1988

### 3.466 POS_MAX_WIDTH_INCREASING_TERRACE

1992

### 3.467 POS_MAX_WIDTH_INFLEXION

1996

### 3.468 POS_MAX_WIDTH_PEAK

2000

### 3.469 POS_MAX_WIDTH_PLAIN

2004

### 3.470 POS_MAX_WIDTH_PLATEAU

2008

### 3.471 POS_MAX_WIDTH_PROPER_PLAIN

2012

### 3.472 POS_MAX_WIDTH_PROPER_PLATEAU

2016

### 3.473 POS_MAX_WIDTH_STEADY_SEQUENCE

2020

### 3.474 POS_MAX_WIDTH_STRICTLY_DECREASING_SEQUENCE

2024

### 3.475 POS_MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE

2028

### 3.476 POS_MAX_WIDTH_SUMMIT

2032

### 3.477 POS_MAX_WIDTH_VALLEY

2036

### 3.478 POS_MAX_WIDTH_ZIGZAG

2040

### 3.479 POS_MIN_HEIGHT_DECREASING_TERRACE

2044

### 3.480 POS_MIN_HEIGHT_INCREASING_TERRACE

2048

### 3.481 POS_MIN_HEIGHT_PLAIN

2052

### 3.482 POS_MIN_HEIGHT_PLATEAU

2056

### 3.483 POS_MIN_HEIGHT_PROPER_PLAIN

2060

### 3.484 POS_MIN_HEIGHT_PROPER_PLATEAU

2064

### 3.485 POS_MIN_HEIGHT_STEADY

2068

### 3.486 POS_MIN_HEIGHT_STEADY_SEQUENCE

2072

### 3.487 POS_MIN_MAX_BUMP_ON_DECREASING_SEQUENCE

2076

### 3.488 POS_MIN_MAX_DECREASING

2080

### 3.489 POS_MIN_MAX_DECREASING_SEQUENCE

2084

### 3.490 POS_MIN_MAX_DIP_ON_INCREASING_SEQUENCE

2088

### 3.491 POS_MIN_MAX_INCREASING

2092

### 3.492 POS_MIN_MAX_INCREASING_SEQUENCE

2096

### 3.493 POS_MIN_MAX_INFLEXION

2100

### 3.494 POS_MIN_MAX_PEAK

2104

### 3.495 POS_MIN_MAX_STRICTLY_DECREASING_SEQUENCE

2108

### 3.496 POS_MIN_MAX_STRICTLY_INCREASING_SEQUENCE

2112
3.497 POS_MIN_MAX_SUMMIT .......................... 2116
3.498 POS_MIN_MAX_ZIGZAG .......................... 2120
3.499 POS_MIN_MAX_BUMP_ON_DECREASING_SEQUENCE .......................... 2124
3.500 POS_MIN_MIN_DECREASING .......................... 2128
3.501 POS_MIN_MIN_DECREASING_SEQUENCE .......................... 2132
3.502 POS_MIN_MIN_DIP_ON_INCREASING_SEQUENCE .......................... 2136
3.503 POS_MIN_MIN_BUMP_ON_DECREASING_SEQUENCE .......................... 2140
3.504 POS_MIN_MIN_INCREASING .......................... 2144
3.505 POS_MIN_MIN_INCREASING_SEQUENCE .......................... 2148
3.506 POS_MIN_MIN_INFLEXION .......................... 2152
3.507 POS_MIN_MIN_STRICTLY_DECREASING_SEQUENCE .......................... 2156
3.508 POS_MIN_MIN_STRICTLY_INCREASING_SEQUENCE .......................... 2160
3.509 POS_MIN_MIN_VALLEY .......................... 2164
3.510 POS_MIN_MIN_GORGE .......................... 2168
3.511 POS_MIN_SURF_BUMP_ON_DECREASING_SEQUENCE .......................... 2172
3.512 POS_MIN_SURF_DECREASING .......................... 2176
3.513 POS_MIN_SURF_DECREASING_SEQUENCE .......................... 2180
3.514 POS_MIN_SURF_DECREASING_TERRACE .......................... 2184
3.515 POS_MIN_SURF_DIP_ON_INCREASING_SEQUENCE .......................... 2188
3.516 POS_MIN_SURF_GORGE .......................... 2192
3.517 POS_MIN_SURF_INCREASING .......................... 2196
3.518 POS_MIN_SURF_INCREASING_SEQUENCE .......................... 2200
3.519 POS_MIN_SURF_INCREASING_TERRACE .......................... 2204
3.520 POS_MIN_SURF_INFLEXION .......................... 2208
3.521 POS_MIN_SURF_PEAK .......................... 2212
3.522 POS_MIN_SURF_PLAIN .......................... 2216
3.523 POS_MIN_SURF_PLATEAU .......................... 2220
3.524 POS_MIN_SURF_PROPER_PLAIN .......................... 2224
3.525 POS_MIN_SURF_PROPER_PLATEAU .......................... 2228
3.526 POS_MIN_SURF_STEADY .......................... 2232
3.527 POS_MIN_SURF_STEADY_SEQUENCE .......................... 2236
3.528 POS_MIN_SURF_STRICTLY_DECREASING_SEQUENCE .......................... 2240
3.529 POS_MIN_SURF_STRICTLY_INCREASING_SEQUENCE .......................... 2244
3.530 POS_MIN_SURF_SUMMIT .......................... 2248
3.531 POS_MIN_SURF_VALLEY .......................... 2252
3.532 POS_MIN_SURF_ZIGZAG .......................... 2256
3.533 POS_MIN_WIDTH_DECREASING_SEQUENCE .......................... 2260
3.534 POS_MIN_WIDTH_DECREASING_TERRACE .......................... 2264
3.535 POS_MIN_WIDTH_GORGE .......................... 2268
3.536 POS_MIN_WIDTH_INCREASING_SEQUENCE .......................... 2272
3.537 POS_MIN_WIDTH_INCREASING_TERRACE .......................... 2276
3.538 POS_MIN_WIDTH_INFLEXION .......................... 2280
3.539 POS_MIN_WIDTH_PEAK .......................... 2284
3. GLOBAL CONSTRAINT CATALOGUE

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.540</td>
<td>POS_MIN_WIDTH_PLAIN</td>
<td>2288</td>
</tr>
<tr>
<td>3.541</td>
<td>POS_MIN_WIDTH_PLATEAU</td>
<td>2292</td>
</tr>
<tr>
<td>3.542</td>
<td>POS_MIN_WIDTH_PROPER_PLAIN</td>
<td>2296</td>
</tr>
<tr>
<td>3.543</td>
<td>POS_MIN_WIDTH_PROPER_PLATEAU</td>
<td>2300</td>
</tr>
<tr>
<td>3.544</td>
<td>POS_MIN_WIDTH_STEADY_SEQUENCE</td>
<td>2304</td>
</tr>
<tr>
<td>3.545</td>
<td>POS_MIN_WIDTH.Strictly_DECREASING_SEQUENCE</td>
<td>2308</td>
</tr>
<tr>
<td>3.546</td>
<td>POS_MIN_WIDTH.Strictly_INCREASING_SEQUENCE</td>
<td>2312</td>
</tr>
<tr>
<td>3.547</td>
<td>POS_MIN_WIDTH_SUMMIT</td>
<td>2316</td>
</tr>
<tr>
<td>3.548</td>
<td>POS_MIN_WIDTH_VALLEY</td>
<td>2320</td>
</tr>
<tr>
<td>3.549</td>
<td>POS_MIN_WIDTH.ZIGZAG</td>
<td>2324</td>
</tr>
<tr>
<td>3.550</td>
<td>SUM_HEIGHT_DECREASING_TERRACE</td>
<td>2328</td>
</tr>
<tr>
<td>3.551</td>
<td>SUM_HEIGHT_INCREASING_TERRACE</td>
<td>2334</td>
</tr>
<tr>
<td>3.552</td>
<td>SUM_HEIGHT_PLAIN</td>
<td>2340</td>
</tr>
<tr>
<td>3.553</td>
<td>SUM_HEIGHT_PLATEAU</td>
<td>2344</td>
</tr>
<tr>
<td>3.554</td>
<td>SUM_HEIGHT_PROPER_PLAIN</td>
<td>2348</td>
</tr>
<tr>
<td>3.555</td>
<td>SUM_HEIGHT_PROPER_PLATEAU</td>
<td>2352</td>
</tr>
<tr>
<td>3.556</td>
<td>SUM_HEIGHT_STEADY</td>
<td>2356</td>
</tr>
<tr>
<td>3.557</td>
<td>SUM_HEIGHT_STEADY.SEQUENCE</td>
<td>2360</td>
</tr>
<tr>
<td>3.558</td>
<td>SUM_MAX_BUMP_ON_DECREASINGSEQUENCE</td>
<td>2364</td>
</tr>
<tr>
<td>3.559</td>
<td>SUM_MAX_DECREASING</td>
<td>2368</td>
</tr>
<tr>
<td>3.560</td>
<td>SUM_MAX_DECREASING.SEQUENCE</td>
<td>2372</td>
</tr>
<tr>
<td>3.561</td>
<td>SUM_MAX_DIP_ON_INCREASINGSEQUENCE</td>
<td>2376</td>
</tr>
<tr>
<td>3.562</td>
<td>SUM_MAX_INCREASING</td>
<td>2380</td>
</tr>
<tr>
<td>3.563</td>
<td>SUM_MAX_INCREASING.SEQUENCE</td>
<td>2384</td>
</tr>
<tr>
<td>3.564</td>
<td>SUM_MAX_INFLEXION</td>
<td>2388</td>
</tr>
<tr>
<td>3.565</td>
<td>SUM_MAX_PEAK</td>
<td>2392</td>
</tr>
<tr>
<td>3.566</td>
<td>SUM_MAX_STRICTLY_DECREASINGSEQUENCE</td>
<td>2396</td>
</tr>
<tr>
<td>3.567</td>
<td>SUM_MAX_STRICTLY_INCREASINGSEQUENCE</td>
<td>2400</td>
</tr>
<tr>
<td>3.568</td>
<td>SUM_MAX_SUMMIT</td>
<td>2404</td>
</tr>
<tr>
<td>3.569</td>
<td>SUM_MAX.ZIGZAG</td>
<td>2408</td>
</tr>
<tr>
<td>3.570</td>
<td>SUM_MIN_BUMP_ON_DECREASINGSEQUENCE</td>
<td>2414</td>
</tr>
<tr>
<td>3.571</td>
<td>SUM_MIN_DECREASING</td>
<td>2418</td>
</tr>
<tr>
<td>3.572</td>
<td>SUM_MIN_DECREASING.SEQUENCE</td>
<td>2422</td>
</tr>
<tr>
<td>3.573</td>
<td>SUM_MIN_DIP_ON_INCREASINGSEQUENCE</td>
<td>2426</td>
</tr>
<tr>
<td>3.574</td>
<td>SUM_MIN_GORGE</td>
<td>2430</td>
</tr>
<tr>
<td>3.575</td>
<td>SUM_MIN_INCREASING</td>
<td>2434</td>
</tr>
<tr>
<td>3.576</td>
<td>SUM_MIN_INCREASING.SEQUENCE</td>
<td>2438</td>
</tr>
<tr>
<td>3.577</td>
<td>SUM_MIN_INFLEXION</td>
<td>2442</td>
</tr>
<tr>
<td>3.578</td>
<td>SUM_MIN_STRICTLY_DECREASINGSEQUENCE</td>
<td>2446</td>
</tr>
<tr>
<td>3.579</td>
<td>SUM_MIN_STRICTLY_INCREASINGSEQUENCE</td>
<td>2450</td>
</tr>
<tr>
<td>3.580</td>
<td>SUM_MIN_VALLEY</td>
<td>2454</td>
</tr>
<tr>
<td>3.581</td>
<td>SUM_MIN.ZIGZAG</td>
<td>2458</td>
</tr>
<tr>
<td>3.582</td>
<td>SUM_RANGE_DECREASING</td>
<td>2464</td>
</tr>
</tbody>
</table>
3.583 SUM_RANGE_DECREASING_SEQUENCE ............... 2468
3.584 SUM_RANGE_INCREASING .......................... 2472
3.585 SUM_RANGE_STRICTLY_DECREASING_SEQUENCE ... 2476
3.586 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE .... 2480
3.587 SUM_RANGE_INCREASING_SEQUENCE .............. 2484
3.588 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2488
3.589 SUM_RANGE_INCREASING_SEQUENCE .............. 2492
3.590 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2496
3.591 SUM_RANGE_INCREASING_SEQUENCE .............. 2500
3.592 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2504
3.593 SUM_RANGE_INCREASING_SEQUENCE .............. 2508
3.594 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2512
3.595 SUM_RANGE_INCREASING_SEQUENCE .............. 2516
3.596 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2520
3.597 SUM_RANGE_INCREASING_SEQUENCE .............. 2524
3.598 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2528
3.599 SUM_RANGE_INCREASING_SEQUENCE .............. 2532
3.600 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2536
3.601 SUM_RANGE_INCREASING_SEQUENCE .............. 2540
3.602 SUM_RANGE_INCREASING_SEQUENCE .............. 2544
3.603 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2548
3.604 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2552
3.605 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2556
3.606 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2560
3.607 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2564
3.608 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2568
3.609 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2572
3.610 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2576
3.611 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2580
3.612 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2584
3.613 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2588
3.614 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2592
3.615 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2596
3.616 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2600
3.617 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2604
3.618 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2608
3.619 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2612
3.620 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2616
3.621 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2620
3.622 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2624
3.623 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2628
3.624 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2632
3.625 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2636
3.626 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2640
3.627 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2644
3.628 SUM_RANGE_STRICTLY_INCREASING_SEQUENCE ... 2648
3.1 ALL_EQUAL_HEIGHT_DECREASING_TERRACE

**DESCRIPTION**

- **Origin**: Based on the DECREASING_TERRACE pattern.
- **Constraint**: ALL_EQUAL_HEIGHT_DECREASING_TERRACE(VARIABLES)
- **Argument**: VARIABLES : collection(var−dvar)
- **Restriction**: required(VARIABLES, var)

**Purpose**

Succeeds if the minima of the values in each occurrence of the DECREASING_TERRACE pattern in the time-series given by the VARIABLES collection are all the same.

An occurrence of the pattern DECREASING_TERRACE is the maximal subsequence which matches the regular expression $> =^+ >$.

Assume that the occurrence of the pattern DECREASING_TERRACE starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i + 1$ to index $j$.

**Example**

$((6, 3, 3, 1, 5, 4, 3, 2, 4, 4, 5, 3, 3, 1, 1))$

Figure 3.1 provides an example where the ALL_EQUAL_HEIGHT_DECREASING_TERRACE $(6, 3, 3, 1, 5, 4, 3, 2, 4, 4, 5, 3, 3, 1, 1))$ constraint holds.

![Graph Illustrating ALL_EQUAL_HEIGHT_DECREASING_TERRACE](image)

Figure 3.1: Illustrating the ALL_EQUAL_HEIGHT_DECREASING_TERRACE constraint of the Example slot
Typical

\[ |\text{VARIABLES}| > 3 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 2 \]
Figure 3.2 depicts the automaton associated with the constraint \textsc{all} \textsc{equal-height-decreasing-terrace}.

\[
\begin{align*}
&\{ C \leftarrow X \\
&\{ D \leftarrow +\infty \\
&\{ F \leftarrow X \\
&\{ R \leftarrow 1 \}
\}
\end{align*}
\begin{align*}
&\leq s \quad \leq \\
&\Rightarrow \;
\end{align*}
\begin{align*}
&\{ D \leftarrow \min(D, \text{VAR}_i) \}
\end{align*}

\text{Figure 3.2: Automaton for the} \textsc{all} \textsc{equal-height-decreasing-terrace} \text{constraint obtained by applying decoration Table 2.36 to the seed transducer of the} \textsc{decreasing-terrace} \text{pattern}
3.2 ALL_EQUAL_HEIGHT_INCREASING_TERRACE

**Origin**
Based on the INCREASING_TERRACE pattern.

**Constraint**
ALL_EQUAL_HEIGHT_INCREASING_TERRACE(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES, var)

Succeeds if the minima of the values in each occurrence of the INCREASING_TERRACE pattern in the time-series given by the VARIABLES collection are all the same.

**Purpose**
An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression $\langle=+\rangle$.
Assume that the occurrence of the pattern INCREASING_TERRACE starts at position $i$ and ends at position $j$. The feature $\text{MIN}$ computes the minimum of the values from index $i+1$ to index $j$.

**Example**

$$\langle(1, 4, 4, 6, 2, 3, 4, 4, 5, 3, 2, 4, 4, 6, 6)\rangle$$

Figure 3.3 provides an example where the ALL_EQUAL_HEIGHT_INCREASING_TERRACE ($\langle[1, 4, 4, 6, 2, 3, 4, 4, 5, 3, 2, 4, 4, 6, 6]\rangle$) constraint holds.

Figure 3.3: Illustrating the ALL_EQUAL_HEIGHT_INCREASING_TERRACE constraint of the Example slot
Typical

\[
|\text{VARIABLES}| > 3 \\
\text{range(VARIABLES.var)} > 2
\]
Figure 3.4 depicts the automaton associated with the constraint \texttt{ALL\_EQUAL\_HEIGHT\_INCREASING\_TERRACE}.

\begin{align*}
\{ C \leftarrow X \\
\{ D \leftarrow +\infty \\
\{ F \leftarrow X \\
\{ R \leftarrow 1 \\
\} \geq s
\end{align*}

\begin{align*}
\{ D \leftarrow \min(D, \text{VAR}_i) \}
\end{align*}

\begin{align*}
\{ C \leftarrow \min(D, \text{VAR}_i) \\
D \leftarrow +\infty \\
F \leftarrow \min(D, \text{VAR}_i) \\
R \leftarrow R \land (F = \min(D, \text{VAR}_i))
\end{align*}

Figure 3.4: Automaton for the \texttt{ALL\_EQUAL\_HEIGHT\_INCREASING\_TERRACE} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \texttt{INCREASING\_TERRACE} pattern.
### 3.3 ALL_EQUAL_HEIGHT_PLAIN

#### DESCRIPTION

**Origin**
Based on the PLAIN pattern.

**Constraint**

\[
\text{ALL_EQUAL_HEIGHT_PLAIN}((\text{VARIABLES}))
\]

**Argument**

\[
\text{VARIABLES} : \text{collection}(\text{var--dvar})
\]

**Restriction**

\[
\text{required}(\text{VARIABLES}, \text{var})
\]

#### AUTOMATON

Succeeds if the minima of the values in each occurrence of the PLAIN pattern in the time-series given by the VARIABLES collection are all the same.

An occurrence of the pattern PLAIN is the \textit{maximal} subsequence which matches the regular expression \texttt{>\*<}.

Assume that the occurrence of the pattern PLAIN starts at position \(i\) and ends at position \(j\). The feature \texttt{MIN} computes the minimum of the values from index \(i + 1\) to index \(j\).

**Purpose**

\[
\text{Succeeds if the minima of the values in each occurrence of the PLAIN pattern in the time-series given by the VARIABLES collection are all the same.}
\]

**Example**

\[
((1, 6, 3, 3, 7, 6, 3, 3, 7, 6, 6, 3, 3, 5, 4, 3, 3, 6, 3))
\]

Figure 3.5 provides an example where the \texttt{ALL_EQUAL_HEIGHT_PLAIN} \(([1, 6, 3, 3, 7, 6, 6, 3, 3, 5, 4, 3, 3, 6, 3])\) constraint holds.

![Diagram](image-url)

**Figure 3.5:** Illustrating the \texttt{ALL_EQUAL_HEIGHT_PLAIN} constraint of the \texttt{Example} slot
Typical

| $|\text{VARIABLES}| > 2$
| $\text{range}(\text{VARIABLES}.\text{var}) > 1$ |
Automaton

Figure 3.6 depicts the automaton associated with the constraint ALL_EQUAL_HEIGHTPLAIN.

\[
\begin{align*}
C & \leftarrow \min(D, \text{VAR}_i) \\
D & \leftarrow +\infty \\
F & \leftarrow \min(D, \text{VAR}_i) \\
R & \leftarrow R \land (F = \min(D, \text{VAR}_i))
\end{align*}
\]

\[
\begin{align*}
C & \leftarrow X \\
D & \leftarrow +\infty \\
F & \leftarrow X \\
R & \leftarrow 1
\end{align*}
\]

Figure 3.6: Automaton for the ALL_EQUAL_HEIGHTPLAIN constraint obtained by applying decoration Table 2.36 to the seed transducer of the PLAIN pattern.
### 3.4 \textbf{ALL\_EQUAL\_HEIGHT\_PLATEAU}

\begin{tabular}{|l|l|}
\hline
\textbf{Origin} & Based on the \textit{PLATEAU} pattern. \\
\hline
\textbf{Constraint} & \texttt{ALL\_EQUAL\_HEIGHT\_PLATEAU} (\texttt{VARIABLES}) \\
\hline
\textbf{Argument} & \texttt{VARIABLES : collection(var\textendash dvar)} \\
\hline
\textbf{Restriction} & \texttt{required(VARIABLES.var)} \\
\hline
\textbf{Purpose} & Succeeds if the minima of the values in each occurrence of the \textit{PLATEAU} pattern in the time-series given by the \texttt{VARIABLES} collection are all the same. \texttt{required} is the \textit{maximal} subsequence which matches the regular expression \texttt{< =* >}. \\
& \textit{An occurrence of the pattern} \textit{PLATEAU} \textit{starts at position} \textit{i} \textit{and ends at position} \textit{j}. \textit{The feature} \texttt{MIN} \textit{computes the minimum of the values from index} \textit{i + 1} \textit{to index} \textit{j}. \\
\hline
\textbf{Example} & \begin{pmatrix}
(7, 2, 5, 5, 1, 2, 5, 3, 4, 5, 5, 2, 5)
\end{pmatrix} \\
\hline
\textbf{Typical} & \begin{pmatrix}
|\texttt{VARIABLES}| > 2 \\
\texttt{range} (\texttt{VARIABLES.var}) > 1
\end{pmatrix} \\
\hline
\end{tabular}

Figure \ref{fig:3.7} provides an example where the \texttt{ALL\_EQUAL\_HEIGHT\_PLATEAU} \begin{pmatrix}
([7, 2, 5, 5, 1, 2, 5, 3, 4, 5, 5, 2, 5])
\end{pmatrix} constraint holds.
Figure 3.7: Illustrating the \texttt{ALL_EQUAL_HEIGHT_PLATEAU} constraint of the \textit{Example} slot
Figure 3.8 depicts the automaton associated with the constraint `ALL_EQUAL_HEIGHT_PLATEAU`.

Figure 3.8: Automaton for the `ALL_EQUAL_HEIGHT_PLATEAU` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `PLATEAU` pattern.
3.5 ALL_EQUAL_HEIGHT_PROPER_PLAIN

Origin: Based on the PROPER_PLAIN pattern.

Constraint: ALL_EQUAL_HEIGHT_PROPER_PLAIN(VARIABLES)

Argument: VARIABLES : collection(var−dvar)

Restriction: required(VARIABLES, var)

Purpose: Succeeds if the minima of the values in each occurrence of the PROPER_PLAIN pattern in the time-series given by the VARIABLES collection are all the same.

An occurrence of the pattern PROPER_PLAIN is the maximal subsequence which matches the regular expression \( \geq \leftarrow \leq \).

Assume that the occurrence of the pattern PROPER_PLAIN starts at position \( i \) and ends at position \( j \). The feature MIN computes the minimum of the values from index \( i + 1 \) to index \( j \).

Example: \((2, 7, 3, 3, 6, 6, 3, 7, 3, 3, 5, 6, 5, 3, 3, 5)\)

Figure 3.9 provides an example where the ALL_EQUAL_HEIGHT_PROPER_PLAIN \((\{2, 7, 3, 3, 6, 6, 3, 7, 3, 3, 5, 6, 5, 3, 3, 5\})\) constraint holds.

Figure 3.9: Illustrating the ALL_EQUAL_HEIGHT_PROPER_PLAIN constraint of the Example slot
Typical

\[ |\text{VARIABLES}| > 3 \]
\[ \text{range}(	ext{VARIABLES.var}) > 1 \]
Figure 3.10 depicts the automaton associated with the constraint \texttt{ALL\_EQUAL\_HEIGHT\_PROPER\_PLAIN}.

Figure 3.10: Automaton for the \texttt{ALL\_EQUAL\_HEIGHT\_PROPER\_PLAIN} constraint obtained by applying decoration Table \texttt{2.36} to the seed transducer of the \texttt{PROPER\_PLAIN} pattern
3.6 ALL_EQUAL_HEIGHT_PROPER_PLATEAU

**DESCRIPTION**

**Origin**
Based on the **PROPER_PLATEAU** pattern.

**Constraint**
ALL_EQUAL_HEIGHT_PROPER_PLATEAU(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES, var)

Succeeds if the minima of the values in each occurrence of the **PROPER_PLATEAU** pattern in the time-series given by the VARIABLES collection are all the same.

**Purpose**
An occurrence of the pattern **PROPER_PLATEAU** is the maximal subsequence which matches the regular expression `< =+ >`.
Assume that the occurrence of the pattern **PROPER_PLATEAU** starts at position i and ends at position j. The feature **MIN** computes the minimum of the values from index i + 1 to index j.

**Example**

```
((7, 1, 5, 5, 2, 2, 5, 1, 5, 5, 3, 2, 3, 5, 5, 3))
```

Figure 3.11 provides an example where the ALL_EQUAL_HEIGHT_PROPER_PLATEAU ((7, 1, 5, 5, 2, 2, 5, 1, 5, 5, 3, 2, 3, 5, 5, 3)) constraint holds.

**Typical**

| VARIABLES | > 3
|-----------|---
| range(VARIABLES.var) | > 1
Figure 3.11: Illustrating the ALL_EQUAL_HEIGHT_PROPER_PLATEAU constraint of the Example slot
Automaton

Figure 3.12 depicts the automaton associated with the constraint \textit{ALL\_EQUAL\_HEIGHT\_PROPER\_PLATEAU}.

![Automaton Diagram]

Figure 3.12: Automaton for the \textit{ALL\_EQUAL\_HEIGHT\_PROPER\_PLATEAU} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textit{PROPER\_PLATEAU} pattern
3.7 ALL_EQUAL_HEIGHT_STEADY

**DESCRIPTION**

**Origin**
Based on the STEADY pattern.

**Constraint**

\[
\text{ALL_EQUAL_HEIGHT_STEADY} (\text{VARIABLES})
\]

**Argument**

\[
\text{VARIABLES : collection} (\text{var - dvar})
\]

**Restriction**

\[
\text{required} (\text{VARIABLES}, \text{var})
\]

**Purpose**
Succeeds if the minima of the values in each occurrence of the STEADY pattern in the time-series given by the VARIABLES collection are all the same.

An occurrence of the pattern \text{STEADY} is the subsequence which matches the regular expression \(=\).

Assume that the occurrence of the pattern \text{STEADY} starts at position \(i\) and ends at position \(j\). The feature \text{MIN} computes the minimum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
(\langle 4, 4, 2, 6, 4, 4, 4, 3, 2, 1, 4, 4, 3, 4 \rangle)
\]

Figure 3.13 provides an example where the \text{ALL_EQUAL_HEIGHT_STEADY} (\([4, 4, 2, 6, 4, 4, 4, 3, 2, 1, 4, 4, 3, 4]\)) constraint holds.

![Figure 3.13: Illustrating the ALL_EQUAL_HEIGHT_STEADY constraint of the Example slot](image)

Figure 3.13: Illustrating the ALL_EQUAL_HEIGHT_STEADY constraint of the Example slot
| Typical | $|\text{VARIABLES}| > 1$ |
Figure 3.14 depicts the automaton associated with the constraint \texttt{ALL\_EQUAL\_HEIGHT\_STEADY}.

\[
\begin{align*}
C &\leftarrow X \\
D &\leftarrow +\infty \\
F &\leftarrow X \\
R &\leftarrow 1
\end{align*}
\]

\[
\begin{align*}
C &\leftarrow \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
D &\leftarrow +\infty \\
F &\leftarrow \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
R &\leftarrow R \land (F = \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}))
\end{align*}
\]

Figure 3.14: Automaton for the \texttt{ALL\_EQUAL\_HEIGHT\_STEADY} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \texttt{STEADY} pattern
3.8 ALL_EQUAL_HEIGHT_STEADY_SEQUENCE

### Description

**Origin**
Based on the STEADY_SEQUENCE pattern.

**Constraint**
ALL_EQUAL_HEIGHT_STEADY_SEQUENCE(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES, var)

Succeeds if the minima of the values in each occurrence of the STEADY_SEQUENCE pattern in the time-series given by the VARIABLES collection are all the same.

**Purpose**
An occurrence of the pattern STEADY_SEQUENCE is the maximal subsequence which matches the regular expression \( =^+ \).
Assume that the occurrence of the pattern STEADY_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature MIN computes the minimum of the values from index \( i \) to index \( j + 1 \).

**Example**

\(((4, 3, 5, 5, 5, 3, 1, 5, 5, 6, 5, 5, 5, 3, 5))\)

Figure 3.15 provides an example where the ALL_EQUAL_HEIGHT_STEADY_SEQUENCE (\([4, 3, 5, 5, 5, 3, 1, 5, 5, 6, 5, 5, 5, 3, 5]\)) constraint holds.

![Figure 3.15: Illustrating the ALL_EQUAL_HEIGHT_STEADY_SEQUENCE constraint of the Example slot](image-url)
| Typical | $|\text{VARIABLES}| > 1$ |
Figure 3.16: Automaton for the `ALL_EQUAL_HEIGHT_STEADYSEQUENCE` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `STEADYSEQUENCE` pattern.
### 3.9 ALL_EQUAL_MAX_BUMP_ON_DECREASING_SEQUENCE

<table>
<thead>
<tr>
<th>Origin</th>
<th>Based on the BUMP_ON_DECREASING_SEQUENCE pattern.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constraint</td>
<td>\texttt{ALL_EQUAL_MAX_BUMP_ON_DECREASING_SEQUENCE(VARIABLES)}</td>
</tr>
<tr>
<td>Argument</td>
<td>VARIABLES : \texttt{collection(var–dvar)}</td>
</tr>
<tr>
<td>Restriction</td>
<td>\texttt{required(VARIABLES, var)}</td>
</tr>
<tr>
<td>Purpose</td>
<td>Succeeds if the maxima of the values in each occurrence of the BUMP_ON_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are all the same. An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence which matches the regular expression \texttt{&gt;&gt;&lt;&gt;&gt;}. Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position (i) and ends at position (j). The feature \texttt{MAX} computes the maximum of the values from index (i+2) to index (j).</td>
</tr>
<tr>
<td>Example</td>
<td>((7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 6, 5, 4, 4))</td>
</tr>
<tr>
<td>Typical</td>
<td>(</td>
</tr>
</tbody>
</table>
Figure 3.17: Illustrating the `ALL_EQUAL_MAX_BUMP_ON_DECREASING_SEQUENCE` constraint of the `Example` slot
Figure 3.18 depicts the automaton associated with the constraint \texttt{ALL\_EQUAL\_MAX\_BUMP\_ON\_DECREASING\_SEQUENCE}.

Figure 3.18: Automaton for the \texttt{ALL\_EQUAL\_MAX\_BUMP\_ON\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \texttt{BUMP\_ON\_DECREASING\_SEQUENCE} pattern.
3.10 ALL_EQUAL_MAX_DECREASING

**Description**

Origin: Based on the DECREASING pattern.

Constraint: `ALL_EQUAL_MAX_DECREASING(VARIABLES)`

Argument: `VARIABLES : collection(var–dvar)`

Restriction: `required(VARIABLES, var)`

Succeeds if the maxima of the values in each occurrence of the DECREASING pattern in the time-series given by the VARIABLES collection are all the same.

**Purpose**

An occurrence of the pattern DECREASING is the subsequence which matches the regular expression `>`. Assume that the occurrence of the pattern DECREASING starts at position `i` and ends at position `j`. The feature MAX computes the maximum of the values from index `i` to index `j + 1`.

**Example**

`((1, 6, 6, 1, 4, 6, 4, 4, 4, 5, 5, 6, 1, 5))`

Figure 3.19 provides an example where the ALL_EQUAL_MAX_DECREASING `((1, 6, 6, 1, 4, 6, 4, 4, 4, 5, 5, 6, 1, 5))` constraint holds.

![Figure 3.19: Illustrating the ALL_EQUAL_MAX_DECREASING constraint of the Example slot](image-url)
Typical

| VARIABLES | > 1
| range(VARIABLES.var) | > 1
Figure 3.20 depicts the automaton associated with the constraint `ALL_EQUAL_MAX_DECREASING`. 

\[
\begin{aligned}
    & C \leftarrow X \\
    & D \leftarrow -\infty \\
    & F \leftarrow X \\
    & R \leftarrow 1 \\
    \{ C \leftarrow \max(\max(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
    & D \leftarrow -\infty \\
    & F \leftarrow \max(\max(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
    & R \leftarrow R \land (F = \max(\max(D, \text{VAR}_i), \text{VAR}_{i+1})) \}
\end{aligned}
\]
### 3.11 **ALL_EQUAL_MAX_DECREASING_SEQUENCE**

<table>
<thead>
<tr>
<th><strong>DESCRIPTION</strong></th>
<th><strong>AUTOMATON</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the <code>DECREASING_SEQUENCE</code> pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td><code>ALL_EQUAL_MAX_DECREASING_SEQUENCE(VARIABLES)</code></td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td><code>VARIABLES : collection(var−dvar)</code></td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td><code>required(VARIABLES.var)</code></td>
</tr>
<tr>
<td><strong>Purpose</strong></td>
<td>Succeeds if the maxima of the values in each occurrence of the <code>DECREASING_SEQUENCE</code> pattern in the time-series given by the <code>VARIABLES</code> collection are all the same. An occurrence of the pattern <code>DECREASING_SEQUENCE</code> is the maximal subsequence which matches the regular expression `&gt;(&gt;</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td><code>((1, 6, 6, 1, 1, 4, 6, 6, 4, 2, 1, 3, 6, 1, 5))</code></td>
</tr>
<tr>
<td><strong>Typical</strong></td>
<td>`</td>
</tr>
</tbody>
</table>

Figure 3.21 provides an example where the `ALL_EQUAL_MAX_DECREASING_SEQUENCE` constraint holds.
Figure 3.21: Illustrating the ALL_EQUAL_MAX_DECREASING_SEQUENCE constraint of the Example slot
Figure 3.22 depicts the automaton associated with the constraint ALL_EQUAL_MAX_DECREASING_SEQUENCE.

Given a sequence constraint, the automaton transitions are defined as follows:

- **Initialization:**
  - $C \leftarrow X$
  - $D \leftarrow -\infty$
  - $F \leftarrow X$
  - $R \leftarrow 1$

- **Input Symbols:**
  - $\leq s$
  - $\geq t$

- **Transitions:**
  - $D \leftarrow -\infty$ if $F < C$
  - $R \leftarrow R \land (F = C)$
  - $C \leftarrow \max(D, \text{VAR}i, \text{VAR}i+1)$
  - $D \leftarrow -\infty$

- **Output Symbols:**
  - $\{D \leftarrow \max(D, \text{VAR}i+1)\}$

Figure 3.22: Automaton for the ALL_EQUAL_MAX_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the DECREASING_SEQUENCE pattern.
### 3.12  ALL_EQUAL_MAX_DIP_ON_INCREASING_SEQUENCE

**Description**

Based on the `DIP_ON_INCREASING_SEQUENCE` pattern.

**Constraint**

`ALL_EQUAL_MAX_DIP_ON_INCREASING_SEQUENCE(VARIABLES)`

**Argument**

`VARIABLES : collection(var−dvar)`

**Restriction**

`required(VARIABLES.var)`

Succeeds if the maxima of the values in each occurrence of the `DIP_ON_INCREASING_SEQUENCE` pattern in the time-series given by the VARIABLES collection are all the same.

**Purpose**

An occurrence of the pattern `DIP_ON_INCREASING_SEQUENCE` is the subsequence which matches the regular expression `<<><<<`. Assume that the occurrence of the pattern `DIP_ON_INCREASING_SEQUENCE` starts at position $i$ and ends at position $j$. The feature `MAX` computes the maximum of the values from index $i + 2$ to index $j$.

**Example**

```plaintext
((0, 1, 2, 1, 5, 6, 7, 3, 0, 2, 3, 5, 0, 2, 3, 3))
```

Figure 3.23 provides an example where the `ALL_EQUAL_MAX_DIP_ON_INCREASING_SEQUENCE` constraint holds.

**Typical**

- `|VARIABLES| > 5`
- `range(VARIABLES.var) > 2`
Figure 3.23: Illustrating the ALL_EQUAL_MAX_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot
Automaton  Figure 3.24 depicts the automaton associated with the constraint `ALL_EQUAL_MAX_DIP_ON_INCREASING_SEQUENCE`.

Figure 3.24: Automaton for the `ALL_EQUAL_MAX_DIP_ON_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `DIP_ON_INCREASING_SEQUENCE` pattern.
ALL_EQUAL_MAX_DIP_ON_INCREASING_SEQUENCE
3.13 **ALL_EQUAL_MAX_INCREASING**

**DESCRIPTION**

**Origin**
Based on the *INCREASING* pattern.

**Constraint**

\[ \text{ALL_EQUAL_MAX_INCREASING(VARIABLES)} \]

**Argument**

\[ \text{VARIABLES} : \text{collection(var–dvar)} \]

**Restriction**

\[ \text{required(VARIABLES, var)} \]

Succeeds if the maxima of the values in each occurrence of the *INCREASING* pattern in the time-series given by the VARIABLES collection are all the same.

**Purpose**

An occurrence of the pattern *INCREASING* is the subsequence which matches the regular expression \(<\).

Assume that the occurrence of the pattern *INCREASING* starts at position \(i\) and ends at position \(j\). The feature \(\text{MAX}\) computes the maximum of the values from index \(i\) to index \(j+1\).

**Example**

\[(6, 1, 1, 6, 6, 6, 5, 3, 3, 6, 4, 1, 6, 2)\]

Figure 3.25 provides an example where the **ALL_EQUAL_MAX_INCREASING** \([(6, 1, 1, 6, 6, 6, 5, 3, 3, 6, 4, 1, 6, 2)]\) constraint holds.

Figure 3.25: Illustrating the **ALL_EQUAL_MAX_INCREASING** constraint of the **Example** slot
<table>
<thead>
<tr>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>(</td>
</tr>
<tr>
<td>(\text{range(VARIABLES.var)} &gt; 1)</td>
</tr>
</tbody>
</table>
Figure 3.26 depicts the automaton associated with the constraint \texttt{ALL_EQUAL_MAX_INCREASING}.

\begin{align*}
C & \leftarrow X \\
D & \leftarrow -\infty \\
F & \leftarrow X \\
R & \leftarrow 1
\end{align*}

\begin{align*}
C & \leftarrow \max(\max(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
D & \leftarrow -\infty \\
F & \leftarrow \max(\max(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
R & \leftarrow R \land (F = \max(\max(D, \text{VAR}_i), \text{VAR}_{i+1}))
\end{align*}

Figure 3.26: Automaton for the \texttt{ALL_EQUAL_MAX_INCREASING} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \texttt{INCREASING} pattern.
ALL_EQUAL_MAX_INCREASING

177
3.14 ALL_EQUAL_MAX_INCREASING_SEQUENCE

DESCRIPTION

Origin
Based on the INCREASING_SEQUENCE pattern.

Constraint
ALL_EQUAL_MAX_INCREASING_SEQUENCE(VARIABLES)

Argument
VARIABLES : collection(var−dvar)

Restriction
required(VARIABLES, var)

Purpose
Succeeds if the maxima of the values in each occurrence of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are all the same. An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression < (< | =)* < | <. Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i to index j + 1.

Example
((6, 1, 1, 6, 6, 3, 1, 3, 3, 5, 6, 4, 1, 6, 2))

Figure 3.27 provides an example where the ALL_EQUAL_MAX_INCREASING_SEQUENCE ([6, 1, 1, 6, 6, 3, 1, 3, 3, 5, 6, 4, 1, 6, 2]) constraint holds.

Figure 3.27: Illustrating the ALL_EQUAL_MAX_INCREASING_SEQUENCE constraint of the Example slot
| Typical                          | $|\text{VARIABLES}| > 1$               |
|--------------------------------|-----------------------------|
|                                | $\text{range}(\text{VARIABLES}.\text{var}) > 1$ |
Figure 3.28 depicts the automaton associated with the constraint ALL_EQUAL_MAX_INCREASING_SEQUENCE.

\[ \begin{align*}
    C & \leftarrow X \\
    D & \leftarrow -\infty \\
    F & \leftarrow X \\
    R & \leftarrow 1
\end{align*} \]

\[ \begin{align*}
    \{ D \leftarrow \max(D, \text{VAR}_i + 1) \} \\
    \{ C \leftarrow \max(C, \max(D, \text{VAR}_i + 1)) \}
\end{align*} \]

Figure 3.28: Automaton for the ALL_EQUAL_MAX_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the INCREASING_SEQUENCE pattern.
3.15 **ALL_EQUAL_MAX_INFLEXION**

**Description**

Origin: Based on the **INFLEXION** pattern.

Constraint: ALL_EQUAL_MAX_INFLEXION(VARIABLES)

Argument: VARIABLES : collection(var−dvar)

Restriction: required(VARIABLES, var)

Purpose: Succeeds if the maxima of the values in each occurrence of the **INFLEXION** pattern in the time-series given by the VARIABLES collection are all the same.

An occurrence of the pattern **INFLEXION** is the maximal subsequence which matches the regular expression `< (< | =) > | > (>| =)* <`.

Assume that the occurrence of the pattern **INFLEXION** starts at position \(i\) and ends at position \(j\). The feature `MAX` computes the maximum of the values from index \(i+1\) to index \(j\).

**Example**

\[\langle 5, 5, 4, 4, 3, 2, 2, 3, 3, 4, 4, 2, 2\rangle\]

Figure 3.29 provides an example where the **ALL_EQUAL_MAX_INFLEXION** \([5, 5, 4, 4, 3, 2, 2, 3, 3, 4, 4, 2, 2]\) constraint holds.

![Figure 3.29: Illustrating the ALL_EQUAL_MAX_INFLEXION constraint of the Example slot](image)

**Typical**

\[|\text{VARIABLES}| > 2\]

\[\text{range}(\text{VARIABLES.var}) > 1\]
Figure 3.30 depicts the automaton associated with the constraint `ALL_EQUAL_MAX_INFLEXION`.

```plaintext
\begin{align*}
C & \leftarrow X \\
D & \leftarrow -\infty \\
F & \leftarrow X \\
R & \leftarrow 1 \\
\{ D \leftarrow \max(D, \text{VAR}_i) \} & \quad \{ D \leftarrow \max(D, \text{VAR}_i) \} \\
\{ C \leftarrow \max(D, \text{VAR}_i) \} & \quad \{ C \leftarrow \max(D, \text{VAR}_i) \} \\
\{ F \leftarrow \max(D, \text{VAR}_i) \} & \quad \{ F \leftarrow \max(D, \text{VAR}_i) \} \\
R & \leftarrow R \land (F = \max(D, \text{VAR}_i)) \\
\end{align*}
```

Figure 3.30: Automaton for the `ALL_EQUAL_MAX_INFLEXION` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `INFLEXION` pattern (transition $r \rightarrow t$ has the same accumulators updates as transition $t \rightarrow r$)
### 3.16 \texttt{ALL\_EQUAL\_MAX\_PEAK}

**Description**

Based on the \texttt{PEAK} pattern.

**Constraint**

\texttt{ALL\_EQUAL\_MAX\_PEAK}([VARIABLES])

**Argument**

\texttt{VARIABLES : collection(var–dvar)}

**Restriction**

\texttt{required(VARIABLES, var)}

Succeeds if the maxima of the values in each occurrence of the \texttt{PEAK} pattern in the time-series given by the \texttt{VARIABLES} collection are all the same.

**Purpose**

An occurrence of the pattern \texttt{PEAK} is the \textit{maximal} subsequence which matches the regular expression \texttt{< (= | <)* (> | =)* >}.

Assume that the occurrence of the pattern \texttt{PEAK} starts at position \textit{i} and ends at position \textit{j}. The feature \texttt{MAX} computes the maximum of the values from index \textit{i} + 1 to index \textit{j}.

**Example**

\[
(1, 2, 2, 7, 7, 1, 2, 4, 5, 7, 1, 2, 3, 6, 7, 1)
\]

Figure 3.31 provides an example where the \texttt{ALL\_EQUAL\_MAX\_PEAK} \((1, 2, 2, 7, 7, 1, 2, 4, 5, 7, 1, 2, 3, 6, 7, 1)) constraint holds.

![Figure 3.31: Illustrating the \texttt{ALL\_EQUAL\_MAX\_PEAK} constraint of the \texttt{Example} slot](image)
Typical

<table>
<thead>
<tr>
<th>VARIABLES</th>
<th>&gt; 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>range(VARIABLES.var)</td>
<td>&gt; 1</td>
</tr>
</tbody>
</table>
Figure 3.32 depicts the automaton associated with the constraint ALL_EQUAL_MAX_PEAK.

\[
\begin{align*}
C & \leftarrow X \\
D & \leftarrow -\infty \\
F & \leftarrow X \\
R & \leftarrow 1
\end{align*}
\geq s
\]

\[
R \land (F = C)
\]

\[
\begin{align*}
C & \leftarrow \max(D, \forall k_i) \\
D & \leftarrow -\infty
\end{align*}
\leq \{ D \leftarrow \max(D, \forall k_i) \}
\]

\[
\begin{align*}
D & \leftarrow -\infty \\
F & \leftarrow C \\
R & \leftarrow R \land (F = C)
\end{align*}
\leq \{ D \leftarrow \max(D, \forall k_i) \}
\]

\[
\begin{align*}
C & \leftarrow \max(C, \max(D, \forall k_i)) \\
D & \leftarrow -\infty
\end{align*}
\leq \{ D \leftarrow \max(D, \forall k_i) \}
\]

\[
\begin{align*}
D & \leftarrow -\infty \\
F & \leftarrow C \\
R & \leftarrow R \land (F = C)
\end{align*}
\leq \{ D \leftarrow \max(D, \forall k_i) \}
\]

\[
\begin{align*}
C & \leftarrow \max(D, \forall k_i) \\
D & \leftarrow -\infty \\
F & \leftarrow X \\
R & \leftarrow 1
\end{align*}
\geq \{ D \leftarrow \max(D, \forall k_i) \}
\]

\[
R \land (F = C)
\]

Figure 3.32: Automaton for the ALL_EQUAL_MAX_PEAK constraint obtained by applying decoration Table 2.36 to the seed transducer of the PEAK pattern.
### 3.17 ALL_EQUAL_MAX STRICTLY DECREASING SEQUENCE

**Description**

Based on the `STRICTLY DECREASING_SEQUENCE` pattern.

**Argument**

`VARIABLES : collection(var−dvar)`

**Restriction**

`required(VARIABLES, var)`

**Purpose**

Succeeds if the maxima of the values in each occurrence of the `STRICTLY DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are all the same.

An occurrence of the pattern `STRICTLY DECREASING_SEQUENCE` is the *maximal* subsequence which matches the regular expression `>`.

Assume that the occurrence of the pattern `STRICTLY DECREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `MAX` computes the maximum of the values from index `i` to index `j + 1`.

**Example**

```
((5, 5, 6, 3, 1, 2, 4, 4, 6, 6, 4, 1, 6, 1))
```

Figure 3.33 provides an example where the `ALL_EQUAL_MAX STRICTLY DECREASING SEQUENCE ([5, 5, 6, 3, 1, 2, 4, 4, 6, 6, 4, 1, 6, 1])` constraint holds.

**Typical**

```
|VARIABLES| > 1
range(VARIABLES.var) > 1
```
Figure 3.33: Illustrating the ALL_EQUAL_MAX_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Figure 3.34 depicts the automaton associated with the constraint `ALL_EQUAL_MAX.StrictlyDecreasing_SEQUENCE`.

```latex
\begin{align*}
C &\leftarrow X \\
D &\leftarrow -\infty \\
F &\leftarrow X \\
R &\leftarrow 1 \\
\end{align*}
\begin{align*}
\{ D &\leftarrow -\infty \\
F &\leftarrow C \\
R &\leftarrow R \land (F = C) \}
\end{align*}
\begin{align*}
C &\leftarrow \max(\max(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
D &\leftarrow -\infty \\
\end{align*}
\begin{align*}
\{ D &\leftarrow -\infty \\
C &\leftarrow \max(C, \max(D, \text{VAR}_{i+1})) \}
\end{align*}
```

Figure 3.34: Automaton for the `ALL_EQUAL_MAX.StrictlyDecreasing_SEQUENCE` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `StrictlyDecreasing.Sequence` pattern.
ALL_EQUAL_MAX_STRICTLY_DECREASING_SEQUENCE
3.18 **ALL_EQUAL_MAX.StrictlyIncreasing_SEQUENCE**

**DESCRIPTION**

**Origin**
Based on the **StrictlyIncreasing.Sequence** pattern.

**Constraint**
ALL_EQUAL_MAX.StrictlyIncreasing.Sequence(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES.var)

**Purpose**
Succeeds if the maxima of the values in each occurrence of the **StrictlyIncreasing.Sequence** pattern in the time-series given by the VARIABLES collection are all the same.

An occurrence of the pattern **StrictlyIncreasing.Sequence** is the *maximal* sub-sequence which matches the regular expression `<+`. Assume that the occurrence of the pattern **StrictlyIncreasing.Sequence** starts at position i and ends at position j. The feature **MAX** computes the maximum of the values from index i to index j + 1.

**Example**

`((6, 1, 1, 6, 6, 3, 1, 1, 3, 4, 5, 6, 4, 1, 6, 2))`

Figure 3.35 provides an example where the **ALL_EQUAL_MAX.StrictlyIncreasing.Sequence** (`[6, 1, 1, 6, 6, 3, 1, 1, 3, 4, 5, 6, 4, 1, 6, 2]`) constraint holds.

**Typical**

`|VARIABLES| > 1`
`range(VARIABLES.var) > 1`
Figure 3.35: Illustrating the ALL_EQUAL_MAX_STRICTLY_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 3.36 depicts the automaton associated with the constraint `ALL_EQUAL_MAX STRICTLY_INCREASING_SEQUENCE`.

\[
\begin{align*}
C & \leftarrow X \\
D & \leftarrow -\infty \\
F & \leftarrow X \\
R & \leftarrow 1
\end{align*}
\]

\[
\begin{align*}
\{ & D \leftarrow -\infty \\
& F \leftarrow C \\
& R \leftarrow R \land (F = C)
\}
\]

\[
\begin{align*}
\{ & C \leftarrow \max(D, \text{VAR}_i, \text{VAR}_{i+1}) \\
& D \leftarrow -\infty
\}
\]

\[
\begin{align*}
\{ & C \leftarrow \max(C, \max(D, \text{VAR}_{i+1})) \\
& D \leftarrow -\infty
\}
\]

Figure 3.36: Automaton for the `ALL_EQUAL_MAX STRICTLY_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `STRICTLY_INCREASING_SEQUENCE` pattern.
ALL_EQUAL_MAX_STRICTLY_INCREASING_SEQUENCE
3.19 **ALL_EQUAL_MAX_SUMMIT**

**DESCRIPTION**

- **Origin**: Based on the SUMMIT pattern.
- **Constraint**: 
  \[ \text{ALL\_EQUAL\_MAX\_SUMMIT}(\text{VARIABLES}) \]
- **Argument**: \[ \text{VARIABLES} : \text{collection(\text{var\_dvar})} \]
- **Restriction**: \[ \text{required} (\text{VARIABLES, var}) \]

**Purpose**

Succeeds if the maxima of the values in each occurrence of the SUMMIT pattern in the time-series given by the VARIABLES collection are all the same.

An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression \(< | < (= | <) > | > (= | >) > \).

Assume that the occurrence of the pattern SUMMIT starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i + 1\) to index \(j\).

**Example**

\[(1, 3, 5, 4, 2, 6, 6, 2, 3, 5, 4, 1, 4, 5, 3, 2)\]

Figure 3.37 provides an example where the ALL_EQUAL_MAX_SUMMIT \([(1, 3, 5, 4, 2, 6, 6, 2, 3, 5, 4, 1, 4, 5, 3, 2)]\) constraint holds.

![Figure 3.37](image-url)

Figure 3.37: Illustrating the ALL_EQUAL_MAX_SUMMIT constraint of the **Example** slot
<table>
<thead>
<tr>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
</tr>
<tr>
<td>$\text{range}(\text{VARIABLES}.\text{var}) &gt; 1$</td>
</tr>
</tbody>
</table>
Automaton

Figure 3.38 depicts the automaton associated with the constraint \texttt{ALL\_EQUAL\_MAX\_SUMMIT}.

Figure 3.38: Automaton for the \texttt{ALL\_EQUAL\_MAX\_SUMMIT} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \texttt{SUMMIT} pattern (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$)
3.20 ALL_EQUAL_MAX_ZIGZAG

DESCRIPTION

Origin
Based on the ZIGZAG pattern.

Constraint
ALL_EQUAL_MAX_ZIGZAG(VARIABLES)

Argument
VARIABLES : collection(var–dvar)

Restriction
required(VARIABLES, var)

Purpose
Succeeds if the maxima of the values in each occurrence of the ZIGZAG pattern in the
time-series given by the VARIABLES collection are all the same.

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the
regular expression $(<>)^+ (<> | <>) | (<>)^+ (> | >>)$. Assume that the occurrence of the pattern ZIGZAG starts at position $i$ and ends at position $j$. The feature MAX computes the maximum of the values from index $i + 1$ to index $j$.

Example
$((1, 6, 1, 2, 7, 5, 1, 6, 3, 4, 2, 5, 6, 3, 4, 7))$

Figure 3.39 provides an example where the ALL_EQUAL_MAX_ZIGZAG $((1, 6, 1, 2, 7, 5, 1, 6, 3, 4, 2, 5, 6, 3, 4, 7))$ constraint holds.

Figure 3.39: Illustrating the ALL_EQUAL_MAX_ZIGZAG constraint of the Example slot
Typical

\[ |\text{VARIABLES}| > 3 \]
\[ \text{range(}\text{VARIABLES.var}) > 1 \]
Automaton Figure 3.40 depicts the automaton associated with the constraint ALL_EQUAL_MAX_ZIGZAG.
Figure 3.40: Automaton for the \texttt{ALL\_EQUAL\_MAX\_ZIGZAG} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \texttt{ZIGZAG} pattern: (1) missing transitions from \texttt{a, b, c, d, e, f} to \texttt{s} are labelled by \texttt{=} ; (2) on transitions from \texttt{b, c, e, f} to \texttt{s} the accumulator \texttt{D} is reset to its initial value; (3) on transitions from \texttt{c, f} to \texttt{s} the accumulator \texttt{F} is reset to \texttt{C}, and the accumulator \texttt{R} is updated wrt \texttt{C} and \texttt{F}
3.21 ALL_EQUAL_MIN_BUMP_ON_DECREASING_SEQUENCE

**Description**

Based on the BUMP_ON_DECREASING_SEQUENCE pattern.

**Constraint**

\[ \text{ALL_EQUAL_MIN_BUMP_ON_DECREASING_SEQUENCE}(\text{VARIABLES}) \]

**Argument**

\[ \text{VARIABLES} : \text{collection}(\text{var}\rightarrow\text{dvar}) \]

**Restriction**

\[ \text{required}(\text{VARIABLES}\text{.var}) \]

Succeeds if the minima of the values in each occurrence of the BUMP_ON_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are all the same.

**Purpose**

An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence which matches the regular expression \[>>><>\].

Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature \text{MIN} computes the minimum of the values from index \(i + 2\) to index \(j\).

**Example**

\[ ((7, 6, 5, 6, 2, 1, 1, 5, 4, 3, 2, 6, 5, 4, 4, 5)) \]

Figure 3.41 provides an example where the ALL_EQUAL_MIN_BUMP_ON_DECREASING_SEQUENCE \[([7, 6, 5, 6, 2, 1, 1, 5, 4, 3, 2, 6, 5, 4, 4, 5])\] constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 5 \]
\[ \text{range}(\text{VARIABLES}\.\text{var}) > 2 \]
Figure 3.41: Illustrating the **ALL_EQUAL_MIN_BUMP_ON_DECREASING_SEQUENCE** constraint of the Example slot.
Automaton

Figure 3.42 depicts the automaton associated with the constraint ALL_EQUAL_MIN_BUMP_ON_DECREASING_SEQUENCE.

Figure 3.42: Automaton for the ALL_EQUAL_MIN_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern
3.22 ALL_EQUAL_MIN_DECREASING

**DESCRIPTION**

**Origin**
Based on the DECREASING pattern.

**Constraint**
ALL_EQUAL_MIN_DECREASING(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES, var)

Succeeds if the minima of the values in each occurrence of the DECREASING pattern in the time-series given by the VARIABLES collection are all the same.

**Purpose**
An occurrence of the pattern DECREASING is the subsequence which matches the regular expression >.
Assume that the occurrence of the pattern DECREASING starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i to index j + 1.

**Example**

\[(1, 6, 6, 1, 1, 4, 4, 5, 5, 5, 1, 3, 6, 1, 5)\]

Figure 3.43 provides an example where the ALL_EQUAL_MIN_DECREASING ([1, 6, 6, 1, 1, 4, 4, 5, 5, 5, 1, 3, 6, 1, 5]) constraint holds.

![Graph illustrating the ALL_EQUAL_MIN_DECREASING constraint](image)

Figure 3.43: Illustrating the ALL_EQUAL_MIN_DECREASING constraint of the Example slot
Typical

| VARIABLES | > 1
| range(VARIABLES.var) | > 1 |
Figure 3.44 depicts the automaton associated with the constraint ALL_EQUAL_MIN_DECREASING.

![Automaton Diagram]

Figure 3.44: Automaton for the ALL_EQUAL_MIN_DECREASING constraint obtained by applying decoration Table 2.36 to the seed transducer of the DECREASING pattern.
3.23 **ALL_EQUAL_MIN_DECREASING_SEQUENCE**

**DESCRIPTION**

**Origin**
Based on the **DECREASING_SEQUENCE** pattern.

**Constraint**

**ALL_EQUAL_MIN_DECREASING_SEQUENCE**(VARIABLES)

**Argument**

VARIABLES : collection(var−dvar)

**Restriction**

required(VARIABLES, var)

**Purpose**
Succeeds if the minima of the values in each occurrence of the **DECREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection are all the same.

An occurrence of the pattern **DECREASING_SEQUENCE** is the *maximal* subsequence which matches the regular expression \(>(>\mid\leq)^*\mid >\).

Assume that the occurrence of the pattern **DECREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **MIN** computes the minimum of the values from index \(i\) to index \(j+1\).

**Example**

\(\langle 1, 6, 6, 1, 4, 5, 5, 4, 2, 1, 3, 4, 1, 5\rangle\)

Figure 3.45 provides an example where the **ALL_EQUAL_MIN_DECREASING_SEQUENCE** (\([1, 6, 6, 1, 4, 5, 5, 4, 2, 1, 3, 4, 1, 5]\)) constraint holds.

![Diagram](image)

**Figure 3.45**: Illustrating the **ALL_EQUAL_MIN_DECREASING_SEQUENCE** constraint of the **Example** slot
Typical

|\text{|VARIABLES|} > 1
\text{range}(\text{VARIABLES.var}) > 1
Figure 3.46 depicts the automaton associated with the constraint \textsc{ALL\_EQUAL\_MIN\_DECREASING\_SEQUENCE}.

\[
\begin{align*}
C &\leftarrow X \\
D &\leftarrow +\infty \\
F &\leftarrow X \\
R &\leftarrow 1
\end{align*}
\]

Figure 3.46: Automaton for the \textsc{ALL\_EQUAL\_MIN\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textsc{DECREASING\_SEQUENCE} pattern
### 3.24 ALL_EQUAL_MIN_DIP_ON_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the DIP_ON_INCREASING_SEQUENCE pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>ALL_EQUAL_MIN_DIP_ON_INCREASING_SEQUENCE(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var–dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES.var)</td>
</tr>
</tbody>
</table>

**Purpose**

Succeeds if the minima of the values in each occurrence of the DIP_ON_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are all the same.

An occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence which matches the regular expression <<><<. Assume that the occurrence of the pattern DIP_ON_INCREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature MIN computes the minimum of the values from index \( i + 2 \) to index \( j \).

**Example**

\[ ((1, 2, 3, 2, 4, 7, 7, 3, 4, 5, 6, 2, 3, 4, 3)) \]

Figure 3.47 provides an example where the ALL_EQUAL_MIN_DIP_ON_INCREASING_SEQUENCE constraint holds.

**Typical**

\[
\text{|VARIABLES| > 5} \\
\text{range(VARIABLES.var) > 2}
\]
Figure 3.47: Illustrating the ALL_EQUAL_MIN_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot
Figure 3.48 depicts the automaton associated with the constraint ALL_EQUAL_MIN_DIP_ON_INCREASING_SEQUENCE.

Figure 3.48: Automaton for the ALL_EQUAL_MIN_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the DIP_ON_INCREASINGSEQUENCE pattern.
3.25 **ALL_EQUAL_MIN_GORGE**

**DESCRIPTION**

Origin: Based on the GORGE pattern.

Constraint: \( \text{ALL_EQUAL_MIN_GORGE(VARIABLES)} \)

Argument: \( \text{VARIABLES : collection(var\_dvar)} \)

Restriction: \( \text{required(VARIABLES, var)} \)

Purpose: Succeeds if the minima of the values in each occurrence of the GORGE pattern in the time-series given by the VARIABLES collection are all the same.

An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression \((>\mid>\mid>^*)<(\mid<\mid<^*)\). Assume that the occurrence of the pattern GORGE starts at position \(i\) and ends at position \(j\). The feature \(\text{MIN}\) computes the minimum of the values from index \(i+1\) to index \(j\).

**Example**

\( ((6, 4, 2, 3, 5, 1, 1, 5, 4, 2, 3, 6, 3, 2, 4, 5)) \)

Figure 3.49 provides an example where the \(\text{ALL_EQUAL_MIN_GORGE}\) \(([6, 4, 2, 3, 5, 1, 1, 5, 4, 2, 3, 6, 3, 2, 4, 5])\) constraint holds.

![Figure 3.49: Illustrating the ALL_EQUAL_MIN_GORGE constraint of the Example slot](image)

**Typical**

\( |\text{VARIABLES}| > 2 \)

\( \text{range(VARIABLES, var)} > 1 \)
Figure 3.50 depicts the automaton associated with the constraint ALL_EQUAL_MIN_GORGE.

Figure 3.50: Automaton for the ALL_EQUAL_MIN_GORGE constraint obtained by applying decoration Table 2.36 to the seed transducer of the GORGE pattern (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$)
### 3.26 ALL_EQUAL_MIN_INCREASING

#### Description

**Origin**
Based on the INCREASING pattern.

**Constraint**

```
ALL_EQUAL_MIN_INCREASING(VARIABLES)
```

**Argument**

```
VARIABLES : collection(var–dvar)
```

**Restriction**

```
required(VARIABLES, var)
```

Succeeds if the minima of the values in each occurrence of the INCREASING pattern in the time-series given by the VARIABLES collection are all the same.

**Purpose**

An occurrence of the pattern INCREASING is the subsequence which matches the regular expression `<.`

Assume that the occurrence of the pattern INCREASING starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i$ to index $j + 1$.

**Example**

```
((7, 2, 2, 6, 6, 3, 2, 2, 7, 4, 4, 3, 3, 2, 5, 1))
```

Figure 3.51 provides an example where the ALL_EQUAL_MIN_INCREASING ((7, 2, 2, 6, 6, 3, 2, 2, 7, 4, 4, 3, 3, 2, 5, 1)) constraint holds.

**Typical**

```
|VARIABLES| > 1
range(VARIABLES.var) > 1
```
ALL_EQUAL_MIN_INCREASING

Figure 3.51: Illustrating the ALL_EQUAL_MIN_INCREASING constraint of the Example slot
Automaton

Figure 3.52 depicts the automaton associated with the constraint \( \text{ALL\_EQUAL\_MIN\_INCREASING} \).

\[
\begin{cases}
C &\leftarrow X \\
D &\leftarrow +\infty \\
F &\leftarrow X \\
R &\leftarrow 1
\end{cases}
\]

\[
\begin{cases}
C &\leftarrow \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
D &\leftarrow +\infty \\
F &\leftarrow \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
R &\leftarrow R \land (F = \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}))
\end{cases}
\]

Figure 3.52: Automaton for the \( \text{ALL\_EQUAL\_MIN\_INCREASING} \) constraint obtained by applying decoration Table 2.36 to the seed transducer of the \( \text{INCREASING} \) pattern.
ALL_EQUAL_MIN_INCREASING
### 3.27 ALL_EQUAL_MIN_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th><strong>DESCRIPTION</strong></th>
<th><strong>AUTOMATON</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the INCREASING_SEQUENCE pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>ALL_EQUAL_MIN_INCREASING_SEQUENCE(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var−dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES, var)</td>
</tr>
<tr>
<td><strong>Purpose</strong></td>
<td>Succeeds if the minima of the values in each occurrence of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are all the same. An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression &lt; (&lt;</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>( ((7, 2, 6, 6, 3, 2, 3, 5, 6, 4, 2, 5, 1)) )</td>
</tr>
<tr>
<td><strong>Typical</strong></td>
<td>(</td>
</tr>
</tbody>
</table>

Figure 3.53 provides an example where the ALL_EQUAL_MIN_INCREASING_SEQUENCE \((7, 2, 2, 6, 6, 3, 2, 3, 5, 6, 4, 2, 5, 1)\) constraint holds.
Figure 3.53: Illustrating the ALL_EQUAL_MIN_INCREASING_SEQUENCE constraint of the Example slot
Figure 3.54 depicts the automaton associated with the constraint \textit{ALL_EQUAL_MIN_INCREASING_SEQUENCE}.

\[
\begin{align*}
&C \leftarrow X \\
&D \leftarrow +\infty \\
&P \leftarrow X \\
&R \leftarrow 1
\end{align*}
\]

\[
\begin{align*}
&\{D \leftarrow +\infty \} \\
&\{F \leftarrow C \} \\
&\{R \leftarrow R \land (F = C) \}
\end{align*}
\]

\[
\begin{align*}
&\{D \leftarrow \min(D, \text{VAR}_{i+1}) \} \\
&\{C \leftarrow \min(C, \min(D, \text{VAR}_{i+1})) \}
\end{align*}
\]

\[
\begin{align*}
&\{C \leftarrow C \min(D, \text{VAR}_{i+1}) \} \\
&\{D \leftarrow +\infty \}
\end{align*}
\]

\[
\begin{align*}
&\{D \leftarrow \min(D, \text{VAR}_{i+1}) \} \\
&\{C \leftarrow \min(C, \min(D, \text{VAR}_{i+1})) \}
\end{align*}
\]

Figure 3.54: Automaton for the \textit{ALL_EQUAL_MIN_INCREASING_SEQUENCE} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textit{INCREASING_SEQUENCE} pattern.
3.28 **ALL_EQUAL_MIN_INFLEXION**

**DESCRIPTION AUTOMATON**

<table>
<thead>
<tr>
<th>Origin</th>
<th>Based on the <strong>INFLEXION</strong> pattern.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constraint</td>
<td><strong>ALL_EQUAL_MIN_INFLEXION</strong>(VARIABLES)</td>
</tr>
<tr>
<td>Argument</td>
<td>VARIABLES : collection(var−dvar)</td>
</tr>
<tr>
<td>Restriction</td>
<td>required(VARIABLES, var)</td>
</tr>
</tbody>
</table>

**Purpose**

Succeeds if the minima of the values in each occurrence of the **INFLEXION** pattern in the time-series given by the VARIABLES collection are all the same.

An occurrence of the pattern **INFLEXION** is the **maximal** subsequence which matches the regular expression \(< (< | =)^* > | > (> | =)^* <\).

Assume that the occurrence of the pattern **INFLEXION** starts at position \(i\) and ends at position \(j\). The feature **MIN** computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**

\(\langle 2, 2, 6, 6, 5, 4, 3, 4, 4, 5 \rangle\)

Figure 3.55 provides an example where the **ALL_EQUAL_MIN_INFLEXION** \(\langle 2, 2, 3, 6, 6, 5, 4, 3, 4, 5 \rangle\) constraint holds.

![Diagram showing the ALL_EQUAL_MIN_INFLEXION constraint](image)

**Example**

Figure 3.55: Illustrating the **ALL_EQUAL_MIN_INFLEXION** constraint of the Example slot
<table>
<thead>
<tr>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
</tr>
<tr>
<td>$\text{range}(\text{VARIABLES}.\text{var}) &gt; 1$</td>
</tr>
</tbody>
</table>
Figure 3.56 depicts the automaton associated with the constraint ALL_EQUAL_MIN_INFLEXION.

\[
\begin{align*}
C &\leftarrow X \\
D &\leftarrow +\infty \\
F &\leftarrow X \\
R &\leftarrow 1 \\
\{ D &\leftarrow \min(D, \text{VAR}_i) \} \\
\end{align*}
\]

\[
\begin{align*}
R \land (F = C) \\
\{ C &\leftarrow \min(D, \text{VAR}_i) \} \\
D &\leftarrow +\infty \\
F &\leftarrow \min(D, \text{VAR}_i) \\
R &\leftarrow R \land (F = \min(D, \text{VAR}_i)) \\
\} \\
\end{align*}
\]

Figure 3.56: Automaton for the ALL_EQUAL_MIN_INFLEXION constraint obtained by applying decoration Table 2.36 to the seed transducer of the INFLEXION pattern (transition \( r \rightarrow t \) has the same accumulators updates as transition \( t \rightarrow r \))
### 3.29 ALL_EQUAL_MIN_STRICTLY_DECREASING_SEQUENCE

**Description**

Based on the `STRICTLY_DECREASING_SEQUENCE` pattern.

**Constraint**

```
ALL_EQUAL_MIN_STRICTLY_DECREASING_SEQUENCE(VARIABLES)
```

**Argument**

```
VARIABLES : collection(var−dvar)
```

**Restriction**

```
required(VARIABLES.var)
```

**Purpose**

Succeeds if the minima of the values in each occurrence of the `STRICTLY_DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are all the same.

An occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` is the *maximal* sub-sequence which matches the regular expression `>^`.

Assume that the occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` starts at position $i$ and ends at position $j$. The feature `MIN` computes the minimum of the values from index $i$ to index $j + 1$.

**Example**

```
((5, 1, 4, 3, 1, 2, 4, 4, 5, 5, 4, 1, 1, 6, 6, 1))
```

Figure 3.57 provides an example where the `ALL_EQUAL_MIN_STRICTLY_DECREASING_SEQUENCE` \([5, 1, 4, 3, 1, 2, 4, 4, 5, 5, 4, 1, 1, 6, 6, 1]) constraint holds.

**Typical**

```
|VARIABLES| > 1
range(VARIABLES.var) > 1
```
Figure 3.57: Illustrating the `ALL_EQUAL_MIN_STRICTLY_DECREASING_SEQUENCE` constraint of the **Example** slot
Figure 3.58 depicts the automaton associated with the constraint `ALL_EQUAL_MIN_STRICTLY_DECREASING_SEQUENCE`.

\[
\begin{array}{l}
C \leftarrow X \\
D \leftarrow +\infty \\
F \leftarrow X \\
R \leftarrow 1
\end{array}
\]

\[
\begin{array}{l}
\leq s \\
\leq \{ C \leftarrow \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}) \}
\end{array}
\]

\[
\begin{array}{l}
D \leftarrow +\infty \\
F \leftarrow C \\
R \leftarrow R \land (F = C)
\end{array}
\]

\[
\begin{array}{l}
\geq r \\
\geq \{ C \leftarrow \min(C, \min(D, \text{VAR}_{i+1})) \}
\end{array}
\]

\[
\begin{array}{l}
D \leftarrow +\infty \\
F \leftarrow C \\
R \leftarrow R \land (F = C)
\end{array}
\]

Figure 3.58: Automaton for the `ALL_EQUAL_MIN_STRICTLY_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `STRICTLY_DECREASING_SEQUENCE` pattern.
### 3.30 ALL_EQUAL_MIN_STRICTLY_INCREASING_SEQUENCE

**Description**

Based on the **STRICTLY_INCREASING_SEQUENCE** pattern.

**Constraint**

ALL_EQUAL_MIN_STRICTLY_INCREASING_SEQUENCE(VARIABLES)

**Argument**

VARIABLES : collection(var−dvar)

**Restriction**

required(VARIABLES.var)

**Purpose**

Succeeds if the minima of the values in each occurrence of the **STRICTLY_INCREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection are all the same.

An occurrence of the pattern **STRICTLY_INCREASING_SEQUENCE** is the maximal sub-sequence which matches the regular expression <+

Assume that the occurrence of the pattern **STRICTLY_INCREASING_SEQUENCE** starts at position i and ends at position j. The feature **MIN** computes the minimum of the values from index i to index j + 1.

**Example**

\((7, 2, 2, 6, 6, 3, 2, 2, 3, 4, 5, 6, 4, 2, 5, 1)\)

Figure 3.59 provides an example where the **ALL_EQUAL_MIN_STRICTLY_INCREASING_SEQUENCE** constraint holds.

**Typical**

\(|VARIABLES| > 1\)

\(range(VARIABLES.var) > 1\)
Figure 3.59: Illustrating the \texttt{ALL\_EQUAL\_MIN\_STRICTLY\_INCREASING\_SEQUENCE} constraint of the \texttt{Example} slot
Figure 3.60 depicts the automaton associated with the constraint \texttt{ALL_EQUAL_MIN_STRICTLY_INCREASING_SEQUENCE}.

\begin{equation*}
\begin{cases}
C \leftarrow X \\
D \leftarrow +\infty \\
F \leftarrow X \\
R \leftarrow 1
\end{cases}
\end{equation*}

Figure 3.60: Automaton for the \texttt{ALL_EQUAL_MIN_STRICTLY_INCREASING_SEQUENCE} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \texttt{STRICTLY_INCREASING_SEQUENCE} pattern.
3.31 **ALL_EQUAL_MIN_VALLEY**

**DESCRIPTION**

**Origin**

Based on the VALLEY pattern.

**Constraint**

\[ \text{ALL_EQUAL_MIN_VALLEY}(\text{VARIABLES}) \]

**Argument**

\[ \text{VARIABLES} : \text{collection}(\text{var} \rightarrow \text{dvar}) \]

**Restriction**

\[ \text{required}(\text{VARIABLES}, \text{var}) \]

Succeeds if the minima of the values in each occurrence of the VALLEY pattern in the time-series given by the VARIABLES collection are all the same.

**Purpose**

An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression \( > (= | >)^* (< | =)^* < \). Assume that the occurrence of the pattern VALLEY starts at position \( i \) and ends at position \( j \). The feature \( \text{MIN} \) computes the minimum of the values from index \( i + 1 \) to index \( j \).

**Example**

\[ (7, 6, 6, 1, 1, 7, 6, 4, 3, 1, 7, 6, 5, 2, 1, 7) \]

Figure 3.61 provides an example where the ALL_EQUAL_MIN_VALLEY \((7, 6, 6, 1, 1, 7, 6, 4, 3, 1, 7, 6, 5, 2, 1, 7)\) constraint holds.

![Diagram of valley values](image)

Figure 3.61: Illustrating the ALL_EQUAL_MIN_VALLEY constraint of the Example slot
| Typical | \[|\text{VARIABLES}| > 2\]  
|         | \[\text{range}(\text{VARIABLES}.\text{var}) > 1\] |
Automaton  Figure 3.62 depicts the automaton associated with the constraint \texttt{ALL\_EQUAL\_MIN\_VALLEY}.

\[
\begin{align*}
C & \leftarrow X \\
D & \leftarrow +\infty \\
F & \leftarrow X \\
R & \leftarrow 1 \\
\end{align*}
\leq s \\
\geq r \\
R \land (F = C)
\leq t
\]

\[
\begin{align*}
D & \leftarrow \min(D, \text{VAR}_i) \\
\end{align*}
\geq \{D \leftarrow \min(D, \text{VAR}_i)\}
\]

\[
\begin{align*}
C & \leftarrow \min(C, \min(D, \text{VAR}_i)) \\
D & \leftarrow +\infty \\
\end{align*}
< \{C \leftarrow \min(C, \min(D, \text{VAR}_i))\}
\]

\[
\begin{align*}
D & \leftarrow +\infty \\
F & \leftarrow C \\
R & \leftarrow R \land (F = C) \\
\end{align*}
\geq \{D \leftarrow \min(D, \text{VAR}_i)\}
\]

\[
\begin{align*}
D & \leftarrow \min(D, \text{VAR}_i) \\
\end{align*}
\leq \{D \leftarrow \min(D, \text{VAR}_i)\}
\]

\[
\begin{align*}
D & \leftarrow +\infty \\
F & \leftarrow C \\
R & \leftarrow R \land (F = C) \\
\end{align*}
\leq \{D \leftarrow \min(D, \text{VAR}_i)\}
\]

Figure 3.62: Automaton for the \texttt{ALL\_EQUAL\_MIN\_VALLEY} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \texttt{VALLEY} pattern.
3.32 ALL_EQUAL_MIN_ZIGZAG

DESCRIPTION AUTOMATON

Origin: Based on the ZIGZAG pattern.

Constraint: ALL_EQUAL_MIN_ZIGZAG(VARIABLES)

Argument: VARIABLES : collection(var–dvar)

Restriction: required(VARIABLES, var)

Purpose: Succeeds if the minima of the values in each occurrence of the ZIGZAG pattern in the time-series given by the VARIABLES collection are all the same.

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression \((<>)^+ (<> | <> ) | (<>)^+ (> | ><> )\).

Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\).

Example: \(((1, 3, 1, 2, 6, 5, 1, 4, 3, 4, 3, 5, 6, 1, 4, 7))\)

Figure 3.63 provides an example where the ALL_EQUAL_MIN_ZIGZAG \(((1, 3, 1, 2, 6, 5, 1, 4, 3, 4, 3, 5, 6, 1, 4, 7))\) constraint holds.

Figure 3.63: Illustrating the ALL_EQUAL_MIN_ZIGZAG constraint of the Example slot
Typical

| VARIABLES | > 3

range(VARIABLES.var) > 1
Automaton

Figure 3.64 depicts the automaton associated with the constraint ALL_EQUAL_MIN_ZIGZAG.
Figure 3.64: Automaton for the ALL_EQUAL_MIN_ZIGZAG constraint obtained by applying decoration Table 2.36 to the seed transducer of the ZIGZAG pattern: (1) missing transitions from a, b, c, d, e, f to s are labelled by =; (2) on transitions from b, c, e, f to s the accumulator $D$ is reset to its initial value; (3) on transitions from c, f to s the accumulator $F$ is reset to $C$, and the accumulator $R$ is updated wrt $C$ and $F$.
### 3.33 ALL_EQUAL_RANGE_DECREASING

#### DESCRIPTION

**Origin**
Based on the `DECREASING` pattern.

**Constraint**

\[ \text{ALL_EQUAL_RANGE_DECREASING} (\text{VARIABLES}) \]

**Argument**

\[ \text{VARIABLES} : \text{collection} (\text{var} - \text{dvar}) \]

**Restriction**

\[ \text{required} (\text{VARIABLES}, \text{var}) \]

Succeeds if the differences between the largest and smallest value in each occurrence of the `DECREASING` pattern in the time-series given by the `VARIABLES` collection are the same.

**Purpose**

An occurrence of the pattern `DECREASING` is the subsequence which matches the regular expression `>`. Assume that the occurrence of the pattern `DECREASING` starts at position \( i \) and ends at position \( j \). The feature `RANGE` computes the range of the values from index \( i \) to index \( j + 1 \).

**Example**

\[ ((1, 4, 6, 2, 2, 4, 4, 5, 5, 5, 1, 3, 6, 2, 5)) \]

Figure 3.65 provides an example where the `ALL_EQUAL_RANGE_DECREASING ([1, 4, 6, 2, 2, 4, 4, 5, 5, 5, 1, 3, 6, 2, 5])` constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 1 \]

\[ \text{range} (\text{VARIABLES}\.\text{var}) > 1 \]
Figure 3.65: Illustrating the ALL_EQUAL_RANGE_DECREASING constraint of the Example slot
Automaton

Figure 3.66 depicts the automaton associated with the constraint `ALL_EQUAL_RANGE_DECREASING`.

\[
\begin{align*}
C & \leftarrow X \\
F & \leftarrow X \\
H & \leftarrow \text{VAR}_1 \\
R & \leftarrow 1 \\
& \quad \{ C \leftarrow |H - \text{VAR}_{i+1}| \\
& \quad F \leftarrow |H - \text{VAR}_{i+1}| \\
& \quad H \leftarrow \text{VAR}_{i+1} \\
& \quad R \leftarrow R \land (F = |H - \text{VAR}_{i+1}|) \\
\end{align*}
\]

Figure 3.66: Automaton for the `ALL_EQUAL_RANGE_DECREASING` constraint obtained by applying decoration Table 2.47 to the seed transducer of the `DECREASING` pattern.
### 3.34 **ALL_EQUAL_RANGE_DECREASING_SEQUENCE**

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th><strong>Automaton</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the <code>DECREASING_SEQUENCE</code> pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td><code>ALL_EQUAL_RANGE_DECREASING_SEQUENCE(VARIABLES)</code></td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td><code>VARIABLES : collection(var-dvar)</code></td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td><code>required(VARIABLES, var)</code></td>
</tr>
</tbody>
</table>

Succeeds if the differences between the largest and smallest value in each occurrence of the `DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are the same.

An occurrence of the pattern `DECREASING_SEQUENCE` is the maximal subsequence which matches the regular expression `>(>|=)*>|>`. Assume that the occurrence of the pattern `DECREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `RANGE` computes the range of the values from index `i` to index `j + 1`.

**Example**

```
((1, 6, 7, 2, 4, 5, 5, 4, 4, 2, 0, 3, 6, 1, 5))
```

Figure 3.67 provides an example where the `ALL_EQUAL_RANGE_DECREASING_SEQUENCE` `([1, 6, 7, 2, 4, 5, 5, 4, 4, 2, 0, 3, 6, 1, 5])` constraint holds.

**Typical**

```
|VARIABLES| > 1
range(VARIABLES.var) > 1
```
Figure 3.67: Illustrating the `ALL_EQUAL_RANGE_DECREASING_SEQUENCE` constraint of the Example slot.
Automaton

Figure 3.68 depicts the automaton associated with the constraint ALL_EQUAL RANGE DECREASING SEQUENCE.

Figure 3.68: Automaton for the ALL_EQUAL_RANGE_DECREEASING_SEQUENCE constraint obtained by applying decoration Table 2.47 to the seed transducer of the DECREASING_SEQUENCE pattern.
3.35 **ALL_EQUAL_RANGE_INCREASING**

<table>
<thead>
<tr>
<th><strong>DESCRIPTION</strong></th>
<th><strong>AUTOMATON</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the <code>INCREASING</code> pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td><code>ALL_EQUAL_RANGE_INCREASING(VARIABLES)</code></td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td><code>VARIABLES : collection(var−dvar)</code></td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td><code>required(VARIABLES,var)</code></td>
</tr>
</tbody>
</table>

Succeeds if the differences between the largest and smallest value in each occurrence of the `INCREASING` pattern in the time-series given by the `VARIABLES` collection are the same.

**Purpose**
An occurrence of the pattern `INCREASING` is the subsequence which matches the regular expression `<`. Assume that the occurrence of the pattern `INCREASING` starts at position $i$ and ends at position $j$. The feature `RANGE` computes the range of the values from index $i$ to index $j+1$.

**Example**

```
((7, 3, 6, 6, 3, 2, 5, 4, 3, 1, 4, 1))
```

Figure 3.69 provides an example where the `ALL_EQUAL_RANGE_INCREASING` `((7, 3, 6, 6, 3, 2, 5, 4, 3, 1, 4, 1))` constraint holds.

**Typical**

```
|VARIABLES| > 1
range(VARIABLES.var) > 1
```
Figure 3.69: Illustrating the **ALL_EQUAL_RANGE_INCREASING** constraint of the **Example** slot
Automaton

Figure 3.70 depicts the automaton associated with the constraint *ALL_EQUAL_RANGE_INCREASING*.

\[
\begin{align*}
C \leftarrow X \\
F \leftarrow X \\
H \leftarrow \text{VAR}_1 \\
R \leftarrow 1
\end{align*}
\]

\[
\begin{align*}
C \leftarrow |H - \text{VAR}_{i+1}| \\
F \leftarrow |H - \text{VAR}_{i+1}| \\
H \leftarrow \text{VAR}_{i+1} \\
R \leftarrow R \land (F = |H - \text{VAR}_{i+1}|)
\end{align*}
\]

\[
\begin{align*}
C \leftarrow X \\
F \leftarrow X \\
H \leftarrow \text{VAR}_1 \\
R \leftarrow 1
\end{align*}
\]

\[
\begin{align*}
C \leftarrow |H - \text{VAR}_{i+1}| \\
F \leftarrow |H - \text{VAR}_{i+1}| \\
H \leftarrow \text{VAR}_{i+1} \\
R \leftarrow R \land (F = |H - \text{VAR}_{i+1}|)
\end{align*}
\]

Figure 3.70: Automaton for the *ALL_EQUAL_RANGE_INCREASING* constraint obtained by applying decoration Table 2.47 to the seed transducer of the *INCREASING* pattern.
ALL_EQUAL_RANGE_INCREASING
### 3.36 ALL_EQUAL_RANGE_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the INCREASING_SEQUENCE pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>ALL_EQUAL_RANGE_INCREASING_SEQUENCE(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var−dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES.var)</td>
</tr>
</tbody>
</table>

**Purpose**

Succeeds if the differences between the largest and smallest value in each occurrence of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are the same.

An occurrence of the pattern INCREASING_SEQUENCE is the *maximal* subsequence which matches the regular expression \(< (\leq | =)\star < | <\).

Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature RANGE computes the range of the values from index \(i\) to index \(j + 1\).

**Example**

\[
((7, 2, 2, 6, 6, 3, 1, 1, 2, 3, 5, 4, 2, 6, 1))
\]

Figure 3.71 provides an example where the ALL_EQUAL_RANGE_INCREASING_SEQUENCE ((7, 2, 2, 6, 6, 3, 1, 1, 2, 3, 5, 4, 2, 6, 1)) constraint holds.

**Typical**

\[
|VARIABLES| > 1 \\
\text{range}(\text{VARIABLES.var}) > 1
\]
Figure 3.71: Illustrating the ALL_EQUAL_RANGE_INCREASING_SEQUENCE constraint of the Example slot
Automaton Figure 3.72 depicts the automaton associated with the constraint \textit{ALL_EQUAL_RANGE_INCREASING_SEQUENCE}.

Figure 3.72: Automaton for the \textit{ALL_EQUAL_RANGE_INCREASING_SEQUENCE} constraint obtained by applying decoration Table 2.47 to the seed transducer of the \textit{INCREASING_SEQUENCE} pattern.
### 3.37 ALL_EQUAL_RANGE_STRICTLY_DECREASING_SEQUENCE

**DESCRIPTION**

**Origin**
Based on the STRICTLY_DECREASING_SEQUENCE pattern.

**Constraint**
ALLEQUAL_RANGE_STRICTLY_DECREASING_SEQUENCE(VARIABLES)

**Argument**
VARIABLES : collection(var – dvar)

**Restriction**
required(VARIABLES, var)

**Purpose**
Succeeds if the differences between the largest and smallest value in each occurrence of the STRICTLY_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are the same.

An occurrence of the pattern STRICTLY_DECREASING_SEQUENCE is the maximal sub-sequence which matches the regular expression \( >^+ \).

Assume that the occurrence of the pattern STRICTLY_DECREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature RANGE computes the range of the values from index \( i \) to index \( j + 1 \).

**Example**

\[ (4, 1, 4, 3, 1, 2, 4, 4, 5, 5, 4, 2, 2, 6, 6, 3) \]

Figure 3.73 provides an example where the ALL_EQUAL_RANGE_STRICTLY_DECREASING_SEQUENCE \((4, 1, 4, 3, 1, 2, 4, 4, 5, 5, 4, 2, 2, 6, 6, 3)\) constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 1 \]
\[ \text{range(VARIABLES.var)} > 1 \]
Figure 3.73: Illustrating the ALL_EQUAL_RANGE_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Figure 3.74 depicts the automaton associated with the constraint ALL_EQUAL_RANGE_STRICTLY_DECREASING_SEQUENCE.

Figure 3.74: Automaton for the ALL_EQUAL_RANGE_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.47 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern.
3.38 ALL_EQUAL_RANGE STRICTLY_INCREASING_SEQUENCE

**DESCRIPTION**

**Origin**: Based on the STRICTLY_INCREASING_SEQUENCE pattern.

**Constraint**: ALL_EQUAL_RANGE STRICTLY_INCREASING_SEQUENCE(VARIABLES)

**Argument**: VARIABLES : collection(var–dvar)

**Restriction**: required(VARIABLES, var)

Succeeds if the differences between the largest and smallest value in each occurrence of the STRICTLY_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are the same.

**Purpose**: An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal sub-sequence which matches the regular expression <^+. Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature RANGE computes the range of the values from index \( i \) to index \( j + 1 \).

**Example**: 

\[
((7, 2, 2, 6, 6, 3, 1, 1, 2, 3, 4, 5, 4, 0, 4, 1))
\]

Figure 3.75 provides an example where the ALL_EQUAL_RANGE STRICTLY_INCREASING_SEQUENCE ((7, 2, 2, 6, 6, 3, 1, 1, 2, 3, 4, 5, 4, 0, 4, 1)) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range(VARIABLES,var)} > 1
\]
ALL_EQUAL_RANGE STRICTLY_INCREASING_SEQUENCE

Figure 3.75: Illustrating the ALL_EQUAL_RANGE_STRICTLY_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 3.76 depicts the automaton associated with the constraint
ALL_EQUAL_RANGE_STRICTLY_INCREASING_SEQUENCE.

{ }
3.39 ALL_EQUAL_SURF_BUMP_ON_DECREASING_SEQUENCE

**DESCRIPTION**

**Origin**
Based on the BUMP_ON_DECREASING_SEQUENCE pattern.

**Constraint**

\[
\text{ALL_EQUAL_SURF_BUMP_ON_DECREASING_SEQUENCE(VARIABLES)}
\]

**Argument**

\[
\text{VARIABLES : collection(var\_dvar)}
\]

**Restriction**

\[
\text{required(VARIABLES, var)}
\]

Succeeds if the surface of all occurrences of the BUMP_ON_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are the same.

**Purpose**

An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence which matches the regular expression \( \text{>>} \text{<<<} \).

Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i + 2 \) to index \( j \).

**Example**

\[
((7, 6, 5, 6, 5, 4, 2, 3, 7, 4, 3, 2, 8, 6, 4, 4))
\]

Figure 3.77 provides an example where the ALL_EQUAL_SURF_BUMP_ON_DECREASING_SEQUENCE ((7, 6, 5, 6, 5, 4, 2, 3, 7, 4, 3, 2, 8, 6, 4, 4)) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 5\quad \text{range(VARIABLES.var)} > 2
\]
Figure 3.77: Illustrating the `ALL_EQUAL_SURF_BUMP_ON_DECREASING_SEQUENCE` constraint of the `Example` slot
Figure 3.78 depicts the automaton associated with the constraint \texttt{ALL_EQUAL_SURF_BUMP_ON_DECREASING_SEQUENCE}.

Figure 3.78: Automaton for the \texttt{ALL_EQUAL_SURF_BUMP_ON_DECREASING_SEQUENCE} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \texttt{BUMP_ON_DECREASING_SEQUENCE} pattern.
3.40 ALL_EQUAL_SURF_DECREASING

**DESCRIPTION**

**Origin**
Based on the **DECREASING** pattern.

**Constraint**
`ALL_EQUAL_SURF_DECREASING(VARIABLES)`

**Argument**
`VARIABLES : collection(var−dvar)`

**Restriction**
`required(VARIABLES, var)`

**Purpose**
Succeeds if the surface of all occurrences of the **DECREASING** pattern in the time-series given by the `VARIABLES` collection are the same.

An occurrence of the pattern **DECREASING** is the subsequence which matches the regular expression `>`.

Assume that the occurrence of the pattern **DECREASING** starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i` to index `j + 1`.

**Example**

```
((1, 5, 5, 2, 3, 4, 6, 1, 2, 3, 4, 5, 2, 3))
```

Figure 3.79 provides an example where the `ALL_EQUAL_SURF_DECREASING` `((1, 5, 5, 2, 3, 4, 6, 1, 2, 3, 4, 5, 2, 3))` constraint holds.

![Graph illustrating the ALL_EQUAL_SURF_DECREASING constraint](image)

Figure 3.79: Illustrating the `ALL_EQUAL_SURF_DECREASING` constraint of the **Example** slot
Typical

- \[ |\text{VARIABLES}| > 1 \]
- \[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]
Figure 3.80 depicts the automaton associated with the constraint `ALL_EQUAL_SURF_DECREASING`.

![Automaton Diagram]

Figure 3.80: Automaton for the `ALL_EQUAL_SURF_DECREASING` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `DECREASING` pattern.
3.41 ALL_EQUAL_SURF_DECREASING_SEQUENCE

**Description**

Based on the DECREASING_SEQUENCE pattern.

**Constraint**

ALL_EQUAL_SURF_DECREASING_SEQUENCE(VARIABLES)

**Argument**

VARIABLES : collection(var–dvar)

**Restriction**

required(VARIABLES, var)

Succeeds if the surface of all occurrences of the DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are the same.

**Purpose**

An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression \( (> | =)^* > | > \). Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i \) to index \( j + 1 \).

**Example**

\( (1, 6, 6, 4, 3, 2, 2, 5, 6, 6, 5, 5, 2, 3, 3) \)

Figure 3.81 provides an example where the ALL_EQUAL_SURF_DECREASING_SEQUENCE constraint holds.

---

Figure 3.81: Illustrating the ALL_EQUAL_SURF_DECREASING_SEQUENCE constraint of the Example slot
Typical

\[ |\text{VARIABLES}| > 1 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]
Figure 3.82 depicts the automaton associated with the constraint `ALL_EQUAL_SURF_DECREASING_SEQUENCE`.

\[
\begin{align*}
C &\leftarrow X \\
D &\leftarrow 0 \\
F &\leftarrow X \\
R &\leftarrow 1 \\
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow 0 \} & \quad \leq s \\
\{ F \leftarrow C \} & \quad \leq \\quad \leq \\
\{ R \leftarrow R \land (F = C) \} & \end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow D + \text{VAR}_{i+1} \} \quad \geq t \\
\{ C \leftarrow D + \text{VAR}_{i+1} \} \\
\{ D \leftarrow 0 \} \\
\end{align*}
\]

Figure 3.82: Automaton for the `ALL_EQUAL_SURF_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `DECREASING_SEQUENCE` pattern.
3.42  ALL_EQUAL_SURF_DECREASING_TERRACE

DESCRIPTION

Origin
Based on the DECREASING_TERRACE pattern.

Constraint
ALL_EQUAL_SURF_DECREASING_TERRACE(VARIABLES)

Argument
VARIABLES : collection(var−dvar)

Restriction
required(VARIABLES, var)

Succeeds if the surface of all occurrences of the DECREASING_TERRACE pattern in the time-series given by the VARIABLES collection are the same.

An occurrence of the pattern DECREASING_TERRACE is the maximal subsequence which matches the regular expression $\geq^{+} \geq$.

Assume that the occurrence of the pattern DECREASING_TERRACE starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 1$ to index $j$.

Purpose

Example

$((5, 4, 4, 1, 6, 6, 2, 2, 2, 1, 6, 4, 4, 2, 2))$

Figure 3.83 provides an example where the ALL_EQUAL_SURF_DECREASING_TERRACE $((5, 4, 4, 1, 6, 6, 2, 2, 2, 1, 6, 4, 4, 2, 2))$ constraint holds.

Figure 3.83: Illustrating the ALL_EQUAL_SURF_DECREASING_TERRACE constraint of the Example slot
Typical

\[
\begin{align*}
|\text{VARIBALES}| & > 3 \\
\text{range(\text{VARIBALES.var})} & > 2
\end{align*}
\]
Automaton Figure 3.84 depicts the automaton associated with the constraint ALL_EQUAL_SURF_DECREASING_TERRACE.

\[
\begin{align*}
C &\leftarrow X \\
D &\leftarrow 0 \\
F &\leftarrow X \\
R &\leftarrow 1 \\
\end{align*}
\leq
\begin{align*}
\{ D \leftarrow D + \text{VAR} \} \\
\{ C \leftarrow D + \text{VAR} \} \\
\{ F \leftarrow D + \text{VAR} \} \\
\{ R \leftarrow R \land (F = D + \text{VAR}) \} \\
\end{align*}
\]

\[
\begin{align*}
R \land (F = C) &\leq \delta \\
\{ D \leftarrow 0 \} &> \\
\{ D \leftarrow D + \text{VAR} \} &< \\
\{ D \leftarrow D + \text{VAR} \} &< \\
\{ D \leftarrow D + \text{VAR} \} &> \\
\end{align*}
\]

Figure 3.84: Automaton for the ALL_EQUAL_SURF_DECREASING_TERRACE constraint obtained by applying decoration Table 2.36 to the seed transducer of the DECREASING_TERRACE pattern.
### 3.43 ALL_EQUAL_SURF_DIP_ON_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th><strong>DESCRIPTION</strong></th>
<th><strong>AUTOMATON</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the <code>DIP_ON_INCREASING_SEQUENCE</code> pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td><code>ALL_EQUAL_SURF_DIP_ON_INCREASING_SEQUENCE(VARIABLES)</code></td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td><code>VARIABLES : collection(var–dvar)</code></td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td><code>required(VARIABLES, var)</code></td>
</tr>
</tbody>
</table>

Succeeds if the surface of all occurrences of the `DIP_ON_INCREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are the same.

#### Purpose

An occurrence of the pattern `DIP_ON_INCREASING_SEQUENCE` is the subsequence which matches the regular expression `<<><<<`. Assume that the occurrence of the pattern `DIP_ON_INCREASING_SEQUENCE` starts at position $i$ and ends at position $j$. The feature `SURF` computes the sum of the values from index $i + 2$ to index $j$.

#### Example

```latex
((1, 2, 3, 2, 3, 4, 6, 5, 1, 4, 5, 6, 0, 2, 4, 4))
```

Figure 3.85 provides an example where the `ALL_EQUAL_SURF_DIP_ON_INCREASING_SEQUENCE` `((1, 2, 3, 2, 3, 4, 6, 5, 1, 4, 5, 6, 0, 2, 4, 4))` constraint holds.

#### Typical

- `|VARIABLES| > 5`
- `range(VARIABLES.var) > 2`
Figure 3.85: Illustrating the ALL_EQUAL_SURF_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot
Figure 3.86 depicts the automaton associated with the constraint \texttt{ALL_EQUAL_SURF_DIP_ON_INCREASING_SEQUENCE}.

Figure 3.86: Automaton for the \texttt{ALL_EQUAL_SURF_DIP_ON_INCREASING_SEQUENCE} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \texttt{DIP_ON_INCREASING_SEQUENCE} pattern
ALL_EQUAL_SURF_DIP_ON_INCREASING_SEQUENCE

293
### 3.44 ALL_EQUAL_SURF_GORGE

**DESCRIPTION**

**Origin**
Based on the GORGE pattern.

**Constraint**

\[
\text{ALL_EQUAL_SURF_GORGE}(\text{VARIABLES})
\]

**Argument**

\[
\text{VARIABLES} : \text{collection}(\text{var} \rightarrow \text{dvar})
\]

**Restriction**

\[
\text{required}(\text{VARIABLES, var})
\]

**Purpose**
Succeeds if the surface of all occurrences of the GORGE pattern in the time-series given by the VARIABLES collection are the same.

An occurrence of the pattern GORGE is the \textit{maximal} subsequence which matches the regular expression \((>|>(=>|>)*)<(>|<(<(|<>)*)<\).

Assume that the occurrence of the pattern GORGE starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i+1\) to index \(j\).

**Example**

\[
\langle 6, 4, 2, 3, 5, 1, 1, 5, 4, 2, 3, 6, 3, 2, 4, 5 \rangle
\]

Figure 3.87 provides an example where the ALL_EQUAL_SURF_GORGE \(([6, 4, 2, 3, 5, 1, 1, 5, 4, 2, 3, 6, 3, 2, 4, 5])\) constraint holds.

![Figure 3.87: Illustrating the ALL_EQUAL_SURF_GORGE constraint of the Example slot](image)

**Typical**

\[
\text{|VARIABLES|} > 2 \\
\text{range}(\text{VARIABLES.var}) > 1
\]
Figure 3.88 depicts the automaton associated with the constraint \textsc{all_equal_surf_gorge}.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{automaton}
\caption{Automaton for the \textsc{all_equal_surf_gorge} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textsc{gorge} pattern (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$)}
\end{figure}
### 3.45 ALL_EQUAL_SURF_INCREASING

<table>
<thead>
<tr>
<th><strong>Origin</strong></th>
<th>Based on the INCREASING pattern.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Constraint</strong></td>
<td>ALL_EQUAL_SURF_INCREASING(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var−dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES, var)</td>
</tr>
</tbody>
</table>

Succeeds if the surface of all occurrences of the INCREASING pattern in the time-series given by the VARIABLES collection are the same.

**Purpose**

An occurrence of the pattern INCREASING is the subsequence which matches the regular expression <.

Assume that the occurrence of the pattern INCREASING starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i\) to index \(j+1\).

**Example**

```
((6, 1, 1, 7, 4, 4, 2, 6, 4, 2, 6, 4, 2, 6, 5, 5, 5))
```

Figure 3.89 provides an example where the ALL_EQUAL_SURF_INCREASING ((6, 1, 1, 7, 4, 4, 2, 6, 4, 2, 6, 4, 2, 6, 5, 3, 5, 5)) constraint holds.

**Typical**

| VARIABLES | > 1 |
| range(VARIABLES.var) | > 1 |
Figure 3.89: Illustrating the ALL_EQUAL_SURF_INCREASING constraint of the Example slot
Automaton

Figure 3.90 depicts the automaton associated with the constraint ALL_EQUAL_SURF_INCREASING.

\[
\begin{cases}
    C & \leftarrow X \\
    D & \leftarrow 0 \\
    F & \leftarrow X \\
    R & \leftarrow 1
\end{cases}
\]

\[
\begin{cases}
    C & \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
    D & \leftarrow 0 \\
    F & \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
    R & \leftarrow R \land (F = D + \text{VAR}_i + \text{VAR}_{i+1})
\end{cases}
\]

Figure 3.90: Automaton for the ALL_EQUAL_SURF_INCREASING constraint obtained by applying decoration Table 2.36 to the seed transducer of the INCREASING pattern.
3.46  **ALL_EQUAL_SURF_INCREASING_SEQUENCE**

**DESCRIPTION**

**Origin**
Based on the *INCREASING_SEQUENCE* pattern.

**Constraint**

\[ \text{ALL_EQUAL_SURF_INCREASING_SEQUENCE(VARIABLES)} \]

**Argument**

\[ \text{VARIABLES : collection(var–dvar)} \]

**Restriction**

\[ \text{required(VARIABLES, var)} \]

Succeeds if the surface of all occurrences of the *INCREASING_SEQUENCE* pattern in the time-series given by the VARIABLES collection are the same.

An occurrence of the pattern *INCREASING_SEQUENCE* is the maximal subsequence which matches the regular expression \(< (< | =)^* < | < \).

Assume that the occurrence of the pattern *INCREASING_SEQUENCE* starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i\) to index \(j+1\).

**Purpose**

Succeeds if the surface of all occurrences of the *INCREASING_SEQUENCE* pattern in the time-series given by the VARIABLES collection are the same.

An occurrence of the pattern *INCREASING_SEQUENCE* is the maximal subsequence which matches the regular expression \(< (< | =)^* < | < \).

Assume that the occurrence of the pattern *INCREASING_SEQUENCE* starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i\) to index \(j+1\).

**Example**

\((6, 1, 1, 3, 4, 4, 5, 5, 2, 1, 1, 5, 5, 6, 4, 4)\)

**Figure 3.91** provides an example where the *ALL_EQUAL_SURF_INCREASING_SEQUENCE* \(([6, 1, 1, 3, 4, 4, 5, 5, 2, 1, 1, 5, 5, 6, 4, 4])\) constraint holds.

**Figure 3.91**: Illustrating the *ALL_EQUAL_SURF_INCREASING_SEQUENCE* constraint of the **Example** slot
Typical

$|\text{VARIABLES}| > 1$

$\text{range}(\text{VARIABLES}\.\text{var}) > 1$
Automaton

Figure 3.92 depicts the automaton associated with the constraint `ALL_EQUAL_SURF_INCREASING_SEQUENCE`.

\[
\begin{align*}
& \{ C \leftarrow X \} \\
& \{ D \leftarrow 0 \} \\
& \{ F \leftarrow X \} \\
& \{ R \leftarrow 1 \} \\
& \geq s \\
& \geq \\
& \{ D \leftarrow 0 \} \\
& \{ F \leftarrow C \} \\
& \{ R \leftarrow R \land (F = C) \} \\
& \{ C \leftarrow D + \text{VAR}_{i} + \text{VAR}_{i+1} \} \\
& \} \land \} \\
& \} \land \} \\
& \} \land \} \\
& \leq t \\
& \leftarrow D + \text{VAR}_{i+1} \} \\
& \{ C \leftarrow C \} \\
& \{ D \leftarrow 0 \} \\
& \leftarrow C + D + \text{VAR}_{i+1} \} \\
& \} \land \} \\
& \} \land \} \\
& \} \land \} \\
& \} \land \}
\end{align*}
\]

Figure 3.92: Automaton for the `ALL_EQUAL_SURF_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `INCREASING_SEQUENCE` pattern.
3.47 ALL_EQUAL_SURF_INCREASING_TERRACE

**DESCRIPTION**

**Origin**
Based on the INCREASING_TERRACE pattern.

**Constraint**
ALL_EQUAL_SURF_INCREASING_TERRACE(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES, var)

**Purpose**
Succeeds if the surface of all occurrences of the INCREASING_TERRACE pattern in the time-series given by the VARIABLES collection are the same.

An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression \(<=+<\).

Assume that the occurrence of the pattern INCREASING_TERRACE starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i+1\) to index \(j\).

**Example**

\(((2, 3, 3, 6, 1, 1, 2, 2, 2, 4, 6, 1, 3, 3, 5, 5))\)

Figure 3.93 provides an example where the ALL_EQUAL_SURF_INCREASING_TERRACE ((2, 3, 3, 6, 1, 1, 2, 2, 2, 4, 6, 1, 3, 3, 5, 5)) constraint holds.

Figure 3.93: Illustrating the ALL_EQUAL_SURF_INCREASING_TERRACE constraint of the Example slot
Typical

\[ |\text{VARIABLES}| > 3 \]
\[ \text{range} (\text{VARIABLES}.\text{var}) > 2 \]
Figure 3.94 depicts the automaton associated with the constraint `ALL_EQUAL_SURF_INCREASING_TERRACE`.

Figure 3.94: Automaton for the `ALL_EQUAL_SURF_INCREASING_TERRACE` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `INCREASING_TERRACE` pattern.
3.48 ALL_EQUAL_SURF_INFLEXION

DESCRIPTION

Origin
Based on the INFLEXION pattern.

Constraint
ALL_EQUAL_SURF_INFLEXION(VARIABLES)

Argument
VARIABLES : collection(var–dvar)

Restriction
required(VARIABLES, var)

Succeeds if the surface of all occurrences of the INFLEXION pattern in the time-series given by the VARIABLES collection are the same.

Purpose
An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression \(< (< | =)^* > | > (> | =)^* <\).
Assume that the occurrence of the pattern INFLEXION starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

Example
\(((2, 2, 3, 6, 6, 5, 5, 3, 2, 4, 5, 6, 3, 3, 3))\)

Figure 3.95 provides an example where the ALL_EQUAL_SURF_INFLEXION ((2, 2, 3, 6, 6, 5, 5, 3, 2, 4, 5, 6, 3, 3, 3)) constraint holds.

Figure 3.95: Illustrating the ALL_EQUAL_SURF_INFLEXION constraint of the Example slot
| Typical | \( |\text{VARIABLES}| > 2 \)  
|         | \( \text{range(VARIABLES.var)} > 1 \)  |
Automaton

Figure 3.96 depicts the automaton associated with the constraint `ALL_EQUAL_SURF_INFLEXION`.

Figure 3.96: Automaton for the `ALL_EQUAL_SURF_INFLEXION` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `INFLEXION` pattern (transition `r → t` has the same accumulators updates as transition `t → r`).
3.49 ALL_EQUAL_SURF_PEAK

**DESCRIPTION**

**Origin**  
Based on the PEAK pattern.

**Constraint**  
ALL_EQUAL_SURF_PEAK(VARIABLES)

**Argument**  
VARIABLES : collection(var-dvar)

**Restriction**  
required(VARIABLES, var)

Succeeds if the surface of all occurrences of the PEAK pattern in the time-series given by the VARIABLES collection are the same.

**Purpose**  
An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression `< (= | <)* (> | =)* >`. Assume that the occurrence of the pattern PEAK starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 1$ to index $j$.

**Example**  

$((1, 2, 7, 7, 1, 2, 4, 5, 7, 1, 2, 3, 6, 7, 1))$

Figure 3.97 provides an example where the ALL_EQUAL_SURF_PEAK ($(1, 2, 7, 7, 1, 2, 4, 5, 7, 1, 2, 3, 6, 7, 1)$) constraint holds.

![Figure 3.97](image-url)

Figure 3.97: Illustrating the ALL_EQUAL_SURF_PEAK constraint of the Example slot
Typical

\[ |\text{VARIABLES}| > 2 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]
Figure 3.98 depicts the automaton associated with the constraint `ALL_EQUAL_SURF_PEAK`.

![Automaton Diagram](image-url)

Figure 3.98: Automaton for the `ALL_EQUAL_SURF_PEAK` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `PEAK` pattern.
### 3.50 ALL_EQUAL_SURF_PLAIN

**Description**
Based on the PLAIN pattern.

**Constraint**
ALL_EQUAL_SURF_PLAIN(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES, var)

**Purpose**
Succeeds if the surface of all occurrences of the PLAIN pattern in the time-series given by the VARIABLES collection are the same.

An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression $>^*<$.

Assume that the occurrence of the pattern PLAIN starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i+1$ to index $j$.

**Example**

$$
((1, 6, 3, 3, 7, 6, 6, 3, 3, 5, 5, 4, 3, 3, 6, 3))
$$

Figure 3.99 provides an example where the ALL_EQUAL_SURF_PLAIN constraint holds.

![Figure 3.99: Illustrating the ALL_EQUAL_SURF_PLAIN constraint of the Example slot](image-url)
Typical

|VARIABLES| > 2
\[\text{range(VARIABLES.var)} > 1\]
Figure 3.100 depicts the automaton associated with the constraint ALL_EQUAL_SURF_PLAIN.

\[
\begin{align*}
C &\leftarrow X \\
D &\leftarrow 0 \\
F &\leftarrow X & (F = D + \text{VAR}_i) \\
R &\leftarrow R \\ 
\leq &\quad \leq &\quad \leq &\quad \leq \\
\{ D \leftarrow D + \text{VAR}_i \} &\quad \{ D \leftarrow D + \text{VAR}_i \} &\quad \{ D \leftarrow D + \text{VAR}_i \} &\quad \{ D \leftarrow D + \text{VAR}_i \} \\
\geq &\quad \geq &\quad \geq &\quad \geq \\
\{ D \leftarrow 0 \} &\quad \{ D \leftarrow 0 \} &\quad \{ D \leftarrow 0 \} &\quad \{ D \leftarrow 0 \} \\
\end{align*}
\]

Figure 3.100: Automaton for the ALL_EQUAL_SURF_PLAIN constraint obtained by applying decoration Table 2.36 to the seed transducer of the PLAIN pattern.
3.51 ALL_EQUAL_SURF_PLATEAU

**DESCRIPTION**

**Origin**
Based on the PLATEAU pattern.

**Constraint**
ALL_EQUAL_SURF_PLATEAU(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES, var)

**Purpose**
Succeeds if the surface of all occurrences of the PLATEAU pattern in the time-series given by the VARIABLES collection are the same.

An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression <="."

Assume that the occurrence of the pattern PLATEAU starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

\[(7, 2, 5, 3, 1, 2, 2, 5, 3, 3, 4, 5, 5, 2, 5)\]

Figure 3.101 provides an example where the ALL_EQUAL_SURF_PLATEAU \([(7, 2, 5, 5, 1, 2, 2, 5, 3, 3, 4, 5, 5, 2, 5))\)] constraint holds.

**Typical**

\[|VARIABLES| > 2\]
\[\text{range}(VARIABLES, \text{var}) > 1\]
Figure 3.101: Illustrating the ALL_EQUAL_SURF_PLATEAU constraint of the Example slot
Figure 3.102 depicts the automaton associated with the constraint ALL_EQUAL_SURF_PLATEAU.

Figure 3.102: Automaton for the ALL_EQUAL_SURF_PLATEAU constraint obtained by applying decoration Table 2.36 to the seed transducer of the PLATEAU pattern.
3.52 ALL_EQUAL_SURF_PROPERPLAIN

**DESCRIPTION**

- **Origin**: Based on the PROPERPLAIN pattern.

- **Constraint**: `ALL_EQUAL_SURF_PROPERPLAIN(VARIABLES)`

- **Argument**: `VARIABLES : collection(var−dvar)`

**Restriction**: `required(VARIABLES, var)`

---

Succeeds if the surface of all occurrences of the PROPERPLAIN pattern in the time-series given by the VARIABLES collection are the same.

An occurrence of the pattern PROPERPLAIN is the maximal subsequence which matches the regular expression `> =+ <`. Assume that the occurrence of the pattern PROPERPLAIN starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Purpose**

**Example**

\[(2,7,3,3,6,6,3,7,3,3,5,6,5,3,3,5)\]

Figure 3.103 provides an example where the ALL_EQUAL_SURF_PROPERPLAIN \([(2,7,3,3,6,6,3,7,3,3,5,6,5,3,3,5)]\) constraint holds.

---

**Example**

Figure 3.103: Illustrating the ALL_EQUAL_SURF_PROPERPLAIN constraint of the Example slot
Typical

\[ |\text{VARIABLES}| > 3 \]

\[ \text{range(VARIABLES.var)} > 1 \]
Figure 3.104 depicts the automaton associated with the constraint
ALL_EQUAL_SURF_PROPER.PLAIN.

Figure 3.104: Automaton for the ALL_EQUAL_SURF_PROPER.PLAIN constraint obtained by applying decoration Table 2.36 to the seed transducer of the PROPER.PLAIN pattern.
### 3.53 ALL_EQUAL_SURF_PROPER_PLATEAU

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the PROPER_PLATEAU pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>ALL_EQUAL_SURF_PROPER_PLATEAU(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : <code>collection(var−dvar)</code></td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td><code>required(VARIABLES, var)</code></td>
</tr>
</tbody>
</table>

Succeeds if the surface of all occurrences of the PROPER_PLATEAU pattern in the time-series given by the VARIABLES collection are the same.

**Purpose**

An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression `< =+ >`.

Assume that the occurrence of the pattern PROPER_PLATEAU starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i+1$ to index $j$.

**Example**

$$\{(7, 1, 5, 5, 2, 2, 5, 1, 5, 5, 3, 2, 3, 5, 5, 3)\}$$

Figure 3.105 provides an example where the ALL_EQUAL_SURF_PROPER_PLATEAU (`\{(7, 1, 5, 5, 2, 2, 5, 1, 5, 5, 3, 2, 3, 5, 5, 3)\}`) constraint holds.

**Typical**

$$|\text{VARIABLES}| > 3$$

$$\text{range}(\text{VARIABLES}.\text{var}) > 1$$
Figure 3.105: Illustrating the ALL_EQUAL_SURF_PROPER_PLATEAU constraint of the Example slot
Figure 3.106 depicts the automaton associated with the constraint ALL_EQUAL_SURF_PROPER_PLATEAU.

Figure 3.106: Automaton for the ALL_EQUAL_SURF_PROPER_PLATEAU constraint obtained by applying decoration Table 2.36 to the seed transducer of the PROPER_PLATEAU pattern.
3.54 **ALL_EQUAL_SURF_STEADY**

**Origin**
Based on the **STEADY** pattern.

**Constraint**

\[ \text{ALL_EQUAL_SURF_STEADY} (\text{VARIABLES}) \]

**Argument**

\[ \text{VARIABLES} : \text{collection}(\text{var} \ldots \text{dvar}) \]

**Restriction**

\[ \text{required} (\text{VARIABLES}, \text{var}) \]

Succeeds if the surface of all occurrences of the **STEADY** pattern in the time-series given by the **VARIABLES** collection are the same.

**Purpose**
An occurrence of the pattern **STEADY** is the subsequence which matches the regular expression '='.
Assume that the occurrence of the pattern **STEADY** starts at position \( i \) and ends at position \( j \). The feature **SURF** computes the sum of the values from index \( i \) to index \( j + 1 \).

**Example**

\[ (4, 4, 4, 2, 6, 4, 4, 4, 3, 2, 1, 4, 4, 3, 4) \]

Figure 3.107 provides an example where the **ALL_EQUAL_SURF_STEADY** ([4, 4, 4, 2, 6, 4, 4, 4, 3, 2, 1, 4, 4, 3, 4]) constraint holds.

Figure 3.107: Illustrating the **ALL_EQUAL_SURF_STEADY** constraint of the **Example** slot.
| Typical | $|\text{VARIABLES}| > 1$ |
Figure 3.108 depicts the automaton associated with the constraint \textit{ALL_EQUAL_SURF_STEADY}.

\begin{align*}
C & \leftarrow X \\
D & \leftarrow 0 \\
F & \leftarrow X \\
R & \leftarrow 1
\end{align*}

\begin{align*}
C & \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
D & \leftarrow 0 \\
F & \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
R & \leftarrow R \land (F = D + \text{VAR}_i + \text{VAR}_{i+1})
\end{align*}

Figure 3.108: Automaton for the \textit{ALL_EQUAL_SURF_STEADY} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textit{STEADY} pattern.
### 3.55 ALL_EQUAL_SURF_STEADY_SEQUENCE

**DESCRIPTION**

**Origin**
Based on the STEADY_SEQUENCE pattern.

**Constraint**
ALL_EQUAL_SURF_STEADY_SEQUENCE(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES, var)

Succeeds if the surface of all occurrences of the STEADY_SEQUENCE pattern in the time-series given by the VARIABLES collection are the same.

**Purpose**
An occurrence of the pattern STEADY_SEQUENCE is the maximal subsequence which matches the regular expression =⁺.

Assume that the occurrence of the pattern STEADY_SEQUENCE starts at position i and ends at position j. The feature SURF computes the sum of the values from index i to index j + 1.

**Example**

```
( (6, 6, 5, 3, 3, 1, 4, 4, 4, 5, 6, 6, 7, 2))
```

Figure 3.109 provides an example where the ALL_EQUAL_SURF_STEADY_SEQUENCE ( [6, 6, 5, 3, 3, 3, 1, 4, 4, 4, 5, 6, 6, 7, 2]) constraint holds.

**Typical**

|VARIABLES| > 1
Figure 3.109: Illustrating the ALL_EQUAL_SURF_STEADY_SEQUENCE constraint of the Example slot
Figure 3.110 depicts the automaton associated with the constraint `ALL_EQUAL_SURF_STEADYSEQUENCE`.

\[
\begin{align*}
C &\leftarrow X \\
D &\leftarrow 0 \\
F &\leftarrow X \\
R &\leftarrow 1
\end{align*}
\]

\[
\begin{align*}
D &\leftarrow 0 \\
F &\leftarrow 0 \\
R &\leftarrow R \land (F = C)
\end{align*}
\]

\[
\begin{align*}
C &\leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
D &\leftarrow 0
\end{align*}
\]

Figure 3.110: Automaton for the `ALL_EQUAL_SURF_STEADYSEQUENCE` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `STEADYSEQUENCE` pattern.
### 3.56 **ALL_EQUAL_SURF_STRICTLY_DECREASING_SEQUENCE**

**Origin**
Based on the **STRICTLY_DECREASING_SEQUENCE** pattern.

**Constraint**

\[
\text{ALL_EQUAL_SURF_STRICTLY_DECREASING_SEQUENCE(VARIABLES)}
\]

**Argument**

\[
\text{VARIABLES} : \text{collection(var-dvar)}
\]

**Restriction**

\[
\text{required(VARIABLES, var)}
\]

Succeeds if the surface of all occurrences of the **STRICTLY_DECREASING_SEQUENCE** pattern in the time-series given by the **VARIABLES** collection are the same.

**Purpose**
An occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** is the maximal sub-sequence which matches the regular expression \( >^+ \).
Assume that the occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** starts at position \( i \) and ends at position \( j \). The feature **SURF** computes the sum of the values from index \( i \) to index \( j+1 \).

**Example**

\[
((3, 6, 5, 4, 5, 6, 6, 5, 3, 1, 2, 6, 5, 3, 1))
\]

Figure 3.111 provides an example where the **ALL_EQUAL_SURF_STRICTLY_DECREASING_SEQUENCE** \([(3, 6, 5, 4, 5, 6, 6, 5, 3, 1, 2, 6, 5, 3, 1)]\) constraint holds.

**Typical**

\[
\text{\{VARIABLES\} > 1} \\
\text{range(VARIABLES.var) > 1}
\]
Figure 3.111: Illustrating the `ALL_EQUAL_SURF STRICTLY DECREASING_SEQUENCE` constraint of the Example slot
Figure 3.112 depicts the automaton associated with the constraint \textsc{ALL\_EQUAL\_SURF\_STRUCTLY\_DECREASING\_SEQUENCE}.

Figure 3.112: Automaton for the \textsc{ALL\_EQUAL\_SURF\_STRUCTLY\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textsc{STRUCTLY\_DECREASING\_SEQUENCE} pattern.
### 3.57 ALL_EQUAL_SURF STRICTLY_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the STRICTLY_INCREASING_SEQUENCE pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>( \text{ALL_EQUAL_SURF_STRICTLY_INCREASING_ SEQUENCE}(\text{VARIABLES}) )</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>( \text{VARIABLES} : \text{collection}(\text{var}-\text{dvar}) )</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>( \text{required}(\text{VARIABLES}, \text{var}) )</td>
</tr>
<tr>
<td><strong>Purpose</strong></td>
<td>Succeeds if the surface of all occurrences of the STRICTLY_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are the same. An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression (&lt;^+). Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at position ( i ) and ends at position ( j ). The feature ( \text{SURF} ) computes the sum of the values from index ( i ) to index ( j+1 ).</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>( ((6, 1, 2, 3, 4, 5, 6, 4, 4)) )</td>
</tr>
</tbody>
</table>
| **Typical** | \( |\text{VARIABLES}| > 1 \)  
\( \text{range}(\text{VARIABLES.var}) > 1 \) |

Figure 3.113 provides an example where the ALL_EQUAL_SURF\_STRICTLY_INCREASING\_SEQUENCE \( ([6, 1, 1, 2, 3, 4, 5, 2, 1, 1, 3, 5, 6, 4, 4]) \) constraint holds.
Figure 3.113: Illustrating the `ALL_EQUAL_SURF STRICTLY_INCREASING_SEQUENCE` constraint of the Example slot
Figure 3.114 depicts the automaton associated with the constraint \textsc{AllEqualSurfStrictlyIncreasingSequence}.

\begin{equation}
\begin{cases}
C \leftarrow X \\
D \leftarrow 0 \\
F \leftarrow X \\
R \leftarrow 1
\end{cases}
\end{equation}

\begin{equation}
\begin{cases}
D \leftarrow 0 \\
F \leftarrow C \\
R \leftarrow R \land (F = C)
\end{cases}
\end{equation}

\begin{equation}
\begin{cases}
C \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
D \leftarrow 0
\end{cases}
\end{equation}

Figure 3.114: Automaton for the \textsc{AllEqualSurfStrictlyIncreasingSequence} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textsc{StrictlyIncreasingSequence} pattern.
### 3.58 ALL_EQUAL_SURF_SUMMIT

**Description**

**Origin**

Based on the **SUMMIT** pattern.

**Constraint**

```
ALL_EQUAL_SURF_SUMMIT(VARIABLES)
```

**Argument**

```
VARIABLES : collection(var−dvar)
```

**Restriction**

```
required(VARIABLES, var)
```

Succeeds if the surface of all occurrences of the **SUMMIT** pattern in the time-series given by the **VARIABLES** collection are the same.

**Purpose**

An occurrence of the pattern **SUMMIT** is the *maximal* subsequence which matches the regular expression `(\( < (\= | <) < (\> | >) (\= | >) \)* >)`. Assume that the occurrence of the pattern **SUMMIT** starts at position `i` and ends at position `j`. The feature **SURF** computes the sum of the values from index `i + 1` to index `j`.

**Example**

```
((1, 3, 5, 4, 2, 6, 6, 2, 3, 5, 4, 1, 4, 5, 3, 2))
```

Figure 3.115 provides an example where the **ALL_EQUAL_SURF_SUMMIT** `(1, 3, 5, 4, 2, 6, 6, 2, 3, 5, 4, 1, 4, 5, 3, 2)` constraint holds.

![Figure 3.115: Illustrating the ALL_EQUAL_SURF_SUMMIT constraint of the Example slot](image-url)
Typical

\[ |\text{VARIABLES}| > 2 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]
Figure 3.116 depicts the automaton associated with the constraint \textsc{ALL_EQUAL_SURF_SUMMIT}.

\begin{figure}
\centering
\includegraphics[width=\textwidth]{figure3116.png}
\caption{Automaton for the \textsc{ALL_EQUAL_SURF_SUMMIT} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textsc{SUMMIT} pattern (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$)}
\end{figure}
### ALL_EQUAL_SURF_VALLEY

**Description**

Based on the VALLEY pattern.

**Constraint**

\[\text{ALL_EQUAL_SURF_VALLEY} (\text{VARIABLES})\]

**Argument**

\[\text{VARIABLES} : \text{collection}(\text{var}-\text{dvar})\]

**Restriction**

\[\text{required} (\text{VARIABLES}, \text{var})\]

Succeeds if the surface of all occurrences of the VALLEY pattern in the time-series given by the VARIABLES collection are the same.

**Purpose**

An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression \[\geq (\leq \geq)* (\leq \geq)* <\].

Assume that the occurrence of the pattern VALLEY starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

\[\langle 7, 6, 6, 1, 1, 7, 6, 4, 3, 1, 7, 6, 5, 2, 1, 7 \rangle\]

Figure 3.117 provides an example where the ALL_EQUAL_SURF_VALLEY (\[7, 6, 6, 1, 1, 7, 6, 4, 3, 1, 7, 6, 5, 2, 1, 7\]) constraint holds.

**Typical**

\[|\text{VARIABLES}| > 2\]

\[\text{range} (\text{VARIABLES}.\text{var}) > 1\]
Figure 3.117: Illustrating the **ALL_EQUAL_SURF_VALLEY** constraint of the **Example** slot
Figure 3.118 depicts the automaton associated with the constraint ALL_EQUAL_SURF_VALLEY.

Automaton Figure 3.118 demonstrates the automaton obtained by applying decoration Table 2.36 to the seed transducer of the VALLEY pattern.
3.60 **ALL_EQUAL_SURF_ZIGZAG**

---

**DESCRIPTION**

**Origin**

Based on the ZIGZAG pattern.

**Constraint**

\[ \text{ALL_EQUAL_SURF_ZIGZAG} (\text{VARIABLES}) \]

**Argument**

\[ \text{VARIABLES} : \text{collection}(\text{var} - \text{dvar}) \]

**Restriction**

\[ \text{required}(\text{VARIABLES}, \text{var}) \]

**Purpose**

Succeeds if the surface of all occurrences of the ZIGZAG pattern in the time-series given by the \text{VARIABLES} collection are the same.

An occurrence of the pattern ZIGZAG is the \textit{maximal} subsequence which matches the regular expression \((<>^+ (< | <>)) | (><^+ (> | >>))\).

Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature \text{SURF} computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

\[ ((1, 6, 1, 2, 6, 5, 1, 2, 1, 2, 1, 4, 5, 2, 4, 7)) \]

Figure 3.119 provides an example where the \text{ALL_EQUAL_SURF_ZIGZAG} constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 3 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]
Figure 3.119: Illustrating the ALL_EQUAL_SURF_ZIGZAG constraint of the Example slot
Automaton Figure 3.120 depicts the automaton associated with the constraint ALL_EQUAL_SURF_ZIGZAG.
Figure 3.120: Automaton for the ALL_EQUAL_SURF_ZIGZAG constraint obtained by applying decoration Table 2.36 to the seed transducer of the ZIGZAG pattern: (1) missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=$; (2) on transitions from $b, c, e, f$ to $s$ the accumulator $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the accumulator $F$ is reset to $C$, and the accumulator $R$ is updated wrt $C$ and $F$.
3.61 **ALL_EQUAL_WIDTH_DECREASING_SEQUENCE**

**DESCRIPTION**

Origin

Based on the **DECREASING_SEQUENCE** pattern.

Constraint

**ALL_EQUAL_WIDTH_DECREASING_SEQUENCE**(VARIABLES)

Argument

VARIABLES : \textit{collection}(\textit{var}\,-\textit{dvar})

Restriction

\texttt{required}(\textit{VARIABLES}, \textit{var})

Purpose

Succeeds if the width of all occurrences of the **DECREASING_SEQUENCE** pattern in the time-series given by the \textit{VARIABLES} collection are the same.

An occurrence of the pattern **DECREASING_SEQUENCE** is the \textit{maximal} subsequence which matches the regular expression \( (> | =)^* > | > \).

Assume that the occurrence of the pattern **DECREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **WIDTH** computes the value \(j - i + 2\).

**Example**

\(((1, 3, 3, 2, 2, 1, 1, 4, 6, 4, 4, 2, 2, 3, 4))\)

Figure 3.121 provides an example where the **ALL_EQUAL_WIDTH_DECREASING_SEQUENCE** ([1, 3, 3, 2, 2, 1, 1, 4, 6, 4, 4, 2, 2, 3, 4]) constraint holds.

![Diagram of values and features](image-url)

Figure 3.121: Illustrating the **ALL_EQUAL_WIDTH_DECREASING_SEQUENCE** constraint of the **Example** slot
Typical

\[ |\text{VARIABLES}| > 1 \]
\[ \text{range(VARIABLES.var)} > 1 \]
Figure 3.122 depicts the automaton associated with the constraint ALL_EQUAL_WIDTH_DECREASING_SEQUENCE.

\[
\begin{align*}
C & \leftarrow X \\
D & \leftarrow 0 \\
F & \leftarrow X \\
R & \leftarrow 1
\end{align*}
\]

\[
\begin{align*}
D & \leftarrow 0 \\
F & \leftarrow C \\
R & \leftarrow R \land (F = C)
\end{align*}
\]

\[
\begin{align*}
D & \leftarrow D + 1
\end{align*}
\]

Figure 3.122: Automaton for the ALL_EQUAL_WIDTH_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the DECREASING_SEQUENCE pattern.
3.62 ALL_EQUAL_WIDTH_DECREASING_TERRACE

**DESCRIPTION**

**Origin**

Based on the DECREASING_TERRACE pattern.

**Constraint**

ALL_EQUAL_WIDTH_DECREASING_TERRACE(VARIABLES)

**Argument**

VARIABLES : collection(var─dvar)

**Restriction**

required(VARIABLES, var)

Succeeds if the width of all occurrences of the DECREASING_TERRACE pattern in the time-series given by the VARIABLES collection are the same.

**Purpose**

An occurrence of the pattern DECREASING_TERRACE is the maximal subsequence which matches the regular expression \( \geq^+ \geq \).

Assume that the occurrence of the pattern DECREASING_TERRACE starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

**Example**

\( ((6, 5, 5, 4, 3, 1, 1, 3, 3, 5, 2, 2, 1, 1, 1)) \)

Figure 3.123 provides an example where the ALL_EQUAL_WIDTH_DECREASING_TERRACE \( ([6, 5, 5, 4, 3, 1, 1, 3, 3, 5, 2, 2, 1, 1, 1]) \) constraint holds.

Figure 3.123: Illustrating the ALL_EQUAL_WIDTH_DECREASING_TERRACE constraint of the Example slot
<table>
<thead>
<tr>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
</tr>
<tr>
<td>$\text{range}(VARIABLES.\text{var}) &gt; 2$</td>
</tr>
</tbody>
</table>
Automaton

Figure 3.124 depicts the automaton associated with the constraint `ALL_EQUAL_WIDTH_DECREASING_TERRACE`.

Figure 3.124: Automaton for the `ALL_EQUAL_WIDTH_DECREASING_TERRACE` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `DECREASING_TERRACE` pattern
3.63 **ALL_EQUAL_WIDTH_GORGE**

**DESCRIPTION AUTOMATON**

**Constraint**

\[ \text{ALL_EQUAL_WIDTH_GORGE}(\text{VARIABLES}) \]

**Argument**

\[ \text{VARIABLES : collection(var–dvar)} \]

**Restriction**

\[ \text{required}(\text{VARIABLES, var}) \]

Succeeds if the width of all occurrences of the \text{GORGE} pattern in the time-series given by the \text{VARIABLES} collection are the same.

An occurrence of the pattern \text{GORGE} is the \textit{maximal} subsequence which matches the regular expression \((>|>(=}|>)^+)<<(=}|>)^+)<\).

Assume that the occurrence of the pattern \text{GORGE} starts at position \(i\) and ends at position \(j\). The feature \text{WIDTH} computes the value \(j - i\).

**Example**

\((6, 4, 2, 3, 5, 1, 1, 5, 4, 2, 3, 6, 3, 2, 4, 5)\)

Figure 3.125 provides an example where the \text{ALL_EQUAL_WIDTH_GORGE} ((6, 4, 2, 3, 5, 1, 1, 5, 4, 2, 3, 6, 3, 2, 4, 5)) constraint holds.

Figure 3.125: Illustrating the \text{ALL_EQUAL_WIDTH_GORGE} constraint of the \textbf{Example} slot
Typical

| VARIABLES | > 2

range(VARIABLES.var) > 1
Automaton Figure 3.126 depicts the automaton associated with the constraint ALL_EQUAL_WIDTH_GORGE.

Figure 3.126: Automaton for the ALL_EQUAL_WIDTH_GORGE constraint obtained by applying decoration Table 2.36 to the seed transducer of the GORGE pattern (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$).
### Description

**Origin**
Based on the `INCREASING_SEQUENCE` pattern.

**Constraint**

\[
\text{ALL_EQUAL_WIDTH_INCREASING_SEQUENCE}(\text{VARIABLES})
\]

**Argument**

\[
\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})
\]

**Restriction**

\[
\text{required}(\text{VARIABLES}, \text{var})
\]

Succeeds if the width of all occurrences of the `INCREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are the same.

**Purpose**

An occurrence of the pattern `INCREASING_SEQUENCE` is the maximal subsequence which matches the regular expression \(< (< | =)^* < | <\).

Assume that the occurrence of the pattern `INCREASING_SEQUENCE` starts at position \(i\) and ends at position \(j\). The feature `WIDTH` computes the value \(j - i + 2\).

**Example**

\[
((6, 4, 4, 5, 5, 6, 6, 3, 1, 1, 3, 3, 5, 5, 4, 3))
\]

Figure 3.127 provides an example where the `ALL_EQUAL_WIDTH_INCREASING_SEQUENCE` constraint holds.

![Figure 3.127: Illustrating the ALL_EQUAL_WIDTH_INCREASING_SEQUENCE constraint of the Example slot](image-url)
Typical

<table>
<thead>
<tr>
<th>VARIABLES</th>
<th>&gt; 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>range(VARIABLES.var)</td>
<td>&gt; 1</td>
</tr>
</tbody>
</table>
Automaton Figure 3.128 depicts the automaton associated with the constraint \texttt{ALL_EQUAL_WIDTH_INCREASING_SEQUENCE}.

\[\begin{align*}
\{ & C \leftarrow X \\
& D \leftarrow 0 \\
& F \leftarrow X \\
& R \leftarrow 1 \} \\
\{ & D \leftarrow 0 \\
& F \leftarrow C \\
& R \leftarrow R \land (F = C) \} \\
\{ & C \leftarrow D + 2 \\
& D \leftarrow 0 \} \\
{ & D \leftarrow D + 1}$
\end{align*}\]

Figure 3.128: Automaton for the \texttt{ALL_EQUAL_WIDTH_INCREASING_SEQUENCE} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \texttt{INCREASING_SEQUENCE} pattern.
3.65  ALL_EQUAL_WIDTH_INCREMENTING_TERRACE

DESCRIPTION AUTOMATON

Origin
Based on the INCREASING_TERRACE pattern.

Constraint
ALL_EQUAL_WIDTH_INCREMENTING_TERRACE(VARIABLES)

Argument
VARIABLES : collection(var–dvar)

Restriction
required(VARIABLES, var)

Purpose
Succeeds if the width of all occurrences of the INCREASING_TERRACE pattern in the time-series given by the VARIABLES collection are the same.
An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression $<=$ $<=$ $<=$.
Assume that the occurrence of the pattern INCREASING_TERRACE starts at position $i$ and ends at position $j$. The feature WIDTH computes the value $j - i$.

Example

$(1, 2, 2, 3, 4, 4, 4, 2, 5, 5, 6, 6, 6)$

Figure 3.129 provides an example where the ALL_EQUAL_WIDTH_INCREMENTING_TERRACE ($(1, 2, 2, 3, 4, 4, 4, 2, 5, 5, 6, 6, 6)$) constraint holds.

![Diagram](#)

Figure 3.129: Illustrating the ALL_EQUAL_WIDTH_INCREMENTING_TERRACE constraint of the Example slot
<table>
<thead>
<tr>
<th>Typical</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td>(\text{range}(\text{VARIABLES}.\text{var}) &gt; 2)</td>
</tr>
</tbody>
</table>
Figure 3.130 depicts the automaton associated with the constraint `ALL_EQUAL_WIDTH_INCREASING_TERRACE`.

Figure 3.130: Automaton for the `ALL_EQUAL_WIDTH_INCREASING_TERRACE` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `INCREASING_TERRACE` pattern.
3.66  **ALL_EQUAL_WIDTH_INFLEXION**

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the INFLEXION pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>ALL_EQUAL_WIDTH_INFLEXION(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var–dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES, var)</td>
</tr>
<tr>
<td><strong>Purpose</strong></td>
<td>Succeeds if the width of all occurrences of the INFLEXION pattern in the time-series given by the VARIABLES collection are the same. An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression $&lt; (&lt;</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>$\langle 2,2,6,6,5,3,4,4,5,3,3,4,6 \rangle$</td>
</tr>
</tbody>
</table>

Figure 3.131 provides an example where the **ALL_EQUAL_WIDTH_INFLEXION** constraint holds.

Figure 3.131: Illustrating the **ALL_EQUAL_WIDTH_INFLEXION** constraint of the **Example** slot

**Typical**

$|\text{VARIABLES}| > 2$

$\text{range}(\text{VARIABLES} \cdot \text{var}) > 1$
Figure 3.132 depicts the automaton associated with the constraint ALL_EQUAL_WIDTH_INFLEXION.

\[
\begin{align*}
C & \leftarrow X \\
D & \leftarrow 0 \\
F & \leftarrow X \\
R & \leftarrow 1
\end{align*}
\]

\[
\begin{align*}
R \land (F = C) \\
& \quad \leq \{ D \leftarrow D + 1 \} \\
& \quad \geq \{ D \leftarrow D + 1 \}
\end{align*}
\]

Figure 3.132: Automaton for the ALL_EQUAL_WIDTH_INFLEXION constraint obtained by applying decoration Table 2.36 to the seed transducer of the INFLEXION pattern (transition \( r \to t \) has the same accumulators updates as transition \( t \to r \))
3.67 **ALL_EQUAL_WIDTH_PEAK**

**DESCRIPTION**

- **Origin**: Based on the PEAK pattern.

- **Constraint**: \( \text{ALL_EQUAL_WIDTH_PEAK} (\text{VARIABLES}) \)

- **Argument**: \( \text{VARIABLES} : \text{collection} (\text{var–dvar}) \)

- **Restriction**: \( \text{required} (\text{VARIABLES} \cdot \text{var}) \)

**Purpose**

Succeeds if the width of all occurrences of the PEAK pattern in the time-series given by the \( \text{VARIABLES} \) collection are the same.

An occurrence of the pattern PEAK is the *maximal* subsequence which matches the regular expression \(< (= | <)* (>| =)* >\).

Assume that the occurrence of the pattern PEAK starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

**Example**

\( (1, 2, 2, 7, 7, 1, 2, 4, 5, 7, 1, 2, 3, 6, 7, 1) \)

Figure 3.133 provides an example where the \( \text{ALL_EQUAL_WIDTH_PEAK} ([1, 2, 2, 7, 7, 1, 2, 4, 5, 7, 1, 2, 3, 6, 7, 1]) \) constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 2 \]

\[ \text{range} (\text{VARIABLES} \cdot \text{var}) > 1 \]
Figure 3.133: Illustrating the ALL_EQUAL_WIDTH_PEAK constraint of the Example slot
Automaton

Figure 3.134 depicts the automaton associated with the constraint \texttt{ALL\_EQUAL\_WIDTH\_PEAK}.

\begin{align*}
\{ C \leftarrow X \} & \rightarrow \{ R \land (F = C) \} \\
\{ D \leftarrow 0 \} & \rightarrow \{ F \leftarrow X \} \\
\{ R \leftarrow 1 \} & \rightarrow \{ R \rightarrow R \land (F = C) \}
\end{align*}

\begin{align*}
\{ D \leftarrow D + 1 \} & \rightarrow \{ C \leftarrow C + D + 1 \} \\
\{ D \leftarrow 0 \} & \rightarrow \{ C \leftarrow C + D + 1 \} \\
\{ F \leftarrow C \} & \rightarrow \{ D \leftarrow D + 1 \}
\end{align*}

Figure 3.134: Automaton for the \texttt{ALL\_EQUAL\_WIDTH\_PEAK} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \texttt{PEAK} pattern.
3.68  ALL_EQUAL_WIDTH_PLAIN

**Origin**
Based on the PLAIN pattern.

**Constraint**
ALL_EQUAL_WIDTH_PLAIN(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES, var)

Succeeds if the width of all occurrences of the PLAIN pattern in the time-series given by the VARIABLES collection are the same.

**Purpose**
An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression \( \geq^* \leq \).
Assume that the occurrence of the pattern PLAIN starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

**Example**
\[
((1, 6, 3, 3, 7, 6, 6, 3, 3, 5, 5, 4, 3, 3, 6, 3))
\]

Figure 3.135 provides an example where the ALL_EQUAL_WIDTH_PLAIN \([(1, 6, 3, 3, 7, 6, 6, 3, 3, 5, 5, 4, 3, 3, 6, 3)]\) constraint holds.

**Typical**
\[
|\text{VARIABLES}| > 2
range(\text{VARIABLES}.var) > 1
\]
Figure 3.135: Illustrating the ALL_EQUAL_WIDTH.PLAIN constraint of the Example slot
Automaton  Figure 3.136 depicts the automaton associated with the constraint ALL_EQUAL_WIDTH_PLAIN.

Figure 3.136: Automaton for the ALL_EQUAL_WIDTH_PLAIN constraint obtained by applying decoration Table 2.36 to the seed transducer of the PLAIN pattern.
### 3.69 ALL_EQUAL_WIDTH_PLATEAU

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th><strong>Automaton</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the PLATEAU pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>ALL_EQUAL_WIDTH_PLATEAU(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var–dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES, var)</td>
</tr>
<tr>
<td><strong>Purpose</strong></td>
<td>Succeeds if the width of all occurrences of the PLATEAU pattern in the time-series given by the VARIABLES collection are the same. An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression (&lt;=*). Assume that the occurrence of the pattern PLATEAU starts at position i and ends at position j. The feature WIDTH computes the value (j - i).</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>((7, 2, 5, 5, 1, 2, 2, 5, 5, 3, 3, 4, 5, 5, 2, 5))</td>
</tr>
</tbody>
</table>

Figure 3.137 provides an example where the ALL_EQUAL_WIDTH_PLATEAU \([(7, 2, 5, 5, 1, 2, 2, 5, 5, 3, 3, 4, 5, 5, 2, 5)]\) constraint holds.

**Typical**

\[|\text{VARIABLES}| > 2\]
\[\text{range}(\text{VARIABLES}.\text{var}) > 1\]
Figure 3.137: Illustrating the `ALL_EQUAL_WIDTH_PLATEAU` constraint of the `Example` slot
Figure 3.138 depicts the automaton associated with the constraint \textsc{ALL}_\textsc{EQUAL}_\textsc{WIDTH}_\textsc{PLATEAU}.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{automaton.png}
\caption{Automaton for the \textsc{ALL}_\textsc{EQUAL}_\textsc{WIDTH}_\textsc{PLATEAU} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textsc{PLATEAU} pattern}
\end{figure}
3.70 ALL_EQUAL_WIDTH_PROPERPLAIN

DESCRIPTION

Origin
Based on the PROPERPLAIN pattern.

Constraint
ALL_EQUAL_WIDTH_PROPERPLAIN(VARIABLES)

Argument
VARIABLES : collection(var − dvar)

Restriction
required(VARIABLES, var)

Purpose
Succeeds if the width of all occurrences of the PROPERPLAIN pattern in the time-series given by the VARIABLES collection are the same.

An occurrence of the pattern PROPERPLAIN is the maximal subsequence which matches the regular expression > = + <.

Assume that the occurrence of the pattern PROPERPLAIN starts at position i and ends at position j. The feature WIDTH computes the value j − i.

Example
((2, 7, 3, 3, 6, 6, 3, 7, 3, 3, 5, 6, 5, 3, 3, 5))

Figure 3.139 provides an example where the ALL_EQUAL_WIDTH_PROPERPLAIN ([2, 7, 3, 3, 6, 6, 3, 7, 3, 3, 5, 6, 5, 3, 3, 5]) constraint holds.

Figure 3.139: Illustrating the ALL_EQUAL_WIDTH_PROPERPLAIN constraint of the Example slot
Typical

|VARIABLES| > 3
range(VARIABLES.var) > 1
Figure 3.140 depicts the automaton associated with the constraint ALL_EQUAL_WIDTH_PROPER_PLAIN.

Figure 3.140: Automaton for the ALL_EQUAL_WIDTH_PROPER_PLAIN constraint obtained by applying decoration Table 2.36 to the seed transducer of the PROPER_PLAIN pattern
### 3.71 ALL_EQUAL_WIDTH_PROPER_PLATEAU

#### DESCRIPTION

**Origin**
Based on the `PROPER_PLATEAU` pattern.

**Constraint**

\[ \text{ALL_EQUAL_WIDTH_PROPER_PLATEAU}(\text{VARIABLES}) \]

**Argument**

\[ \text{VARIABLES} : \text{collection}(\text{var} - \text{dvar}) \]

**Restriction**

\[ \text{required(}\text{VARIABLES.}\text{var}) \]

Purpose

Succeeds if the width of all occurrences of the `PROPER_PLATEAU` pattern in the time-series given by the `VARIABLES` collection are the same.

An occurrence of the pattern `PROPER_PLATEAU` is the *maximal* subsequence which matches the regular expression `<=+>`.

Assume that the occurrence of the pattern `PROPER_PLATEAU` starts at position \( i \) and ends at position \( j \). The feature `WIDTH` computes the value \( j - i \).

#### Example

\[ ([7, 1, 5, 5, 2, 2, 5, 1, 5, 5, 3, 2, 3, 5, 5, 3]) \]

Figure 3.141 provides an example where the `ALL_EQUAL_WIDTH_PROPER_PLATEAU` \(([7, 1, 5, 5, 2, 2, 5, 1, 5, 5, 3, 2, 3, 5, 5, 3]) \) constraint holds.

#### Typical

\[ |\text{VARIABLES}| > 3 \]

\[ \text{range(}\text{VARIABLES.}\text{var}) > 1 \]
Figure 3.141: Illustrating the ALL_EQUAL_WIDTH_PROPER_PLATEAU constraint of the Example slot
Figure 3.142 depicts the automaton associated with the constraint $\text{ALL_EQUAL_WIDTH_PROPER_PLATEAU}$. 

Figure 3.142: Automaton for the $\text{ALL_EQUAL_WIDTH_PROPER_PLATEAU}$ constraint obtained by applying decoration Table 2.36 to the seed transducer of the $\text{PROPER_PLATEAU}$ pattern
3.72 \textbf{ALL\_EQUAL\_WIDTH\_STEADY\_SEQUENCE}

\textbf{DESCRIPTION}

\textbf{Origin} \quad Based on the \texttt{STEADY\_SEQUENCE} pattern.

\textbf{Constraint} \quad \texttt{ALL\_EQUAL\_WIDTH\_STEADY\_SEQUENCE(VARIABLES)}

\textbf{Argument} \quad \texttt{VARIABLES} : \texttt{collection(var\_dvar)}

\textbf{Restriction} \quad \texttt{required(VARIABLES, var)}

\textbf{Purpose} \quad Succeeds if the width of all occurrences of the \texttt{STEADY\_SEQUENCE} pattern in the time-series given by the \texttt{VARIABLES} collection are the same.

An occurrence of the pattern \texttt{STEADY\_SEQUENCE} is the \textit{maximal} subsequence which matches the regular expression \texttt{=}^+.

Assume that the occurrence of the pattern \texttt{STEADY\_SEQUENCE} starts at position \(i\) and ends at position \(j\). The feature \texttt{WIDTH} computes the value \(j - i + 2\).

\textbf{Example} \quad \((3, 5, 5, 5, 3, 1, 4, 4, 6, 5, 3, 2, 2, 5)\)

Figure 3.143 provides an example where the \texttt{ALL\_EQUAL\_WIDTH\_STEADY\_SEQUENCE} \((3, 5, 5, 5, 3, 1, 4, 4, 6, 5, 3, 2, 2, 5)\) constraint holds.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure3143}
\caption{Illustrating the \texttt{ALL\_EQUAL\_WIDTH\_STEADY\_SEQUENCE} constraint of the \textbf{Example} slot}
\end{figure}
Typical | $|\text{VARIABLES}| > 1$
Automaton

Figure 3.144 depicts the automaton associated with the constraint \texttt{ALL_EQUAL_WIDTH_STEADY_SEQUENCE}.

\begin{align*}
&\\{ \begin{array}{l}
C \leftarrow X \\
D \leftarrow 0 \\
F \leftarrow X \\
R \leftarrow 1
\end{array} \}\quad \xrightarrow{\text{\textcircled{S}}}
\\&\{ \begin{array}{l}
D \leftarrow 0 \\
F \leftarrow C \\
R \leftarrow R \land (F = C)
\end{array} \}\quad \xrightarrow{\text{\textcircled{S}}}
\\&\{ \begin{array}{l}
C \leftarrow D + 2 \\
D \leftarrow 0
\end{array} \}\quad \xrightarrow{\text{\textcircled{S}}}
\\&\{ \begin{array}{l}
C \leftarrow C + D + 1 \\
D \leftarrow 0
\end{array} \}
\end{align*}

Figure 3.144: Automaton for the \texttt{ALL_EQUAL_WIDTH_STEADY_SEQUENCE} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \texttt{STEADY_SEQUENCE} pattern.
3.73 ALL_EQUAL_WIDTH_STRICTLY_DECREASING_SEQUENCE

**Description**

Based on the `STRICTLY_DECREASING_SEQUENCE` pattern.

**Constraint**

`ALL_EQUAL_WIDTH_STRICTLY_DECREASING_SEQUENCE(VARIABLES)`

**Argument**

`VARIABLES : collection(var−dvar)`

**Restriction**

`required(VARIABLES, var)`

Succeeds if the width of all occurrences of the `STRICTLY_DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are the same. An occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` is the *maximal* subsequence which matches the regular expression `>+`. Assume that the occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` starts at position $i$ and ends at position $j$. The feature `WIDTH` computes the value $j - i + 2$.

**Example**

`((4, 3, 2, 2, 4, 4, 6, 6, 4, 1, 1, 2, 2, 4, 3, 1))`

Figure 3.145 provides an example where the `ALL_EQUAL_WIDTH_STRICTLY_DECREASING_SEQUENCE` `((4, 3, 2, 2, 4, 4, 6, 6, 4, 1, 1, 2, 2, 4, 3, 1))` constraint holds.

![Diagram](image_url)

Figure 3.145: Illustrating the `ALL_EQUAL_WIDTH_STRICTLY_DECREASING_SEQUENCE` constraint of the Example slot.
Typical

| VARIABLES | > 1
| range(VARIABLES.var) | > 1
Automaton  Figure 3.146 depicts the automaton associated with the constraint \texttt{ALL\_EQUAL\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE}.

\[
\begin{align*}
\text{\{ } & C \leftarrow X \\
& D \leftarrow 0 \\
& F \leftarrow X \\
& R \leftarrow 1 \\
\text{\{ } & \leq s \\
\end{align*}
\]

\[
\begin{align*}
\text{\{ } & \{ } \\
& D \leftarrow 0 \\
& F \leftarrow C \\
& R \leftarrow R \land (F = C) \\
\text{\{ } & \leq \text{\{ } \\
\end{align*}
\]

\[
\begin{align*}
\text{\{ } & \{ } \\
& C \leftarrow D + 2 \\
& D \leftarrow 0 \\
\text{\{ } & \geq \text{\{ } \\
\end{align*}
\]

\[
\begin{align*}
\text{\{ } & \{ } \\
& C \leftarrow C + D + 1 \\
& D \leftarrow 0 \\
\text{\{ } & > r \\
\end{align*}
\]

Figure 3.146: Automaton for the \texttt{ALL\_EQUAL\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \texttt{STRICTLY\_DECREASING\_SEQUENCE} pattern.
### ALL_EQUAL_WIDTH STRICTLY_INCREASING_SEQUENCE

**Origin**
Based on the STRICTLY_INCREASING_SEQUENCE pattern.

**Constraint**
ALL_EQUAL_WIDTH_STRICTLY_INCREASING_SEQUENCE(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES, var)

Succeeds if the width of all occurrences of the STRICTLY_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are the same.

An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal sub-sequence which matches the regular expression <+

Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature `WIDTH` computes the value \( j - i + 2 \).

**Example**
\[(6, 3, 3, 4, 5, 6, 6, 3, 1, 1, 2, 3, 5, 5, 4, 3)\]

Figure 3.147 provides an example where the ALL_EQUAL_WIDTH_STRICTLY_INCREASING_SEQUENCE ([6, 3, 3, 4, 5, 6, 6, 3, 1, 1, 2, 3, 5, 5, 4, 3]) constraint holds.

Figure 3.147: Illustrating the ALL_EQUAL_WIDTH_STRICTLY_INCREASING_SEQUENCE constraint of the **Example** slot
<table>
<thead>
<tr>
<th>Typical</th>
</tr>
</thead>
</table>
| $|\text{VARIABLES}| > 1$
| $\text{range}(\text{VARIABLES}.\text{var}) > 1$ |
Figure 3.148 depicts the automaton associated with the constraint `ALL_EQUAL_WIDTH_STRICTLY_INCREASING_SEQUENCE`.

Figure 3.148: Automaton for the `ALL_EQUAL_WIDTH_STRICTLY_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `STRICTLY_INCREASING_SEQUENCE` pattern
ALL_EQUAL_WIDTH_STRICTLY_INCREASING_SEQUENCE
3.75 ALL_EQUAL_WIDTH_SUMMIT

**DESCRIPTION**

**Constraint**

\begin{equation}
\text{ALL_EQUAL_WIDTH_SUMMIT}(\text{VARIABLES})
\end{equation}

**Argument**

\text{VARIABLES} : \text{collection}(\text{var} – \text{dvar})

**Restriction**

\text{required}(\text{VARIABLES}, \text{var})

**Purpose**

Succeeds if the width of all occurrences of the \text{SUMMIT} pattern in the time-series given by the \text{VARIABLES} collection are the same.

An occurrence of the pattern \text{SUMMIT} is the \textit{maximal} subsequence which matches the regular expression $(< | < = | < ) ( > | > = | > )^* >$.

Assume that the occurrence of the pattern \text{SUMMIT} starts at position $i$ and ends at position $j$. The feature \text{WIDTH} computes the value $j - i$.

**Example**

$((1, 3, 5, 4, 2, 6, 6, 2, 3, 5, 4, 1, 4, 5, 3, 2))$

Figure 3.149 provides an example where the \text{ALL_EQUAL_WIDTH_SUMMIT} $(1, 3, 5, 4, 2, 6, 6, 2, 3, 5, 4, 1, 4, 5, 3, 2)$ constraint holds.

![Diagram of Example](image-url)
Typical

| VARIABLES | > 2
\[\text{range}(\text{VARIABLES}.\text{var}) > 1\]
Figure 3.150 depicts the automaton associated with the constraint ALL_EQUAL_WIDTH_SUMMIT.

\[
\begin{align*}
C &\leftarrow X \\
D &\leftarrow 0 \\
F &\leftarrow X \\
R &\leftarrow 1 \\
\{C \leftarrow D + 1, D \leftarrow 0\} &\quad (<) \\
\{D \leftarrow D + 1\} &\quad (\geq) \\
\{D \leftarrow D + 1\} &\quad (=)
\end{align*}
\]

Figure 3.150: Automaton for the ALL_EQUAL_WIDTH_SUMMIT constraint obtained by applying decoration Table 2.36 to the seed transducer of the SUMMIT pattern (transition \( u \rightarrow r \) has the same accumulator update as transition \( r \rightarrow u \)).
### ALL_EQUAL_WIDTH_VALLEY

**Origin**
Based on the VALLEY pattern.

**Constraint**
\[ \text{ALL_EQUAL_WIDTH_VALLEY}(\text{VARIABLES}) \]

**Argument**
\[ \text{VARIABLES} : \text{collection}(\text{var} - \text{dvar}) \]

**Restriction**
\[ \text{required}(\text{VARIABLES}, \text{var}) \]

**Purpose**
Succeeds if the width of all occurrences of the VALLEY pattern in the time-series given by the VARIABLES collection are the same.

An occurrence of the pattern VALLEY is the *maximal* subsequence which matches the regular expression \[ > (= | >)^* (< | =)^* < \].

Assume that the occurrence of the pattern VALLEY starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

**Example**
\[ ((7, 6, 6, 1, 1, 7, 6, 4, 3, 1, 7, 6, 5, 2, 1, 7)) \]

Figure 3.151 provides an example where the ALL_EQUAL_WIDTH_VALLEY \([7, 6, 6, 1, 1, 7, 6, 4, 3, 1, 7, 6, 5, 2, 1, 7]) \] constraint holds.

**Typical**
\[ |\text{VARIABLES}| > 2 \]
\[ \text{range}(\text{VARIABLES.var}) > 1 \]
Figure 3.151: Illustrating the \texttt{ALL_EQUAL_WIDTH_VALLEY} constraint of the \texttt{Example} slot.
Figure 3.152 depicts the automaton associated with the constraint \textit{ALL\_EQUAL\_WIDTH\_VALLEY}.

Figure 3.152: Automaton for the \textit{ALL\_EQUAL\_WIDTH\_VALLEY} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textit{VALLEY} pattern
### 3.77 ALL_EQUAL_WIDTH_ZIGZAG

<table>
<thead>
<tr>
<th></th>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the ZIGZAG pattern.</td>
<td><img src="image" alt="ZIGZAG pattern" /></td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>ALL_EQUAL_WIDTH_ZIGZAG(VARIABLES)</td>
<td></td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var−dvar)</td>
<td></td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES, var)</td>
<td></td>
</tr>
</tbody>
</table>

**Purpose**

Succeeds if the width of all occurrences of the ZIGZAG pattern in the time-series given by the VARIABLES collection are the same.

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression $(<>)^+ (< | <>) | (<>)^+ (> | ><>).$

Assume that the occurrence of the pattern ZIGZAG starts at position $i$ and ends at position $j$. The feature WIDTH computes the value $j - i$.

**Example**

```
((3, 4, 1, 4, 3, 3, 4, 2, 6, 3, 2, 0, 3, 1, 2, 3))
```

Figure 3.153 provides an example where the ALL_EQUAL_WIDTH_ZIGZAG ([3, 4, 1, 4, 3, 3, 4, 2, 6, 3, 2, 0, 3, 1, 2, 3]) constraint holds.

**Typical**

- $|\text{VARIABLES}| > 3$
- $\text{range}(\text{VARIABLES}.\text{var}) > 1$
Figure 3.153: Illustrating the \texttt{ALL_EQUAL_WIDTH_ZIGZAG} constraint of the \textbf{Example} slot
Automaton Figure 3.154 depicts the automaton associated with the constraint ALL_EQUAL_WIDTH_ZIGZAG.
Figure 3.154: Automaton for the `ALL_EQUAL_WIDTH_ZIGZAG` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `ZIGZAG` pattern; (1) missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=$; (2) on transitions from $b, c, e, f$ to $s$ the accumulator $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the accumulator $F$ is reset to $C$, and the accumulator $R$ is updated wrt $C$ and $F$. 
3.78 **DECREASING_HEIGHT_DECREASING_TERRACE**

**Origin**
Based on the `DECREASING_TERRACE` pattern.

**Constraint**

\[
\text{DECREASING_HEIGHT\_DECREASING\_TERRACE}(\text{VARIABLES})
\]

**Argument**

\[
\text{VARIABLES : collection} (\text{var\_dvar})
\]

**Restriction**

\[
\text{required}(\text{VARIABLES, var})
\]

**Purpose**

Succeeds if the minima of the values in each occurrence of the `DECREASING_TERRACE` pattern in the time-series given by the `VARIABLES` collection are decreasing.

An occurrence of the pattern `DECREASING_TERRACE` is the maximal subsequence which matches the regular expression `\( >=^+ > \)`.

Assume that the occurrence of the pattern `DECREASING_TERRACE` starts at position \( i \) and ends at position \( j \). The feature `MIN` computes the minimum of the values from index \( i + 1 \) to index \( j \).

**Example**

\[
((7, 6, 6, 4, 3, 3, 2, 2, 4, 4, 6, 3, 3, 1, 1))
\]

Figure 3.155 provides an example where the `DECREASING_HEIGHT\_DECREASING\_TERRACE` constraint holds.

**Typical**

\[
\text{|VARIABLES|} > 3 \\
\text{range(VARIABLES.var)} > 2
\]
Figure 3.155: Illustrating the DECREASING_HEIGHT DECREASING_TERRACE constraint of the Example slot
Automaton

Figure 3.156 depicts the automaton associated with the constraint DECREASING_HEIGHT_DECREASING_TERRACE.

Figure 3.156: Automaton for the DECREASING_HEIGHT_DECREASING_TERRACE constraint obtained by applying decoration Table 2.36 to the seed transducer of the DECREASING_TERRACE pattern.
3.79 DECREASING_HEIGHT_INCREASING_TERRACE

**DESCRIPTION**

**Origin**
Based on the INCREASING_TERRACE pattern.

**Constraint**
DECREASING_HEIGHT_INCREASING_TERRACE(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES, var)

Succeeds if the minima of the values in each occurrence of the INCREASING_TERRACE pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**
An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression <\=^+=<.
Assume that the occurrence of the pattern INCREASING_TERRACE starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**
\((2, 5, 5, 6, 2, 3, 4, 4, 5, 3, 1, 2, 2, 3)\)

Figure 3.157 provides an example where the DECREASING_HEIGHT_INCREASING_TERRACE \(([2, 5, 5, 6, 2, 3, 4, 4, 5, 3, 1, 2, 2, 3])\) constraint holds.

**Typical**
\(|\text{VARIABLES}| > 3\
\text{range}(\text{VARIABLES.var}) > 2\)
Figure 3.157: Illustrating the DECREASING_HEIGHT_INCREASING_TERRACE constraint of the Example slot
Automaton Figure 3.158 depicts the automaton associated with the constraint DECREASING_HEIGHT_INCREASING_TERRACE.

Figure 3.158: Automaton for the DECREASING_HEIGHT_INCREASING_TERRACE constraint obtained by applying decoration Table 2.36 to the seed transducer of the INCREASING_TERRACE pattern
3.80 DECREASING_HEIGHT_PLAIN

**DESCRIPTION**

**Origin**
Based on the PLAIN pattern.

**Constraint**
DECREASING_HEIGHT_PLAIN(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES, var)

**Purpose**
Succeeds if the minima of the values in each occurrence of the PLAIN pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression \( >=^*< \).

Assume that the occurrence of the pattern PLAIN starts at position \( i \) and ends at position \( j \). The feature MIN computes the minimum of the values from index \( i + 1 \) to index \( j \).

**Example**

\((2, 3, 6, 5, 5, 7, 6, 6, 4, 5, 5, 4, 3, 6, 6, 3)\)

Figure 3.159 provides an example where the DECREASING_HEIGHT_PLAIN \(([2, 3, 6, 5, 5, 7, 6, 6, 4, 5, 5, 4, 3, 6, 6, 3])\) constraint holds.

![Illustration of DECREASING_HEIGHT_PLAIN constraint](image)

Figure 3.159: Illustrating the DECREASING_HEIGHT_PLAIN constraint of the **Example** slot
Typical

|VARIABLES| > 2
\[\text{range}(\text{VARIABLES}.\text{var}) > 1\]
DECREASING_HEIGHTPLAIN

Automaton

Figure 3.160 depicts the automaton associated with the constraint DECREASING_HEIGHTPLAIN.

<image of automaton>

Figure 3.160: Automaton for the DECREASING_HEIGHTPLAIN constraint obtained by applying decoration Table 2.36 to the seed transducer of the PLAIN pattern.
3.81 DECREASING_HEIGHT_PLATEAU

Description

Based on the PLATEAU pattern.

Constraint

DECREASING_HEIGHT_PLATEAU(VARIABLES)

Argument

VARIABLES : collection(var−dvar)

Restriction

required(VARIABLES, var)

Purpose

Succeeds if the minima of the values in each occurrence of the PLATEAU pattern in the time-series given by the VARIABLES collection are decreasing. An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression \(<^{*}\)\).

Assume that the occurrence of the pattern PLATEAU starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\).

Example

\(((5, 2, 2, 5, 5, 4, 3, 3, 4, 2, 2, 1, 3, 2, 5, 7))\)

Figure 3.161 provides an example where the DECREASING_HEIGHT_PLATEAU (\([5, 2, 2, 5, 5, 4, 3, 3, 4, 2, 2, 1, 3, 2, 5, 7]\)) constraint holds.

Typical

\(|\text{VARIABLES}| > 2\)

\(\text{range}(\text{VARIABLES,var}) > 1\)
Figure 3.161: Illustrating the `DECREASING_HEIGHT_PLATEAU` constraint of the `Example` slot
Automaton

Figure 3.162 depicts the automaton associated with the constraint \textit{DECREASING\_HEIGHT\_PLATEAU}.

\[
\begin{align*}
C & \leftarrow \min(D, \forall \var{r}_i) \\
D & \leftarrow +\infty \\
F & \leftarrow \min(D, \forall \var{r}_i) \\
R & \leftarrow R \land (F \geq \min(D, \forall \var{r}_i)) \\
\{ D \leftarrow \min(D, \forall \var{r}_i) \} & \geq \{ D \leftarrow +\infty \} \\
\{ D \leftarrow +\infty \} & < \\
\{ D \leftarrow +\infty \} & < \\
\end{align*}
\]

Figure 3.162: Automaton for the \textit{DECREASING\_HEIGHT\_PLATEAU} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textit{PLATEAU} pattern.
3.82 DECREASING_HEIGHT_PROPERPLAIN

**Description**

Based on the PROPERPLAIN pattern.

**Constraint**

DECREASING_HEIGHT_PROPERPLAIN(VARIABLES)

**Argument**

VARIABLES : collection(var–dvar)

**Restriction**

required(VARIABLES, var)

Succeeds if the minima of the values in each occurrence of the PROPERPLAIN pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**

An occurrence of the pattern PROPERPLAIN is the maximal subsequence which matches the regular expression > = + <.

Assume that the occurrence of the pattern PROPERPLAIN starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i + 1 to index j.

**Example**

\((2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5)\)

Figure 3.163 provides an example where the DECREASING_HEIGHT_PROPERPLAIN \((2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5)\) constraint holds.

**Typical**

\(|\text{VARIABLES}| > 3\)

\(\text{range}(\text{VARIABLES}.\text{var}) > 1\)
Figure 3.163: Illustrating the DECREASING_HEIGHT_PROPER_PLAIN constraint of the Example slot.
Automaton Figure 3.164 depicts the automaton associated with the constraint DECREASING_HEIGHT_PROPERPLAIN.

Figure 3.164: Automaton for the DECREASING_HEIGHT_PROPERPLAIN constraint obtained by applying decoration Table 2.36 to the seed transducer of the PROPERPLAIN pattern.
### 3.83 DECREASING_HEIGHT PROPER PLATEAU

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the PROPER PLATEAU pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>DECREASING_HEIGHT_PROPER_PLATEAU(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var–dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES, var)</td>
</tr>
</tbody>
</table>

Succeeds if the minima of the values in each occurrence of the PROPER PLATEAU pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**
An occurrence of the pattern PROPER PLATEAU is the maximal subsequence which matches the regular expression \(<=+>\).
Assume that the occurrence of the pattern PROPER_PLATEAU starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**
\[(3, 5, 5, 5, 3, 2, 3, 4, 4, 1, 5, 2, 3, 3, 1, 7)\]

Figure 3.165 provides an example where the DECREASING_HEIGHT_PROPER_PLATEAU ([3, 5, 5, 5, 3, 2, 3, 4, 4, 1, 5, 2, 3, 3, 1, 7]) constraint holds.

**Typical**
- \(|\text{VARIABLES}| > 3\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)
Figure 3.165: Illustrating the \texttt{DECREASING\_HEIGHT\_PROPER\_PLATEAU} constraint of the \texttt{Example} slot
Automaton Figure 3.166 depicts the automaton associated with the constraint DECREASING_HEIGHT_PROPER_PLATEAU.

Figure 3.166: Automaton for the DECREASING_HEIGHT_PROPER_PLATEAU constraint obtained by applying decoration Table 2.36 to the seed transducer of the PROPER_PLATEAU pattern.
### 3.84 DECREASING_HEIGHT_STEADY

#### DESCRIPTION

**Origin**
Based on the **STEADY** pattern.

**Constraint**
`DECREASING_HEIGHT_STEADY(VARIABLES)`

**Argument**
`VARIABLES : collection(var−dvar)`

**Restriction**
`required(VARIABLES, var)`

Succeeds if the minima of the values in each occurrence of the **STEADY** pattern in the time-series given by the `VARIABLES` collection are decreasing.

#### Purpose

An occurrence of the pattern **STEADY** is the subsequence which matches the regular expression `=`. Assume that the occurrence of the pattern **STEADY** starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i` to index `j + 1`.

**Example**

```
((3, 6, 6, 2, 7, 5, 5, 6, 5, 5, 3, 3, 7))
```

Figure 3.167 provides an example where the `DECREASING_HEIGHT_STEADY` constraint holds.

---

**Figure 3.167:** Illustrating the `DECREASING_HEIGHT_STEADY` constraint of the **Example** slot.
| Typical | $|\text{VARIABLES}| > 1$ |
Automaton Figure 3.168 depicts the automaton associated with the constraint `DECREASING_HEIGHT_STEADY`.

Figure 3.168: Automaton for the `DECREASING_HEIGHT_STEADY` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `STEADY` pattern.
3.85 **DECREASING_HEIGHT_STEADY_SEQUENCE**

**DESCRIPTION**

**Origin**
Based on the **STEADY_SEQUENCE** pattern.

**Constraint**
**DECREASING_HEIGHT_STEADY_SEQUENCE(VARIABLES)**

**Argument**

VARIABLES : collection(var–dvar)

**Restriction**

required(VARIABLES.var)

**Purpose**

Succeeds if the minima of the values in each occurrence of the **STEADY_SEQUENCE** pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern **STEADY_SEQUENCE** is the maximal subsequence which matches the regular expression $=^\oplus$.

Assume that the occurrence of the pattern **STEADY_SEQUENCE** starts at position $i$ and ends at position $j$. The feature **MIN** computes the minimum of the values from index $i$ to index $j + 1$.

**Example**

$((4, 3, 2, 5, 5, 3, 2, 4, 4, 6, 5, 3, 2, 2, 6))$

Figure 3.169 provides an example where the **DECREASING_HEIGHT_STEADY_SEQUENCE** ($((4, 3, 2, 5, 5, 3, 2, 4, 4, 6, 5, 3, 2, 2, 6))$) constraint holds.

![Figure 3.169: Illustrating the DECREASING_HEIGHT_STEADY_SEQUENCE constraint of the Example slot](image-url)
Typical | $|\text{VARIABLES}| > 1$
Automaton

Figure 3.170 depicts the automaton associated with the constraint DECREASING_HEIGHT_STEADY_SEQUENCE.

\[
\begin{align*}
  C & \leftarrow +\infty \\
  D & \leftarrow +\infty \\
  F & \leftarrow +\infty \\
  R & \leftarrow 1
\end{align*}
\]

\[
\begin{align*}
  D & \leftarrow +\infty \\
  F & \leftarrow C \\
  R & \leftarrow R \land (F \geq C)
\end{align*}
\]

\[
\begin{align*}
  C & \leftarrow \min(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
  D & \leftarrow +\infty
\end{align*}
\]

\[
\begin{align*}
  C & \leftarrow \min(C, \min(D, \text{VAR}_{i+1})) \\
  D & \leftarrow +\infty
\end{align*}
\]

Figure 3.170: Automaton for the DECREASING_HEIGHT_STEADY_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the STEADY_SEQUENCE pattern
3.86  DECREASING_MAX_BUMP_ON_DECREASING_SEQUENCE

<table>
<thead>
<tr>
<th>Origin</th>
<th>Based on the BUMP_ON_DECREASING_SEQUENCE pattern.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constraint</td>
<td>DECREASING_MAX_BUMP_ON_DECREASING_SEQUENCE(VARIABLES)</td>
</tr>
<tr>
<td>Argument</td>
<td>VARIABLES : collection(var−dvar)</td>
</tr>
<tr>
<td>Restriction</td>
<td>required(VARIABLES, var)</td>
</tr>
<tr>
<td>Purpose</td>
<td>Succeeds if the maxima of the values in each occurrence of the BUMP_ON_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing.</td>
</tr>
<tr>
<td></td>
<td>An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence which matches the regular expression &gt;&gt;&lt;&lt;&gt;.</td>
</tr>
<tr>
<td></td>
<td>Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i + 2 to index j.</td>
</tr>
<tr>
<td>Example</td>
<td>((7, 6, 5, 6, 5, 4, 1, 2, 7, 5, 4, 2, 5, 4, 3, 3))</td>
</tr>
<tr>
<td>Typical</td>
<td></td>
</tr>
<tr>
<td></td>
<td>range(VARIABLES.var) &gt; 2</td>
</tr>
</tbody>
</table>

Figure 3.171 provides an example where the DECREASING_MAX_BUMP_ON_DECREASING_SEQUENCE constraint holds.
DECREASING_MAX_BUMP_ON_DECREASING_SEQUENCE

Figure 3.171: Illustrating the DECREASING_MAX_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 3.172 depicts the automaton associated with the constraint `DECREASING_MAX_BUMP_ON_DECREASING_SEQUENCE`.

Figure 3.172: Automaton for the `DECREASING_MAX_BUMP_ON_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `BUMP_ON_DECREASING_SEQUENCE` pattern.
3.87 DECREASING_MAX_DECREASING

**DESCRIPTION**

**Origin**
Based on the DECREASING pattern.

**Constraint**
DECREASING_MAX_DECREASING(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES, var)

Succeeds if the maxima of the values in each occurrence of the DECREASING pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**
An occurrence of the pattern DECREASING is the subsequence which matches the regular expression >.

Assume that the occurrence of the pattern DECREASING starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i\) to index \(j + 1\).

**Example**
\(([1, 1, 3, 4, 5, 7, 7, 5, 4, 4, 2, 2, 5, 5])\)

Figure 3.173 provides an example where the DECREASING_MAX_DECREASING \(([1, 1, 3, 4, 5, 7, 7, 5, 4, 4, 2, 2, 5, 5])\) constraint holds.

**Typical**
\(|\text{VARIABLES}| > 1\)
range(VARIABLES.var) > 1
DECREASING_MAX_DECREASING

Figure 3.173: Illustrating the DECREASING_MAX_DECREASING constraint of the Example slot
Automaton

Figure 3.174 depicts the automaton associated with the constraint DECREASING_MAX_DECREASING.

\[
\begin{align*}
C &\leftarrow +\infty \\
D &\leftarrow -\infty \\
F &\leftarrow +\infty \\
R &\leftarrow 1
\end{align*}
\]

Figure 3.174: Automaton for the \textit{DECREASING\_MAX\_DECREASING} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textit{DECREASING} pattern.
## 3.88 \texttt{DECREASING_MAX\_DECREASING\_SEQUENCE}

### Description

Origin
- Based on the \texttt{DECREASING\_SEQUENCE} pattern.

Constraint
- \texttt{DECREASING\_MAX\_DECREASING\_SEQUENCE(VARIABLES)}

Argument
- \texttt{VARIABLES : collection(var–dvar)}

Restriction
- \texttt{required(VARIABLES, var)}

Purpose
- Succeeds if the maxima of the values in each occurrence of the \texttt{DECREASING\_SEQUENCE} pattern in the time-series given by the \texttt{VARIABLES} collection are decreasing.

- An occurrence of the pattern \texttt{DECREASING\_SEQUENCE} is the maximal subsequence which matches the regular expression \( (> | =)* > | > \).

- Assume that the occurrence of the pattern \texttt{DECREASING\_SEQUENCE} starts at position \( i \) and ends at position \( j \). The feature \texttt{MAX} computes the maximum of the values from index \( i \) to index \( j + 1 \).

Example
- \((4, 7, 4, 2, 3, 4, 6, 5, 4, 3, 2, 3, 2)\)

Typical
- \( |\texttt{VARIABLES}| > 1 \)
- \( \texttt{range(VARIABLES.var)} > 1 \)

---

Figure 3.175 provides an example where the \texttt{DECREASING\_MAX\_DECREASING\_SEQUENCE} constraint holds.
Figure 3.175: Illustrating the \texttt{DECREASING\_MAX\_DECREASING\_SEQUENCE} constraint of the \texttt{Example} slot
Figure 3.176 depicts the automaton associated with the constraint \textsc{decreasing_max_decreasing_sequence}.

\begin{align*}
C & \leftarrow +\infty \\
D & \leftarrow -\infty \\
F & \leftarrow +\infty \\
R & \leftarrow 1
\end{align*}

Figure 3.176: Automaton for the \textsc{decreasing_max_decreasing_sequence} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textsc{decreasing_sequence} pattern.
3.89 DECREASING_MAX_DIP_ON_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th></th>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the DIP_ON_INCREASING_SEQUENCE pattern.</td>
<td></td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>DECREASING_MAX_DIP_ON_INCREASING_SEQUENCE(VARIABLES)</td>
<td></td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var−dvar)</td>
<td></td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES, var)</td>
<td></td>
</tr>
</tbody>
</table>

Succeeds if the maxima of the values in each occurrence of the DIP_ON_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**
An occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence which matches the regular expression <<><<<. Assume that the occurrence of the pattern DIP_ON_INCREASING_SEQUENCE starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i + 2 to index j.

**Example**

(⟨4, 5, 6, 0, 2, 4, 1, 2, 3, 2, 3, 4, 6, 5, 1⟩)

Figure 3.177 provides an example where the DECREASING_MAX_DIP_ON_INCREASING_SEQUENCE constraint holds.

**Typical**

| VARIABLES | > 5 |
| range(VARIABLES.var) | > 2 |
DECREASING\_MAX\_DIP\_ON\_INCREASING\_SEQUENCE

Figure 3.177: Illustrating the **DECREASING\_MAX\_DIP\_ON\_INCREASING\_SEQUENCE** constraint of the **Example** slot
Automaton

Figure 3.178 depicts the automaton associated with the constraint **DECREASING_MAX_DIP_ON_INCREASING_SEQUENCE**.

\[
\begin{align*}
\{ C \leftarrow +\infty \\ D \leftarrow -\infty \\ F \leftarrow +\infty \\ R \leftarrow 1 \}\quad \rightarrow \quad &\geq \quad < \\ &\quad \rightarrow \quad < \\ &\quad \rightarrow \quad < \\
\{ D \leftarrow \max(D, \text{VAR}_i) \} \\
\{ D \leftarrow -\infty \} \\
\{ D \leftarrow \max(D, \text{VAR}_i) \} \\
\{ D \leftarrow -\infty \} \\
\end{align*}
\]

Figure 3.178: Automaton for the **DECREASING_MAX_DIP_ON_INCREASING_SEQUENCE** constraint obtained by applying decoration Table 2.36 to the seed transducer of the **DIP_ON_INCREASING_SEQUENCE** pattern.
DECREASING_MAX_DIP_ON_INCREASING_SEQUENCE
3.90 **DECREASING_MAX_INCREASING**

**DESCRIPTION**

**Origin**

Based on the **INCREASING** pattern.

**Constraint**

DECREASING_MAX_INCREASING(VARIABLES)

**Argument**

VARIABLES : collection(var−dvar)

**Restriction**

required(VARIABLES, var)

Succeeds if the maxima of the values in each occurrence of the **INCREASING** pattern in the time-series given by the VARIABLES collection are decreasing. An occurrence of the pattern **INCREASING** is the subsequence which matches the regular expression <.

**Purpose**

Assume that the occurrence of the pattern **INCREASING** starts at position $i$ and ends at position $j$. The feature MAX computes the maximum of the values from index $i$ to index $j+1$.

**Example**

$((4, 4, 5, 5, 4, 3, 3, 1, 4, 2, 4, 3, 3, 1, 2, 1))$

Figure 3.179 provides an example where the DECREASING_MAX_INCREASING ($(4, 4, 5, 5, 4, 3, 3, 1, 4, 2, 4, 3, 3, 1, 2, 1)$) constraint holds.

Figure 3.179: Illustrating the DECREASING_MAX_INCREASING constraint of the Example slot
| Typical | $|\text{VARIABLES}| > 1$ |
|---------|-----------------|
|         | $\text{range(}\text{VARIABLES.var}) > 1$ |
Figure 3.180 depicts the automaton associated with the constraint DECREASING_MAX_INCREASING.

$$\begin{align*}
\begin{cases}
C & \leftarrow \infty \\
D & \leftarrow -\infty \\
F & \leftarrow +\infty \\
R & \leftarrow 1
\end{cases}
\end{align*}$$

\[ \overset{s}{\rightarrow} \]

\[ \overset{R \land (F \geq C)}{\rightarrow} \]

Figure 3.180: Automaton for the DECREASING_MAX_INCREASING constraint obtained by applying decoration Table 2.36 to the seed transducer of the INCREASING pattern.
3.91 DECREASING_MAX_INCREASING_SEQUENCE

**DESCRIPTION**

Based on the INCREASING_SEQUENCE pattern.

**ARGUMENT**

VARIABLES : collection(var-dvar)

**RESTRICTION**

required(VARIABLES, var)

Succeeds if the maxima of the values in each occurrence of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing.

**PURPOSE**

An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression $< (< | =)^* < | <$.

Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position $i$ and ends at position $j$. The feature MAX computes the maximum of the values from index $i$ to index $j + 1$.

**EXAMPLE**

$((3, 4, 5, 4, 3, 1, 2, 2, 4, 3, 3, 1, 3, 1))$

Figure 3.181 provides an example where the DECREASING_MAX_INCREASING_SEQUENCE ($(3, 4, 5, 4, 3, 1, 2, 2, 4, 3, 3, 1, 3, 1)$) constraint holds.

Figure 3.181: Illustrating the DECREASING_MAX_INCREASING_SEQUENCE constraint of the Example slot
Typical

\[ |\text{VARIABLES}| > 1 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]
Automaton

Figure 3.182 depicts the automaton associated with the constraint DECREASING_MAX_INCREASING_SEQUENCE.

Figure 3.182: Automaton for the DECREASING_MAX_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the INCREASING_SEQUENCE pattern
DECREASING_MAX_INCREASING_SEQUENCE
3.92 DECREASING_MAX_INFLEXION

**Description**
Based on the INFLEXION pattern.

**Constraint**
DECREASING_MAX_INFLEXION(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES, var)

Succeeds if the maxima of the values in each occurrence of the INFLEXION pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**
An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression < (< | =)∗ | > | (>| =)∗ | <. Assume that the occurrence of the pattern INFLEXION starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i + 1 to index j.

**Example**
\((\langle 3, 4, 3, 2, 2, 1, 1, 2, 2, 5, 5 \rangle)\)

Figure 3.183 provides an example where the DECREASING_MAX_INFLEXION (\([3, 4, 3, 2, 2, 1, 1, 2, 2, 5, 5]\)) constraint holds.

Figure 3.183: Illustrating the DECREASING_MAX_INFLEXION constraint of the Example slot
Typical

\[
|\text{VARIABLES}| > 2 \\
\text{range} (\text{VARIABLES}.\text{var}) > 1
\]
Figure 3.184 depicts the automaton associated with the constraint DECREEASING_MAX_INFLEXION.

\[
\begin{align*}
&\{ C \leftarrow +\infty \} \quad \{ D \leftarrow -\infty \} \quad \{ F \leftarrow +\infty \} \quad R \leftarrow 1 \\
&\{ D \leftarrow \max(D, \text{VAR}_i) \} \quad \{ \text{VAR}_i \leftarrow \min(\text{VAR}_i, D) \} \\
\end{align*}
\]

\[
\begin{align*}
&\{ C \leftarrow \max(D, \text{VAR}_i) \} \\
&\{ D \leftarrow -\infty \} \\
&\{ F \leftarrow \max(D, \text{VAR}_i) \} \\
&\{ R \leftarrow R \land (F \geq \max(D, \text{VAR}_i)) \}
\end{align*}
\]

Figure 3.184: Automaton for the DECREEASING_MAX_INFLEXION constraint obtained by applying decoration Table 2.36 to the seed transducer of the INFLEXION pattern (transition \( r \rightarrow t \) has the same accumulators updates as transition \( t \rightarrow r \))
DECREASING_MAX_INFLEXION
3.93 DECREASING_MAX_PEAK

**DESCRIPTION**

Origin: Based on the PEAK pattern.

Constraint: \( \text{DECREASING\_MAX\_PEAK}(\text{VARIABLES}) \)

Argument: \( \text{VARIABLES} : \text{collection}(\text{var–dvar}) \)

Restriction: \( \text{required}(\text{VARIABLES, var}) \)

Purpose: Succeeds if the maxima of the values in each occurrence of the PEAK pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression \( < (= | <)^* (> | =)^* > \).

Assume that the occurrence of the pattern PEAK starts at position \( i \) and ends at position \( j \). The feature MAX computes the maximum of the values from index \( i + 1 \) to index \( j \).

Example: \( (1, 6, 6, 6, 2, 5, 4, 3, 2, 3, 2, 1, 5, 5, 7) \)

Figure 3.185 provides an example where the \( \text{DECREASING\_MAX\_PEAK} \) constraint holds.

![Figure 3.185: Illustrating the DECREASING_MAX_PEAK constraint of the Example slot](image_url)
| Typical | $|\text{VARIABLES}| > 2$
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>\textit{range} (VARIABLES.var) &gt; 1</td>
</tr>
</tbody>
</table>
Automaton  Figure 3.186 depicts the automaton associated with the constraint DECREASING_MAX_PEAK.

$$\begin{aligned}
&\{ C \leftarrow +\infty \} \\
&\{ D \leftarrow -\infty \} \\
&\{ F \leftarrow +\infty \} \\
&\{ R \leftarrow 1 \}
\end{aligned}$$

$$\begin{aligned}
&\geq s \\
&\leq \{ D \leftarrow \max(D, V_{ikr}) \}
\end{aligned}$$

$$\begin{aligned}
&\leq R \\
&\geq t
\end{aligned}$$

$$\begin{aligned}
&\{ C \leftarrow \max(D, V_{ikr}) \} \\
&\{ D \leftarrow -\infty \}
\end{aligned}$$

$$\begin{aligned}
&\{ C \leftarrow \max(C, \max(D, V_{ikr})) \} \\
&\{ D \leftarrow -\infty \}
\end{aligned}$$

$$\begin{aligned}
&\leq \{ D \leftarrow \max(D, V_{ikr}) \}
\end{aligned}$$

$$\begin{aligned}
&\{ D \leftarrow -\infty \} \\
&\{ F \leftarrow C \} \\
&\{ R \leftarrow R \land (F \geq C) \}
\end{aligned}$$

Figure 3.186: Automaton for the DECREASING_MAX_PEAK constraint obtained by applying decoration Table 2.36 to the seed transducer of the PEAK pattern.
### DESCRIPTION

**Origin**
Based on the `STRICTLY_DECREASING_SEQUENCE` pattern.

**Constraint**
`DECREASING_MAX_STRICTLY_DECREASING_SEQUENCE(VARIABLES)`

**Argument**
`VARIABLES : collection(var−dvar)`

**Restriction**
`required(VARIABLES, var)`

**Purpose**
Succeeds if the maxima of the values in each occurrence of the `STRICTLY_DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are decreasing.

An occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` is the *maximal* sub-sequence which matches the regular expression `>^`. Assume that the occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` starts at position \( i \) and ends at position \( j \). The feature `MAX` computes the maximum of the values from index \( i \) to index \( j + 1 \).

**Example**

\[
(6, 4, 6, 5, 5, 3, 5, 4, 4, 3, 2, 3, 4)
\]

Figure 3.187 provides an example where the `DECREASING_MAX_STRICTLY_DECREASING_SEQUENCE` constraint holds.

**Typical**

\[
|VARIABLES| > 1 \\
range(VARIABLES.var) > 1
\]
Figure 3.187: Illustrating the DECREASING_MAX_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 3.188 depicts the automaton associated with the constraint DECREASING_MAX STRICTLY_DECREASING_SEQUENCE.

\[
\begin{align*}
C & \leftarrow +\infty \\
D & \leftarrow -\infty \\
F & \leftarrow +\infty \\
R & \leftarrow 1
\end{align*}
\]

\[
\begin{align*}
D & \leftarrow -\infty \\
F & \leftarrow C \\
R & \leftarrow R \land (F \geq C)
\end{align*}
\]

\[
\begin{align*}
C & \leftarrow \max(D, \text{VAR}_i, \text{VAR}_{i+1}) \\
D & \leftarrow -\infty
\end{align*}
\]

\[
\begin{align*}
C & \leftarrow \max(C, \max(D, \text{VAR}_{i+1})) \\
D & \leftarrow -\infty
\end{align*}
\]

Figure 3.188: Automaton for the DECREASING_MAX STRICTLY DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the STRICTLY DECREASING_SEQUENCE pattern.
DECREASING_MAX_STRICTLY_DECREASING_SEQUENCE

493
### 3.95 **DECREASING_MAX.Strictly_INCREASING_SEQUENCE**

**Description**

Based on the **Strictly_INCREASING_SEQUENCE** pattern.

**Constraint**

\[
\text{DECREASING_MAX.Strictly_INCREASING_SEQUENCE(VARIABLES)}
\]

**Argument**

\[
\text{VARIABLES : collection(var–dvar)}
\]

**Restriction**

\[
\text{required(VARIABLES, var)}
\]

**Purpose**

Succeeds if the maxima of the values in each occurrence of the **Strictly_INCREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern **Strictly_INCREASING_SEQUENCE** is the maximal sub-sequence which matches the regular expression \(<^+\).

Assume that the occurrence of the pattern **Strictly_INCREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
((3, 4, 5, 5, 4, 3, 1, 2, 3, 4, 3, 3, 1, 3, 1))
\]

Figure 3.189 provides an example where the **DECREASING_MAX.Strictly_INCREASING_SEQUENCE** \([(3, 4, 5, 5, 4, 3, 1, 2, 3, 4, 3, 3, 1, 3, 1)]\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1
\]

\[
\text{range}(\text{VARIABLES.var}) > 1
\]
Figure 3.189: Illustrating the DECREASING_MAX.Strictly_INCREMENTING_SEQUENCE constraint of the Example slot
Figure 3.190 depicts the automaton associated with the constraint \textsc{decreasing\_max\_strictly\_increasing\_sequence}.

\[
\begin{align*}
&\begin{cases}
C \leftarrow +\infty \\
D \leftarrow -\infty \\
F \leftarrow +\infty \\
R \leftarrow 1
\end{cases} \\
&\begin{cases}
D \leftarrow -\infty \\
F \leftarrow C \\
R \leftarrow R \wedge (F \geq C)
\end{cases}
\end{align*}
\]

Figure 3.190: Automaton for the \textsc{decreasing\_max\_strictly\_increasing\_sequence} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textsc{strictly\_increasing\_sequence} pattern.
3.96 **DECREASING_MAX_SUMMIT**

**Description**

Based on the **SUMMIT** pattern.

**Constraint**

DECREASING_MAX_SUMMIT(VARIABLES)

**Argument**

VARIABLES : collection(var-dvar)

**Restriction**

required(VARIABLES, var)

Succeeds if the maxima of the values in each occurrence of the **SUMMIT** pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**

An occurrence of the pattern **SUMMIT** is the *maximal* subsequence which matches the regular expression \(< | < (= | <)^* <)(>| > (= | >)^* >\).

Assume that the occurrence of the pattern **SUMMIT** starts at position \(i\) and ends at position \(j\). The feature \(\text{MAX}\) computes the maximum of the values from index \(i + 1\) to index \(j\).

**Example**

\(((1, 4, 4, 5, 4, 2, 6, 6, 2, 3, 5, 4, 1, 2, 4, 2))\)

Figure 3.191 provides an example where the DECREASING_MAX_SUMMIT constraint holds.

**Typical**

| VARIABLES | > 2  
| range(VARIABLES.var) | > 1  

Figure 3.191: Illustrating the DECREASING_MAX_SUMMIT constraint of the Example slot
Automaton

Figure 3.192 depicts the automaton associated with the constraint DECREASING_MAX_SUMMIT.

Figure 3.192: Automaton for the DECREASING_MAX_SUMMIT constraint obtained by applying decoration Table 2.36 to the seed transducer of the SUMMIT pattern (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$)
### 3.97 DECREASING_MAX_ZIGZAG

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the ZIGZAG pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>DECREASING_MAX_ZIGZAG(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var–dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES, var)</td>
</tr>
<tr>
<td><strong>Purpose</strong></td>
<td>Succeeds if the maxima of the values in each occurrence of the ZIGZAG pattern in the time-series given by the VARIABLES collection are decreasing. An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression ((&lt;&gt;)^+ (&lt;</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>((6, 5, 7, 4, 5, 1, 1, 6, 3, 6, 4, 5, 2, 7, 7))</td>
</tr>
<tr>
<td><strong>Typical</strong></td>
<td>(</td>
</tr>
</tbody>
</table>

Figure 3.193 provides an example where the DECREASING_MAX_ZIGZAG constraint holds.
Figure 3.193: Illustrating the DECREASING_MAX_ZIGZAG constraint of the Example slot
Automaton

Figure 3.194 depicts the automaton associated with the constraint DECREASING_MAX_ZIGZAG.
Figure 3.194: Automaton for the \texttt{DECREASING\_MAX\_ZIGZAG} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \texttt{ZIGZAG} pattern; (1) missing transitions from \(a\), \(b\), \(c\), \(d\), \(e\), \(f\) to \(s\) are labelled by \(=\); (2) on transitions from \(b\), \(c\), \(e\), \(f\) to \(s\) the accumulator \(D\) is reset to its initial value; (3) on transitions from \(c\), \(f\) to \(s\) the accumulator \(F\) is reset to \(C\), and the accumulator \(R\) is updated wrt \(C\) and \(F\).
### 3.98 DECREASING_MIN_BUMP_ON_DECREASING_SEQUENCE

**DESCRIPTION**

Based on the BUMP_ON_DECREASING_SEQUENCE pattern.

**AUTOMATON**

<table>
<thead>
<tr>
<th>Origin</th>
<th>Based on the BUMP_ON_DECREASING_SEQUENCE pattern.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constraint</td>
<td>DECREASING_MIN_BUMP_ON_DECREASING_SEQUENCE(VARIABLES)</td>
</tr>
<tr>
<td>Argument</td>
<td>VARIABLES : collection(var−dvar)</td>
</tr>
<tr>
<td>Restriction</td>
<td>required(VARIABLES, var)</td>
</tr>
</tbody>
</table>

Succeeds if the minima of the values in each occurrence of the BUMP_ON_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**

An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence which matches the regular expression >><<. Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 2\) to index \(j\).

**Example**

\[ ((7, 6, 5, 6, 4, 1, 1, 5, 4, 2, 6, 5, 4, 5)) \]

Figure 3.195 provides an example where the DECREASING_MIN_BUMP_ON_DECREASING_SEQUENCE \((7, 6, 5, 6, 4, 1, 1, 5, 4, 2, 6, 5, 4, 5))\) constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 5 \]

\[ \text{range}(\text{VARIABLES} \text{var}) > 2 \]
DECREASING_MIN_BUMP_ON_DECREASING_SEQUENCE

Figure 3.195: Illustrating the DECREASING_MIN_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot
Figure 3.196 depicts the automaton associated with the constraint DECREASING_MIN_BUMP_ON_DECREASING_SEQUENCE.

Automaton

Figure 3.196: Automaton for the DECREASING_MIN_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern.
DECREASING_MIN_BUMP_ON_DECREASING_SEQUENCE
3.99 DECREASING_MIN_DECREASING

**Origin**
Based on the DECREASING pattern.

**Constraint**
DECREASING_MIN_DECREASING(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES, var)

**Purpose**
Succeeds if the minima of the values in each occurrence of the DECREASING pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern DECREASING is the subsequence which matches the regular expression >.

Assume that the occurrence of the pattern DECREASING starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i\) to index \(j+1\).

**Example**

\[
((4, 7, 6, 4, 5, 5, 6, 4, 4, 3, 3, 2, 2, 3, 1))
\]

Figure 3.197 provides an example where the DECREASING_MIN_DECREASING \([(4, 7, 6, 4, 5, 5, 6, 4, 4, 3, 3, 2, 2, 3, 1)]\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1
\]

\[
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
Figure 3.197: Illustrating the DECREASING_MIN_DECREASING constraint of the Example slot
Automaton

Figure 3.198 depicts the automaton associated with the constraint DECREASING\_MIN\_DECREASING.

\[ C \leftarrow \min(D, \text{VAR}_i, \text{VAR}_{i+1}) \\
D \leftarrow +\infty \\
F \leftarrow \min(D, \text{VAR}_i, \text{VAR}_{i+1}) \\
R \leftarrow R \land (F \geq \min(D, \text{VAR}_i, \text{VAR}_{i+1})) \]

Figure 3.198: Automaton for the DECREASING\_MIN\_DECREASING constraint obtained by applying decoration Table 2.36 to the seed transducer of the DECREASING pattern
3.100 DECREASING_MIN_DECREASING_SEQUENCE

**DESCRIPTION**

Origin | Based on the DECREASING_SEQUENCE pattern.

Constraint | DECREASING_MIN_DECREASING_SEQUENCE(VARIABLES)

Argument | VARIABLES : collection(var–dvar)

Restriction | required(VARIABLES, var)

**Purpose**

Succeeds if the minima of the values in each occurrence of the DECREASINGSEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern DECREASINGSEQUENCE is the maximal subsequence which matches the regular expression > (>|=)*>|>.

Assume that the occurrence of the pattern DECREASINGSEQUENCE starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i to index j + 1.

Example | ((4, 7, 6, 4, 5, 5, 6, 5, 4, 4, 2, 2, 3, 1))

Figure 3.199 provides an example where the DECREASING_MIN_DECREASING_SEQUENCE ((4, 7, 6, 4, 5, 5, 6, 5, 4, 4, 2, 2, 3, 1)) constraint holds.

Typical | | | | |

$|VARIABLES| > 1$

$range(VARIABLES.var) > 1$
Figure 3.199: Illustrating the DECREASING_MIN_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 3.200 depicts the automaton associated with the constraint DECREASING_MIN_DECENDING_SEQUENCE.

\[
\begin{align*}
    &\begin{align*}
        C &\leftarrow +\infty \\
        D &\leftarrow +\infty \\
        F &\leftarrow +\infty \\
        R &\leftarrow 1
    \end{align*} \\
\end{align*}
\]

\[
\begin{align*}
    \leq s &\quad \leq \\
    \begin{cases}
        D &\leftarrow +\infty \\
        F &\leftarrow C \\
        R &\leftarrow R \land (F \geq C)
    \end{cases} \\
\end{align*}
\]

\[
\begin{align*}
    \geq t &\quad \begin{cases}
        C &\leftarrow \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
        D &\leftarrow +\infty
    \end{cases} \\
\end{align*}
\]

\[
\begin{align*}
    \{ D &\leftarrow \min(D, \text{VAR}_{i+1}) \} \\
    \{ C &\leftarrow \min(C, \min(D, \text{VAR}_{i+1})) \} \\
\end{align*}
\]

Figure 3.200: Automaton for the DECREASING_MIN_DECENDING_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the DECREASING_SEQUENCE pattern.
3.101 DECREASING_MIN_DIP_ON_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Origin</td>
</tr>
<tr>
<td>Constraint</td>
</tr>
<tr>
<td>Argument</td>
</tr>
<tr>
<td>Restriction</td>
</tr>
</tbody>
</table>

Succeeds if the minima of the values in each occurrence of the DIP_ON_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence which matches the regular expression <<><<<. Assume that the occurrence of the pattern DIP_ON_INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature \(\text{MIN}\) computes the minimum of the values from index \(i + 2\) to index \(j\).

Example

\[
(\langle 4, 5, 6, 1, 2, 4, 4, 1, 2, 3, 0, 3, 4, 6, 5, 1 \rangle)
\]

Figure 3.201 provides an example where the DECREASING_MIN_DIP_ON_INCREASING_SEQUENCE constraint holds.

Typical

\[
\text{|VARIABLES|} > 5 \\
\text{range}(\text{VARIABLES, var}) > 2
\]
Figure 3.201: Illustrating the DECREASING_MIN_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot
Automaton Figure 3.202 depicts the automaton associated with the constraint DECREASING_MIN_DIP_ON_INCREASING_SEQUENCE.

Figure 3.202: Automaton for the DECREASING_MIN_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern.
DECREASING_MIN_DIP_ON_INCREASING_SEQUENCE
3.102 DECREASING_MINGORGE

**Description**

Based on the GORGE pattern.

**Constraint**

DECREASING_MINGORGE(VARIABLES)

**Argument**

VARIABLES : \texttt{collection(var-dvar)}

**Restriction**

\text{required}(\text{VARIABLES}, \text{var})

**Purpose**

Succeeds if the minima of the values in each occurrence of the GORGE pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression \((\geq | > (\leq | >)^* >) (\leq | < (\leq | <)^* <))\).

Assume that the occurrence of the pattern GORGE starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**

\[(6, 2, 3, 4, 5, 6, 1, 1, 5, 4, 2, 3, 6, 1, 4, 5)\]

Figure 3.203 provides an example where the DECREASING_MINGORGE \(([6, 2, 3, 4, 5, 6, 1, 1, 5, 4, 2, 3, 6, 1, 4, 5])\) constraint holds.

![Graph illustarting the DECREASING_MINGORGE constraint](image-url)

Figure 3.203: Illustrating the DECREASING_MINGORGE constraint of the Example slot
### Typical

<table>
<thead>
<tr>
<th>VARIABLES</th>
<th>&gt; 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>range(VARIABLES.var)</td>
<td>&gt; 1</td>
</tr>
</tbody>
</table>
Figure 3.204 depicts the automaton associated with the constraint \textsc{decreasing\_min\_gorge}.

\begin{align*}
\{ C \leftarrow +\infty, \\
D \leftarrow +\infty, \\
F \leftarrow +\infty, \\
R \leftarrow 1 \} & \quad \leq S \\
\{ D \leftarrow +\infty \} & \quad \leq \quad \{ D \leftarrow \min(D, \text{VAR}_i) \}
\end{align*}

\begin{align*}
\{ D \leftarrow +\infty \} & \quad > \quad \{ C \leftarrow \min(D, \text{VAR}_i), \\
D \leftarrow +\infty \} \\
R \land (F \geq C) & \quad > \quad \{ D \leftarrow \min(D, \text{VAR}_i) \}
\end{align*}

\begin{align*}
\{ D \leftarrow +\infty \} & \quad = \quad \{ C \leftarrow \min(C, \min(D, \text{VAR}_i)), \\
D \leftarrow +\infty \} \\
\{ D \leftarrow \min(D, \text{VAR}_i) \} & \quad = \quad \{ D \leftarrow \min(D, \text{VAR}_i) \}
\end{align*}

\begin{align*}
\{ D \leftarrow \min(D, \text{VAR}_i) \} & \quad = \quad \{ D \leftarrow \min(D, \text{VAR}_i) \}
\end{align*}

Figure 3.204: Automaton for the \textsc{decreasing\_min\_gorge} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textsc{gorge} pattern (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$)
### 3.103 DECREASING_MIN_INCREASING

**DESCRIPTION**

Based on the INCREASING pattern.

**Constraint**

DECREASING_MIN_INCREASING(VARIABLES)

**Argument**

VARIABLES : collection(var−dvar)

**Restriction**

required(VARIABLES, var)

Succeeds if the minima of the values in each occurrence of the INCREASING pattern in the time-series given by the VARIABLES collection are decreasing. An occurrence of the pattern INCREASING is the subsequence which matches the regular expression <.

**Purpose**

Assume that the occurrence of the pattern INCREASING starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i\) to index \(j + 1\).

**Example**

\(((4, 4, 5, 5, 4, 3, 3, 2, 4, 3, 2, 2, 1, 3, 1))\)

Figure 3.205 provides an example where the DECREASING_MIN_INCREASING \(((4, 4, 5, 5, 4, 3, 3, 2, 4, 3, 2, 2, 1, 3, 1))\) constraint holds.

![Figure 3.205](image-url)

**Figure 3.205:** Illustrating the DECREASING_MIN_INCREASING constraint of the Example slot
| Typical | $|\text{VARIABLES}| > 1$  
|         | $\text{range}(\text{VARIABLES}.\text{var}) > 1$ |
Automaton Figure 3.206 depicts the automaton associated with the constraint DECREASING_MIN_INCREASING.

\[
\begin{array}{l}
C \leftarrow \min (\min (D, VAR_i), VAR_{i+1}) \\
D \leftarrow +\infty \\
F \leftarrow +\infty \\
R \leftarrow R \land (F \geq \min (\min (D, VAR_i), VAR_{i+1})) \\
\end{array}
\]

Figure 3.206: Automaton for the DECREASING_MIN_INCREASING constraint obtained by applying decoration Table 2.36 to the seed transducer of the INCREASING pattern.
3.104  DECREASING_MIN_INCREASING_SEQUENCE

**DESCRIPTION**

**Origin**
Based on the INCREASINGSEQUENCE pattern.

**Constraint**
DECREASING_MIN_INCREASING_SEQUENCE(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES, var)

**Purpose**
Succeeds if the minima of the values in each occurrence of the INCREASINGSEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern INCREASINGSEQUENCE is the maximal subsequence which matches the regular expression < (< | =)∗ < | <.

Assume that the occurrence of the pattern INCREASINGSEQUENCE starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i$ to index $j + 1$.

**Example**

$((3, 4, 5, 5, 4, 3, 1, 4, 3, 3, 2, 2, 1, 3, 1))$

Figure 3.207 provides an example where the DECREASING_MIN_INCREASING_SEQUENCE ($[(3, 4, 5, 5, 4, 3, 1, 4, 3, 3, 2, 2, 1, 3, 1)]$) constraint holds.

![Diagram of values and feature values](image)

Figure 3.207: Illustrating the DECREASING_MIN_INCREASING_SEQUENCE constraint of the Example slot
Typical

\[ |\text{VARIABLES}| > 1 \]

\[ \text{range} (\text{VARIABLES}\text{.var}) > 1 \]
Automaton

Figure 3.208 depicts the automaton associated with the constraint `DECREASING_MIN_INCREASING_SEQUENCE`.

\[
\begin{align*}
C & \leftarrow +\infty \\
D & \leftarrow +\infty \\
F & \leftarrow +\infty \\
R & \leftarrow 1 \\
\end{align*}
\]

Figure 3.208: Automaton for the `DECREASING_MIN_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `INCREASING_SEQUENCE` pattern
DECREASING_MIN_INCREASING_SEQUENCE
3.105  DECREASING_MIN_INFLEXION

DESCRIPTION  AUTOMATON

Origin  Based on the INFLEXION pattern.

Constraint  DECREASING_MIN_INFLEXION(VARIABLES)

Argument  VARIABLES : collection(var–dvar)

Restriction  required(VARIABLES, var)

Purpose  Succeeds if the minima of the values in each occurrence of the INFLEXION pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression \(<(\langle | =\rangle)^* \rangle > \rangle | \rangle =\rangle)^* <\). Assume that the occurrence of the pattern INFLEXION starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\).

Example  \((3, 4, 3, 2, 2, 1, 1, 2, 2, 5, 5)\)

Figure 3.209 provides an example where the DECREASING_MIN_INFLEXION \((3, 4, 3, 2, 2, 1, 1, 2, 2, 5, 5)\) constraint holds.

Figure 3.209: Illustrating the DECREASING_MIN_INFLEXION constraint of the Example slot
Typical

$|\text{VARIABLES}| > 2$

$\text{range}(\text{VARIABLES}\.\text{var}) > 1$
Automaton

Figure 3.210 depicts the automaton associated with the constraint DECREASING_MIN_INFLEXION.

\[
\begin{align*}
&\{ C \leftarrow +\infty \\
&\quad D \leftarrow +\infty \\
&\quad F \leftarrow +\infty \\
&\quad R \leftarrow 1 \\
&\{ D \leftarrow \min(D, \text{VAR}_i) \}
\end{align*}
\]

Figure 3.210: Automaton for the DECREASING_MIN_INFLEXION constraint obtained by applying decoration Table 2.36 to the seed transducer of the INFLEXION pattern (transition \( r \rightarrow t \) has the same accumulators updates as transition \( t \rightarrow r \))
### 3.106 DECREASING_MIN_STRICTLY_DECREASING_SEQUENCE

**DESCRIPTION**

**Origin**

Based on the STRICTLY_DECREASING_SEQUENCE pattern.

**Constraint**

DECREASING_MIN_STRICTLY_DECREASING_SEQUENCE(VARIABLES)

**Argument**

VARIABLES: collection(var−dvar)

**Restriction**

required(VARIABLES, var)

Succeeds if the minima of the values in each occurrence of the STRICTLY_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**

An occurrence of the pattern STRICTLY_DECREASING_SEQUENCE is the maximal sub-sequence which matches the regular expression >⁺. Assume that the occurrence of the pattern STRICTLY_DECREASING_SEQUENCE starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i to index j + 1.

**Example**

\[((6, 5, 6, 5, 4, 4, 3, 6, 2, 4, 3, 2, 3, 4))\]

Figure 3.211 provides an example where the DECREASING_MIN_STRICTLY_DECREASING_SEQUENCE ((6, 5, 6, 5, 4, 4, 3, 6, 2, 4, 3, 2, 3, 4)) constraint holds.

**Typical**

\[|\text{VARIABLES}| > 1\]

\[\text{range}(\text{VARIABLES}.\text{var}) > 1\]
Figure 3.211: Illustrating the `DECREASING_MIN STRICTLY DECREASING_SEQUENCE` constraint of the `Example` slot
Automaton

Figure 3.212 depicts the automaton associated with the constraint DECREASING_MIN_STRICTLY_DECREASING_SEQUENCE.

Figure 3.212: Automaton for the DECREASING_MIN_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern.
3.107  DECREASING_MIN_STRICTLY_INCREASING_SEQUENCE

**DESCRIPTION**

**Origin**
Based on the STRICTLY_INCREASING_SEQUENCE pattern.

**Constraint**
DECREASING_MIN_STRICTLY_INCREASING_SEQUENCE(VARIABLES)

**Argument**
VARIABLES : collection variance

**Restriction**
required(VARIABLES, var)

**Purpose**
Succeeds if the minima of the values in each occurrence of the STRICTLY_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression <+.

Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i to index j + 1.

**Example**
((3, 4, 5, 5, 4, 3, 1, 4, 3, 3, 2, 2, 1, 3, 1))

Figure 3.213 provides an example where the DECREASING_MIN_STRICTLY_INCREASING_SEQUENCE ((3, 4, 5, 5, 4, 3, 1, 4, 3, 3, 2, 2, 1, 3, 1)) constraint holds.

**Typical**
|VARIABLES| > 1
range(VARIABLES, var) > 1
Figure 3.213: Illustrating the DECREASING_MIN STRICTLY_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 3.214 depicts the automaton associated with the constraint \texttt{DECREASING\_MIN\_STRICLY\_INCREASING\_SEQUENCE}.

\begin{align*}
C & \leftarrow +\infty \\
D & \leftarrow +\infty \\
F & \leftarrow +\infty \\
R & \leftarrow 1 \\
\end{align*}

\begin{align*}
\{ D & \leftarrow +\infty \\
F & \leftarrow C \\
R & \leftarrow R \wedge (F \geq C) \}
\end{align*}

\begin{align*}
\{ C & \leftarrow \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
D & \leftarrow +\infty \\
\}
\end{align*}

\begin{align*}
\{ C & \leftarrow \min(C, \min(D, \text{VAR}_{i+1})) \\
D & \leftarrow +\infty \\
\}
\end{align*}

Figure 3.214: Automaton for the \texttt{DECREASING\_MIN\_STRICLY\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \texttt{STRICLY\_INCREASING\_SEQUENCE} pattern.
3.108 DECREASING_MIN_VALLEY

**DESCRIPTION AUTOMATON**

**Origin**
Based on the VALLEY pattern.

**Constraint**
DECREASING_MIN_VALLEY(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES.var)

**Purpose**
Succeeds if the minima of the values in each occurrence of the VALLEY pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern VALLEY is the *maximal* subsequence which matches the regular expression \( > (= | >)^* (< | =)^* < \).

Assume that the occurrence of the pattern VALLEY starts at position \( i \) and ends at position \( j \). The feature MIN computes the minimum of the values from index \( i + 1 \) to index \( j \).

**Example**
\[ ((1, 3, 3, 7, 6, 5, 4, 6, 6, 5, 4, 3, 6, 2, 2, 7)) \]

Figure 3.215 provides an example where the DECREASING_MIN_VALLEY ((1, 3, 3, 7, 6, 5, 4, 6, 6, 5, 4, 3, 6, 2, 2, 7)) constraint holds.

**Typical**
\[ |VARIABLES| > 2 \]
\[ range(VARIABLES.var) > 1 \]
Figure 3.215: Illustrating the **DECREASING_MIN_Valley** constraint of the **Example** slot.
Figure 3.216 depicts the automaton associated with the constraint `DECREASING_MIN_VALLEY`.

Automaton

Figure 3.216: Automaton for the `DECREASING_MIN_VALLEY` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `VALLEY` pattern.
DECREASING_MIN_VALLEY
### 3.109 DECREASING_MIN_ZIGZAG

<table>
<thead>
<tr>
<th><strong>DESCRIPTION</strong></th>
<th><strong>AUTOMATON</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the ZIGZAG pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>DECREASING_MIN_ZIGZAG(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var–dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES, var)</td>
</tr>
</tbody>
</table>

**Purpose**

Succeeds if the minima of the values in each occurrence of the ZIGZAG pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern ZIGZAG is the *maximal* subsequence which matches the regular expression: $(<>)^+(<> | (<>)^+ (> | >>))$. Assume that the occurrence of the pattern ZIGZAG starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i + 1$ to index $j$.

**Example**

$$\text{Example} = (6, 5, 7, 4, 5, 1, 1, 6, 3, 6, 4, 3, 5, 2, 7, 7)$$

Figure 3.217 provides an example where the DECREASING_MIN_ZIGZAG ($(6, 5, 7, 4, 5, 1, 1, 6, 3, 6, 4, 3, 5, 2, 7, 7)$) constraint holds.

**Typical**

$$|\text{VARIABLES}| > 3$$

$$\text{range}(\text{VARIABLES.var}) > 1$$
Figure 3.217: Illustrating the DECREASING_MIN_ZIGZAG constraint of the Example slot
Automaton  Figure 3.218 depicts the automaton associated with the constraint DECREASING_MIN_ZIGZAG.
Figure 3.218: Automaton for the DECREASING\_MIN\_ZIGZAG constraint obtained by applying decoration Table 2.36 to the seed transducer of the ZIGZAG pattern; (1) missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=;$ (2) on transitions from $b, c, e, f$ to $s$ the accumulator $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the accumulator $F$ is reset to $C$, and the accumulator $R$ is updated wrt $C$ and $F$. 
### 3.110 DECREASING_RANGE_DECREASING

<table>
<thead>
<tr>
<th>Origin</th>
<th>Based on the <strong>DECREASING</strong> pattern.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constraint</td>
<td><strong>DECREASING_RANGE_DECREASING</strong>(<code>VARIABLES</code>)</td>
</tr>
<tr>
<td>Argument</td>
<td><code>VARIABLES : collection(var−dvar)</code></td>
</tr>
<tr>
<td>Restriction</td>
<td><code>required(VARIABLES.var)</code></td>
</tr>
</tbody>
</table>

Succeeds if the differences between the largest and smallest value in each occurrence of the **DECREASING** pattern in the time-series given by the `VARIABLES` collection are decreasing.

** Purpose **
An occurrence of the pattern **DECREASING** is the subsequence which matches the regular expression `>`. Assume that the occurrence of the pattern **DECREASING** starts at position $i$ and ends at position $j$. The feature **RANGE** computes the range of the values from index $i$ to index $j + 1$.

** Example **
```
(⟨4, 8, 6, 4, 5, 5, 8, 6, 5, 5, 4, 6, 5⟩)
```
Figure 3.219 provides an example where the **DECREASING_RANGE_DECREASING** (`[4, 8, 6, 4, 5, 5, 8, 6, 5, 5, 4, 4, 6, 5]`) constraint holds.

** Typical **
```
|VARIABLES| > 1  
\text{range}(\text{VARIABLES.var}) > 1 
```
Figure 3.219: Illustrating the DECREASING_RANGE_DECREASING constraint of the Example slot
Automaton Figure 3.220 depicts the automaton associated with the constraint DECREASING_RANGE_DECREASING.

\[
\begin{align*}
&\{ C \leftarrow +\infty \\ & F \leftarrow +\infty \\ & H \leftarrow \text{VAR}_i \\ & R \leftarrow 1 \} \\
&\{ C \leftarrow |H - \text{VAR}_{i+1}| \\ & F \leftarrow |H - \text{VAR}_{i+1}| \\ & H \leftarrow \text{VAR}_{i+1} \\ & R \leftarrow R \land (F \geq |H - \text{VAR}_{i+1}|) \} \\
&\{ H \leftarrow \text{VAR}_{i+1} \} \\
&\{ R \land (F \geq C) \}
\end{align*}
\]

Figure 3.220: Automaton for the DECREASING_RANGE_DECREASING constraint obtained by applying decoration Table 2.47 to the seed transducer of the DECREASING pattern.
### 3.111 DECREASING\_RANGE\_DECREASING\_SEQUENCE

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the DECREASING_SEQUENCE pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>DECREASING_RANGE_DECREASING_SEQUENCE(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var–dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES, var)</td>
</tr>
</tbody>
</table>

Succeeds if the differences between the largest and smallest value in each occurrence of the DECREASING\_SEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern DECREASING\_SEQUENCE is the maximal subsequence which matches the regular expression \( (> | =)^* > | > \).

Assume that the occurrence of the pattern DECREASING\_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature RANGE computes the range of the values from index \( i \) to index \( j + 1 \).

**Example**

\[
(4, 7, 6, 1, 5, 5, 6, 5, 5, 4, 4, 2, 2, 3, 1)
\]

Figure 3.221 provides an example where the DECREASING\_RANGE\_DECREASING\_SEQUENCE \( ([4, 7, 6, 1, 5, 5, 6, 5, 5, 4, 4, 2, 2, 3, 1]) \) constraint holds.

**Typical**

\[
| \text{VARIABLES} | > 1 \\
| \text{range(VARIABLES.var)} & > 1
\]
Figure 3.221: Illustrating the DECREASING_RANGE_DECREASING_SEQUENCE constraint of the Example slot
Figure 3.222 depicts the automaton associated with the constraint `DECREASING RANGE DECREASING SEQUENCE`.

\begin{equation}
\begin{cases}
C \leftarrow +\infty \\
F \leftarrow +\infty \\
H \leftarrow \text{VAR}_1 \\
R \leftarrow 1
\end{cases}
\end{equation}

\begin{equation}
\begin{cases}
C \leftarrow |H - \text{VAR}_i| \\
H \leftarrow \text{VAR}_{i+1} \\
R \leftarrow R \land (F \geq C)
\end{cases}
\end{equation}

\begin{equation}
\begin{cases}
F \leftarrow |H - \text{VAR}_i| \\
H \leftarrow \text{VAR}_{i+1} \\
R \leftarrow R \land (F \geq C)
\end{cases}
\end{equation}

Figure 3.222: Automaton for the `DECREASING RANGE DECREASING SEQUENCE` constraint obtained by applying decoration Table 2.47 to the seed transducer of the `DECREASING SEQUENCE` pattern.
### 3.112 DECREASING\_RANGE\_INCREASING

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the <strong>INCREASING</strong> pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td><strong>DECREASING_RANGE_INCREASING</strong>(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : \textit{collection}(var−dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>\textit{required}(VARIABLES, var)</td>
</tr>
</tbody>
</table>

Succeeds if the differences between the largest and smallest value in each occurrence of the **INCREASING** pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**

An occurrence of the pattern **INCREASING** is the subsequence which matches the regular expression \(<\).
Assume that the occurrence of the pattern **INCREASING** starts at position \(i\) and ends at position \(j\). The feature **RANGE** computes the range of the values from index \(i\) to index \(j+1\).

**Example**

\[((4, 6, 8, 4, 4, 5, 6, 6, 7, 5, 5, 5, 6, 4))\]

Figure 3.223 provides an example where the **DECREASING\_RANGE\_INCREASING** ((4, 6, 8, 4, 4, 5, 6, 6, 7, 5, 5, 5, 6, 4)) constraint holds.

**Typical**

\[|\text{VARIABLES}| > 1\]
\[\text{range}(\text{VARIABLES.var}) > 1\]
Figure 3.223: Illustrating the DECREASING_RANGE_INCREASING constraint of the Example slot
Figure 3.224 depicts the automaton associated with the constraint `DECREASING RANGE INCREASING`.

Figure 3.224: Automaton for the `DECREASING RANGE INCREASING` constraint obtained by applying decoration Table 2.47 to the seed transducer of the `INCREASING` pattern.
### 3.113 DECREASING_RANGE_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th><strong>DESCRIPTION</strong></th>
<th><strong>AUTOMATON</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the INCREASING_SEQUENCE pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>DECREASING_RANGE_INCREASING_SEQUENCE(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var–dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES, var)</td>
</tr>
</tbody>
</table>

**Purpose**
Succeeds if the differences between the largest and smallest value in each occurrence of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression $(< | =)^* < | <$.
Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position $i$ and ends at position $j$. The feature RANGE computes the range of the values from index $i$ to index $j + 1$.

**Example**

$$((1, 4, 5, 5, 4, 3, 1, 4, 3, 3, 2, 2, 1, 3, 1))$$

Figure 3.225 provides an example where the DECREASING_RANGE_INCREASING_SEQUENCE constraint holds.

**Typical**

$$|\text{VARIABLES}| > 1$$

$$\text{range}(\text{VARIABLES.var}) > 1$$
Figure 3.225: Illustrating the **DECREASING_RANGE_INCREASING_SEQUENCE** constraint of the **Example** slot
Figure 3.226 depicts the automaton associated with the constraint \texttt{DECREASING\_RANGE\_INCREASING\_SEQUENCE}.

\[
\begin{array}{c}
\left\{ \begin{array}{l}
C \leftarrow +\infty \\
F \leftarrow +\infty \\
H \leftarrow \text{VAR}_1 \\
R \leftarrow 1
\end{array} \right. \\
\geq s
\end{array}
\]

\[
\Rightarrow \left\{ H \leftarrow \text{VAR}_{i+1} \right. \right. \\
\geq \left\{ \begin{array}{l}
F \leftarrow |H - \text{VAR}_i| \\
H \leftarrow \text{VAR}_{i+1} \\
R \leftarrow R \land (F \geq C)
\end{array} \right. \\
\begin{array}{c}
C \leftarrow |H - \text{VAR}_{i+1}| \\
\leq t \right. \right. \\
\end{array}
\]

Figure 3.226: Automaton for the \texttt{DECREASING\_RANGE\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.47 to the seed transducer of the \texttt{INCREASING\_SEQUENCE} pattern.
3.114  **DECREASING_RANGE STRICTLY DECREASING_SEQUENCE**

**Description**

**Origin**
Based on the **STRICTLY DECREASING_SEQUENCE** pattern.

**Constraint**
`DECREASING_RANGE.StrictlyDecreasing.Sequence(VARIABLES)`

**Argument**
`VARIABLES : collection(var - dvar)`

**Restriction**
`required(VARIABLES, var)`

**Purpose**
Succeeds if the differences between the largest and smallest value in each occurrence of the **STRICTLY DECREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern **STRICTLY DECREASING_SEQUENCE** is the maximal sub-sequence which matches the regular expression `>+`. Assume that the occurrence of the pattern **STRICTLY DECREASING_SEQUENCE** starts at position `i` and ends at position `j`. The feature `RANGE` computes the range of the values from index `i` to index `j + 1`.

**Example**

```
((6, 3, 3, 5, 4, 2, 7, 6, 5, 4, 4, 3, 3, 2, 6))
```

Figure 3.227 provides an example where the **DECREASING_RANGE STRICTLY DECREASING_SEQUENCE** `([6, 3, 3, 5, 4, 2, 7, 6, 5, 4, 4, 3, 3, 2, 6])` constraint holds.

**Typical**

```
|VARIABLES| > 1
range(VARIABLES.var) > 1
```
Figure 3.227: Illustrating the \texttt{DECREASING\_RANGE\_STRICTLY\_DECREASING\_SEQUENCE} constraint of the \texttt{Example} slot
Automaton  Figure 3.228 depicts the automaton associated with the constraint \textsc{decreasing_range_strictly_decreasing_sequence}.

\begin{align*}
C &\leftarrow +\infty \\
F &\leftarrow +\infty \\
H &\leftarrow \text{VAR}_i \\
R &\leftarrow 1
\end{align*}

\begin{align*}
&\leq S \\
&\{ H \leftarrow \text{VAR}_{i+1} \}
\end{align*}

\begin{align*}
&\leq R < (F \geq C) \\
&\{ C \leftarrow |H - \text{VAR}_{i+1}| \} \\
&\geq F < (H \leftarrow |H - \text{VAR}_{i+1}|) \\
&\{ C \leftarrow |H - \text{VAR}_{i+1}| \}
\end{align*}

Figure 3.228: Automaton for the \textsc{decreasing_range_strictly_decreasing_sequence} constraint obtained by applying decoration Table 2.47 to the seed transducer of the \textsc{strictly_decreasing_sequence} pattern.
DECREASING\_RANGE\_STRICTLY\_DECREASING\_SEQUENCE
### 3.115 DECREASING_RANGE_STRICTLY_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the STRICTLY_INCREASING_SEQUENCE pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>DECREASING_RANGE_STRICTLY_INCREASING_SEQUENCE(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var−dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES, var)</td>
</tr>
<tr>
<td><strong>Purpose</strong></td>
<td>Succeeds if the differences between the largest and smallest value in each occurrence of the STRICTLY_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing. An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression &lt;+. Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at position i and ends at position j. The feature RANGE computes the range of the values from index i to index j + 1.</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>((1, 4, 5, 5, 4, 1, 0, 2, 3, 6, 2, 3, 4, 4, 6))</td>
</tr>
<tr>
<td><strong>Typical</strong></td>
<td></td>
</tr>
</tbody>
</table>
DECREASING_RANGE_STRICTLY_INCREASING_SEQUENCE

Figure 3.229: Illustrating the DECREASING_RANGE_STRICTLY_INCREASING_SEQUENCE constraint of the Example slot
Automaton Figure 3.230 depicts the automaton associated with the constraint `DECREASING_RANGE_STRICTLY_INCREASING_SEQUENCE`.

\[
\begin{align*}
&\{ C \leftarrow +\infty \} \\
&\{ F \leftarrow +\infty \} \\
&\{ H \leftarrow \text{VAR}_1 \} \\
&\{ R \leftarrow 1 \} \\
\end{align*}
\]

\[
\begin{align*}
&\{ H \leftarrow \text{VAR}_{i+1} \} \\
&\{ C \leftarrow |H - \text{VAR}_{i+1}| \} \\
&\{ F \leftarrow |H - \text{VAR}_i| \} \\
&\{ H \leftarrow \text{VAR}_{i+1} \} \\
\end{align*}
\]

Figure 3.230: Automaton for the `DECREASING_RANGE_STRICTLY_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 2.47 to the seed transducer of the `STRICTLY_INCREASING_SEQUENCE` pattern.
DECREASING RANGE STRICTLY INCREASING SEQUENCE
### 3.116 DECREASING_SURF_BUMP_ON_DECREASING_SEQUENCE

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the BUMP_ON_DECREASING_SEQUENCE pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>DECREASING_SURF_BUMP_ON_DECREASING_SEQUENCE(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var–dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES, var)</td>
</tr>
</tbody>
</table>

Succeeds if the values denoting the surface of each occurrence of the BUMP_ON_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**

An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence which matches the regular expression >><<. Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 2$ to index $j$.

**Example**

$$((7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3))$$

Figure 3.231 provides an example where the DECREASING_SURF_BUMP_ON_DECREASING_SEQUENCE ($[7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3]$) constraint holds.

**Typical**

$$|\text{VARIABLES}| > 5$$

$$\text{range}(\text{VARIABLES}.\text{var}) > 2$$
Figure 3.231: Illustrating the \textit{DECREASING\_SURF\_BUMP\_ON\_DECREASING\_SEQUENCE} constraint of the \textbf{Example} slot
Figure 3.232: Automaton for the \textsc{decreasing_surf_bump_on_decreasing_sequence} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textsc{bump_on_decreasing_sequence} pattern.
3.117 **DECREASING_SURF_DECREASING**

**DESCRIPTION**

**Origin**
Based on the *DECREASING* pattern.

**Constraint**

`DECREASING_SURF_DECREASING(VARIABLES)`

**Argument**

`VARIABLES : collection(var−dvar)`

**Restriction**

`required(VARIABLES, var)`

Succeeds if the values denoting the surface of each occurrence of the *DECREASING* pattern in the time-series given by the `VARIABLES` collection are decreasing.

**Purpose**

An occurrence of the pattern *DECREASING* is the subsequence which matches the regular expression `>`.

Assume that the occurrence of the pattern *DECREASING* starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i` to index `j + 1`.

**Example**

`((2, 6, 5, 2, 3, 3, 3, 2, 4, 4, 5, 7, 7))`

Figure 3.233 provides an example where the *DECREASING_SURF_DECREASING* constraint holds.

![Graph](image_url)

Figure 3.233: Illustrating the *DECREASING_SURF_DECREASING* constraint of the **Example** slot
Typical

| VARIABLES | > 1

range(VARIABLES.var) > 1
Automaton

Figure 3.234 depicts the automaton associated with the constraint `DECREASING_SURF_DECREASING`.

\[
\begin{align*}
C & \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
D & \leftarrow 0 \\
F & \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
R & \leftarrow R \land (F \geq D + \text{VAR}_i + \text{VAR}_{i+1})
\end{align*}
\]

Figure 3.234: Automaton for the `DECREASING_SURF_DECREASING` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `DECREASING` pattern
3.118 DECREASING_SURF_DECREASING_SEQUENCE

**DESCRIPTION**

**Origin**
Based on the DECREASING_SEQUENCE pattern.

**Constraint**
DECREASING_SURF_DECREASING_SEQUENCE(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES, var)

Succeeds if the values denoting the surface of each occurrence of the DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**
An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression > (> | =)* > | >.
Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i$ to index $j + 1$.

**Example**

$(<2, 6, 5, 2, 3, 3, 4, 3, 3, 2, 4, 4, 5, 7, 5>)$

Figure 3.235 provides an example where the DECREASING_SURF_DECREASING_SEQUENCE ($(2, 6, 5, 2, 3, 3, 4, 3, 3, 2, 4, 4, 5, 7, 5)$) constraint holds.

**Typical**

$|\text{VARIABLES}| > 1$

range(VARIABLES.var) > 1
Figure 3.235: Illustrating the decreasing sequence constraint of the Example slot
Automaton  

Figure 3.236 depicts the automaton associated with the constraint `DECREASING_SURF_DECREASING_SEQUENCE`.

\[
\begin{align*}
&C \leftarrow +\infty \\
&D \leftarrow 0 \\
&R \leftarrow +\infty \\
&F \leftarrow +\infty \\
\end{align*}
\]

\[
\begin{align*}
&D \leftarrow 0 \\
&F \leftarrow C \\
&R \leftarrow R \land (F \geq C) \\
\end{align*}
\]

\[
\begin{align*}
&\{ D \leftarrow D + \text{VAR}_{i+1} \} \\
&\{ C \leftarrow C + D + \text{VAR}_{i+1} \} \\
\end{align*}
\]

\[
\begin{align*}
&\{ D \leftarrow 0 \} \\
&\{ C \leftarrow +\infty \} \\
&R \leftarrow 0 \\
\end{align*}
\]

Figure 3.236: Automaton for the `DECREASING_SURF_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `DECREASING_SEQUENCE` pattern.
DECREASING_SURF DECREASING_SEQUENCE
### 3.119 DECREASING_SURF_DECREASING_TERRACE

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the DECREASING_TERRACE pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>DECREASING_SURF_DECREASING_TERRACE(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var–dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES, var)</td>
</tr>
<tr>
<td><strong>Purpose</strong></td>
<td>Succeeds if the values denoting the surface of each occurrence of the DECREASING_TERRACE pattern in the time-series given by the VARIABLES collection are decreasing. An occurrence of the pattern DECREASING_TERRACE is the maximal subsequence which matches the regular expression ( \geq^{+} ). Assume that the occurrence of the pattern DECREASING_TERRACE starts at position ( i ) and ends at position ( j ). The feature SURF computes the sum of the values from index ( i + 1 ) to index ( j ).</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>( ((7, 6, 6, 6, 4, 3, 2, 2, 4, 4, 6, 3, 3, 1, 1)) )</td>
</tr>
</tbody>
</table>
| **Typical** | \( |\text{VARIABLES}| > 3 \)
\( \text{range}(\text{VARIABLES}.\text{var}) > 2 \) |

Figure 3.237 provides an example where the DECREASING_SURF_DECREASING_TERRACE \( ([7, 6, 6, 6, 4, 3, 2, 2, 4, 4, 6, 3, 3, 1, 1]) \) constraint holds.
Figure 3.237: Illustrating the `DECREASING_SURF_DECREASING_TERRACE` constraint of the Example slot
Automaton

Figure 3.238 depicts the automaton associated with the constraint DECREASING_SURF_DECREASING_TERRACE.

\[
\begin{align*}
&\begin{cases}
C &\leftarrow +\infty \\
D &\leftarrow 0 \\
F &\leftarrow +\infty \\
R &\leftarrow 1
\end{cases} \\
\begin{cases}
D &\leftarrow 0 \\
F &\leftarrow +\infty \\
R &\leftarrow 1
\end{cases} \\
\begin{cases}
C &\leftarrow D + VAR_i \\
D &\leftarrow 0 \\
F &\leftarrow D + VAR_i \\
R &\leftarrow R \land (F \geq D + VAR_i)
\end{cases}
\end{align*}
\]

Figure 3.238: Automaton for the DECREASING_SURF_DECREASING_TERRACE constraint obtained by applying decoration Table 2.36 to the seed transducer of the DECREASING_TERRACE pattern
3.120  **DECREASING_SURF_DIP_ON_INCREASING_SEQUENCE**

**DESCRIPTION**

- **Origin**: Based on the DIP_ON_INCREASING_SEQUENCE pattern.

- **Constraint**: DECREASING_SURF_DIP_ON_INCREASING_SEQUENCE(VARIABLES)

- **Argument**: VARIABLES : collection(var−dvar)

- **Restriction**: required(VARIABLES, var)

**Purpose**

Succeeds if the values denoting the surface of each occurrence of the DIP_ON_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence which matches the regular expression <<><<.

Assume that the occurrence of the pattern DIP_ON_INCREASING_SEQUENCE starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 2 to index j.

**Example**

```plaintext
((4, 5, 6, 0, 3, 4, 1, 2, 3, 2, 3, 4, 6, 5, 1))
```

Figure 3.239 provides an example where the DECREASING_SURF_DIP_ON_INCREASING_SEQUENCE ((4, 5, 6, 0, 3, 4, 1, 2, 3, 2, 3, 4, 6, 5, 1)) constraint holds.

**Typical**

- |VARABLES| > 5
- range(VARIABLES.var) > 2
DECREASING\_SURF\_DIP\_ON\_INCREASING\_SEQUENCE

Figure 3.239: Illustrating the DECREASING\_SURF\_DIP\_ON\_INCREASING\_SEQUENCE constraint of the Example slot
Automaton Figure 3.240 depicts the automaton associated with the constraint DECREASING_SURF_DIP_ON_INCREASING_SEQUENCE.

Figure 3.240: Automaton for the DECREASING_SURF_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern.
3.121 DECREASING_SURF_GORGE

**DESCRIPTION**

Origin: Based on the GORGE pattern.

**Constraint**

DECREASING_SURF_GORGE(VARIABLES)

**Argument**

VARIABLES : collection(var−dvar)

**Restriction**

required(VARIABLES, var)

**Purpose**

Succeeds if the values denoting the surface of each occurrence of the GORGE pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression \((> | > (= | >)^* (> | < | <)^* <)\).

Assume that the occurrence of the pattern GORGE starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

\((6, 2, 3, 4, 5, 6, 1, 1, 5, 4, 2, 3, 6, 1, 4, 5)\)

Figure 3.241 provides an example where the DECREASING_SURF_GORGE \(([6, 2, 3, 4, 5, 6, 1, 1, 5, 4, 2, 3, 6, 1, 4, 5])\) constraint holds.
Typical

\[ |\text{VARIABLES}| > 2 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]
Figure 3.242 depicts the automaton associated with the constraint DECREASING_SURF_GORGE.

Automaton

Figure 3.242: Automaton for the DECREASING_SURF_GORGE constraint obtained by applying decoration Table 2.36 to the seed transducer of the GORGE pattern (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$)
3.122  DECREASING_SURF_INCREASING

**DESCRIPTION**

**Origin**
Based on the INCREASING pattern.

**Constraint**

\[
\text{DECREASING\_SURF\_INCREASING}(\text{VARIABLES})
\]

**Argument**

\[
\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})
\]

**Restriction**

\[
\text{required}(\text{VARIABLES}, \text{var})
\]

Succeeds if the values denoting the surface of each occurrence of the INCREASING pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**

An occurrence of the pattern INCREASING is the subsequence which matches the regular expression \(<\).

Assume that the occurrence of the pattern INCREASING starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
((4, 4, 5, 5, 4, 3, 1, 4, 3, 3, 2, 1, 3, 1))
\]

Figure 3.243 provides an example where the DECREASING_SURF_INCREASING \(((4, 4, 5, 5, 4, 3, 1, 4, 3, 3, 2, 1, 3, 1))\) constraint holds.

![Figure 3.243: Illustrating the DECREASING_SURF_INCREASING constraint of the Example slot](image)

**Typical**

\[
|\text{VARIABLES}| > 1
\]

\[
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
Automaton

Figure 3.244 depicts the automaton associated with the constraint DECREASING_SURF_INCREASING.

\[
\begin{align*}
C & \leftarrow +\infty \\
D & \leftarrow 0 \\
F & \leftarrow +\infty \\
R & \leftarrow 1
\end{align*}
\]

Figure 3.244: Automaton for the DECREASING_SURF_INCREASING constraint obtained by applying decoration Table 2.36 to the seed transducer of the INCREASING pattern.
### 3.123 DECREASING_SURF_INCREASING_SEQUENCE

#### DESCRIPTION

**Origin**
Based on the INCREASING_SEQUENCE pattern.

**Constraint**
`DECREASING_SURF_INCREASING_SEQUENCE(VARIABLES)`

**Argument**
`VARIABLES : collection(var-dvar)`

**Restriction**
`required(VARIABLES, var)`

**Purpose**
Succeeds if the values denoting the surface of each occurrence of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression `< (< | =)* < |`.

Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position `i` and ends at position `j`. The feature SURF computes the sum of the values from index `i` to index `j + 1`.

**Example**

```
((3, 4, 5, 5, 4, 3, 1, 4, 3, 3, 2, 2, 1, 3, 1))
```

Figure 3.245 provides an example where the DECREASING_SURF_INCREASING_SEQUENCE (\([3, 4, 5, 5, 4, 3, 1, 4, 3, 3, 2, 2, 1, 3, 1]\)) constraint holds.

**Typical**

- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`
Figure 3.245: Illustrating the DECREASING_SURF_INCREASING_SEQUENCE constraint of the Example slot
Automaton Figure 3.246 depicts the automaton associated with the constraint `DECREASING_SURF_INCREASING_SEQUENCE`.

\[
\begin{align*}
C &\leftarrow +\infty \\
D &\leftarrow 0 \\
F &\leftarrow +\infty \\
R &\leftarrow 1 \\
\end{align*}
\]

\[
\begin{align*}
D &\leftarrow 0 \\
F &\leftarrow C \\
R &\leftarrow R \land (F \geq C) \\
\end{align*}
\]

\[
\begin{align*}
D &\leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
\end{align*}
\]

\[
\begin{align*}
C &\leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
D &\leftarrow 0 \\
\end{align*}
\]

\[
\begin{align*}
D &\leftarrow D + \text{VAR}_{i+1} \\
\end{align*}
\]

\[
\begin{align*}
C &\leftarrow C + D + \text{VAR}_{i+1} \\
D &\leftarrow 0 \\
\end{align*}
\]

Figure 3.246: Automaton for the `DECREASING_SURF_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `INCREASING_SEQUENCE` pattern.
### 3.124 DECREASING_SURF_INCREASING_TERRACE

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th><strong>Automaton</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the INCREASING_TERRACE pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>DECREASING_SURF_INCREASING_TERRACE(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var–dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES,var)</td>
</tr>
<tr>
<td><strong>Purpose</strong></td>
<td>Succeeds if the values denoting the surface of each occurrence of the INCREASING_TERRACE pattern in the time-series given by the VARIABLES collection are decreasing. An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression $&lt;=$+$. Assume that the occurrence of the pattern INCREASING_TERRACE starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 1$ to index $j$.</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>$((2, 5, 5, 5, 6, 2, 3, 4, 4, 5, 3, 3, 1, 2, 2, 3))$</td>
</tr>
</tbody>
</table>
| **Typical**     | $|VARIABLES| > 3$  
|                  | range(VARIABLES.var) > 2 |

Figure 3.247 provides an example where the DECREASING_SURF_INCREASING_TERRACE ($[2, 5, 5, 5, 6, 2, 3, 4, 4, 5, 3, 3, 1, 2, 2, 3]$) constraint holds.
Figure 3.247: Illustrating the \texttt{DECREASING\_SURF\_INCREASING\_TERRACE} constraint of the \texttt{Example} slot
Figure 3.248 depicts the automaton associated with the constraint \textsc{decreasing_surf_increasing_terrace}.

\begin{equation}
\begin{cases}
C \leftarrow +\infty \\
D \leftarrow 0 \\
F \leftarrow +\infty \\
R \leftarrow 1
\end{cases}
\end{equation}

\begin{equation}
\begin{cases}
C \leftarrow D + \text{VAR}_i \\
D \leftarrow 0 \\
F \leftarrow D + \text{VAR}_i \\
R \leftarrow R \land (F \geq D + \text{VAR}_i)
\end{cases}
\end{equation}

Figure 3.248: Automaton for the \textsc{decreasing_surf_increasing_terrace} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textsc{increasing_terrace} pattern.
3.125 DECREASING_SURF_INFLEXION

| Origin | Based on the INFLEXION pattern. |
| Constraint | DECREASING_SURF_INFLEXION(VARIABLES) |
| Argument | VARIABLES : collection(var−dvar) |
| Restriction | required(VARIABLES, var) |

Succeeds if the values denoting the surface of each occurrence of the INFLEXION pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression $< (< | =) | > | ( > | =) >$.< 

Assume that the occurrence of the pattern INFLEXION starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 1$ to index $j$.

Example

$$((6, 5, 5, 4, 3, 2, 3, 3, 4, 6, 3, 3, 1, 2, 1, 1))$$

Figure 3.249 provides an example where the DECREASING_SURF_INFLEXION ($[6, 5, 5, 4, 3, 2, 3, 3, 4, 6, 3, 3, 1, 2, 1, 1]$) constraint holds.

Typical

$$|VARIABLES| > 2$$
$$range(VARIABLES.var) > 1$$
Figure 3.249: Illustrating the DECREASING_SURF_INFLEXION constraint of the Example slot
Automaton

Figure 3.250 depicts the automaton associated with the constraint DECREASING_SURF_INFLEXION.

\[
\begin{align*}
&\{ C \leftarrow +\infty \} \quad D \leftarrow 0 \quad F \leftarrow +\infty \quad R \leftarrow 1 \\
&\{ D \leftarrow D + \text{VAR}_i \} \quad \{ D \leftarrow D + \text{VAR}_i \} \\
&\{ C \leftarrow D + \text{VAR}_i \} \quad D \leftarrow 0 \quad F \leftarrow D + \text{VAR}_i \quad R \leftarrow R \land (F \geq D + \text{VAR}_i) \\
\end{align*}
\]

Figure 3.250: Automaton for the DECREASING_SURF_INFLEXION constraint obtained by applying decoration Table 2.36 to the seed transducer of the INFLEXION pattern (transition \( r \rightarrow t \) has the same accumulators updates as transition \( t \rightarrow r \))
3.126 DECREASING_SURF_PEAK

**DESCRIPTION**

**Origin**
Based on the PEAK pattern.

**Constraint**
DECREASING_SURF_PEAK(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES, var)

**Purpose**
Succeeds if the values denoting the surface of each occurrence of the PEAK pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression < (= | <)∗ (> | =)∗ >.

Assume that the occurrence of the pattern PEAK starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 1 to index j.

**Example**

\[
((1, 6, 6, 6, 2, 5, 4, 3, 2, 2, 3, 2, 1, 5, 5, 7))
\]

Figure 3.251 provides an example where the DECREASING_SURF_PEAK \(([1, 6, 6, 6, 2, 5, 4, 3, 2, 2, 3, 2, 1, 5, 5, 7])\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 2
\]

\[
\text{range}(\text{VARIABLES}.var) > 1
\]
Figure 3.251: Illustrating the DECREASING_SURF_PEAK constraint of the Example slot
Automaton Figure 3.252 depicts the automaton associated with the constraint `DECREASING_SURF_PEAK`.

Figure 3.252: Automaton for the `DECREASING_SURF_PEAK` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `PEAK` pattern.
3.127 DECREASING_SURF_PLAIN

**DESCRIPTION**

**Origin**
Based on the PLAIN pattern.

**Constraint**
DECREASING_SURF_PLAIN(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES, var)

Succeeds if the values denoting the surface of each occurrence of the PLAIN pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**
An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression \( > = ^* < \).

Assume that the occurrence of the pattern PLAIN starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i + 1 \) to index \( j \).

**Example**

\(((2, 3, 6, 5, 5, 7, 6, 6, 4, 5, 5, 4, 3, 6, 6, 3))\)

Figure 3.253 provides an example where the DECREASING_SURF_PLAIN \(((2, 3, 6, 5, 5, 7, 6, 6, 4, 5, 5, 4, 3, 6, 6, 3))\) constraint holds.

Figure 3.253: Illustrating the DECREASING_SURF_PLAIN constraint of the Example slot.
<table>
<thead>
<tr>
<th>Typical</th>
</tr>
</thead>
</table>
| $|\text{VARIABLES}| > 2$  
| $\text{range}(\text{VARIABLES}.\text{var}) > 1$ |
Automaton Figure 3.254 depicts the automaton associated with the constraint DECREASING_SURF_PLAIN.

\[
\begin{align*}
C &\leftarrow +\infty \\
D &\leftarrow 0 \\
F &\leftarrow +\infty \\
R &\leftarrow 1 \\
\end{align*}
\]

\[
\begin{align*}
R \land (F \geq C) &\leq s \\
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow D + \text{VAR}_i \} \\
\{ D \leftarrow 0 \} \\
\end{align*}
\]

Figure 3.254: Automaton for the DECREASING_SURF_PLAIN constraint obtained by applying decoration Table 2.36 to the seed transducer of the PLAIN pattern.
DECREASING_SURF_PLAIN
3.128 DECREASING_SURF_PLATEAU

### DESCRIPTION

**Origin**
Based on the PLATEAU pattern.

**Constraint**
DECREASING_SURF_PLATEAU(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES, var)

Succeeds if the values denoting the surface of each occurrence of the PLATEAU pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**
An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression < =∗ >.
Assume that the occurrence of the pattern PLATEAU starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i+1\) to index \(j\).

**Example**
\((5, 2, 2, 5, 5, 4, 3, 3, 4, 2, 1, 3, 2, 5, 7))

Figure 3.255 provides an example where the DECREASING_SURF_PLATEAU \((5, 2, 2, 5, 5, 4, 3, 3, 4, 2, 2, 1, 3, 2, 5, 7))\) constraint holds.

**Typical**

\[|\text{VARIABLES}| > 2\]
\[\text{range(VARIABLES.var)} > 1\]
Figure 3.255: Illustrating the DECREASING_SURF_PLATEAU constraint of the Example slot
Automaton

Figure 3.256 depicts the automaton associated with the constraint DECREASING_SURF_PLATEAU.

$$\begin{align*}
\{ C \leftarrow \infty \} & \quad \{ D \leftarrow 0 \} \\
\{ F \leftarrow +\infty \} & \quad \{ R \leftarrow 1 \}
\end{align*}$$

$$\begin{align*}
C & \leftarrow D + \text{VAR}_i \\
D & \leftarrow 0 \\
F & \leftarrow D + \text{VAR}_i \\
R & \leftarrow R \land (F \geq D + \text{VAR}_i)
\end{align*}$$

$$\begin{align*}
\{ D \leftarrow D + \text{VAR}_i \} & \quad \{ D \leftarrow 0 \} \\
\{ D \leftarrow 0 \} & \quad \{ D \leftarrow D + \text{VAR}_i \}
\end{align*}$$

$$\begin{align*}
R \land (F \geq C) & \quad \{ D \leftarrow 0 \}
\end{align*}$$

Figure 3.256: Automaton for the DECREASING_SURF_PLATEAU constraint obtained by applying decoration Table 2.36 to the seed transducer of the PLATEAU pattern.
DECREASING_SURF_PLATEAU
### 3.129 DECREASING_SURF_PROPER.PLAIN

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the PROPER.PLAIN pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>DECREASING_SURF_PROPER_PLAIN(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var−dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES, var)</td>
</tr>
</tbody>
</table>

Succeeds if the values denoting the surface of each occurrence of the PROPER.PLAIN pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**

An occurrence of the pattern PROPER.PLAIN is the maximal subsequence which matches the regular expression > =^+ <.

Assume that the occurrence of the pattern PROPER.PLAIN starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 1 to index j.

**Example**

$$((2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5))$$

Figure 3.257 provides an example where the DECREASING_SURF_PROPER_PLAIN $$((2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5))$$ constraint holds.

**Typical**

$$|\text{VARIABLES}| > 3$$

$$\text{range(VARIABLES.var)} > 1$$
Figure 3.257: Illustrating the DECREASING_SURF_PROPER_PLAIN constraint of the Example slot
Automaton

Figure 3.258 depicts the automaton associated with the constraint DECREASING_SURF_PROPER_PLAIN.

\[
\begin{align*}
C & \leftarrow D + \text{VAR}_i \\
D & \leftarrow 0 \\
F & \leftarrow D + \text{VAR}_i \\
R & \leftarrow R \land (F \geq D + \text{VAR}_i)
\end{align*}
\]

\[
\begin{align*}
C & \leftarrow +\infty \\
D & \leftarrow 0 \\
F & \leftarrow +\infty \\
R & \leftarrow 1
\end{align*}
\]

\[
\begin{align*}
\text{\{D \leftarrow D + \text{VAR}_i\}} & \leq s \\
\text{\{D \leftarrow 0\}} & > \text{\{D \leftarrow 0\}} \\
\text{\{D \leftarrow D + \text{VAR}_i\}} & \geq t
\end{align*}
\]

Figure 3.258: Automaton for the DECREASING_SURF_PROPER_PLAIN constraint obtained by applying decoration Table 2.36 to the seed transducer of the PROPER_PLAIN pattern
### 3.130 DECREASING_SURF_PROPER_PLATEAU

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th><strong>Automaton</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the PROPER_PLATEAU pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>DECREASING_SURF_PROPER_PLATEAU(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var–dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES, var)</td>
</tr>
</tbody>
</table>

Succeeds if the values denoting the surface of each occurrence of the PROPER_PLATEAU pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression < = + >.

Assume that the occurrence of the pattern PROPER_PLATEAU starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i + 1 \) to index \( j \).

#### Example

\[
(3, 5, 5, 3, 2, 3, 4, 4, 1, 5, 2, 3, 3, 1, 7)
\]

Figure 3.259 provides an example where the DECREASING_SURF_PROPER_PLATEAU constraint holds.

#### Typical

\[
| \text{VARIABLES} | > 3  \\
| \text{range(VARIABLES.var)} | > 1
\]
Figure 3.259: Illustrating the DECREASING_SURF_PROPER_PLATEAU constraint of the Example slot
Automaton Figure 3.260 depicts the automaton associated with the constraint \textsc{decreasing_surf_proper_plateau}.

Figure 3.260: Automaton for the \textsc{decreasing_surf_proper_plateau} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textsc{proper_plateau} pattern.
3.131 DECREASING_SURF_STEADY

DESCRIPTION AUTOMATON

Origin
Based on the STEADY pattern.

Constraint
DECREASING_SURF_STEADY(VARIABLES)

Argument
VARIABLES : collection(var−dvar)

Restriction
required(VARIABLES, var)

Succeeds if the values denoting the surface of each occurrence of the STEADY pattern in the time-series given by the VARIABLES collection are decreasing.

Purpose
An occurrence of the pattern STEADY is the subsequence which matches the regular expression =.
Assume that the occurrence of the pattern STEADY starts at position i and ends at position j. The feature SURF computes the sum of the values from index i to index j + 1.

Example
(⟨3, 6, 6, 2, 7, 5, 5, 5, 6, 5, 5, 3, 3, 7⟩)

Figure 3.261 provides an example where the DECREASING_SURF_STEADY ([3, 6, 6, 2, 7, 5, 5, 6, 5, 5, 3, 3, 7]) constraint holds.

Figure 3.261: Illustrating the DECREASING_SURF_STEADY constraint of the Example slot
Typical \[ |\text{VARIABLES}| > 1 \]
Automaton

Figure 3.262 depicts the automaton associated with the constraint DECREASING_SURF_STEADY.

\[
\begin{align*}
\{ C & \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
D & \leftarrow 0 \\
F & \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
R & \leftarrow R \land (F \geq D + \text{VAR}_i + \text{VAR}_{i+1}) 
\end{align*}
\]

Figure 3.262: Automaton for the DECREASING_SURF_STEADY constraint obtained by applying decoration Table 2.36 to the seed transducer of the STEADY pattern.
### 3.132 DECREASING_SURF_STEADY_SEQUENCE

**DESCRIPTION**

**Origin**
Based on the `STEADY_SEQUENCE` pattern.

**Constraint**
`DECREASING_SURF_STEADY_SEQUENCE(VARIABLES)`

**Argument**
```
VARIABLES : collection(var—dvar)
```

**Restriction**
```
required(VARIABLES, var)
```

Succeeds if the values denoting the surface of each occurrence of the `STEADY_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are decreasing.

**Purpose**
An occurrence of the pattern `STEADY_SEQUENCE` is the maximal subsequence which matches the regular expression `=\+`. Assume that the occurrence of the pattern `STEADY_SEQUENCE` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i` to index `j + 1`.

**Example**
```
((4, 3, 2, 5, 5, 3, 2, 4, 4, 6, 5, 3, 2, 2, 6))
```

Figure 3.263 provides an example where the `DECREASING_SURF_STEADY_SEQUENCE` constraint holds.

![Diagram](image)

Figure 3.263: Illustrating the `DECREASING_SURF_STEADY_SEQUENCE` constraint of the Example slot
Typical $|\text{VARIABLES}| > 1$
Automaton

Figure 3.264 depicts the automaton associated with the constraint DECREASING_SURF_STEADY_SEQUENCE.

Figure 3.264: Automaton for the DECREASING_SURF_STEADY_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the STEADY_SEQUENCE pattern
3.133  **DECREASING_SURF_STRICTLY_DECREASING_SEQUENCE**

**DESCRIPTION**

**Origin**  
Based on the **STRICTLY_DECREASING_SEQUENCE** pattern.

**Constraint**  
**DECREASING_SURF_STRICTLY_DECREASING_SEQUENCE**(VARIABLES)

**Argument**  
VARIABLES : collection(var − dvar)

**Restriction**  
required(VARIABLES, var)

**Purpose**  
Succeeds if the values denoting the surface of each occurrence of the **STRICTLY_DECREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** is the *maximal* sub-sequence which matches the regular expression >*

Assume that the occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i$ to index $j + 1$.

**Example**  

((7, 4, 6, 5, 5, 4, 4, 3, 6, 1, 4, 2, 1, 2, 3, 4))

Figure 3.265 provides an example where the **DECREASING_SURF_STRICTLY_DECREASING_SEQUENCE** constraint holds.

**Typical**  

|VARIABLES| > 1  
|range(VARIABLES.var)| > 1
DECREASING_SURF_STRICTLY_DECREASING_SEQUENCE

Figure 3.265: Illustrating the DECREASING_SURF_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Automaton Figure 3.266 depicts the automaton associated with the constraint `DECREASING_SURF_STRICTLY_DECREASING_SEQUENCE`.

\[
\begin{align*}
C & \leftarrow +\infty \\
D & \leftarrow 0 \\
F & \leftarrow +\infty \\
R & \leftarrow 1
\end{align*}
\]

\[
\begin{align*}
C & \leftarrow \text{VAR}_i + \text{VAR}_{i+1} \\
D & \leftarrow 0
\end{align*}
\]

\[
\begin{align*}
C & \leftarrow C + D + \text{VAR}_{i+1} \\
D & \leftarrow 0
\end{align*}
\]

Figure 3.266: Automaton for the `DECREASING_SURF_STRICTLY_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `STRICTLY_DECREASING_SEQUENCE` pattern.
DECREASING_SURF_STRICTLY_DECREASING_SEQUENCE
3.134 DECREASING_SURF STRICTLY_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td>Origin</td>
<td>Based on the STRICTLY_INCREASING_SEQUENCE pattern.</td>
</tr>
<tr>
<td>Constraint</td>
<td>DECREASING_SURF_STRICTLY_INCREASING_SEQUENCE(VARIABLES)</td>
</tr>
<tr>
<td>Argument</td>
<td>VARIABLES : collection(var–dvar)</td>
</tr>
<tr>
<td>Restriction</td>
<td>required(VARIABLES.var)</td>
</tr>
</tbody>
</table>

Succeeds if the values denoting the surface of each occurrence of the STRICTLY_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**

An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the *maximal* sub-sequence which matches the regular expression <+. Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i$ to index $j + 1$.

**Example**

$((3, 4, 5, 5, 4, 3, 1, 4, 3, 3, 2, 2, 1, 3, 1))$

Figure 3.267 provides an example where the DECREASING_SURF STRICTLY_INCREASING_SEQUENCE $((3, 4, 5, 5, 4, 3, 1, 4, 3, 3, 2, 2, 1, 3, 1))$ constraint holds.

**Typical**

$|$VARIABLES$| > 1$

$\text{range}(\text{VARIABLES.var}) > 1$
Figure 3.267: Illustrating the \textsc{decreasing_surf_strictly_increasing_sequence} constraint of the \textit{Example} slot
Automaton

Figure 3.268 depicts the automaton associated with the constraint `DECREASING_SURF_STRICTLY_INCREASING_SEQUENCE`.

\[
\begin{align*}
C & \leftarrow +\infty \\
D & \leftarrow 0 \\
F & \leftarrow +\infty \\
R & \leftarrow 1 \\
\end{align*}
\]

\[
\begin{align*}
\{ & D \leftarrow 0 \\
& F \leftarrow C \\
& R \leftarrow R \land (F \geq C) \} \\
\{ & C \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
& D \leftarrow 0 \}
\end{align*}
\]

Figure 3.268: Automaton for the `DECREASING_SURF_STRICTLY_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `STRICTLY_INCREASING_SEQUENCE` pattern.
DECREASING, SURF, STRICTLY, INCREASING, SEQUENCE
### 3.135 **DECREASING_SURF_SUMMIT**

<table>
<thead>
<tr>
<th><strong>DESCRIPTION</strong></th>
<th><strong>AUTOMATON</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Origin</td>
<td>Based on the SUMMIT pattern.</td>
</tr>
<tr>
<td>Constraint</td>
<td>DECREASING_SURF_SUMMIT(VARIABLES)</td>
</tr>
<tr>
<td>Argument</td>
<td>VARIABLES : collection(var–dvar)</td>
</tr>
<tr>
<td>Restriction</td>
<td>required(VARIABLES, var)</td>
</tr>
</tbody>
</table>

**Purpose**

Succeeds if the values denoting the surface of each occurrence of the SUMMIT pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression $(< | < (= | <)^* <)(>| > (= | >)^*) >)$.

Assume that the occurrence of the pattern SUMMIT starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 1$ to index $j$.

**Example**

$((1, 4, 4, 5, 4, 2, 6, 6, 2, 3, 5, 4, 1, 2, 4, 2))$

Figure 3.269 provides an example where the DECREASING_SURF_SUMMIT($(1, 4, 4, 5, 4, 2, 6, 6, 2, 3, 5, 4, 1, 2, 4, 2)$) constraint holds.

**Typical**

$|\text{VARIABLES}| > 2$

$\text{range}(\text{VARIABLES}.\text{var}) > 1$
Figure 3.269: Illustrating the DECREASING_SURF_SUMMIT constraint of the Example slot
Automaton

Figure 3.270 depicts the automaton associated with the constraint DECREASING_SURF_SUMMIT.

Figure 3.270: Automaton for the DECREASING_SURF_SUMMIT constraint obtained by applying decoration Table 2.36 to the seed transducer of the SUMMIT pattern (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$)
3.136 DECREASING_SURF_VALLEY

**Origin**
Based on the VALLEY pattern.

**Constraint**
DECREASING_SURF_VALLEY(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES, var)

**Purpose**
Succeeds if the values denoting the surface of each occurrence of the VALLEY pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern VALLEY is the *maximal* subsequence which matches the regular expression \( > (= | >)^* (< | =)^* < \).

Assume that the occurrence of the pattern VALLEY starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i + 1 \) to index \( j \).

**Example**
\[
((1, 3, 3, 7, 6, 5, 4, 6, 6, 5, 4, 3, 6, 2, 2, 7))
\]

Figure 3.271 provides an example where the DECREASING_SURF_VALLEY \(((1, 3, 3, 7, 6, 5, 4, 6, 6, 5, 4, 3, 6, 2, 2, 7))\) constraint holds.

**Typical**
\[
|\text{VARIABLES}| > 2
\]
\[
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
Figure 3.271: Illustrating the DECREASING_SURF_VALLEY constraint of the Example slot.
Figure 3.272 depicts the automaton associated with the constraint `DECREASING_SURF_VALLEY`.

Figure 3.272: Automaton for the `DECREASING_SURF_VALLEY` constraint obtained by applying decoration Table 2.36 to the seed transducer of the VALLEY pattern.
### 3.137 DECREASING_SURF_ZIGZAG

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
</table>

**Origin**
Based on the ZIGZAG pattern.

**Constraint**
DECREASING_SURF_ZIGZAG(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES, var)

Succeeds if the values denoting the surface of each occurrence of the ZIGZAG pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**
An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression \((<>)^+ (<> | <>) | (<>|^)+ | >><\).
Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**
\(((6, 5, 7, 4, 5, 1, 1, 6, 3, 6, 4, 3, 5, 2, 7, 7))\)

Figure 3.273 provides an example where the DECREASING_SURF_ZIGZAG \(\{6, 5, 7, 4, 5, 1, 1, 6, 3, 6, 4, 3, 5, 2, 7, 7\}\) constraint holds.

**Typical**
\[|\text{VARIABLES}| > 3\]
\[\text{range}(\text{VARIABLES}.\text{var}) > 1\]
DECREASING_SURF_ZIGZAG

Figure 3.273: Illustrating the DECREASING_SURF_ZIGZAG constraint of the Example slot
Figure 3.274 depicts the automaton associated with the constraint DECREASING_SURF_ZIGZAG.
Figure 3.274: Automaton for the DECREASING_SURF_ZIGZAG constraint obtained by applying decoration Table 2.36 to the seed transducer of the ZIGZAG pattern; (1) missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=$; (2) on transitions from $b, c, e, f$ to $s$ the accumulator $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the accumulator $F$ is reset to $C$, and the accumulator $R$ is updated wrt $C$ and $F$. 
### 3.138 DECREASING_WIDTH_DECREASING_SEQUENCE

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the DECREASING_SEQUENCE pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>DECREASING_WIDTH_DECREASING_SEQUENCE(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var–dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES, var)</td>
</tr>
</tbody>
</table>

Succeeds if the values denoting the width of each occurrence of the DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**

An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression \( (> | =)^* > | > \). Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i + 2 \).

**Example**

\[ ((2, 6, 5, 2, 3, 3, 4, 3, 2, 3, 4, 4, 5, 7, 5)) \]

Figure 3.275 provides an example where the DECREASING_WIDTH_DECREASING_SEQUENCE constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 1 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]
Figure 3.275: Illustrating the \textsc{decreasing}\_\textsc{width}\_\textsc{decreasing}\_\textsc{sequence} constraint of the \textsc{Example} slot
Automaton

Figure 3.276 depicts the automaton associated with the constraint \( \text{DECREASING_WIDTH\_DECREASING\_SEQUENCE} \).

\[
\begin{align*}
C & \leftarrow +\infty \\
D & \leftarrow 0 \\
F & \leftarrow +\infty \\
R & \leftarrow 1
\end{align*}
\]

\[
\begin{aligned}
\text{\( \leq s \)} \quad \rightarrow \\
\{D \leftarrow 0, F \leftarrow C, R \leftarrow R \wedge (F \geq C)\} \\
\{C \leftarrow D + 1\}
\end{aligned}
\]

\[
\begin{aligned}
\text{\( \geq t \)} \quad \rightarrow \\
\{C \leftarrow +\infty, D \leftarrow 0\} \\
\{D \leftarrow 0\}
\end{aligned}
\]

Figure 3.276: Automaton for the \( \text{DECREASING_WIDTH\_DECREASING\_SEQUENCE} \) constraint obtained by applying decoration Table 2.36 to the seed transducer of the \( \text{DECREASING\_SEQUENCE} \) pattern.
3.139 DECREASING_WIDTH_DECREASING_TERRACE

Description

Origin
Based on the DECREASING_TERRACE pattern.

Constraint
DECREASING_WIDTH_DECREASING_TERRACE(VARIABLES)

Argument
VARIABLES : collection(var−dvar)

Restriction
required(VARIABLES, var)

Purpose
Succeeds if the values denoting the width of each occurrence of the DECREASING_TERRACE pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern DECREASING_TERRACE is the maximal subsequence which matches the regular expression $\geq^+$. Assume that the occurrence of the pattern DECREASING_TERRACE starts at position $i$ and ends at position $j$. The feature WIDTH computes the value $j − i$.

Example

Figure 3.277 provides an example where the DECREASING_WIDTH_DECREASING_TERRACE ($[7, 6, 6, 6, 3, 3, 2, 2, 4, 4, 6, 3, 3, 1, 1]$) constraint holds.

Typical

$|VARIABLES| > 3$

range(VARIABLES.var) > 2
Figure 3.277: Illustrating the `DECREASING_WIDTH_DECREASING_TERRACE` constraint of the *Example* slot
Automaton

Figure 3.278 depicts the automaton associated with the constraint DECREASING_WIDTH_DECREASING_TERRACE.

\[
\begin{align*}
&\{ C \leftarrow +\infty \} \\
&\{ D \leftarrow 0 \} \\
&\{ F \leftarrow +\infty \} \\
&\{ R \leftarrow 1 \} \\
\end{align*}
\]

\[
\begin{align*}
&\{ D \leftarrow D + 1 \} \\
&\{ D \leftarrow 0 \} \\
&\{ F \leftarrow D + 1 \} \\
&\{ R \leftarrow R \land (F \geq D + 1) \} \\
\end{align*}
\]

Figure 3.278: Automaton for the DECREASING_WIDTH_DECREASING_TERRACE constraint obtained by applying decoration Table 2.36 to the seed transducer of the DECREASING_TERRACE pattern.
3.140 **DECREASING_WIDTH_GORGE**

### Origin
Based on the **GORGE** pattern.

### Constraint
`DECREASING_WIDTH_GORGE(VARIABLES)`

### Argument
`VARIABLES : collection(var−dvar)`

### Restriction
`required(VARIABLES, var)`

**Purpose**
Succeeds if the values denoting the width of each occurrence of the **GORGE** pattern in the time-series given by the **VARIABLES** collection are decreasing.

An occurrence of the pattern **GORGE** is the *maximal* subsequence which matches the regular expression `(> | > (≡ | >)* >)(< | < (≡ | <)* <)`. Assume that the occurrence of the pattern **GORGE** starts at position `i` and ends at position `j`. The feature **WIDTH** computes the value `j − i`.

### Example
`((6, 2, 3, 4, 5, 6, 1, 1, 5, 4, 2, 3, 6, 1, 4, 5))`

Figure 3.279 provides an example where the **DECREASING_WIDTH_GORGE** `((6, 2, 3, 4, 5, 6, 1, 1, 5, 4, 2, 3, 6, 1, 4, 5))` constraint holds.
<table>
<thead>
<tr>
<th>Typical</th>
</tr>
</thead>
</table>
| $|\text{VARIABLES}| > 2$
| $\text{range}(\text{VARIABLES.var}) > 1$ |
Automaton Figure 3.280 depicts the automaton associated with the constraint DECREASING_WIDTH_GORGE.

Figure 3.280: Automaton for the DECREASING_WIDTH_GORGE constraint obtained by applying decoration Table 2.36 to the seed transducer of the GORGE pattern (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$)
DECREASING_WIDTH_INCREASING_SEQUENCE

3.141 DECREASING_WIDTH_INCREASING_SEQUENCE

**DESCRIPTION**

**Origin**

Based on the INCREASING_SEQUENCE pattern.

**Constraint**

DECREASING_WIDTH_INCREASING_SEQUENCE(VARIABLES)

**Argument**

VARIABLES : collection(var–dvar)

**Restriction**

required(VARIABLES, var)

Succeeds if the values denoting the width of each occurrence of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**

An occurrence of the pattern INCREASING_SEQUENCE is the *maximal* subsequence which matches the regular expression `< (< | =) * < | <`. Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position $i$ and ends at position $j$. The feature WIDTH computes the value $j - i + 2$.

**Example**

`((3, 4, 5, 5, 4, 3, 3, 1, 4, 3, 3, 2, 2, 1, 3, 1))`

Figure 3.281 provides an example where the DECREASING_WIDTH_INCREASING_SEQUENCE constraint holds.

Figure 3.281: Illustrating the DECREASING_WIDTH_INCREASING_SEQUENCE constraint of the Example slot
Typical

\[ |\text{VARIABLES}| > 1 \]
\[ \text{range(\text{VARIABLES.var})} > 1 \]
Automaton

Figure 3.282 depicts the automaton associated with the constraint `DECREASING_WIDTH_INCREASING_SEQUENCE`.

\[
\begin{align*}
C & \leftarrow +\infty \\
D & \leftarrow 0 \\
F & \leftarrow +\infty \\
R & \leftarrow 1
\end{align*}
\]

\[
\begin{align*}
D & \leftarrow 0 \\
F & \leftarrow C \\
R & \leftarrow R \land (F \geq C)
\end{align*}
\]

\[
\begin{align*}
C & \leftarrow D + 2 \\
D & \leftarrow 0
\end{align*}
\]

\[
\begin{align*}
D & \leftarrow D + 1
\end{align*}
\]

\[
\begin{align*}
C & \leftarrow C + D + 1 \\
D & \leftarrow 0
\end{align*}
\]

Figure 3.282: Automaton for the `DECREASING_WIDTH_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `INCREASING_SEQUENCE` pattern.
### 3.142 DECREASING_WIDTH_INCREASING_TERRACE

<table>
<thead>
<tr>
<th>Origin</th>
<th>Based on the INCREASING_TERRACE pattern.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constraint</td>
<td>DECREASING_WIDTH_INCREASING_TERRACE(VARIABLES)</td>
</tr>
<tr>
<td>Argument</td>
<td>VARIABLES : collection(var–dvar)</td>
</tr>
<tr>
<td>Restriction</td>
<td>required(VARIABLES, var)</td>
</tr>
</tbody>
</table>

**Purpose**

Succeeds if the values denoting the width of each occurrence of the INCREASING_TERRACE pattern in the time-series given by the VARIABLES collection are decreasing.

Assume that the occurrence of the pattern INCREASING_TERRACE starts at position $i$ and ends at position $j$. The feature WIDTH computes the value $j - i$.

**Example**

$$((2, 5, 5, 5, 6, 2, 3, 4, 4, 5, 3, 3, 1, 2, 2, 3))$$

Figure 3.283 provides an example where the DECREASING_WIDTH_INCREASING_TERRACE $((2, 5, 5, 5, 6, 2, 3, 4, 4, 5, 3, 3, 1, 2, 2, 3))$ constraint holds.

**Typical**

$|\text{VARIABLES}| > 3$

$$\text{range}(\text{VARIABLES.var}) > 2$$
Figure 3.283: Illustrating the DECREASING_WIDTH_INCREASING_TERRACE constraint of the Example slot
Figure 3.284 depicts the automaton associated with the constraint `DECREASING_WIDTH_INCREASING_TERRACE`.

\begin{align*}
\begin{cases}
C & \leftarrow +\infty \\
D & \leftarrow 0 \\
F & \leftarrow +\infty \\
R & \leftarrow 1 
\end{cases}
\end{align*}

\[ \begin{array}{c}
\{ D \leftarrow D + 1 \} \\
\{ D \leftarrow D + 1 \} \\
\{ D \leftarrow 0 \} \\
{ R \leftarrow R \land (F \geq D + 1) } \\
\end{array} \]

Figure 3.284: Automaton for the `DECREASING_WIDTH_INCREASING_TERRACE` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `INCREASING_TERRACE` pattern.
3.143 DECREASING_WIDTH_INFLEXION

**DESCRIPTION**

Origin: Based on the INFLEXION pattern.

Constraint: DECREASING_WIDTH_INFLEXION(VARIABLES)

Argument: VARIABLES : collection(var−dvar)

Restriction: required(VARIABLES, var)

Purpose: Succeeds if the values denoting the width of each occurrence of the INFLEXION pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression \(< (< | =)^* > | > (>| =)^* <\).

Assume that the occurrence of the pattern INFLEXION starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

**Example**

\(((6, 5, 5, 4, 3, 2, 3, 3, 4, 6, 3, 3, 1, 2, 1, 1))\)

Figure 3.285 provides an example where the DECREASING_WIDTH_INFLEXION (\([6, 5, 5, 4, 3, 2, 3, 3, 4, 6, 3, 3, 1, 2, 1, 1]\)) constraint holds.

![Diagram](image)

Figure 3.285: Illustrating the DECREASING_WIDTH_INFLEXION constraint of the Example slot
Typical

\(|\text{VARIABLES}| > 2\)

\(\text{range}(\text{VARIABLES}.\text{var}) > 1\)
Automaton

Figure 3.286 depicts the automaton associated with the constraint `DECREASING_WIDTH_INFLEXION`.

\[
\begin{align*}
C &\leftarrow +\infty \\
D &\leftarrow 0 \\
F &\leftarrow +\infty \\
R &\leftarrow 1 \\
\end{align*}
\]

\[
\begin{align*}
\left\{ D \leftarrow D + 1 \right\} \\
\left\{ C \leftarrow D + 1 \right\} \\
\left\{ F \leftarrow D + 1 \right\} \\
\left\{ R \leftarrow R \land (F \geq D + 1) \right\}
\end{align*}
\]

Figure 3.286: Automaton for the `DECREASING_WIDTH_INFLEXION` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `INFLEXION` pattern (transition `r \rightarrow t` has the same accumulators updates as transition `t \rightarrow r`)
### 3.144 DECREASING_WIDTH_PEAK

#### Description

**Origin**
Based on the PEAK pattern.

**Constraint**
`DECREASING_WIDTH_PEAK(VARIABLES)`

**Argument**
`VARIABLES : collection(var−dvar)`

**Restriction**
`required(VARIABLES, var)`

Succeeds if the values denoting the width of each occurrence of the PEAK pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**
An occurrence of the pattern PEAK is the *maximal* subsequence which matches the regular expression `< (= | <)* (> | =)* >`.

Assume that the occurrence of the pattern PEAK starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j − i`.

**Example**

```
((1, 6, 6, 6, 2, 5, 4, 3, 2, 2, 2, 2, 1, 5, 5, 7))
```

Figure 3.287 provides an example where the `DECREASING_WIDTH_PEAK` constraint holds.

**Typical**

```
|VARIABLES| > 2
range(VARIABLES.var) > 1
```
Figure 3.287: Illustrating the DECREASING_WIDTH.Peak constraint of the Example slot
Automaton

Figure 3.288 depicts the automaton associated with the constraint `DECREASING_WIDTH_PEAK`.

Figure 3.288: Automaton for the `DECREASING_WIDTH_PEAK` constraint obtained by applying decoration Table 2.36 to the seed transducer of the PEAK pattern.
3.145 DECREASING_WIDTH_PLAIN

**DESCRIPTION**

**Origin**
Based on the PLAIN pattern.

**Constraint**
DECREASING_WIDTH_PLAIN(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES, var)

**Purpose**
Succeeds if the values denoting the width of each occurrence of the PLAIN pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression \( >^* < \).

Assume that the occurrence of the pattern PLAIN starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

**Example**

\(((2, 3, 6, 5, 5, 7, 6, 4, 5, 5, 4, 3, 6, 6, 3))\)

Figure 3.289 provides an example where the DECREASING_WIDTH_PLAIN \(((2, 3, 6, 5, 5, 7, 6, 4, 5, 5, 4, 3, 6, 6, 3))\) constraint holds.

![Figure 3.289: Illustrating the DECREASING_WIDTH_PLAIN constraint of the Example slot](image-url)

Figure 3.289: Illustrating the DECREASING_WIDTH_PLAIN constraint of the Example slot.
Typical

$|\text{VARIABLES}| > 2$

$\text{range}(\text{VARIABLES}.\text{var}) > 1$
Automaton

Figure 3.290 depicts the automaton associated with the constraint DECREASING_WIDTH_PLAIN.

Figure 3.290: Automaton for the DECREASING_WIDTH_PLAIN constraint obtained by applying decoration Table 2.36 to the seed transducer of the PLAIN pattern.
3.146 DECREASING_WIDTH_PLATEAU

**DESCRIPTION**

**Origin**
Based on the PLATEAU pattern.

**Constraint**
DECREASING_WIDTH_PLATEAU(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES, var)

**Purpose**
Succeeds if the values denoting the width of each occurrence of the PLATEAU pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression \(<=\ast\rangle\).

Assume that the occurrence of the pattern PLATEAU starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

**Example**

\(((5, 2, 2, 5, 5, 4, 3, 3, 4, 2, 2, 1, 3, 2, 5, 7))\)

Figure 3.291 provides an example where the DECREASING_WIDTH_PLATEAU ((5, 2, 2, 5, 5, 4, 3, 3, 4, 2, 2, 1, 3, 2, 5, 7)) constraint holds.

**Typical**

\(\text{|VARIABLES|} > 2\)
\(\text{range(VARIABLES.var)} > 1\)
Figure 3.291: Illustrating the `DECREASING_WIDTH_PLATEAU` constraint of the `Example` slot
Figure 3.292 depicts the automaton associated with the constraint \textsc{decreasing_width_plateau}.

Figure 3.292: Automaton for the \textsc{decreasing_width_plateau} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textsc{plateau} pattern.
3.147 DECREASING_WIDTH_PROPER.PLAIN

**DESCRIPTION AUTOMATON**

**Origin**
Based on the PROPER.PLAIN pattern.

**Constraint**
DECREASING_WIDTH_PROPER.PLAIN(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES, var)

Succeeds if the values denoting the width of each occurrence of the PROPER.PLAIN pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**
An occurrence of the pattern PROPER.PLAIN is the maximal subsequence which matches the regular expression > = + <.
Assume that the occurrence of the pattern PROPER.PLAIN starts at position i and ends at position j. The feature WIDTH computes the value j − i.

**Example**

```
((2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5))
```

Figure 3.293 provides an example where the DECREASING_WIDTH_PROPER.PLAIN constraint holds.

![Diagram of variables and feature values](image)

Figure 3.293: Illustrating the DECREASING_WIDTH_PROPER.PLAIN constraint of the Example slot
Typical

| VARIABLES | > 3
\[\text{range(}\text{VARIABLES.var}) > 1 \]
Automaton

Figure 3.294 depicts the automaton associated with the constraint DECREASING_WIDTH_PROPERPlain.

Figure 3.294: Automaton for the DECREASING_WIDTH_PROPERPlain constraint obtained by applying decoration Table 2.36 to the seed transducer of the PROPERPlain pattern.
3.148 DECREASING_WIDTH_PROPER_PLATEAU

DESCRIPTION

Origin

Based on the PROPER_PLATEAU pattern.

Constraint

DECREASING_WIDTH_PROPER_PLATEAU(VARIABLES)

Argument

VARIABLES : collection(var−dvar)

Restriction

required(VARIABLES.var)

Purpose

Succeeds if the values denoting the width of each occurrence of the PROPER_PLATEAU pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression < =+ >.

Assume that the occurrence of the pattern PROPER_PLATEAU starts at position i and ends at position j. The feature WIDTH computes the value j − i.

Example

((3, 5, 5, 3, 2, 3, 4, 4, 1, 5, 2, 3, 3, 1, 7))

Figure 3.295 provides an example where the DECREASING_WIDTH_PROPER_PLATEAU ([(3, 5, 5, 3, 2, 3, 4, 4, 1, 5, 2, 3, 3, 1, 7)]) constraint holds.

Typical

|VARIABLES| > 3
range(VARIABLES.var) > 1
Figure 3.295: Illustrating the DECREASING_WIDTH_PROPER_PLATEAU constraint of the Example slot
Automaton Figure 3.296 depicts the automaton associated with the constraint `DECREASING_WIDTH_PROPER_PLATEAU`.

\[
\begin{align*}
\text{\{C } & \leftarrow +\infty \\
\text{\{D } & \leftarrow 0 \\
\text{\{F } & \leftarrow +\infty \\
\text{\{R } & \leftarrow 1 \\
\end{align*}
\text{\{D } & \leftarrow D + 1 \\
\text{\{F } & \leftarrow D + 1 \\
\text{\{R } & \leftarrow R \land (F \geq D + 1) \\
\end{align*}
\]

Figure 3.296: Automaton for the `DECREASING_WIDTH_PROPER_PLATEAU` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `PROPER_PLATEAU` pattern
3.149  DECREASING_WIDTH_STEADY_SEQUENCE

DESCRIPTION AUTOMATON

Origin
Based on the STEADY_SEQUENCE pattern.

Constraint
DECREASING_WIDTH_STEADY_SEQUENCE(VARIABLES)

Argument
VARIABLES : collection(var−dvar)

Restriction
required(VARIABLES, var)

Purpose
Succeeds if the values denoting the width of each occurrence of the STEADY_SEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern STEADY_SEQUENCE is the maximal subsequence which matches the regular expression \( ^* \).

Assume that the occurrence of the pattern STEADY_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i + 2 \).

Example
\((4, 3, 2, 5, 5, 3, 2, 4, 4, 6, 5, 3, 2, 2, 6)\)

Figure 3.297 provides an example where the DECREASING_WIDTH_STEADY_SEQUENCE ([4, 3, 2, 5, 5, 3, 2, 4, 4, 6, 5, 3, 2, 2, 6]) constraint holds.

Figure 3.297: Illustrating the DECREASING_WIDTH_STEADY_SEQUENCE constraint of the Example slot

Typical
\(|VARIABLES| > 1\)
DECREASING_WIDTH_STEADY_SEQUENCE

Automaton

Figure 3.298 depicts the automaton associated with the constraint DECREASING_WIDTH_STEADY_SEQUENCE.

\[
\begin{align*}
&\begin{cases}
C \leftarrow +\infty \\
D \leftarrow 0 \\
F \leftarrow +\infty \\
R \leftarrow 1
\end{cases} \\
&\begin{cases}
D \leftarrow 0 \\
F \leftarrow C \\
R \leftarrow R \land (F \geq C)
\end{cases}
\end{align*}
\]

Figure 3.298: Automaton for the DECREASING_WIDTH_STEADY_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the STEADY_SEQUENCE pattern.
### 3.150 DECREASING_WIDTH STRICTLY_DECREASING_SEQUENCE

#### DESCRIPTION

**Origin**
Based on the `STRICTLY_DECREASING_SEQUENCE` pattern.

**Constraint**
```
DECREASING_WIDTH_STRICTLY_DECREASING_SEQUENCE(VARIABLES)
```

**Argument**
```
VARIABLES : collection(var−dvar)
```

**Restriction**
```
required(VARIABLES, var)
```

Succeeds if the values denoting the width of each occurrence of the `STRICTLY_DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are decreasing.

**Purpose**
An occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` is the *maximal* subsequence which matches the regular expression `>⁺`. Assume that the occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j − i + 2`.

**Example**
```
((1, 1, 2, 2, 3, 2, 1, 5, 6, 6, 5, 4, 4, 6, 6, 1))
```

Figure 3.299 provides an example where the `DECREASING_WIDTH_STRICTLY_DECREASING_SEQUENCE` `((1, 1, 2, 2, 3, 2, 1, 5, 6, 6, 5, 4, 4, 6, 6, 1))` constraint holds.

**Typical**
```
|VARIABLES| > 1
range(VARIABLES.var) > 1
```
DECREASING
WIDTH
STRICTLY
DECREASING
SEQUENCE

3

= < < > > < = > = < >

VARIABLES

values

feature values (WIDTH)

Figure 3.299: Illustrating the DECREASING_WIDTH_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Automaton  Figure 3.300 depicts the automaton associated with the constraint \texttt{DECREASING_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE}.

\begin{align*}
C & \leftarrow +\infty \\
D & \leftarrow 0 \\
F & \leftarrow +\infty \\
R & \leftarrow 1
\end{align*}

\begin{align*}
\leq s & \leq \\
\begin{cases}
D & \leftarrow 0 \\
F & \leftarrow C \\
R & \leftarrow R \land (F \geq C)
\end{cases} & \begin{cases}
C & \leftarrow D + 2 \\
D & \leftarrow 0
\end{cases} \\
> R & >
\end{align*}

\begin{align*}
\{ D \leftarrow 0 \\
F \leftarrow C \\
R \leftarrow R \land (F \geq C) \} & \begin{cases}
C & \leftarrow C + D + 1 \\
D & \leftarrow 0
\end{cases}
\end{align*}

Figure 3.300: Automaton for the \texttt{DECREASING_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \texttt{STRICTLY\_DECREASING\_SEQUENCE} pattern.
### 3.151 DECREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th><strong>Automaton</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the STRICTLY_INCREASING_SEQUENCE pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>DECREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var−dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES, var)</td>
</tr>
</tbody>
</table>

Succeeds if the values denoting the width of each occurrence of the STRICTLY_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are decreasing.

**Purpose**
An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression \(<^+\).
Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i + 2\).

**Example**
\[
((3, 4, 5, 5, 4, 3, 1, 4, 3, 3, 2, 2, 1, 3, 1))
\]

Figure 3.301 provides an example where the DECREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE ((3, 4, 5, 5, 4, 3, 1, 4, 3, 3, 2, 2, 1, 3, 1)) constraint holds.

**Typical**
\[
|\text{VARIABLES}| > 1
\]
\[
\text{range}(\text{VARIABLES}\.var) > 1
\]
Figure 3.301: Illustrating the **DECREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE** constraint of the **Example** slot
Automaton

Figure 3.302 depicts the automaton associated with the constraint `DECREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE`.

\[
\begin{align*}
&\begin{cases}
C &\leftarrow +\infty \\
D &\leftarrow 0 \\
F &\leftarrow +\infty \\
R &\leftarrow 1
\end{cases} \\
\end{align*}
\]

\[
\begin{align*}
&\begin{cases}
D &\leftarrow 0 \\
F &\leftarrow C \\
R &\leftarrow R \land (F \geq C)
\end{cases} \\
\end{align*}
\]

\[
\begin{align*}
&\begin{cases}
C &\leftarrow D + 2 \\
D &\leftarrow 0
\end{cases} \\
\end{align*}
\]

\[
\begin{align*}
&\begin{cases}
C &\leftarrow C + D + 1 \\
D &\leftarrow 0
\end{cases} \\
\end{align*}
\]

Figure 3.302: Automaton for the `DECREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `STRICTLY_INCREASING_SEQUENCE` pattern.
### 3.152 DECREASING_WIDTH_SUMMIT

**DESCRIPTION**

Origin: Based on the SUMMIT pattern.

Constraint: \( \text{DECREASING\_WIDTH\_SUMMIT}(\text{VARIABLES}) \)

Argument: \( \text{VARIABLES} : \text{collection}(\text{var}\rightarrow\text{dvar}) \)

Restriction: \( \text{required}(\text{VARIABLES}, \text{var}) \)

**Purpose**

Succeeds if the values denoting the width of each occurrence of the SUMMIT pattern in the time-series given by the \( \text{VARIABLES} \) collection are decreasing.

An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression \( (< | (= | <) (< | > | > (= | >) ^* >) \).

Assume that the occurrence of the pattern SUMMIT starts at position \( i \) and ends at position \( j \). The feature \( \text{WIDTH} \) computes the value \( j - i \).

**Example**

\( \langle 1, 4, 4, 5, 4, 2, 6, 6, 2, 3, 5, 4, 1, 2, 4, 2 \rangle \)

Figure 3.303 provides an example where the DECREASING_WIDTH_SUMMIT \( \langle 1, 4, 4, 5, 4, 2, 6, 6, 2, 3, 5, 4, 1, 2, 4, 2 \rangle \) constraint holds.

![Figure 3.303: Illustrating the DECREASING_WIDTH_SUMMIT constraint of the Example slot](image-url)
Typical

\|VARIABLES\| > 2
\text{range}(\text{VARIABLES}.\text{var}) > 1
Automaton Figure 3.304 depicts the automaton associated with the constraint \textsc{decreasing_width_summit}.

\[
\begin{align*}
&\begin{cases}
C \leftarrow +\infty \\
D \leftarrow 0 \\
F \leftarrow +\infty \\
R \leftarrow 1
\end{cases} \\
\{D \leftarrow D + 1\} \\
\{C \leftarrow D + 1, D \leftarrow 0\} \\
= u \\
\{C \leftarrow D + 1, D \leftarrow 0\} \\
\{D \leftarrow D + 1\} \\
\{D \leftarrow D + 1\} \\
= u \\
\end{align*}
\]

\[
\begin{align*}
&\{D \leftarrow D + 1\} \\
\{C \leftarrow +\infty, D \leftarrow 0, F \leftarrow +\infty, R \leftarrow 1\} \\
\geq s \\
\end{align*}
\]

\[
\begin{align*}
\leq t \\
\end{align*}
\]

\[
\begin{align*}
\{D \leftarrow D + 1\} \\
\{C \leftarrow +\infty, D \leftarrow 0, F \leftarrow +\infty, R \leftarrow 1\} \\
\geq s \\
\end{align*}
\]

Figure 3.304: Automaton for the \textsc{decreasing_width_summit} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textsc{summit} pattern (transition \(u \rightarrow r\) has the same accumulator update as transition \(r \rightarrow u\))
3.153 **DECREASING_WIDTH_VALLEY**

**Origin**  
Based on the **VALLEY** pattern.

**Constraint**  
**DECREASING_WIDTH_VALLEY**(VARIABLES)

**Argument**  
VARIABLES : *collection*(var−dvar)

**Restriction**  
required(VARIABLES, var)

**Purpose**  
Succeeds if the values denoting the width of each occurrence of the **VALLEY** pattern in the time-series given by the VARIABLES collection are decreasing.  
An occurrence of the pattern **VALLEY** is the maximal subsequence which matches the regular expression \( > (= | >)^* (< | =)^* < \).  
Assume that the occurrence of the pattern **VALLEY** starts at position \( i \) and ends at position \( j \). The feature **WIDTH** computes the value \( j - i \).

**Example**  
\[ (1, 3, 3, 7, 6, 5, 4, 6, 6, 5, 4, 3, 6, 2, 2, 7) \]

Figure 3.305 provides an example where the **DECREASING_WIDTH_VALLEY** \([1, 3, 3, 7, 6, 5, 4, 6, 6, 5, 4, 3, 6, 2, 2, 7]\) constraint holds.

**Typical**  
\[ |VARIABLES| > 2 \]
\[ range(VARIABLES.var) > 1 \]
Figure 3.305: Illustrating the DECREASING_WIDTH_VALLEY constraint of the Example slot
Automaton

Figure 3.306 depicts the automaton associated with the constraint **DECREASING_WIDTH_VALLEY**.

![Automaton Diagram]

Figure 3.306: Automaton for the **DECREASING_WIDTH_VALLEY** constraint obtained by applying decoration Table 2.36 to the seed transducer of the VALLEY pattern.
### 3.154 DECREASING_WIDTH_ZIGZAG

<table>
<thead>
<tr>
<th><strong>DESCRIPTION</strong></th>
<th><strong>AUTOMATON</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the ZIGZAG pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>DECREASING_WIDTH_ZIGZAG(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var−dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES,var)</td>
</tr>
</tbody>
</table>

#### Purpose

Succeeds if the values denoting the width of each occurrence of the ZIGZAG pattern in the time-series given by the VARIABLES collection are decreasing.

An occurrence of the pattern ZIGZAG is the *maximal* subsequence which matches the regular expression \((<>)^+ (< | <>)(<>)^+ (> | >>)\).

Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

#### Example

```
((6, 5, 7, 4, 5, 1, 6, 3, 6, 4, 3, 5, 2, 7, 7))
```

Figure 3.307 provides an example where the DECREASING_WIDTH_ZIGZAG \(([6, 5, 7, 4, 5, 1, 6, 3, 6, 4, 3, 5, 2, 7, 7])\) constraint holds.

#### Typical

```
| VARIABLES | > 3
range(VARIABLES.var) | > 1
```
DECREASING_WIDTH_ZIGZAG

Figure 3.307: Illustrating the DECREASING_WIDTH_ZIGZAG constraint of the Example slot
Automaton Figure 3.308 depicts the automaton associated with the constraint DECREASING_WIDTH_ZIGZAG.
Figure 3.308: Automaton for the DECREASING_WIDTH_ZIGZAG constraint obtained by applying decoration Table 2.36 to the seed transducer of the ZIGZAG pattern; (1) missing transitions from \(a, b, c, d, e, f\) to \(s\) are labelled by \(=\); (2) on transitions from \(b, c, e, f\) to \(s\) the accumulator \(D\) is reset to its initial value; (3) on transitions from \(c, f\) to \(s\) the accumulator \(F\) is reset to \(C\), and the accumulator \(R\) is updated wrt \(C\) and \(F\).
3.155 INCREASING_HEIGHT_DECREATING_TERRACE

**DESCRIPTION**

**Origin**
Based on the DECREASING_TERRACE pattern.

**Constraint**
\[ \text{INCREASING_HEIGHT_DECREATING_TERRACE}(\text{VARIABLES}) \]

**Argument**
\[ \text{VARIABLES : collection}(\text{var} \rightarrow \text{dvar}) \]

**Restriction**
\[ \text{required}(\text{VARIABLES}, \text{var}) \]

**Purpose**
Succeeds if the minima of the values in each occurrence of the DECREASING_TERRACE pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern DECREASING_TERRACE is the maximal subsequence which matches the regular expression \[ \geq + > \].

Assume that the occurrence of the pattern DECREASING_TERRACE starts at position \( i \) and ends at position \( j \). The feature \( \text{MIN} \) computes the minimum of the values from index \( i + 1 \) to index \( j \).

**Example**
\[ ((5, 2, 2, 1, 5, 4, 3, 3, 2, 4, 4, 6, 5, 5, 4)) \]

Figure 3.309 provides an example where the INCREASING_HEIGHT_DECREATING_TERRACE \([(5, 2, 2, 1, 5, 4, 3, 3, 2, 4, 4, 6, 5, 5, 4)]) constraint holds.

**Typical**
\[ |\text{VARIABLES}| > 3 \]
\[ \text{range}(\text{VARIABLES.var}) > 2 \]
Figure 3.309: Illustrating the INCREASING_HEIGHT_DECREASING_TERRACE constraint of the Example slot
Automaton

Figure 3.310 depicts the automaton associated with the constraint INCREASING_HEIGHT_DECREASING_TERRACE.

![Automaton Diagram]

Figure 3.310: Automaton for the INCREASING_HEIGHT_DECREASING_TERRACE constraint obtained by applying decoration Table 2.36 to the seed transducer of the DECREASING_TERRACE pattern
### 3.156 INCREASING_HEIGHT_INCREASING_TERRACE

<table>
<thead>
<tr>
<th><strong>DESCRIPTION</strong></th>
<th><strong>AUTOMATON</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the INCREASING_TERRACE pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>INCREASING_HEIGHT_INCREASING_TERRACE(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var–dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES, var)</td>
</tr>
</tbody>
</table>

Succeeds if the minima of the values in each occurrence of the INCREASING_TERRACE pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression \(<=^+<\).

**Purpose**
Assume that the occurrence of the pattern INCREASING_TERRACE starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**

\[\{(1, 2, 2, 4, 5, 5, 6, 6, 4, 4, 2, 5, 5, 7, 7)\}\]

Figure 3.311 provides an example where the INCREASING_HEIGHT_INCREASING_TERRACE \([(1, 2, 2, 4, 5, 5, 6, 6, 4, 4, 2, 5, 5, 7, 7)]\) constraint holds.

**Typical**

\[|\text{VARIABLES}| > 3\]
\[\text{range}(\text{VARIABLES}.\text{var}) > 2\]
Figure 3.311: Illustrating the \texttt{INCREASING\_HEIGHT\_INCREASING\_TERRACE} constraint of the \texttt{Example} slot
Figure 3.312 depicts the automaton associated with the constraint INCREASING_HEIGHT_INCREASING_TERRACE.

\[
\begin{align*}
C & \leftarrow -\infty \\
D & \leftarrow +\infty \\
F & \leftarrow -\infty \\
R & \leftarrow 1 \\
\{D \leftarrow \min(D, \text{VAR}_i)\} & \quad \{D \leftarrow \min(D, \text{VAR}_i)\} \\
\{C \leftarrow \min(D, \text{VAR}_i)\} & \quad \{D \leftarrow \min(D, \text{VAR}_i)\} \\
\end{align*}
\]

The automaton is defined by the transition conditions:

- \(R \land (F \leq C)\)
- \(R \lor (F > C)\)
- \(R \lor (F = C)\)
- \(R \lor (F > C)\)
- \(R \lor (F = C)\)

Figure 3.312: Automaton for the INCREASING_HEIGHT_INCREASING_TERRACE constraint obtained by applying decoration Table 2.36 to the seed transducer of the INCREASING_TERRACE pattern.
3.157 INCREASING_HEIGHT.PLAIN

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
</table>

**Origin**
Based on the PLAIN pattern.

**Constraint**
INCREASING_HEIGHT.PLAIN(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES, var)

Succeeds if the minima of the values in each occurrence of the PLAIN pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**
An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression \( > = ^* < \).
Assume that the occurrence of the pattern PLAIN starts at position \( i \) and ends at position \( j \). The feature \( \text{MIN} \) computes the minimum of the values from index \( i + 1 \) to index \( j \).

**Example**
\( (3, 6, 6, 3, 4, 5, 5, 4, 6, 6, 7, 5, 6, 3, 2) \)

Figure 3.313 provides an example where the INCREASING_HEIGHT.PLAIN (\([3, 6, 6, 3, 4, 5, 5, 4, 6, 6, 7, 5, 6, 3, 2]\)) constraint holds.

![Diagram](image-url)
| Typical | \(|VARIABLES| > 2\)  
| range(VARIABLES.var) > 1 |
Figure 3.314 depicts the automaton associated with the constraint INCREASING_HEIGHTPLAIN.

Figure 3.314: Automaton for the INCREASING_HEIGHTPLAIN constraint obtained by applying decoration Table 2.36 to the seed transducer of the PLAIN pattern.
3.158 INCREASING_HEIGHT_PLATEAU

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
</table>

**Origin**

Based on the PLATEAU pattern.

**Constraint**

INCREASING_HEIGHT_PLATEAU(VARIABLES)

**Argument**

VARIABLES : collection(var-dvar)

**Restriction**

required(VARIABLES, var)

Succeeds if the minima of the values in each occurrence of the PLATEAU pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**

An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression `<=*`.

Assume that the occurrence of the pattern PLATEAU starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i + 1$ to index $j$.

**Example**

$$((7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5))$$

Figure 3.315 provides an example where the INCREASING_HEIGHT_PLATEAU constraint holds.

**Typical**

$$|\text{VARIABLES}| > 2$$

$$\text{range(VARIABLES.var)} > 1$$
INCREASING_HEIGHT_PLATEAU

Figure 3.315: Illustrating the INCREASING_HEIGHT_PLATEAU constraint of the Example slot
Automaton Figure 3.316 depicts the automaton associated with the constraint INCREASING_HEIGHT_PLATEAU.

\[
\begin{align*}
\{ D \leftarrow \min(D, \forall R_i) \} & \quad \{ D \leftarrow +\infty \} \\
\{ D \leftarrow \min(D, \forall R_i) \} & \quad \{ D \leftarrow +\infty \} \\
\{ D \leftarrow \min(D, \forall R_i) \} & \quad \{ D \leftarrow +\infty \} \\
\{ D \leftarrow \min(D, \forall R_i) \} & \quad \{ D \leftarrow +\infty \} \\
\end{align*}
\]

Figure 3.316: Automaton for the INCREASING_HEIGHT_PLATEAU constraint obtained by applying decoration Table 2.36 to the seed transducer of the PLATEAU pattern.
### 3.159 INCREASING_HEIGHT_PROPER_PLAIN

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td>Origin</td>
<td>Based on the PROPER_PLAIN pattern.</td>
</tr>
<tr>
<td>Constraint</td>
<td>INCREASING_HEIGHT_PROPER_PLAIN(VARIABLES)</td>
</tr>
<tr>
<td>Argument</td>
<td>VARIABLES : collection(var−dvar)</td>
</tr>
<tr>
<td>Restriction</td>
<td>required(VARIABLES.var)</td>
</tr>
</tbody>
</table>

**Purpose**

Succeeds if the minima of the values in each occurrence of the PROPER_PLAIN pattern in the time-series given by the VARIABLES collection are increasing. An occurrence of the pattern PROPER_PLAIN is the maximal subsequence which matches the regular expression > = <.

Assume that the occurrence of the pattern PROPER_PLAIN starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**

\(((5, 3, 3, 5, 6, 5, 4, 4, 7, 3, 6, 5, 5, 5, 7, 2))\)

Figure 3.317 provides an example where the INCREASING_HEIGHT_PROPER_PLAIN ([5, 3, 3, 5, 6, 5, 4, 4, 7, 3, 6, 5, 5, 5, 7, 2]) constraint holds.

**Typical**

\(|\text{VARIABLES}| > 3\)

\(\text{range(VARIABLES.var)} > 1\)
Figure 3.317: Illustrating the INCREASING_HEIGHT_PROPER_PLAIN constraint of the Example slot
Figure 3.318 depicts the automaton associated with the constraint increasing_height_proper_plain.

\[
\begin{align*}
\begin{cases}
C \leftarrow -\infty \\
D \leftarrow +\infty \\
F \leftarrow -\infty \\
R \leftarrow 1
\end{cases}
\end{align*}
\leq r = \{D \leftarrow \min(D, \text{VARi})\}
\leq \{D \leftarrow \min(D, \text{VARi})\}
\text{then}
\begin{cases}
C \leftarrow \text{min}(D, \text{VARi}) \\
D \leftarrow +\infty \\
F \leftarrow \text{min}(D, \text{VARi}) \\
R \leftarrow R \land (F \leq \text{min}(D, \text{VARi}))
\end{cases}
\end{align*}
\]
**3.160 INCREASING_HEIGHT_PROPER_PLATEAU**

<table>
<thead>
<tr>
<th>ORIGIN</th>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the PROPER_PLATEAU pattern.</td>
<td>$&lt; =^+ &gt;$</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>INCREASING_HEIGHT_PROPER_PLATEAU(VARIABLES)</td>
<td></td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var_dvar)</td>
<td></td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES, var)</td>
<td></td>
</tr>
</tbody>
</table>

Succeeds if the minima of the values in each occurrence of the PROPER_PLATEAU pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression $< =^+ >$.

Assume that the occurrence of the pattern PROPER_PLATEAU starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i + 1$ to index $j$.

**Example**

$((7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 5, 3))$

Figure 3.319 provides an example where the INCREASING_HEIGHT_PROPER_PLATEAU ($([7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 5])$) constraint holds.

**Typical**

$|\text{VARIABLES}| > 3$

$\text{range}(\text{VARIABLES.var}) > 1$
Figure 3.319: Illustrating the INCREASING_HEIGHT_PROPER_PLATEAU constraint of the Example slot.
Automaton

Figure 3.320 depicts the automaton associated with the constraint INCREASING_HEIGHT_PROPER_PLATEAU.
### 3.161 INCREASING_HEIGHT_STEADY

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the <strong>STEADY</strong> pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td><strong>INCREASING_HEIGHT_STEADY</strong>(<em>VARIABLES</em>)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td><strong>VARIABLES</strong> : <em>collection</em>(var–dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td><strong>required</strong>(VARIABLES, var)</td>
</tr>
</tbody>
</table>

Succeeds if the minima of the values in each occurrence of the **STEADY** pattern in the time-series given by the **VARIABLES** collection are increasing.

#### Purpose
An occurrence of the pattern **STEADY** is the subsequence which matches the regular expression `=`.

Assume that the occurrence of the pattern **STEADY** starts at position `i` and ends at position `j`. The feature **MIN** computes the minimum of the values from index `i` to index `j + 1`.

#### Example

```
((1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6))
```

Figure 3.321 provides an example where the **INCREASING_HEIGHT_STEADY** `([1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6])` constraint holds.

**Typical**

```
|VARIABLES| > 1
```
Figure 3.321: Illustrating the \texttt{INCREASING\_HEIGHT\_STEADY} constraint of the \texttt{Example} slot
Automaton Figure 3.322 depicts the automaton associated with the constraint INCREASING_HEIGHT_STEADY.

\[
\begin{align*}
&\begin{cases}
C \leftarrow -\infty \\
D \leftarrow +\infty \\
F \leftarrow -\infty \\
R \leftarrow 1
\end{cases} \\
\bigcup

&\begin{cases}
C \leftarrow \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
D \leftarrow +\infty \\
F \leftarrow \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
R \leftarrow R \land (F \leq \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}))
\end{cases}
\end{align*}
\]

Figure 3.322: Automaton for the INCREASING_HEIGHT_STEADY constraint obtained by applying decoration Table 2.36 to the seed transducer of the STEADY pattern
3.162  INCREASING_HEIGHT_STEADY_SEQUENCE

DESCRIPTION AUTOMATON

Origin
Based on the STEADY_SEQUENCE pattern.

Constraint
INCREASING_HEIGHT_STEADY_SEQUENCE(VARIABLES)

Argument
VARIABLES : collection(var–dvar)

Restriction
required(VARIABLES.var)

Purpose
Succeeds if the minima of the values in each occurrence of the STEADY_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern STEADY_SEQUENCE is the maximal subsequence which matches the regular expression $=^+$. Assume that the occurrence of the pattern STEADY_SEQUENCE starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i$ to index $j + 1$.

Example
$((6, 2, 2, 3, 5, 6, 4, 4, 2, 3, 5, 5, 2, 3, 4))$

Figure 3.323 provides an example where the INCREASING_HEIGHT_STEADY_SEQUENCE ($[6, 2, 2, 3, 5, 6, 4, 4, 2, 3, 5, 5, 2, 3, 4]$) constraint holds.

Figure 3.323: Illustrating the INCREASING_HEIGHT_STEADY_SEQUENCE constraint of the Example slot
Typical | $|\text{VARIABLES}| > 1$
Automaton

Figure 3.324 depicts the automaton associated with the constraint \textit{INCREASING\_HEIGHT\_STEADY\_SEQUENCE}.

\begin{align*}
C &\leftarrow -\infty \\
D &\leftarrow +\infty \\
F &\leftarrow -\infty \\
R &\leftarrow 1 \\
\end{align*}

\begin{align*}
C &\leftarrow \min(C, \min(D, \text{VAR}_i)) \\
D &\leftarrow +\infty
\end{align*}

Figure 3.324: Automaton for the \textit{INCREASING\_HEIGHT\_STEADY\_SEQUENCE} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textit{STEADY\_SEQUENCE} pattern
### 3.163 INCREASING_MAX_BUMP_ON_DECREASING_SEQUENCE

**DESCRIPTION**

Origin: Based on the `BUMP_ON_DECREASING_SEQUENCE` pattern.

Constraint: `INCREASING_MAX_BUMP_ON_DECREASING_SEQUENCE(VARIABLES)`

Argument: `VARIABLES : collection(var−dvar)`

Restriction: `required(VARIABLES, var)`

**Purpose**

Succeeds if the maxima of the values in each occurrence of the `BUMP_ON_DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are increasing.

An occurrence of the pattern `BUMP_ON_DECREASING_SEQUENCE` is the subsequence which matches the regular expression `>><<>`. Assume that the occurrence of the pattern `BUMP_ON_DECREASING_SEQUENCE` starts at position $i$ and ends at position $j$. The feature `MAX` computes the maximum of the values from index $i + 2$ to index $j$.

**Example**

```
(⟨7, 5, 4, 2, 5, 4, 3, 3, 5, 7, 6, 5, 6, 5, 4, 1⟩)
```

Figure 3.325 provides an example where the `INCREASING_MAX_BUMP_ON_DECREASING_SEQUENCE` constraint holds.

**Typical**

- `|VARIABLES| > 5`
- `range(VARIABLES.var) > 2`
Figure 3.325: Illustrating the \texttt{INCREASING\_MAX\_BUMP\_ON\_DECREASING\_SEQUENCE} constraint of the \texttt{Example} slot
Automaton Figure 3.326 depicts the automaton associated with the constraint INCREASING_MAX_BUMP_ON_DECREASING_SEQUENCE.

Figure 3.326: Automaton for the INCREASING_MAX_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern
INCREASING_MAX_BUMP_ON_DECREASING_SEQUENCE
3.164 INCREASING_MAX_DECREASING

**DESCRIPTION**

**Origin**
Based on the DECREASING pattern.

**Constraint**
INCREASING_MAX_DECREASING(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES, var)

Succeeds if the maxima of the values in each occurrence of the DECREASING pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**
An occurrence of the pattern DECREASING is the subsequence which matches the regular expression >.
Assume that the occurrence of the pattern DECREASING starts at position $i$ and ends at position $j$. The feature MAX computes the maximum of the values from index $i$ to index $j + 1$.

**Example**

\[
((2, 1, 1, 3, 2, 2, 3, 1, 1, 3, 4, 5, 6, 5))
\]

Figure 3.327 provides an example where the INCREASING_MAX_DECREASING ([2, 1, 1, 3, 2, 2, 3, 1, 1, 5, 3, 4, 5, 6, 5]) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1
\]
\[
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
Figure 3.327: Illustrating the INCREASING_MAX_DECREASING constraint of the Example slot
Automaton

Figure 3.328 depicts the automaton associated with the constraint INCREASING_MAX_DECREASING.

\[
\begin{align*}
  & C \leftarrow -\infty \\
  & D \leftarrow -\infty \\
  & F \leftarrow -\infty \\
  & R \leftarrow 1 \\
  \end{align*}
\]

\[
\begin{align*}
  C & \leftarrow \max(\max(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
  D & \leftarrow -\infty \\
  F & \leftarrow \max(\max(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
  R & \leftarrow R \land (F \leq \max(\max(D, \text{VAR}_i), \text{VAR}_{i+1})) \\
\end{align*}
\]

Figure 3.328: Automaton for the INCREASING_MAX_DECREASING constraint obtained by applying decoration Table 2.36 to the seed transducer of the DECREASING pattern.
### 3.165 INCREASING_MAX_DECREEASING_SEQUENCE

**Description**

- **Origin**: Based on the `DECREASING_SEQUENCE` pattern.
- **Constraint**: `INCREASING_MAX_DECREEASING_SEQUENCE(VARIABLES)`
- **Argument**: `VARIABLES : collection(var–dvar)`
- **Restriction**: `required(VARIABLES, var)`

**Purpose**

Succeeds if the maxima of the values in each occurrence of the `DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are increasing.

- An occurrence of the pattern `DECREASING_SEQUENCE` is the maximal subsequence which matches the regular expression `>(>|=)*>|`. Assume that the occurrence of the pattern `DECREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `MAX` computes the maximum of the values from index `i` to index `j + 1`.

**Example**

```plaintext
((4, 3, 2, 3, 4, 4, 6, 5, 5, 3, 4, 4, 6, 4, 6))
```

Figure 3.329 provides an example where the `INCREASING_MAX_DECREEASING_SEQUENCE` constraint holds.

**Typical**

- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`
Figure 3.329: Illustrating the INCREASING_MAX_DECREASING_SEQUENCE constraint of the Example slot
**Automaton**

Figure 3.330 depicts the automaton associated with the constraint **INCREASING_MAX_DECREASING_SEQUENCE**.

\[
\begin{align*}
C &\leftarrow -\infty \\
D &\leftarrow -\infty \\
F &\leftarrow -\infty \\
R &\leftarrow 1 \\
F &\leftarrow C \\
R &\leftarrow R \land (F \leq C) \\
\{D \leftarrow \max(D, \text{VAR}_{i+1})\} \\
C &\leftarrow \max(C, \max(D, \text{VAR}_{i+1})) \\
D &\leftarrow -\infty
\end{align*}
\]

Figure 3.330: Automaton for the **INCREASING_MAX_DECREASING_SEQUENCE** constraint obtained by applying decoration Table 2.36 to the seed transducer of the **DECREASING_SEQUENCE** pattern.
### 3.166 INCREASING_MAX_DIP_ON_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th>Origin</th>
<th>Based on the DIP_ON_INCREASING_SEQUENCE pattern.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constraint</td>
<td>\texttt{INCREASING_MAX_DIP_ON_INCREASING_SEQUENCE(VARIABLES)}</td>
</tr>
<tr>
<td>Argument</td>
<td>\texttt{VARIABLES : collection(var_dvar)}</td>
</tr>
<tr>
<td>Restriction</td>
<td>\texttt{required(VARIABLES, var)}</td>
</tr>
</tbody>
</table>

Succeeds if the maxima of the values in each occurrence of the \texttt{DIP\_ON\_INCREASING\_SEQUENCE} pattern in the time-series given by the \texttt{VARIABLES} collection are increasing.

**Purpose**

An occurrence of the pattern \texttt{DIP\_ON\_INCREASING\_SEQUENCE} is the subsequence which matches the regular expression \texttt{<><><}.

Assume that the occurrence of the pattern \texttt{DIP\_ON\_INCREASING\_SEQUENCE} starts at position \(i\) and ends at position \(j\). The feature \texttt{MAX} computes the maximum of the values from index \(i + 2\) to index \(j\).

**Example**

\[
((1, 2, 3, 2, 3, 4, 6, 5, 1, 4, 5, 6, 0, 2, 4, 4))
\]

Figure 3.331 provides an example where the \texttt{INCREASING\_MAX\_DIP\_ON\_INCREASING\_SEQUENCE} \([1, 2, 3, 2, 3, 4, 6, 5, 1, 4, 5, 6, 0, 2, 4, 4]) constraint holds.

**Typical**

\[
|\texttt{VARIABLES}| > 5
\]
\[
\texttt{range(VARIABLES.var)} > 2
\]
Figure 3.331: Illustrating the **INCREASING_MAX_DIP_ON_INCREASING_SEQUENCE** constraint of the **Example** slot
Automaton

Figure 3.332 depicts the automaton associated with the constraint \textit{INCREASING_MAX_DIP_ON_INCREASING_SEQUENCE}.

Figure 3.332: Automaton for the \textit{INCREASING_MAX_DIP_ON_INCREASING_SEQUENCE} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textit{DIP_ON_INCREASING_SEQUENCE} pattern.
INCREASING_MAX_DIP_ON_INCREASING_SEQUENCE
### INCREASING_MAX_INCREASING

#### Description

**Origin**
Based on the **INCREASING** pattern.

**Constraint**

\[ \text{INCREASING}\_\text{MAX}\_\text{INCREASING}(\text{VARIABLES}) \]

**Argument**

\[ \text{VARIABLES} : \text{collection}(\text{var} - \text{dvar}) \]

**Restriction**

\[ \text{required}(\text{VARIABLES}, \text{var}) \]

Succeeds if the maxima of the values in each occurrence of the **INCREASING** pattern in the time-series given by the **VARIABLES** collection are increasing.

**Purpose**

An occurrence of the pattern **INCREASING** is the subsequence which matches the regular expression \(<\).

Assume that the occurrence of the pattern **INCREASING** starts at position \(i\) and ends at position \(j\). The feature **MAX** computes the maximum of the values from index \(i\) to index \(j + 1\).

**Example**

\[ ((2, 1, 2, 0, 2, 3, 1, 1, 3, 4, 2, 1, 1, 0, 6, 7)) \]

Figure 3.333 provides an example where the **INCREASING_MAX_INCREASING ([2, 1, 2, 0, 2, 3, 1, 1, 3, 4, 2, 1, 1, 0, 6, 7])** constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 1 \]

\[ \text{range}(\text{VARIABLES} \cdot \text{var}) > 1 \]
Figure 3.333: Illustrating the INCREASING_MAX_INCREASING constraint of the Example slot
Figure 3.334 depicts the automaton associated with the constraint INCREASING_MAX_INCREASING.

\[
\begin{align*}
\text{Initial State: } & S \\
\text{Transitions: } & R \leftarrow R \land (F \leq \max(\max(D, \text{VAR}_i), \text{VAR}_{i+1})) \\
\text{States: } & \{ C \leftarrow -\infty \}
\end{align*}
\]

Figure 3.334: Automaton for the INCREASING_MAX_INCREASING constraint obtained by applying decoration Table 2.36 to the seed transducer of the INCREASING pattern.
### 3.168 INCREASING_MAX_INCREASING_SEQUENCE

#### Description

**Origin**
Based on the INCREASING_SEQUENCE pattern.

**Constraint**

\[
\text{INCREASING\_MAX\_INCREASING\_SEQUENCE(VARIABLES)}
\]

**Argument**

\[
\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})
\]

**Restriction**

\[
\text{required(VARIABLES, var)}
\]

Succeeds if the maxima of the values in each occurrence of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**

An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression \(< (< | =)^* < | <\). Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
((4, 1, 4, 5, 5, 4, 4, 2, 3, 3, 4, 5, 6, 6, 5, 6))
\]

Figure 3.335 provides an example where the INCREASING_MAX_INCREASING_SEQUENCE ([4, 1, 4, 5, 5, 4, 4, 2, 3, 3, 4, 5, 6, 6, 5, 6]) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1
\]

\[
\text{range(VARIABLES.var)} > 1
\]
Figure 3.335: Illustrating the INCREASING_MAX_INCREASING_SEQUENCE constraint of the Example slot
Figure 3.336 depicts the automaton associated with the constraint \textsc{increasing\_max\_increasing\_sequence}.

\[
\begin{align*}
C & \leftarrow -\infty \\
D & \leftarrow -\infty \\
F & \leftarrow -\infty \\
R & \leftarrow 1
\end{align*}
\]

\[
\{ D \leftarrow \max(D, \text{VAR}_{i+1}) \}
\]

\[
\{ C \leftarrow \max(C, \max(D, \text{VAR}_{i+1})) \}
\]

\[
\{ C \leftarrow \max(D, \text{VAR}_{i+1}) \}
\]

\[
\{ D \leftarrow -\infty \}
\]

Figure 3.336: Automaton for the \textsc{increasing\_max\_increasing\_sequence} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textsc{increasing\_sequence} pattern
3.169 INCREASING_MAX_INFLEXION

**DESCRIPTION**

Origin

Based on the INFLEXION pattern.

Constraint

\[ \text{INCREASING\_MAX\_INFLEXION}(\text{VARIABLES}) \]

Argument

\[ \text{VARIABLES} : \text{collection} (\text{var} - \text{dvar}) \]

Restriction

\[ \text{required}(\text{VARIABLES}, \text{var}) \]

Purpose

Succeeds if the maxima of the values in each occurrence of the INFLEXION pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression \(< (< | =) > | > (| =) > \) <.

Assume that the occurrence of the pattern INFLEXION starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i + 1\) to index \(j\).

Example

\[ (5, 5, 2, 2, 1, 2, 2, 3, 4, 3) \]

Figure 3.337 provides an example where the INCREASING_MAX_INFLEXION ([5, 5, 2, 2, 1, 2, 2, 3, 4, 3]) constraint holds.

![Diagram](figure3.337.png)

Figure 3.337: Illustrating the INCREASING_MAX_INFLEXION constraint of the **Example** slot
Typical

<table>
<thead>
<tr>
<th>VARIABLES</th>
<th>&gt; 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>range(VARIABLES.var)</td>
<td>&gt; 1</td>
</tr>
</tbody>
</table>
Figure 3.338 depicts the automaton associated with the constraint INCREASING_MAX_INFLEXION.

Figure 3.338: Automaton for the INCREASING_MAX_INFLEXION constraint obtained by applying decoration Table 2.36 to the seed transducer of the INFLEXION pattern (transition $r \rightarrow t$ has the same accumulators updates as transition $t \rightarrow r$)
3.170 INCREASING_MAX_PEAK

**DESCRIPTION**

**Origin**
Based on the PEAK pattern.

**Constraint**
INCREASING_MAX_PEAK(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES, var)

**Purpose**
Succeeds if the maxima of the values in each occurrence of the PEAK pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression < (= | <) *( > | =) >.

Assume that the occurrence of the pattern PEAK starts at position $i$ and ends at position $j$. The feature MAX computes the maximum of the values from index $i + 1$ to index $j$.

**Example**

$((7, 5, 5, 1, 2, 3, 2, 3, 4, 5, 2, 6, 6, 6, 1))$

Figure 3.339 provides an example where the INCREASING_MAX_PEAK ([(7, 5, 5, 1, 2, 3, 2, 3, 4, 5, 2, 6, 6, 6, 1)]) constraint holds.

![Figure 3.339: Illustrating the INCREASING_MAX_PEAK constraint of the Example slot](image)
**Typical**

| VARIABLES | > 2  
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>range(VARIABLES.var)</td>
<td>&gt; 1</td>
</tr>
</tbody>
</table>
Automaton

Figure 3.340 depicts the automaton associated with the constraint INCREASING_MAX_PEAK.

Figure 3.340: Automaton for the INCREASING_MAX_PEAK constraint obtained by applying decoration Table 2.36 to the seed transducer of the PEAK pattern.
### 3.171 INCREASING_MAX_STRICTLY_DECREASING_SEQUENCE

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the <strong>STRICTLY_DECREASING_SEQUENCE</strong> pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td><strong>INCREASING_MAX_STRICTLY_DECREASING_SEQUENCE</strong>(<code>VARIABLES</code>)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td><code>VARIABLES : collection(var−dvar)</code></td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td><code>required(VARIABLES, var)</code></td>
</tr>
<tr>
<td><strong>Purpose</strong></td>
<td>Succeeds if the maxima of the values in each occurrence of the <strong>STRICTLY_DECREASING_SEQUENCE</strong> pattern in the time-series given by the <code>VARIABLES</code> collection are increasing. An occurrence of the pattern <strong>STRICTLY_DECREASING_SEQUENCE</strong> is the <code>maximal</code> subsequence which matches the regular expression <code>&gt;^</code>. Assume that the occurrence of the pattern <strong>STRICTLY_DECREASING_SEQUENCE</strong> starts at position <code>i</code> and ends at position <code>j</code>. The feature <code>MAX</code> computes the maximum of the values from index <code>i</code> to index <code>j + 1</code>.</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td><code>((2, 3, 2, 3, 4, 5, 5, 6, 4, 3, 2, 4, 7, 4))</code></td>
</tr>
</tbody>
</table>
| **Typical** | `|VARIABLES| > 1`  
`range(VARIABLES.var) > 1` |

Figure 3.341 provides an example where the **INCREASING_MAX_STRICTLY_DECREASING_SEQUENCE** constraint holds.
Figure 3.341: Illustrating the `INCREASING_MAX_STRICTLY_DECREASING_SEQUENCE` constraint of the `Example` slot
Automaton Figure 3.342 depicts the automaton associated with the constraint INCREASING_MAX_STRICTLY_DECREASING_SEQUENCE.

\[
\begin{align*}
&\{ C \leftarrow -\infty \} \\
&\{ D \leftarrow -\infty \} \\
&\{ F \leftarrow -\infty \} \\
&\{ R \leftarrow 1 \}
\end{align*}
\]

Figure 3.342: Automaton for the INCREASING_MAX_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the STRICTLY_DECREEASING_SEQUENCE pattern.
INCREASING_MAX.StrictLY_DECRESSING_SEQUENCE
3.172 INCREASING_MAX_STRICTLY_INCREASING_SEQUENCE

**DESCRIPTION**

**Origin**
Based on the STRICTLY_INCREASING_SEQUENCE pattern.

**Constraint**
INCREASING_MAX_STRICTLY_INCREASING_SEQUENCE(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES, var)

**Purpose**
Succeeds if the maxima of the values in each occurrence of the STRICTLY_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression <+
Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i to index j + 1.

**Example**

$((4, 1, 4, 5, 5, 4, 4, 2, 3, 4, 5, 6, 6, 5, 6))$

Figure 3.343 provides an example where the INCREASING_MAX_STRICTLY_INCREASING_SEQUENCE $((4, 1, 4, 5, 5, 4, 4, 2, 2, 3, 4, 5, 6, 6, 5, 6))$ constraint holds.

**Typical**

$|\text{VARIABLES}| > 1$

\text{range}(\text{VARIABLES}.var) > 1$
Figure 3.343: Illustrating the INCREASING_MAX, STRICTLY_INCREASING_SEQUENCE constraint of the Example slot
Figure 3.344 depicts the automaton associated with the constraint INCREASING_MAX_STRICTLY_INCREASING_SEQUENCE.

\[
\begin{align*}
D & \leftarrow -\infty \\
F & \leftarrow -\infty \\
R & \leftarrow 1 \\
C & \leftarrow -\infty \\
\end{align*}
\]

Figure 3.344: Automaton for the INCREASING_MAX_STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern.
### 3.173 INCREASING_MAX_SUMMIT

<table>
<thead>
<tr>
<th><strong>DESCRIPTION</strong></th>
<th><strong>AUTOMATON</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>In the given time-series, the maxima of the values in each occurrence of the pattern SUMMIT are increasing.</td>
<td></td>
</tr>
</tbody>
</table>

- **Origin**: Based on the SUMMIT pattern.
- **Constraint**: `INCREASING_MAX_SUMMIT(VARIABLES)`
- **Argument**: `VARIABLES : collection(var–dvar)`
- **Restriction**: `required(VARIABLES, var)`

Succeeds if the maxima of the values in each occurrence of the SUMMIT pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression `(\(< | (\backslash = | <)\)\)\(\backslash > | (\backslash = | >)\)\)`.

Assume that the occurrence of the pattern SUMMIT starts at position `i` and ends at position `j`. The feature MAX computes the maximum of the values from index `i + 1` to index `j`.

**Example**

```
((1, 5, 2, 1, 6, 6, 2, 3, 5, 4, 1, 4, 6, 4, 3, 2))
```

Figure 3.345 provides an example where the `INCREASING_MAX_SUMMIT` constraint holds.

**Typical**

- `|VARIABLES| > 2`
- `range(VARIABLES.var) > 1`
Figure 3.345: Illustrating the INCREASING_MAX_SUMMIT constraint of the Example slot
Automaton

Figure 3.346 depicts the automaton associated with the constraint `INCREASING_MAX_SUMMIT`.

\[
\begin{align*}
C & \leftarrow -\infty \\
D & \leftarrow -\infty \\
F & \leftarrow -\infty \\
R & \leftarrow 1
\end{align*}
\]

\[
\begin{align*}
{\{ D \leftarrow \max(D, \text{VAR}_i) \}} \\
{\{ D \leftarrow \max(D, \text{VAR}_i) \} \\
{\{ C \leftarrow \max(C, \max(D, \text{VAR}_i)) \} \\
{\{ D \leftarrow -\infty \}}
\end{align*}
\]

\[
\begin{align*}
{\{ C \leftarrow \max(D, \text{VAR}_i) \} \\
{\{ D \leftarrow \max(D, \text{VAR}_i) \} \\
{\{ C \leftarrow \max(C, \max(D, \text{VAR}_i)) \} \\
{\{ D \leftarrow -\infty \}}
\end{align*}
\]

\[
\begin{align*}
\geq & \quad \{ D \leftarrow \max(D, \text{VAR}_i) \} \\
\leq & \quad \{ D \leftarrow \max(D, \text{VAR}_i) \} \\
= & \quad \{ D \leftarrow \max(D, \text{VAR}_i) \} \\
< & \quad \{ D \leftarrow \max(D, \text{VAR}_i) \}
\end{align*}
\]

\[
\begin{align*}
R \land (F \leq C) & \quad \{ D \leftarrow \max(D, \text{VAR}_i) \} \\
R \land (F \leq C) & \quad \{ D \leftarrow \max(D, \text{VAR}_i) \} \\
R \land (F \leq C) & \quad \{ D \leftarrow \max(D, \text{VAR}_i) \} \\
R \land (F \leq C) & \quad \{ D \leftarrow \max(D, \text{VAR}_i) \}
\end{align*}
\]

\[
\begin{align*}
R \land (F \leq C) & \quad \{ D \leftarrow \max(D, \text{VAR}_i) \} \\
R \land (F \leq C) & \quad \{ D \leftarrow \max(D, \text{VAR}_i) \} \\
R \land (F \leq C) & \quad \{ D \leftarrow \max(D, \text{VAR}_i) \} \\
R \land (F \leq C) & \quad \{ D \leftarrow \max(D, \text{VAR}_i) \}
\end{align*}
\]

Figure 3.346: Automaton for the `INCREASING_MAX_SUMMIT` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `SUMMIT` pattern (transition \( u \rightarrow r \) has the same accumulator update as transition \( r \rightarrow u \) )
<table>
<thead>
<tr>
<th><strong>DESCRIPTION</strong></th>
<th><strong>AUTOMATON</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the ZIGZAG pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>INCREASING_MAX_ZIGZAG(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var−dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES, var)</td>
</tr>
</tbody>
</table>

**Purpose**

Succeeds if the maxima of the values in each occurrence of the ZIGZAG pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression \((<>)^+ (<|<>)|(<>)^+ (>|><))\).

Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i + 1\) to index \(j\).

**Example**

\[ (7, 7, 2, 5, 3, 4, 6, 3, 6, 1, 1, 5, 4, 7, 5, 6) \]

Figure 3.347 provides an example where the INCREASING_MAX_ZIGZAG \(([7, 7, 2, 5, 3, 4, 6, 3, 6, 1, 1, 5, 4, 7, 5, 6])\) constraint holds.

**Typical**

<table>
<thead>
<tr>
<th>VARIABLES</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>range(VARIABLES.var)</td>
<td>1</td>
</tr>
</tbody>
</table>
Figure 3.347: Illustrating the INCREASING_MAX_ZIGZAG constraint of the Example slot
Automaton

Figure 3.348 depicts the automaton associated with the constraint INCREASING_MAX_ZIGZAG.
Figure 3.348: Automaton for the INCREASING_MAX_ZIGZAG constraint obtained by applying decoration Table 2.36 to the seed transducer of the ZIGZAG pattern; (1) missing transitions from a, b, c, d, e, f to s are labelled by =; (2) on transitions from b, c, e, f to s the accumulator D is reset to its initial value; (3) on transitions from c, f to s the accumulator F is reset to C, and the accumulator R is updated wrt C and F.
3.175 **INCREASING_MIN_BUMP_ON_DECREASING_SEQUENCE**

**Description**

**Origin**
Based on the `BUMP_ON_DECREASING_SEQUENCE` pattern.

**Constraint**

\[
\text{INCREASING_MIN_BUMP_ON_DECREASING_SEQUENCE(\text{VARIABLES})}
\]

**Argument**

\[
\text{VARIABLES : collection(var−dvar)}
\]

**Restriction**

\[
\text{required(\text{VARIABLES, var})}
\]

Succeeds if the minima of the values in each occurrence of the `BUMP_ON_DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are increasing.

**Purpose**

An occurrence of the pattern `BUMP_ON_DECREASING_SEQUENCE` is the subsequence which matches the regular expression `>><<>>`. Assume that the occurrence of the pattern `BUMP_ON_DECREASING_SEQUENCE` starts at position \(i\) and ends at position \(j\). The feature `MIN` computes the minimum of the values from index \(i + 2\) to index \(j\).

**Example**

\[
((5, 5, 4, 2, 6, 5, 4, 5, 7, 6, 5, 6, 4, 1, 1))
\]

Figure 3.349 provides an example where the `INCREASING_MIN_BUMP_ON_DECREASING_SEQUENCE (\{5, 5, 4, 2, 6, 5, 4, 5, 7, 6, 5, 6, 4, 1, 1\})` constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 5 \\
\text{range(\text{VARIABLES.var})} > 2
\]
INCREASING_MIN_BUMP_ON_DECREASING_SEQUENCE

Figure 3.349: Illustrating the INCREASING_MIN_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot
Figure 3.350 depicts the automaton associated with the constraint INCREASING_MIN_BUMP_ON_DECREASING_SEQUENCE.

Figure 3.350: Automaton for the INCREASING_MIN_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern.
### 3.176 INCREASING_MIN_DECENDING

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the <strong>DECREASING</strong> pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td><strong>INCREASING_MIN_DECENDING</strong>(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var−dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES, var)</td>
</tr>
<tr>
<td><strong>Purpose</strong></td>
<td>Succeeds if the minima of the values in each occurrence of the <strong>DECREASING</strong> pattern in the time-series given by the VARIABLES collection are increasing. An occurrence of the pattern <strong>DECREASING</strong> is the subsequence which matches the regular expression &gt;. Assume that the occurrence of the pattern <strong>DECREASING</strong> starts at position i and ends at position j. The feature <strong>MIN</strong> computes the minimum of the values from index i to index j + 1.</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>$((3, 6, 6, 1, 1, 3, 3, 4, 5, 6, 1, 5, 3, 3, 6, 5))$</td>
</tr>
<tr>
<td><strong>Typical</strong></td>
<td>$</td>
</tr>
</tbody>
</table>

Figure 3.351 provides an example where the INCREASING_MIN_DECENDING $((3, 6, 6, 1, 1, 3, 3, 4, 5, 6, 1, 5, 3, 3, 6, 5))$ constraint holds.
Figure 3.351: Illustrating the INCREASING_MIN_DECREASING constraint of the Example slot
Figure 3.352 depicts the automaton associated with the constraint INCREASING_MIN_DECREASING.

\[
\begin{array}{l}
\{ C ← \infty, \\
D ← +\infty, \\
F ← -\infty, \\
R ← 1 \}
\end{array}
\]

\[
\begin{array}{l}
C ← \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
D ← +\infty \\
F ← \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
R ← R \land (F ≤ \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}))
\end{array}
\]

Figure 3.352: Automaton for the INCREASING_MIN_DECREASING constraint obtained by applying decoration Table 2.36 to the seed transducer of the DECREASING pattern.
3.177 INCREASING_MIN_DECREASING_SEQUENCE

**DESCRIPTION AUTOMATON**

**Origin**
Based on the DECREASING_SEQUENCE pattern.

**Constraint**

INCREASING_MIN_DECREASING_SEQUENCE(VARIABLES)

**Argument**

VARIABLES : collection(var−dvar)

**Restriction**

required(VARIABLES, var)

Succeeds if the minima of the values in each occurrence of the DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**

An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression > (> | =)* > | >.

Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i$ to index $j + 1$.

**Example**

$((4, 3, 2, 3, 4, 4, 6, 3, 4, 4, 5, 5, 6, 4))$

Figure 3.353 provides an example where the INCREASING_MIN_DECREASING_SEQUENCE $([4, 3, 2, 3, 4, 4, 6, 3, 4, 4, 5, 5, 6, 4])$ constraint holds.

Figure 3.353: Illustrating the INCREASING_MIN_DECREASING_SEQUENCE constraint of the Example slot
Typical

\[
|\text{VARIABLES}| > 1 \\
\text{range(VARIABLES.var)} > 1
\]
Automaton

Figure 3.354 depicts the automaton associated with the constraint INCREASING_MIN_DECREASING_SEQUENCE.

\[
\begin{align*}
C & \leftarrow -\infty \\
D & \leftarrow +\infty \\
F & \leftarrow -\infty \\
R & \leftarrow 1
\end{align*}
\]

\[
\begin{align*}
& \leq s \\
& \begin{cases}
D \leftarrow +\infty \\
F \leftarrow C \\
R \leftarrow R \land (F \leq C)
\end{cases} \\
& \begin{cases}
C \leftarrow \min(D, \text{VAR}_i) \\
D \leftarrow +\infty
\end{cases} \\
& \begin{cases}
D \leftarrow \min(D, \text{VAR}_i + 1)
\end{cases}
\]

\[
\begin{align*}
& \geq t \\
& \begin{cases}
C \leftarrow \min(C, \min(D, \text{VAR}_i + 1)) \\
D \leftarrow +\infty
\end{cases} \\
& \begin{cases}
D \leftarrow \min(D, \text{VAR}_i + 1)
\end{cases}
\]

Figure 3.354: Automaton for the INCREASING_MIN_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the DECREASING_SEQUENCE pattern
### 3.178 INCREASING_MIN_DIP_ON_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the <code>DIP_ON_INCREASING_SEQUENCE</code> pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td><code>INCREASING_MIN_DIP_ON_INCREASING_SEQUENCE(VARIABLES)</code></td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td><code>VARIABLES : collection(var–dvar)</code></td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td><code>required(VARIABLES,var)</code></td>
</tr>
</tbody>
</table>
| **Purpose** | Succeeds if the minima of the values in each occurrence of the `DIP_ON_INCREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are increasing. 
An occurrence of the pattern `DIP_ON_INCREASING_SEQUENCE` is the subsequence which matches the regular expression `<><<<`. 
Assume that the occurrence of the pattern `DIP_ON_INCREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i + 2` to index `j`. |
| **Example** | `((1, 2, 3, 0, 3, 4, 6, 5, 1, 4, 5, 6, 1, 2, 4, 4))` |
| **Typical** | `|VARIABLES| > 5` 
`range(VARIABLES.var) > 2` |

Figure 3.355 provides an example where the `INCREASING_MIN_DIP_ON_INCREASING_SEQUENCE` constraint holds.
Figure 3.355: Illustrating the **INCREASING_MIN_DIP_ON_INCREASING_SEQUENCE** constraint of the **Example** slot
Figure 3.356 depicts the automaton associated with the constraint INCREASING_MIN_DIP_ON_INCREASING_SEQUENCE.

Automaton

\[
\begin{align*}
&\{ C \leftarrow -\infty \\
& D \leftarrow +\infty \\
& F \leftarrow -\infty \\
& R \leftarrow 1 \\
\} \\
\begin{array}{c}
\geq \\
\{ D \leftarrow \min(D, \text{VAR}_i) \}
\end{array} \rightarrow
\begin{array}{c}
\leq \\
\{ D \leftarrow +\infty \}
\end{array} \rightarrow
\begin{array}{c}
\geq \\
\{ D \leftarrow \min(D, \text{VAR}_i) \}
\end{array} \rightarrow
\begin{array}{c}
\leq \\
\{ D \leftarrow +\infty \}
\end{array}
\]

Figure 3.356: Automaton for the INCREASING_MIN_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern.
3.179 INCREASING_MIN_GORGE

**DESCRIPTION**

**Origin**

Based on the GORGE pattern.

**Constraint**

INCREASING_MIN_GORGE(VARIABLES)

**Argument**

VARIABLES : collection(var−dvar)

**Restriction**

required(VARIABLES, var)

**Purpose**

Succeeds if the minima of the values in each occurrence of the GORGE pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression \((> | > (= | >)^* >) (< | < (= | <)^* <)\).

Assume that the occurrence of the pattern GORGE starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**

\(((6, 2, 5, 1, 1, 5, 4, 2, 3, 6, 5, 4, 3, 5, 5))\)

Figure 3.357 provides an example where the INCREASING_MIN_GORGE \(([6, 2, 5, 1, 1, 5, 4, 2, 3, 6, 5, 4, 3, 5, 5])\) constraint holds.

![Figure 3.357: Illustrating the INCREASING_MIN_GORGE constraint of the Example slot](image-url)
**Typical**

- $|\text{VARIABLES}| > 2$
- $\text{range}(\text{VARIABLES.var}) > 1$
Figure 3.358 depicts the automaton associated with the constraint INCREASING_MIN_GORGE.

Figure 3.358: Automaton for the INCREASING_MIN_GORGE constraint obtained by applying decoration Table 2.36 to the seed transducer of the GORGE pattern (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$)
### 3.180 INCREASING_MIN_INCREASING

**Description**

Based on the INCREASING pattern.

**Constraint**

INCREASING_MIN_INCREASING(VARIABLES)

**Argument**

VARIABLES : collection(var–dvar)

**Restriction**

required(VARIABLES, var)

Succeeds if the minima of the values in each occurrence of the INCREASING pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**

An occurrence of the pattern INCREASING is the subsequence which matches the regular expression <.

Assume that the occurrence of the pattern INCREASING starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i to index j + 1.

**Example**

\( ((4, 1, 2, 4, 3, 3, 2, 3, 3), 4, 4, 6, 6, 5, 7) \)

Figure 3.359 provides an example where the INCREASING_MIN_INCREASING \(([4, 1, 2, 4, 3, 3, 2, 3, 3, 4, 4, 6, 6, 5, 7])\) constraint holds.

**Typical**

\(|\text{VARIABLES}| > 1\)

\(\text{range}(\text{VARIABLES.var}) > 1\)
Figure 3.359: Illustrating the INCREASING_MIN_INCREASING constraint of the Example slot
Automaton

Figure 3.360 depicts the automaton associated with the constraint INCREASING_MIN_INCREASING.

\[
\begin{align*}
C &\leftarrow -\infty \\
D &\leftarrow +\infty \\
F &\leftarrow -\infty \\
R &\leftarrow 1
\end{align*}
\]

\[
\begin{align*}
\{ C &\leftarrow \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
D &\leftarrow +\infty \\
F &\leftarrow \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
R &\leftarrow R \land (F \leq \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1})) \}
\end{align*}
\]

Figure 3.360: Automaton for the INCREASING_MIN_INCREASING constraint obtained by applying decoration Table 2.36 to the seed transducer of the INCREASING pattern.
### 3.181 INCREASING_MIN_INCREASING_SEQUENCE

**Description**

Based on the INCREASING_SEQUENCE pattern.

**Constraint**

`INCREASING_MIN_INCREASING_SEQUENCE(VARIABLES)`

**Argument**

`VARIABLES : collection(var−dvar)`

**Restriction**

`required(VARIABLES, var)`

**Purpose**

Succeeds if the minima of the values in each occurrence of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern `INCREASING_SEQUENCE` is the *maximal* subsequence which matches the regular expression `< (< | =)* < | <`. Assume that the occurrence of the pattern `INCREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i` to index `j + 1`.

**Example**

```
((4, 1, 2, 4, 3, 3, 2, 3, 4, 4, 6, 6, 5, 7))
```

Figure 3.361 provides an example where the `INCREASING_MIN_INCREASING_SEQUENCE` (`[4, 1, 2, 4, 3, 3, 2, 3, 4, 4, 6, 6, 5, 7]`) constraint holds.

**Typical**

`|VARIABLES| > 1`

`range(VARIABLES.var) > 1`
Figure 3.361: Illustrating the INCREASING_MIN_INCREMENTING_SEQUENCE constraint of the Example slot
Automaton

Figure 3.362 depicts the automaton associated with the constraint INCREASING_MIN_INCREASING_SEQUENCE.

\[
\begin{align*}
C &\leftarrow -\infty \\
D &\leftarrow +\infty \\
F &\leftarrow -\infty \\
R &\leftarrow 1
\end{align*}
\]

\[
\begin{cases}
\geq s &
\begin{cases}
D &\leftarrow +\infty \\
F &\leftarrow C \\
R &\leftarrow R \land (F \leq C)
\end{cases} \\
< &
\begin{cases}
C &\leftarrow \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
D &\leftarrow +\infty
\end{cases}
\end{cases}
\]

\[
\begin{cases}
\leq t &
\begin{cases}
D &\leftarrow \min(D, \text{VAR}_{i+1}) \\
C &\leftarrow \min(C, \min(D, \text{VAR}_{i+1}))
\end{cases}
\end{cases}
\]

Figure 3.362: Automaton for the INCREASING_MIN_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the INCREASING_SEQUENCE pattern.
INCREASING_MIN_INCREMENTING_SEQUENCE

837
3.182 INCREASING_MIN_INFLEXION

**DESCRIPTION**

**Origin**
Based on the INFLEXION pattern.

**Constraint**
INCREASING_MIN_INFLEXION(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES, var)

Succeeds if the minima of the values in each occurrence of the INFLEXION pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern INFLEXION is the *maximal* subsequence which matches the regular expression $< (\langle | = \rangle^* > | > > | = \rangle^* \langle$.

Assume that the occurrence of the pattern INFLEXION starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i + 1$ to index $j$.

**Purpose**

Figure 3.363 provides an example where the INCREASING_MIN_INFLEXION ([1, 2, 4, 5, 3, 3, 4, 5, 6]) constraint holds.

**Example**

$((1, 1, 2, 4, 5, 3, 3, 4, 5, 6))$

**Typical**

$|\text{VARIABLES}| > 2$

$\text{range(}\text{VARIABLES.var}) > 1$
Figure 3.363: Illustrating the INCREASING_MIN_INFLEXION constraint of the Example slot
Automaton

Figure 3.364 depicts the automaton associated with the constraint `INCREASING_MIN_INFLEXION`.

\[
\begin{align*}
\{ C \leftarrow -\infty \} & \quad \{ D \leftarrow +\infty \} \\
\{ F \leftarrow -\infty \} & \quad \{ R \leftarrow 1 \}
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow \min(D, \text{VAR}_i) \} & \quad \{ D \leftarrow \min(D, \text{VAR}_i) \} \\
\{ C \leftarrow \min(D, \text{VAR}_i) \} & \quad \{ F \leftarrow \min(D, \text{VAR}_i) \} \\
\{ F \leftarrow -\infty \} & \quad \{ F \leftarrow -\infty \} \\
\{ R \leftarrow R \land (F \leq C) \} & \quad \{ R \leftarrow R \land (F \leq \min(D, \text{VAR}_i)) \}
\end{align*}
\]

Figure 3.364: Automaton for the `INCREASING_MIN_INFLEXION` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `INFLEXION` pattern (transition $r \to t$ has the same accumulators updates as transition $t \to r$)
INCREASING_MIN_INFLEXION
### 3.183 INCREASING_MIN_STRICTLY_DECREASING_SEQUENCE

#### DESCRIPTION

**Origin**
Based on the `STRICTLY_DECREASING_SEQUENCE` pattern.

**Constraint**
`INCREASING_MIN_STRICTLY_DECREASING_SEQUENCE(VARIABLES)`

**Argument**
`VARIABLES : collection(var–dvar)`

**Restriction**
`required(VARIABLES, var)`

**Purpose**
Succeeds if the minima of the values in each occurrence of the `STRICTLY_DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are increasing.

An occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` is the `maximal` subsequence which matches the regular expression `>^+`. Assume that the occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i` to index `j + 1`.

**Example**

```plaintext
((1, 3, 2, 2, 4, 4, 5, 5, 6, 3, 5, 5, 4, 6, 7, 4))
```

Figure 3.365 provides an example where the `INCREASING_MIN_STRICTLY_DECREASING_SEQUENCE` constraint holds.

**Typical**

- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`
Figure 3.365: Illustrating the \textsc{increasing,min,strictly,decreasing,sequence} constraint of the \textbf{Example} slot
Figure 3.366 depicts the automaton associated with the constraint INCREASING_MIN_STRICTLY_DECREASING_SEQUENCE.

\[
\begin{cases}
C \leftarrow -\infty \\
D \leftarrow +\infty \\
E \leftarrow -\infty \\
R \leftarrow 1
\end{cases}
\]

\[
C \leftarrow \min(C, \min(D, \text{VAR}_{i+1})) \\
D \leftarrow +\infty
\]

Figure 3.366: Automaton for the INCREASING_MIN_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern.
INCREASING_MIN_STRICTLY_DECREASING_SEQUENCE
3.184 INCREASING_MIN_STRICTLY_INCREASING_SEQUENCE

**DESCRIPTION**

**Origin**
Based on the STRICTLY_INCREASING_SEQUENCE pattern.

**Constraint**
INCREASING_MIN_STRICTLY_INCREASING_SEQUENCE(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES, var)

**Purpose**
Succeeds if the minima of the values in each occurrence of the STRICTLY_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression <+

Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i to index j + 1.

**Example**

\[(4, 1, 2, 4, 3, 3, 2, 3, 3, 4, 5, 6, 6, 5, 7)\]

Figure 3.367 provides an example where the INCREASING_MIN_STRICTLY_INCREASING_SEQUENCE \((4, 1, 2, 4, 3, 3, 2, 3, 3, 4, 5, 6, 6, 5, 7)\) constraint holds.

**Typical**

\[|\text{VARIABLES}| > 1\]

\[\text{range}(|\text{VARIABLES}.\text{var}|) > 1\]
Figure 3.367: Illustrating the `INCREASING_MIN_STRICTLY_INCREASING_SEQUENCE` constraint of the Example slot
**Automaton**

Figure 3.368 depicts the automaton associated with the constraint **INCREASING_MIN STRICTLY INCREASING_SEQUENCE**.

\[
\begin{align*}
C & \leftarrow -\infty \\
D & \leftarrow +\infty \\
F & \leftarrow -\infty \\
R & \leftarrow 1
\end{align*}
\]

Figure 3.368: Automaton for the **INCREASING_MIN STRICTLY INCREASING_SEQUENCE** constraint obtained by applying decoration Table 2.36 to the seed transducer of the **STRICTLY INCREASING_SEQUENCE** pattern.
3.185 INCREASING_MIN_VALLEY

**DESCRIPTION**

Based on the VALLEY pattern.

**AUTOMATON**

![Valley Pattern Diagram]

**Constraint**

INCREASING_MIN_VALLEY(VARIABLES)

**Argument**

VARIABLES : collection(var–dvar)

**Restriction**

required(VARIABLES, var)

**Purpose**

Succeeds if the minima of the values in each occurrence of the VALLEY pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern VALLEY is the *maximal* subsequence which matches the regular expression \( > (= | >)^* (< | =)^* < \).

Assume that the occurrence of the pattern VALLEY starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**

\(((7, 2, 2, 6, 3, 4, 5, 6, 6, 4, 4, 6, 7, 3, 3, 1))\)

Figure 3.369 provides an example where the INCREASING_MIN_VALLEY constraint holds.

**Typical**

\(|\text{VARIABLES}| > 2\)

\(\text{range(VARIABLES.var)} > 1\)
Figure 3.369: Illustrating the INCREASING_MIN_VALLEY constraint of the Example slot
Figure 3.370 depicts the automaton associated with the constraint `INCREASING_MIN_VALLEY`.

\[ \begin{align*}
\{ C \leftarrow -\infty \} \\
\{ D \leftarrow +\infty \} \\
\{ F \leftarrow -\infty \} \\
R \leftarrow 1 \\
\{ D \leftarrow \min(D, \text{VAR}_i) \} \\
\{ C \leftarrow \min(D, \text{VAR}_i) \} \\
\{ D \leftarrow +\infty \} \\
\{ F \leftarrow C \} \\
\{ R \leftarrow R \wedge (F \leq C) \}
\end{align*} \]

Figure 3.370: Automaton for the `INCREASING_MIN_VALLEY` constraint obtained by applying decoration Table 2.36 to the seed transducer of the VALLEY pattern.
### 3.186 INCREASING_MIN_ZIGZAG

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the ZIGZAG pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>INCREASING_MIN_ZIGZAG(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var−dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES, var)</td>
</tr>
</tbody>
</table>

Succeeds if the minima of the values in each occurrence of the ZIGZAG pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression \((<>)^+ (<> | <>) | (><)^+ (> | ><<)\).

Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**

\[ ((7, 7, 2, 5, 3, 4, 6, 3, 6, 1, 1, 5, 4, 7, 5, 6)) \]

Figure 3.371 provides an example where the INCREASING_MIN_ZIGZAG ((7, 7, 2, 5, 3, 4, 6, 3, 6, 1, 1, 5, 4, 7, 5, 6)) constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 3 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]
Figure 3.371: Illustrating the INCREASING_MIN_ZIGZAG constraint of the **Example** slot
Automaton Figure 3.372 depicts the automaton associated with the constraint INCREASING_MIN_ZIGZAG.
Figure 3.372: Automaton for the INCREASING_MIN_ZIGZAG constraint obtained by applying decoration Table 2.36 to the seed transducer of the ZIGZAG pattern; (1) missing transitions from a, b, c, d, e, f to s are labelled by =; (2) on transitions from b, c, e, f to s the accumulator D is reset to its initial value; (3) on transitions from c, f to s the accumulator F is reset to C, and the accumulator R is updated wrt C and F.
### 3.187 INCREASING_RANGE_DECREASING

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th><strong>Automaton</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the <em>DECREASING</em> pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td><code>INCREASING_RANGE_DECREASING(VARIABLES)</code></td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td><code>VARIABLES : collection(var−dvar)</code></td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td><code>required(VARIABLES, var)</code></td>
</tr>
</tbody>
</table>
| **Purpose**     | Succeeds if the differences between the largest and smallest value in each occurrence of the *DECREASING* pattern in the time-series given by the *VARIABLES* collection are increasing. An occurrence of the pattern *DECREASING* is the subsequence which matches the regular expression `>`.
Assume that the occurrence of the pattern *DECREASING* starts at position $i$ and ends at position $j$. The feature *RANGE* computes the range of the values from index $i$ to index $j + 1$. |
| **Example**     | $((3, 6, 6, 5, 4, 4, 6, 4, 2, 2, 0, 5, 2, 6, 3, 3))$ |
| **Typical**     | $|VARIABLES| > 1$
$\text{range}(\text{VARIABLES}.\text{var}) > 1$ |

Figure 3.373 provides an example where the `INCREASING_RANGE_DECREASING` ($(3, 6, 6, 5, 4, 4, 6, 4, 2, 2, 0, 5, 2, 6, 3, 3)$) constraint holds.
Figure 3.373: Illustrating the INCREASING_RANGE_DECREASING constraint of the Example slot
Automaton

Figure 3.374 depicts the automaton associated with the constraint `INCREASING_RANGE_DECREASING`.

\[
\begin{align*}
\begin{cases}
    C & \leftarrow -\infty \\
    F & \leftarrow -\infty \\
    H & \leftarrow \text{VAR}_1 \\
    R & \leftarrow 1 \\
\end{cases}
\end{align*}
\]

\[
\begin{align*}
\begin{cases}
    C & \leftarrow |H - \text{VAR}_{i+1}| \\
    F & \leftarrow |H - \text{VAR}_{i+1}| \\
    H & \leftarrow \text{VAR}_{i+1} \\
    R & \leftarrow R \land (F \leq |H - \text{VAR}_{i+1}|) \\
\end{cases}
\end{align*}
\]

\[
\begin{align*}
{H} & \leftarrow \text{VAR}_{i+1} \\
\end{align*}
\]

\[
{R \land (F \leq C)}
\]

Figure 3.374: Automaton for the `INCREASING_RANGE_DECREASING` constraint obtained by applying decoration Table 2.47 to the seed transducer of the `DECREASING` pattern.
3.188 INCREASING RANGE DECREASING SEQUENCE

**DESCRIPTION**  
**AUTOMATON**

<table>
<thead>
<tr>
<th>Origin</th>
<th>Based on the DECREASING_SEQUENCE pattern.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constraint</td>
<td>INCREASING_RANGE_DECREASING_SEQUENCE(VARIABLES)</td>
</tr>
<tr>
<td>Argument</td>
<td>VARIABLES : collection(var–dvar)</td>
</tr>
<tr>
<td>Restriction</td>
<td>required(VARIABLES, var)</td>
</tr>
</tbody>
</table>

Succeeds if the differences between the largest and smallest value in each occurrence of the DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**

An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression $>$ (> | =)* $>$ | $>$. Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position $i$ and ends at position $j$. The feature RANGE computes the range of the values from index $i$ to index $j + 1$.

**Example**

$((4, 3, 2, 0, 0, 3, 4, 6, 2, 4, 4, 6, 5, 1))$

Figure 3.375 provides an example where the INCREASING_RANGE_DECREASING_SEQUENCE $((4, 3, 2, 2, 0, 0, 3, 4, 6, 2, 4, 4, 6, 5, 1))$ constraint holds.

**Typical**

<table>
<thead>
<tr>
<th>VARIABLES</th>
<th>$&gt; 1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>range(VARIABLES.var)</td>
<td>$&gt; 1$</td>
</tr>
</tbody>
</table>
Figure 3.375: Illustrating the INCREASING\_RANGE\_DECREASING\_SEQUENCE constraint of the Example slot
Automaton

Figure 3.376 depicts the automaton associated with the constraint INCREASING_RANGE_DECREASING_SEQUENCE.

\[
\begin{align*}
C &\leftarrow -\infty \\
F &\leftarrow -\infty \\
H &\leftarrow \text{VAR}_1 \\
R &\leftarrow 1
\end{align*}
\]

Figure 3.376: Automaton for the INCREASING_RANGE_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.47 to the seed transducer of the DECREASING_SEQUENCE pattern.
### 3.189 INCREASING_RANGE_INCREASING

**Description**

Based on the INCREASING pattern.

**Constraint**

INCREASING_RANGE_INCREASING(VARIABLES)

**Argument**

VARIABLES : collection(var−dvar)

**Restriction**

required(VARIABLES.var)

**Purpose**

Succeeds if the differences between the largest and smallest value in each occurrence of the INCREASING pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern INCREASING is the subsequence which matches the regular expression <.

Assume that the occurrence of the pattern INCREASING starts at position \(i\) and ends at position \(j\). The feature RANGE computes the range of the values from index \(i\) to index \(j + 1\).

**Example**

\[((4, 5, 6, 6, 3, 3, 4, 6, 2, 2, 1, 4, 7, 0, 5, 5))\]

Figure 3.377 provides an example where the INCREASING_RANGE_INCREASING \([(4, 5, 6, 6, 3, 3, 4, 6, 2, 2, 1, 4, 7, 0, 5, 5))\) constraint holds.

**Typical**

\[|\text{VARIABLES}| > 1\]

\[\text{range(VARIABLES.var)} > 1\]
Figure 3.377: Illustrating the \textsc{Increasing\_Range\_Increasing} constraint of the \textbf{Example} slot
Figure 3.378 depicts the automaton associated with the constraint \texttt{INCREASING\_RANGE\_INCREASING}.

\begin{align*}
\begin{array}{l}
C &\leftarrow -\infty \\
F &\leftarrow -\infty \\
H &\leftarrow \text{VAR}_1 \\
R &\leftarrow 1 \\
\{ C \leftarrow |H - \text{VAR}_{i+1}| \\
F \leftarrow |H - \text{VAR}_{i+1}| \\
H \leftarrow \text{VAR}_{i+1} \\
R \leftarrow R \land (F \leq |H - \text{VAR}_{i+1}|) \}
\end{array}
\end{align*}

Figure 3.378: Automaton for the \texttt{INCREASING\_RANGE\_INCREASING} constraint obtained by applying decoration Table 2.47 to the seed transducer of the \texttt{INCREASING} pattern.
### 3.190 INCREASING_RANGE_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the INCREASING_SEQUENCE pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>INCREASING_RANGE_INCREASING_SEQUENCE(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var−dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES,var)</td>
</tr>
<tr>
<td><strong>Purpose</strong></td>
<td>Succeeds if the differences between the largest and smallest value in each occurrence of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing. An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression &lt; (&lt;</td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>$((4, 1, 2, 4, 3, 3, 2, 3, 3, 4, 4, 6, 6, 3, 7))$</td>
</tr>
</tbody>
</table>
| **Typical** | $|\text{VARIABLES}| > 1$  
$\text{range}(\text{VARIABLES.var}) > 1$ |

Figure 3.379 provides an example where the increasing_range_increasing_sequence $([4, 1, 2, 4, 3, 3, 2, 3, 4, 4, 6, 6, 3, 7])$ constraint holds.
Figure 3.379: Illustrating the **INCREASING_RANGE_INCREASING_SEQUENCE** constraint of the **Example** slot
Figure 3.380 depicts the automaton associated with the constraint \texttt{INCREASING RANGE INCREASING SEQUENCE}.

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{automaton.png}
\caption{Automaton for the \texttt{INCREASING RANGE INCREASING SEQUENCE} constraint obtained by applying decoration Table 2.47 to the seed transducer of the \texttt{INCREASING SEQUENCE} pattern}
\end{figure}

\begin{align*}
&\begin{cases}
C &\leftarrow -\infty \\
F &\leftarrow -\infty \\
H &\leftarrow {\text{VAR}}_1 \\
R &\leftarrow 1 \\
\end{cases} \\
&\geq s \quad \{H \leftarrow {\text{VAR}}_{i+1}\} \\
&\begin{cases}
F &\leftarrow \lfloor H - {\text{VAR}}_i \rfloor \\
H &\leftarrow {\text{VAR}}_{i+1} \\
R &\leftarrow R \land (F \leq C) \\
\end{cases} \\
&\leq t \quad \{C \leftarrow \lfloor H - {\text{VAR}}_{i+1} \rfloor\}
\end{align*}
## 3.191 INCREASING_RANGE_STRICTLY_DECREASING_SEQUENCE

### Description

**Origin**
Based on the `STRICTLY_DECREASING_SEQUENCE` pattern.

**Constraint**
`INCREASING_RANGE_STRICTLY_DECREASING_SEQUENCE(VARIABLES)`

**Argument**
`VARIABLES : collection(var−dvar)`

**Restriction**
`required(VARIABLES.var)`

Succeeds if the differences between the largest and smallest value in each occurrence of the `STRICTLY_DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are increasing.

### Purpose

An occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` is the *maximal* sub-sequence which matches the regular expression `>^`. Assume that the occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` starts at position $i$ and ends at position $j$. The feature `RANGE` computes the range of the values from index $i$ to index $j + 1$.

**Example**

```plaintext
((1, 3, 2, 5, 6, 5, 3, 3, 0, 1, 3, 5, 1, 1, 2))
```

Figure 3.381 provides an example where the `INCREASING_RANGE_STRICTLY_DECREASING_SEQUENCE` constraint holds.

### Typical

<table>
<thead>
<tr>
<th>VARIABLES</th>
<th>&gt; 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>range(VARIABLES.var)</td>
<td>&gt; 1</td>
</tr>
</tbody>
</table>
Figure 3.381: Illustrating the INCREASING_RANGE STRICTLY DECREASING_SEQUENCE constraint of the Example slot
Figure 3.382 depicts the automaton associated with the constraint INCREASING_RANGE_STRICTLY_DECREASING_SEQUENCE.

Automaton

\[ \begin{aligned}
C & \leftarrow -\infty \\
F & \leftarrow -\infty \\
H & \leftarrow \text{VAR}_1 \\
R & \leftarrow 1
\end{aligned} \]

\[ \begin{aligned}
C & \leftarrow |H - \text{VAR}_{i+1}| \\
F & \leftarrow |H - \text{VAR}_i| \\
H & \leftarrow \text{VAR}_{i+1} \\
R & \leftarrow R \land (F \leq C)
\end{aligned} \]

\[ \begin{aligned}
C & \leftarrow |H - \text{VAR}_{i+1}| \\
H & \leftarrow \text{VAR}_{i+1} \\
R & \leftarrow R \land (F \leq C)
\end{aligned} \]

\[ \begin{aligned}
C & \leftarrow |H - \text{VAR}_{i+1}| \\
F & \leftarrow |H - \text{VAR}_i| \\
H & \leftarrow \text{VAR}_{i+1} \\
R & \leftarrow R \land (F \leq C)
\end{aligned} \]

Figure 3.382: Automaton for the INCREASING_RANGE_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.47 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern
INCREASING_RANGE_STRICTLY_DECENDING_SEQUENCE  877
### INCREASING_RANGE_STRICTLY_INCREASING_SEQUENCE

#### Description

**Origin**
Based on the `STRICTLY_INCREASING_SEQUENCE` pattern.

**Constraint**
`INCREASING_RANGE_STRICTLY_INCREASING_SEQUENCE(VARIABLES)`

**Argument**
`VARIABLES : collection(var−dvar)`

**Restriction**
`required(VARIABLES, var)`

Succeeds if the differences between the largest and smallest value in each occurrence of the `STRICTLY_INCREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are increasing.

**Purpose**
An occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` is the maximal sub-sequence which matches the regular expression `<+`. Assume that the occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `RANGE` computes the range of the values from index `i` to index `j + 1`.

**Example**

```
((4, 1, 2, 3, 0, 1, 1, 4, 1, 2, 3, 4, 4, 8))
```

Figure 3.383 provides an example where the `INCREASING_RANGE_STRICTLY_INCREASING_SEQUENCE` constraint holds.

**Typical**

```
|VARIABLES| > 1
range(VARIABLES.var) > 1
```
Figure 3.383: Illustrating the \texttt{INCREASING\_RANGE\_STRICTLY\_INCREASING\_SEQUENCE} constraint of the \texttt{Example} slot
Automaton

Figure 3.384 depicts the automaton associated with the constraint INCREASING_RANGE_STRICTLY_INCREASING_SEQUENCE.

\[
\begin{aligned}
& C \leftarrow -\infty \\
& F \leftarrow -\infty \\
& H \leftarrow \text{VAR}_1 \\
& R \leftarrow 1
\end{aligned}
\]

\[
\begin{aligned}
& F \leftarrow |H - \text{VAR}_i| \\
& H \leftarrow \text{VAR}_{i+1} \\
& R \leftarrow R \land (F \leq C)
\end{aligned}
\]

\[
\begin{aligned}
& C \leftarrow |H - \text{VAR}_{i+1}|
\end{aligned}
\]

\[
\begin{aligned}
& H \leftarrow \text{VAR}_{i+1} \\
& R \leftarrow R \land (F \leq C)
\end{aligned}
\]

\[
\begin{aligned}
& C \leftarrow |H - \text{VAR}_{i+1}|
\end{aligned}
\]

Figure 3.384: Automaton for the INCREASING_RANGE_STRICTLY_INCREASINGSEQUENCE constraint obtained by applying decoration Table 2.47 to the seed transducer of the STRICTLY_INCREASINGSEQUENCE pattern.
### 3.193 INCREASING_SURF_BUMP_ON_Decreasing_SEQUENCE

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the BUMP_ON_Decreasing_SEQUENCE pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>INCREASING_SURF_BUMP_ON_Decreasing_SEQUENCE(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var--dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES.var)</td>
</tr>
</tbody>
</table>

**Purpose**

Succeeds if the values denoting the surface of each occurrence of the BUMP_ON_Decreasing_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern BUMP_ON_Decreasing_SEQUENCE is the subsequence which matches the regular expression >><<>. Assume that the occurrence of the pattern BUMP_ON_Decreasing_SEQUENCE starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 2$ to index $j$.

**Example**

```
((7, 6, 4, 2, 5, 4, 3, 3, 5, 7, 6, 5, 6, 5, 4, 1))
```

Figure 3.385 provides an example where the INCREASING_SURF_BUMP_ON_Decreasing_SEQUENCE ($\{7, 6, 4, 2, 5, 4, 3, 3, 5, 7, 6, 5, 6, 5, 4, 1\}$) constraint holds.

**Typical**

```
|VARIABLES| > 5
range(VARIABLES.var) > 2
```
Figure 3.385: Illustrating the INCREASING_SURF_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 3.386 depicts the automaton associated with the constraint INCREASING_SURF_BUMP_ON_DECREASING_SEQUENCE.

\[
\begin{align*}
&\{ C \leftarrow -\infty, \\
&\quad D \leftarrow 0, \\
&\quad F \leftarrow -\infty, \\
&\quad R \leftarrow 1 \}\rightarrow \\
&\quad \leq \\
&\quad > \\
&\quad \leq \\
&\quad > \\
&\quad \leq \\
&\quad > \\
&\{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
&\quad \{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
&\quad \leq \\
&\quad > \\
&\quad \leq \\
&\quad > \\
&\quad > \\
&\quad \leq \\
&\quad > \\
&\quad \leq \\
&\quad > \\
&\quad \leq \\
&\quad > \\
&\quad \leq
\end{align*}
\]

\[
\begin{align*}
&\quad \{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
&\quad \{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\quad \{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\quad \{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\quad \{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

Figure 3.386: Automaton for the INCREASING_SURF_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern
INCREASING_SURF_BUMP_ON_DECREASING_SEQUENCE
3.194 INCREASING_SURF_DECREASING

**Description**

Based on the DECREASING pattern.

**Argument**

VARIABLES : collection(var–dvar)

**Restriction**

required(VARIABLES, var)

Succeeds if the values denoting the surface of each occurrence of the DECREASING pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**

An occurrence of the pattern DECREASING is the subsequence which matches the regular expression >. Assume that the occurrence of the pattern DECREASING starts at position \( i \) and ends at position \( j \). The feature \( \text{SURF} \) computes the sum of the values from index \( i \) to index \( j + 1 \).

**Example**

\((\langle 3, 3, 2, 2, 3, 4, 4, 6, 3, 4, 5, 5, 6, 4, 6 \rangle)\)

Figure 3.387 provides an example where the INCREASING_SURF_DECREASING (\([3, 3, 2, 2, 3, 4, 4, 6, 3, 4, 5, 5, 6, 4, 6]\)) constraint holds.

Figure 3.387: Illustrating the INCREASING_SURF_DECREASING constraint of the Example slot

**Typical**

\(|\text{VARIABLES}| > 1\)

\(\text{range}(\text{VARIABLES} \cdot \text{var}) > 1\)
Automaton

Figure 3.388 depicts the automaton associated with the constraint INCREASING_SURF_DECREASING.

\[
\begin{cases}
  C \leftarrow \infty \\
  D \leftarrow 0 \\
  F \leftarrow \infty \\
  R \leftarrow 1
\end{cases}
\]

\[
\begin{cases}
  C \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
  D \leftarrow 0 \\
  F \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
  R \leftarrow R \land (F \leq D + \text{VAR}_i + \text{VAR}_{i+1})
\end{cases}
\]

Figure 3.388: Automaton for the INCREASING_SURF_DECREASING constraint obtained by applying decoration Table 2.36 to the seed transducer of the DECREASING pattern.
### Description

**Origin**
Based on the `DECREASING_SEQUENCE` pattern.

**Constraint**
`INCREASING_SURF_DECREASING_SEQUENCE(VARIABLES)`

**Argument**
`VARIABLES : collection(var−dvar)`

**Restriction**
`required(VARIABLES.var)`

**Purpose**
Succeeds if the values denoting the surface of each occurrence of the `DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are increasing.

An occurrence of the pattern `DECREASING_SEQUENCE` is the *maximal* subsequence which matches the regular expression `>(>|=)*>|>`.

Assume that the occurrence of the pattern `DECREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i` to index `j + 1`.

**Example**

```plaintext
((4, 3, 2, 3, 4, 6, 3, 4, 4, 5, 6, 4, 6))
```

Figure 3.389 provides an example where the `INCREASING_SURF_DECREASING_SEQUENCE ([4, 3, 2, 3, 4, 6, 3, 4, 4, 5, 6, 4, 6])` constraint holds.

**Typical**

- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`
Figure 3.389: Illustrating the INCREASING_SURF_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 3.390 depicts the automaton associated with the constraint INCREASING_SURF_DECREASING_SEQUENCE.

\[
\begin{align*}
\{ & C \leftarrow -\infty \\
& D \leftarrow 0 \\
& F \leftarrow -\infty \\
& R \leftarrow 1 \}\left\{ \begin{array}{l}
\leq s \\
\leq \end{array} \right\}
\end{align*}
\]

\[
\begin{align*}
\{ & D \leftarrow 0 \\
& F \leftarrow C \\
& R \leftarrow R \wedge (F \leq C) \}\left\{ \begin{array}{l}
\leq \end{array} \right\}
\end{align*}
\]

\[
\begin{align*}
\{ & C \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
& D \leftarrow 0 \}\left\{ \begin{array}{l}
> \end{array} \right\}
\end{align*}
\]

\[
\begin{align*}
\{ & D \leftarrow D + \text{VAR}_{i+1} \}\left\{ \begin{array}{l}
\leq \end{array} \right\}
\end{align*}
\]

\[
\begin{align*}
\{ & C \leftarrow C + D + \text{VAR}_{i+1} \\
& D \leftarrow 0 \}\left\{ \begin{array}{l}
> \end{array} \right\}
\end{align*}
\]

\[
\begin{align*}
\{ & C \leftarrow -\infty \\
& D \leftarrow 0 \\
& F \leftarrow -\infty \\
& R \leftarrow 1 \}\left\{ \begin{array}{l}
\leq s \\
\leq \end{array} \right\}
\end{align*}
\]

Figure 3.390: Automaton for the INCREASING_SURF_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the DECREASING_SEQUENCE pattern.
### 3.196 INCREASING_SURF_DECREASING_TERRACE

**Description**

Based on the `DECREASING_TERRACE` pattern.

**Constraint**

`INCREASING_SURF_DECREASING_TERRACE(VARIABLES)`

**Argument**

`VARIABLES : collection(var−dvar)`

**Restriction**

`required(VARIABLES, var)`

Succeeds if the values denoting the surface of each occurrence of the `DECREASING_TERRACE` pattern in the time-series given by the `VARIABLES` collection are increasing.

**Purpose**

An occurrence of the pattern `DECREASING_TERRACE` is the *maximal* subsequence which matches the regular expression `>=+>`. Assume that the occurrence of the pattern `DECREASING_TERRACE` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i+1` to index `j`.

**Example**

```
((5, 2, 2, 1, 5, 4, 3, 3, 2, 4, 4, 6, 5, 5, 4))
```

Figure 3.391 provides an example where the `INCREASING_SURF_DECREASING_TERRACE` constraint holds.

**Typical**

```
|VARIABLES| > 3
range(VARIABLES, var) > 2
```
Figure 3.391: Illustrating the `INCREASING_SURF_DECREASING_TERRACE` constraint of the `Example` slot
Automaton  
Figure 3.392 depicts the automaton associated with the constraint \text{INCREASING\_SURF\_DECREASING\_TERRACE}.

\[
\begin{align*}
C &\leftarrow -\infty \\
D &\leftarrow 0 \\
F &\leftarrow -\infty \\
R &\leftarrow 1
\end{align*}
\]

\[
\begin{align*}
\{ C \leftarrow -\infty, D \leftarrow 0, F \leftarrow -\infty, R \leftarrow 1 \} &\rightarrow \leq s &\leq \\
\{ D \leftarrow 0 \} &\rightarrow \text{≤} \\
\{ D \leftarrow D + \text{VAR}_i \} &\rightarrow > \\
\{ D \leftarrow D + \text{VAR}_i \} &\rightarrow > \\
\{ C \leftarrow D + \text{VAR}_i, D \leftarrow 0, F \leftarrow D + \text{VAR}_i, R \leftarrow R \land (F \leq C) \} &\rightarrow > \\
\end{align*}
\]

Figure 3.392: Automaton for the \text{INCREASING\_SURF\_DECREASING\_TERRACE} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \text{DECREASING\_TERRACE} pattern.
INCREASING_SURF_DECREASING_TERRACE
## 3.197 INCREASING_SURF_DIP_ON_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th></th>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the DIP_ON_INCREASING_SEQUENCE pattern.</td>
<td><img src="image.png" alt="Diagram" /></td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>INCREASING_SURF_DIP_ON_INCREASING_SEQUENCE(VARIABLES)</td>
<td></td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var–dvar)</td>
<td></td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES, var)</td>
<td></td>
</tr>
</tbody>
</table>

**Purpose**

Succeeds if the values denoting the surface of each occurrence of the DIP_ON_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence which matches the regular expression <<><<.

Assume that the occurrence of the pattern DIP_ON_INCREASING_SEQUENCE starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i+2$ to index $j$.

**Example**

```
((1, 2, 3, 0, 3, 4, 6, 5, 1, 4, 5, 6, 1, 2, 4, 4))
```

Figure 3.393 provides an example where the INCREASING_SURF_DIP_ON_INCREASING_SEQUENCE constraint holds.

**Typical**

- $|\text{VARIABLES}| > 5$
- $\text{range}(\text{VARIABLES.var}) > 2$
INCREASING_SURF_DIP_ON_INCREASING_SEQUENCE

Figure 3.393: Illustrating the INCREASING_SURF_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot
Figure 3.394 depicts the automaton associated with the constraint \textsc{increasing\_surf\_dip\_on\_increasing\_sequence}.

Figure 3.394: Automaton for the \textsc{increasing\_surf\_dip\_on\_increasing\_sequence} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textsc{dip\_on\_increasing\_sequence} pattern.
### INCREASING_SURF_GORGE

**Description Automaton**

\[(> | > (= >) >) (< | < (= <) <)\]

**Origin**

Based on the GORGE pattern.

**Constraint**

\[\text{INCREASING\_SURF\_GORGE(vARIABLES)}\]

**Argument**

\[\text{VARIABLES : collection(var–dvar)}\]

**Restriction**

\[\text{required(VARIABLES, var)}\]

Succeeds if the values denoting the surface of each occurrence of the GORGE pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression \((> | > (= >) >) (< | < (= <) <))\).

Assume that the occurrence of the pattern GORGE starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Purpose**

Succeeds if the values denoting the surface of each occurrence of the GORGE pattern in the time-series given by the VARIABLES collection are increasing.

**Example**

\[(6, 2, 5, 1, 1, 5, 4, 2, 3, 6, 5, 4, 3, 5, 5)\]

Figure 3.395 provides an example where the INCREASING_SURF_GORGE \([6, 2, 5, 1, 1, 5, 4, 2, 3, 6, 5, 4, 3, 5, 5]\) constraint holds.

Figure 3.395: Illustrating the INCREASING_SURF_GORGE constraint of the Example slot
| Typical                  | \( |\text{VARIABLES}| > 2 \) |
|-------------------------|---------------------------------|
|                         | \( \text{range(}\text{VARIABLES.var}) > 1 \) |
Automaton Figure 3.396 depicts the automaton associated with the constraint \textsc{increasing\_surf\_gorge}.

Figure 3.396: Automaton for the \textsc{increasing\_surf\_gorge} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textsc{gorge} pattern (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$).
INCREASING_SURF_GORGE
3.199 INCREASING_SURF_INCREASING

DESCRIPTION AUTOMATON

Origin
Based on the INCREASING pattern.

Constraint
INCREASING_SURF_INCREASING(VARIABLES)

Argument
VARIABLES : collection(var–dvar)

Restriction
required(VARIABLES, var)

Purpose
Succeeds if the values denoting the surface of each occurrence of the INCREASING pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern INCREASING is the subsequence which matches the regular expression <.

Assume that the occurrence of the pattern INCREASING starts at position i and ends at position j. The feature SURF computes the sum of the values from index i to index j + 1.

Example
((6, 1, 2, 3, 5, 5, 4, 5, 6, 4, 3, 3, 2, 1, 1))

Figure 3.397 provides an example where the INCREASING_SURF_INCREASING ((6, 1, 2, 3, 5, 5, 4, 5, 6, 4, 3, 3, 2, 1, 1)) constraint holds.

Figure 3.397: Illustrating the INCREASING_SURF_INCREASING constraint of the Example slot
Typical

<table>
<thead>
<tr>
<th>VARIABLES</th>
<th>&gt; 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>range(VARIABLES.var)</td>
<td>&gt; 1</td>
</tr>
</tbody>
</table>
Figure 3.398 depicts the automaton associated with the constraint INCREASING_SURF_INCREASING.

Figure 3.398: Automaton for the INCREASING_SURF_INCREASING constraint obtained by applying decoration Table 2.36 to the seed transducer of the INCREASING pattern.
### 3.200 Increasing_Surf_Increasing_Sequence

**Description**

Based on the *Increasing_Sequence* pattern.

**Constraint**

`Increasing_Surf_Increasing_Sequence(VARIABLES)`

**Argument**

`VARIABLES : collection(var−dvar)`

**Restriction**

`required(VARIABLES.var)`

**Purpose**

Succeeds if the values denoting the surface of each occurrence of the *Increasing_Sequence* pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern *Increasing_Sequence* is the *maximal* subsequence which matches the regular expression `< (< | =)∗ | < | <`. Assume that the occurrence of the pattern *Increasing_Sequence* starts at position $i$ and ends at position $j$. The feature `SURF` computes the sum of the values from index $i$ to index $j + 1$.

**Example**

```plaintext
((6, 2, 3, 6, 5, 5, 4, 5, 6, 1, 3, 3, 4, 5, 5))
```

Figure 3.399 provides an example where the `Increasing_Surf_Increasing_Sequence` `((6, 2, 3, 6, 5, 5, 4, 5, 6, 1, 3, 3, 4, 5, 5))` constraint holds.

**Typical**

- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`
Figure 3.399: Illustrating the INCREASING_SURF_INCREASING_SEQUENCE constraint of the Example slot
Figure 3.400 depicts the automaton associated with the constraint **INCREASING_SURF**. The automaton is defined as follows:

\[
\begin{align*}
C & \leftarrow -\infty \\
D & \leftarrow 0 \\
F & \leftarrow -\infty \\
R & \leftarrow 1
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow 0 \} & \quad \{ C \leftarrow C + D + \text{VAR}_{i+1} \} \\
\{ F \leftarrow C \} & \quad \{ C \leftarrow C + \text{VAR}_i + \text{VAR}_{i+1} \} \\
\{ R \leftarrow R \land (F \leq C) \} & \quad \{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow D + \text{VAR}_{i+1} \} & \quad \{ C \leftarrow C + D + \text{VAR}_{i+1} \}
\end{align*}
\]

Figure 3.400: Automaton for the **INCREASING_SURF** constraint obtained by applying decoration Table 2.36 to the seed transducer of the **INCREASING_SEQUENCE** pattern.
3.201 INCREASING_SURF_INCREASING_TERRACE

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the INCREASING_TERRACE pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>INCREASING_SURF_INCREASING_TERRACE(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var–dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES, var)</td>
</tr>
</tbody>
</table>

Succeeds if the values denoting the surface of each occurrence of the INCREASING_TERRACE pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**

An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression $\leq^+ \leq$.

Assume that the occurrence of the pattern INCREASING_TERRACE starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 1$ to index $j$.

**Example**

$$\left(\langle 1, 2, 2, 4, 5, 5, 6, 6, 4, 4, 2, 5, 5, 7, 7 \rangle\right)$$

Figure 3.401 provides an example where the INCREASING_SURF_INCREASING_TERRACE $$\left(\langle 1, 2, 2, 4, 5, 5, 6, 6, 4, 4, 2, 5, 5, 7, 7 \rangle\right)$$ constraint holds.

**Typical**

$$|\text{VARIABLES}| > 3$$

$$\text{range(VARIABLES.var)} > 2$$
Figure 3.401: Illustrating the INCREASING_SURF_INCREASING_TERRACE constraint of the Example slot
Automaton

Figure 3.402 depicts the automaton associated with the constraint INCREASING_SURF_INCREASING_TERRACE.

\[
\begin{align*}
\{ C \leftarrow -\infty \\ D \leftarrow 0 \\ F \leftarrow -\infty \\ R \leftarrow 1 \} \\
\{ D \leftarrow D + \text{VAR}_i \} \\
\{ C \leftarrow D + \text{VAR}_i \\ D \leftarrow 0 \\ F \leftarrow D + \text{VAR}_i \\ R \leftarrow R \land (F \leq D + \text{VAR}_i) \}
\end{align*}
\]

Figure 3.402: Automaton for the INCREASING_SURF_INCREASING_TERRACE constraint obtained by applying decoration Table 2.36 to the seed transducer of the INCREASING_TERRACE pattern
### 3.202 INCREASING_SURF_INFLEXION

<table>
<thead>
<tr>
<th></th>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the <strong>INFLEXION</strong> pattern.</td>
<td><img src="image" alt="Diagram" /></td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td><code>INCREASING_SURF_INFLEXION(VARIABLES)</code></td>
<td></td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td><code>VARIABLES : collection(var–dvar)</code></td>
<td></td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td><code>required(VARIABLES, var)</code></td>
<td></td>
</tr>
</tbody>
</table>

**Purpose**

Succeeds if the values denoting the surface of each occurrence of the **INFLEXION** pattern in the time-series given by the **VARIABLES** collection are increasing.

An occurrence of the pattern **INFLEXION** is the maximal subsequence which matches the regular expression `< (< | =)* > | > | =)* <`. Assume that the occurrence of the pattern **INFLEXION** starts at position `i` and ends at position `j`. The feature **SURF** computes the sum of the values from index `i + 1` to index `j`.

**Example**

```plaintext
((1, 1, 2, 3, 3, 2, 2, 2, 2, 1, 6, 6, 5, 5, 5))
```

Figure 3.403 provides an example where the `INCREASING_SURF_INFLEXION ((1, 1, 2, 3, 3, 2, 2, 2, 2, 1, 6, 6, 5, 5, 5))` constraint holds.

**Typical**

- `|VARIABLES| > 2`
- `range(VARIABLES, var) > 1`
INCREASING_SURF_INFLEXION

Figure 3.403: Illustrating the INCREASING_SURF_INFLEXION constraint of the Example slot
Figure 3.404 depicts the automaton associated with the constraint INCREASING_SURF_INFLEXION.

Figure 3.404: Automaton for the INCREASING_SURF_INFLEXION constraint obtained by applying decoration Table 2.36 to the seed transducer of the INFLEXION pattern (transition \( r \to t \) has the same accumulators updates as transition \( t \to r \))
3.203 INCREASING_SURF_PEAK

**DESCRIPTION**

Origin: Based on the PEAK pattern.

Constraint: INCREASING_SURF_PEAK(VARIABLES)

Argument: VARIABLES : collection(var−dvar)

Restriction: required(VARIABLES, var)

**AUTOMATON**

Succeeds if the values denoting the surface of each occurrence of the PEAK pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression $< (= | < )^* ( > | = )^* >$.

Assume that the occurrence of the pattern PEAK starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 1$ to index $j$.

**Example**

$$((7, 5, 5, 1, 2, 3, 2, 3, 4, 5, 2, 6, 6, 6, 1))$$

Figure 3.405 provides an example where the INCREASING_SURF_PEAK ($([7, 5, 5, 1, 2, 3, 2, 3, 4, 5, 2, 6, 6, 6, 1])$) constraint holds.

Figure 3.405: Illustrating the INCREASING_SURF_PEAK constraint of the Example slot
INCREASING_SURF_PEAK

Typical

$|\text{VARIABLES}| > 2$

$\text{range(VARIABLES.var)} > 1$
Automaton

Figure 3.406 depicts the automaton associated with the constraint INCREASING_SURF_PEAK.

\[
\begin{align*}
&\left\{ \begin{array}{l}
C \leftarrow -\infty \\
D \leftarrow 0 \\
F \leftarrow -\infty \\
R \leftarrow 1
\end{array} \right. \quad \geq S \\
< \\
\{ D \leftarrow D + \text{VAR}_i \}
\end{align*}
\]

\[
\begin{align*}
&\left\{ \begin{array}{l}
C \leftarrow D + \text{VAR}_i \\
D \leftarrow 0
\end{array} \right. \quad \{ C \leftarrow C + D + \text{VAR}_i \}
\end{align*}
\]

\[
\left\{ \begin{array}{l}
D \leftarrow 0 \\
F \leftarrow C \\
R \leftarrow R \land (F \leq C)
\end{array} \right. \quad \{ D \leftarrow D + \text{VAR}_i \}
\]

Figure 3.406: Automaton for the INCREASING_SURF_PEAK constraint obtained by applying decoration Table 2.36 to the seed transducer of the PEAK pattern
3.204 \textbf{INCREASING\_SURF\_PLAIN}

\begin{description}
\item[Origin] Based on the \texttt{PLAIN} pattern.
\item[Constraint] \texttt{INCREASING\_SURF\_PLAIN(VARIABLES)}
\item[Argument] \texttt{VARIABLES : collection(var–dvar)}
\item[Restriction] \texttt{required(VARIABLES, var)}
\end{description}

\textbf{Succeeds if the values denoting the surface of each occurrence of the \texttt{PLAIN} pattern in the time-series given by the \texttt{VARIABLES} collection are increasing.}

\textbf{An occurrence of the pattern \texttt{PLAIN} is the maximal subsequence which matches the regular expression \texttt{\textgreater \ast \textless}.}

\textbf{Assume that the occurrence of the pattern \texttt{PLAIN} starts at position \textit{i} and ends at position \textit{j}. The feature \texttt{SURF} computes the sum of the values from index \textit{i} + 1 to index \textit{j}.}

\textbf{Example} \hfill\hfill (⟨3, 6, 6, 3, 4, 5, 5, 4, 6, 6, 7, 5, 6, 3, 2⟩)

Figure 3.407 provides an example where the \texttt{INCREASING\_SURF\_PLAIN ([3, 6, 6, 3, 4, 5, 5, 4, 6, 6, 7, 5, 6, 3, 2])} constraint holds.
Typical

\[|\text{VARIABLES}| > 2\]
\[\text{range}(\text{VARIABLES}.\text{var}) > 1\]
Automaton

Figure 3.408 depicts the automaton associated with the constraint INCREASING_SURFPLAIN.

Figure 3.408: Automaton for the INCREASING_SURFPLAIN constraint obtained by applying decoration Table 2.36 to the seed transducer of the PLAIN pattern.
3.205  INCREASING_SURF_PLATEAU

**DESCRIPTION**

**Origin**
Based on the PLATEAU pattern.

**Constraint**
INCREASING_SURF_PLATEAU(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES, var)

**Purpose**
Succeeds if the values denoting the surface of each occurrence of the PLATEAU pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression < =∗ >.

Assume that the occurrence of the pattern PLATEAU starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**
\[((7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5))\]

Figure 3.409 provides an example where the INCREASING_SURF_PLATEAU ([7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5]) constraint holds.

**Typical**
| VARIABLES | > 2
| range(VARIABLES.var) | > 1
Figure 3.409: Illustrating the **INCREASING_SURF_PLATEAU** constraint of the **Example** slot
Figure 3.410 depicts the automaton associated with the constraint INCREASING_SURF_PLATEAU.

\[
\begin{align*}
C &\leftarrow \infty \\
D &\leftarrow 0 \\
F &\leftarrow 0 \\
R &\leftarrow 1 \\
\end{align*}
\]

\[
\begin{align*}
C &\leftarrow \infty \\
D &\leftarrow 0 \\
F &\leftarrow -\infty \\
R &\leftarrow 1 \\
\end{align*}
\]

\[
\begin{align*}
C &\leftarrow D + \text{VAR}_i \\
D &\leftarrow 0 \\
F &\leftarrow D + \text{VAR}_i \\
R &\leftarrow R \land (F \leq D + \text{VAR}_i) \\
\end{align*}
\]

\[
\begin{align*}
C &\leftarrow \text{VAR}_i \\
D &\leftarrow 0 \\
F &\leftarrow D + \text{VAR}_i \\
R &\leftarrow R \land (F \leq D + \text{VAR}_i) \\
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow D + \text{VAR}_i \} \\
\{ D \leftarrow 0 \} \\
\{ D \leftarrow D + \text{VAR}_i \} \\
\{ D \leftarrow 0 \} \\
\end{align*}
\]

Figure 3.410: Automaton for the INCREASING_SURF_PLATEAU constraint obtained by applying decoration Table 2.36 to the seed transducer of the PLATEAU pattern
### 3.206 INCREASING_SURF_PROPER_PLAIN

<table>
<thead>
<tr>
<th></th>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the PROPER_PLAIN pattern.</td>
<td></td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>INCREASING_SURF_PROPER_PLAIN(VARIABLES)</td>
<td></td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var–dvar)</td>
<td></td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES.var)</td>
<td></td>
</tr>
<tr>
<td><strong>Purpose</strong></td>
<td>Succeeds if the values denoting the surface of each occurrence of the PROPER_PLAIN pattern in the time-series given by the VARIABLES collection are increasing. An occurrence of the pattern PROPER_PLAIN is the maximal subsequence which matches the regular expression ( &gt; =^+ &lt; ). Assume that the occurrence of the pattern PROPER_PLAIN starts at position ( i ) and ends at position ( j ). The feature SURF computes the sum of the values from index ( i + 1 ) to index ( j ).</td>
<td></td>
</tr>
<tr>
<td><strong>Example</strong></td>
<td>(((5, 3, 3, 5, 6, 5, 4, 4, 7, 3, 6, 5, 5, 7, 2)))</td>
<td></td>
</tr>
<tr>
<td><strong>Typical</strong></td>
<td>(</td>
<td>\text{VARIABLES}</td>
</tr>
</tbody>
</table>

Figure 3.411 provides an example where the INCREASING_SURF_PROPER_PLAIN \(((5, 3, 3, 5, 6, 5, 4, 4, 7, 3, 6, 5, 5, 7, 2))\) constraint holds.
Figure 3.411: Illustrating the INCREASING_SURF_PROPER.PLAIN constraint of the Example slot
Automaton

Figure 3.412 depicts the automaton associated with the constraint INCREASING_SURF_PROPER_PLAIN.

\[
\begin{align*}
\text{\{D} & \text{\leftarrow D + VAR_i\}} \\
\text{\{D} & \text{\leftarrow 0\}} \\
R & \text{\leftarrow R \land (F \leq D + VAR_i)} \\
R & \text{\leftarrow 1} \\
F & \text{\leftarrow D + VAR_i} \\
C & \text{\leftarrow D + VAR_i} \\
\end{align*}
\]

Figure 3.412: Automaton for the INCREASING_SURF_PROPER_PLAIN constraint obtained by applying decoration Table 2.36 to the seed transducer of the PROPER_PLAIN pattern.
3.207 INCREASING_SURF_PROPER_PLATEAU

**DESCRIPTION**

**Origin**
Based on the PROPER_PLATEAU pattern.

**Constraint**
INCREASING_SURF_PROPER_PLATEAU(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES, var)

**Purpose**
Succeeds if the values denoting the surface of each occurrence of the PROPER_PLATEAU pattern in the time-series given by the VARIABLES collection are increasing. An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression < =+ >. Assume that the occurrence of the pattern PROPER_PLATEAU starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i+1 \) to index \( j \).

**Example**

\[
((7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3))
\]

Figure 3.413 provides an example where the INCREASING_SURF_PROPER_PLATEAU ((7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3)) constraint holds.

**Typical**

\[|\text{VARIABLES}| > 3\]
\[\text{range}(\text{VARIABLES}.\text{var}) > 1\]
Figure 3.413: Illustrating the \textit{INCREASING\_SURF\_PROPER\_PLATEAU} constraint of the \textbf{Example} slot
Automaton

Figure 3.414 depicts the automaton associated with the constraint INCREASING_SURF_PROPER_PLATEAU.

Figure 3.414: Automaton for the INCREASING_SURF_PROPER_PLATEAU constraint obtained by applying decoration Table 2.36 to the seed transducer of the PROPER_PLATEAU pattern
INCREASING_SURF_PROPER_PLATEAU

939
**3.208 INCREASING_SURF_STEADY**

**Description**

**Origin**
Based on the STEADY pattern.

**Constraint**
INCREASING_SURF_STEADY(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES, var)

**Purpose**
Succeeds if the values denoting the surface of each occurrence of the STEADY pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern STEADY is the subsequence which matches the regular expression =.

Assume that the occurrence of the pattern STEADY starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i \) to index \( j + 1 \).

**Example**

\[ ((1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6)) \]

Figure 3.415 provides an example where the INCREASING_SURF_STEADY \(((1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6))\) constraint holds.

**Typical**

\[ |VARIABLES| > 1 \]
Figure 3.415: Illustrating the INCREASING_SURF steadiness constraint of the Example slot
Automaton Figure 3.416 depicts the automaton associated with the constraint INCREASING_SURF_STEADY.

\[
\begin{align*}
\text{states} & = \{ C \leftarrow -\infty, D \leftarrow 0, F \leftarrow -\infty, R \leftarrow 1 \} \\
\end{align*}
\]

\[
\begin{align*}
C & \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
D & \leftarrow 0 \\
F & \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
R & \leftarrow R \land (F \leq D + \text{VAR}_i + \text{VAR}_{i+1}) \\
\end{align*}
\]

Figure 3.416: Automaton for the INCREASING_SURF_STEADY constraint obtained by applying decoration Table 2.36 to the seed transducer of the STEADY pattern.
3.209 \textbf{INCREASING\_SURF\_STEADY\_SEQUENCE}

\textbf{DESCRIPTION}

Origin
Based on the \textit{STEADY\_SEQUENCE} pattern.

Constraint
\textbf{INCREASING\_SURF\_STEADY\_SEQUENCE(VARIABLES)}

Argument
\textbf{VARIABLES : collection(var\textminus dvar)}

Restriction
\textbf{required(VARIABLES, var)}

Purpose
Succeeds if the values denoting the surface of each occurrence of the \textit{STEADY\_SEQUENCE} pattern in the time-series given by the \textbf{VARIABLES} collection are increasing.

An occurrence of the pattern \textit{STEADY\_SEQUENCE} is the \textit{maximal} subsequence which matches the regular expression =$^+$.

Assume that the occurrence of the pattern \textit{STEADY\_SEQUENCE} starts at position $i$ and ends at position $j$. The feature \textit{SURF} computes the sum of the values from index $i$ to index $j + 1$.

Example
\((\langle 6, 2, 2, 3, 5, 6, 4, 4, 2, 3, 5, 5, 2, 3, 4 \rangle)\)

Figure 3.417 provides an example where the \textit{INCREASING\_SURF\_STEADY\_SEQUENCE} ((6, 2, 2, 3, 5, 6, 4, 4, 2, 3, 5, 5, 2, 3, 4)) constraint holds.

![Diagram illustrating the INCREASING\_SURF\_STEADY\_SEQUENCE constraint]

Figure 3.417: Illustrating the \textbf{INCREASING\_SURF\_STEADY\_SEQUENCE} constraint of the \textbf{Example} slot.
| Typical | $|\text{VARIABLES}| > 1$ |
Automaton Figure 3.418 depicts the automaton associated with the constraint INCREASING_SURF_STEADY_SEQUENCE.

Figure 3.418: Automaton for the INCREASING_SURF_STEADY_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the STEADY_SEQUENCE pattern.
3.210 INCREASING_SURF_STRICTLY_DECREASING_SEQUENCE

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
<tbody>
<tr>
<td>Origin</td>
<td>Based on the STRICTLY_DECREASING_SEQUENCE pattern.</td>
</tr>
<tr>
<td>Constraint</td>
<td>INCREASING_SURF_STRICTLY_DECREASING_SEQUENCE(VARIABLES)</td>
</tr>
<tr>
<td>Argument</td>
<td>VARIABLES : collection(var~dvar)</td>
</tr>
<tr>
<td>Restriction</td>
<td>required(VARIABLES.var)</td>
</tr>
<tr>
<td>Purpose</td>
<td>Succeeds if the values denoting the surface of each occurrence of the STRICTLY_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing. An occurrence of the pattern STRICTLY_DECREASING_SEQUENCE is the maximal sub-sequence which matches the regular expression &gt;+. Assume that the occurrence of the pattern STRICTLY_DECREASING_SEQUENCE starts at position (i) and ends at position (j). The feature SURF computes the sum of the values from index (i) to index (j+1).</td>
</tr>
<tr>
<td>Example</td>
<td>((1, 4, 3, 2, 7, 3, 3, 2, 7, 3, 3, 4, 4, 5, 7, 5))</td>
</tr>
<tr>
<td>Typical</td>
<td>(</td>
</tr>
<tr>
<td></td>
<td>range(VARIABLES.var) &gt; 1</td>
</tr>
</tbody>
</table>
Figure 3.419: Illustrating the INCREASING_SURF_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Automaton Figure 3.420 depicts the automaton associated with the constraint INCREASING_SURF_STRICTLY_DECREASING_SEQUENCE.

\[
\begin{align*}
C &\leftarrow -\infty \\
D &\leftarrow 0 \\
F &\leftarrow -\infty \\
R &\leftarrow 1 \\
\end{align*}
\]

\[
\begin{align*}
C &\leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
D &\leftarrow 0 \\
\end{align*}
\]

Figure 3.420: Automaton for the INCREASING_SURF_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern.
INCREASING_SURF_STRICTLY_DECREASING_SEQUENCE
3.211 INCREASING_SURF STRICTLY_INCREASING_SEQUENCE

**Origin**
Based on the STRICTLY_INCREASING_SEQUENCE pattern.

**Constraint**
INCREASING_SURF_STRICTLY_INCREASING_SEQUENCE(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES, var)

Succeeds if the values denoting the surface of each occurrence of the STRICTLY_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**
An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression <+. Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at position i and ends at position j. The feature SURF computes the sum of the values from index i to index j + 1.

**Example**

```plaintext
((6, 2, 3, 6, 5, 5, 4, 5, 6, 1, 2, 3, 4, 5, 5))
```

Figure 3.421 provides an example where the INCREASING_SURF_STRICTLY_INCREASING_SEQUENCE ((6, 2, 3, 6, 5, 5, 4, 5, 6, 1, 2, 3, 4, 5, 5)) constraint holds.

**Typical**

```plaintext
|VARIABLES| > 1
range(VARIABLES.var) > 1
```
Figure 3.421: Illustrating the $\text{INCREASING\_SURF\_STRICTLY\_INCREASING\_SEQUENCE}$ constraint of the $\text{Example}$ slot
Automaton Figure 3.422 depicts the automaton associated with the constraint `INCREASING_SURF_STRICTLY_INCREASING_SEQUENCE`.

```plaintext
\begin{align*}
&\{ C \leftarrow -\infty \} \\
&\{ D \leftarrow 0 \} \\
&\{ F \leftarrow -\infty \} \\
&\{ R \leftarrow 1 \}
\end{align*}
```

\[ \geq s \]

\[ \geq \]

\[ D \leftarrow 0 \]
\[ F \leftarrow C \]
\[ R \leftarrow R \land (F \leq C) \]

\[ \{ C \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \} \]
\[ \{ D \leftarrow 0 \} \]

\[ \{ C \leftarrow C + D + \text{VAR}_{i+1} \} \]
\[ \{ D \leftarrow 0 \} \]

Figure 3.422: Automaton for the `INCREASING_SURF_STRICTLY_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `STRICTLY_INCREASING_SEQUENCE` pattern.
3.212 INCREASING_SURF_SUMMIT

**DESCRIPTION**

**Origin**
Based on the SUMMIT pattern.

**Constraint**
INCREASING_SURF_SUMMIT(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES,var)

**Purpose**
Succeeds if the values denoting the surface of each occurrence of the SUMMIT pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression (< | = ( | < )∗ < | > | = | > )∗ >.

Assume that the occurrence of the pattern SUMMIT starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

\[
((1, 5, 2, 1, 6, 6, 2, 3, 5, 4, 1, 4, 6, 4, 3, 2))
\]

Figure 3.423 provides an example where the INCREASING_SURF_SUMMIT \([(1, 5, 2, 1, 6, 6, 2, 3, 5, 4, 1, 4, 6, 4, 3, 2)]\) constraint holds.

**Typical**

\(|\text{VARIABLES}| > 2\)

range(VARIABLES,var) > 1
Figure 3.423: Illustrating the INCREASING_SURF_SUMMIT constraint of the Example slot
Automaton

Figure 3.424 depicts the automaton associated with the constraint INCREASING_SURF_SUMMIT.

\[
\begin{align*}
&\left\{ C \leftarrow -\infty \\
&\quad D \leftarrow 0 \\
&\quad F \leftarrow -\infty \\
&\quad R \leftarrow 1 \\
\right\} \quad \rightarrow \quad \geq S \\
\end{align*}
\]

\( R \land (F \leq C) \)

\[
\begin{align*}
&\left\{ \begin{array}{l}
D \leftarrow D + \text{VAR}_i \\
\end{array} \right\} \\
&\left\{ \begin{array}{l}
C \leftarrow D + \text{VAR}_i \\
D \leftarrow 0 \\
\end{array} \right\} \\
\end{align*}
\]

\[
\begin{align*}
&\left\{ \begin{array}{l}
D \leftarrow D + \text{VAR}_i \\
\end{array} \right\} \\
&\left\{ \begin{array}{l}
C \leftarrow D + \text{VAR}_i \\
D \leftarrow 0 \\
\end{array} \right\} \\
\end{align*}
\]

\[
\begin{align*}
&\left\{ \begin{array}{l}
D \leftarrow D + \text{VAR}_i \\
\end{array} \right\} \\
&\left\{ \begin{array}{l}
C \leftarrow D + \text{VAR}_i \\
D \leftarrow 0 \\
\end{array} \right\} \\
\end{align*}
\]

\[
\begin{align*}
&\left\{ \begin{array}{l}
D \leftarrow D + \text{VAR}_i \\
\end{array} \right\} \\
&\left\{ \begin{array}{l}
C \leftarrow D + \text{VAR}_i \\
D \leftarrow 0 \\
\end{array} \right\} \\
\end{align*}
\]

\[
\begin{align*}
&\left\{ \begin{array}{l}
D \leftarrow D + \text{VAR}_i \\
\end{array} \right\} \\
&\left\{ \begin{array}{l}
C \leftarrow D + \text{VAR}_i \\
D \leftarrow 0 \\
\end{array} \right\} \\
\end{align*}
\]

\[
\begin{align*}
&\left\{ \begin{array}{l}
D \leftarrow D + \text{VAR}_i \\
\end{array} \right\} \\
&\left\{ \begin{array}{l}
C \leftarrow D + \text{VAR}_i \\
D \leftarrow 0 \\
\end{array} \right\} \\
\end{align*}
\]

\[
\begin{align*}
&\left\{ \begin{array}{l}
D \leftarrow D + \text{VAR}_i \\
\end{array} \right\} \\
&\left\{ \begin{array}{l}
C \leftarrow D + \text{VAR}_i \\
D \leftarrow 0 \\
\end{array} \right\} \\
\end{align*}
\]

\[
\begin{align*}
&\left\{ \begin{array}{l}
D \leftarrow D + \text{VAR}_i \\
\end{array} \right\} \\
&\left\{ \begin{array}{l}
C \leftarrow D + \text{VAR}_i \\
D \leftarrow 0 \\
\end{array} \right\} \\
\end{align*}
\]

Figure 3.424: Automaton for the INCREASING_SURF_SUMMIT constraint obtained by applying decoration Table 2.36 to the seed transducer of the SUMMIT pattern (transition \( u \rightarrow r \) has the same accumulator update as transition \( r \rightarrow u \))
### 3.213 INCREASING_SURF_VALLEY

#### Description

**Origin**

Based on the VALLEY pattern.

**Constraint**

INCREASING_SURF_VALLEY(VARIABLES)

**Argument**

VARIABLES : collection(var−dvar)

**Restriction**

required(VARIABLES, var)

Succeeds if the values denoting the surface of each occurrence of the VALLEY pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**

An occurrence of the pattern VALLEY is the *maximal* subsequence which matches the regular expression > (= | >)*(< | =)*<.

Assume that the occurrence of the pattern VALLEY starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 1$ to index $j$.

**Example**

$((7, 2, 2, 6, 3, 4, 5, 6, 6, 4, 4, 6, 7, 3, 3, 1))$

Figure 3.425 provides an example where the INCREASING_SURF_VALLEY ($(7, 2, 2, 6, 3, 4, 5, 6, 6, 4, 4, 6, 7, 3, 3, 1))$ constraint holds.

**Typical**

$|\text{VARIABLES}| > 2$

$\text{range}(\text{VARIABLES}\cdot\text{var}) > 1$
Figure 3.425: Illustrating the INCREASING_SURF_VALLEY constraint of the Example slot
Automaton

Figure 3.426 depicts the automaton associated with the constraint INCREASING_SURF_VALLEY.

Figure 3.426: Automaton for the INCREASING_SURF_VALLEY constraint obtained by applying decoration Table 2.36 to the seed transducer of the VALLEY pattern
3.214 INCREASING_SURF_ZIGZAG

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the ZIGZAG pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>( \text{INCREASING_SURF_ZIGZAG}(\text{VARIABLES}) )</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>( \text{VARIABLES} : \text{collection}(\text{var} - \text{dvar}) )</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>\text{required}(\text{VARIABLES}, \text{var})</td>
</tr>
</tbody>
</table>

Succeeds if the values denoting the surface of each occurrence of the ZIGZAG pattern in the time-series given by the \( \text{VARIABLES} \) collection are increasing.

**Purpose**

An occurrence of the pattern ZIGZAG is the \textit{maximal} subsequence which matches the regular expression \( (<>)^+ (< | <> | (>)^+ (> | >>) \)
Assume that the occurrence of the pattern ZIGZAG starts at position \( i \) and ends at position \( j \). The feature \textit{SURF} computes the sum of the values from index \( i + 1 \) to index \( j \).

**Example**

\( [(7, 7, 2, 5, 3, 4, 6, 3, 6, 1, 1, 5, 4, 7, 5, 6)] \)

Figure 3.427 provides an example where the \text{INCREASING_SURF_ZIGZAG} \([(7, 7, 2, 5, 3, 4, 6, 3, 6, 1, 1, 5, 4, 7, 5, 6)] \) constraint holds.

**Typical**

\( |\text{VARIABLES}| > 3 \)
\( \text{range}(\text{VARIABLES}.\text{var}) > 1 \)
Figure 3.427: Illustrating the \textit{INCREASING\_SURF\_ZIGZAG} constraint of the \textbf{Example} slot
Automaton

Figure 3.428 depicts the automaton associated with the constraint INCREASING_SURF_ZIGZAG.
Figure 3.428: Automaton for the INCREASING_SURF_ZIGZAG constraint obtained by applying decoration Table 2.36 to the seed transducer of the ZIGZAG pattern; (1) missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=;$ (2) on transitions from $b, c, e, f$ to $s$ the accumulator $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the accumulator $F$ is reset to $C$, and the accumulator $R$ is updated wrt $C$ and $F$
### 3.215 INCREASING_WIDTH_DECREASING_SEQUENCE

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the DECREASING_SEQUENCE pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>INCREASING_WIDTH_DECREASING_SEQUENCE(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var−dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES, var)</td>
</tr>
</tbody>
</table>

Succeeds if the values denoting the width of each occurrence of the DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**
An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression \( (> | =)^* > | > \). Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i + 2 \).

**Example**
\[
((1, 6, 6, 4, 4, 5, 6, 6, 5, 2, 3, 3, 2, 2, 1, 1))
\]

Figure 3.429 provides an example where the INCREASING_WIDTH_DECREASING_SEQUENCE \((|1, 6, 6, 4, 4, 5, 6, 6, 5, 2, 3, 3, 2, 2, 1, 1|)\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES} . \text{var}) > 1
\]
Figure 3.429: Illustrating the INCREASING_WIDTH_DECREASING_SEQUENCE constraint of the Example slot
Figure 3.430 depicts the automaton associated with the constraint `INCREASING_WIDTH_DECREASING_SEQUENCE`.

Figure 3.430: Automaton for the `INCREASING_WIDTH_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 2.36 to the seed transducer of the `DECREASING_SEQUENCE` pattern
## 3.2.16 INCREASING_WIDTH_DECREASING_TERRACE

### Description

<table>
<thead>
<tr>
<th>Origin</th>
<th>Based on the DECREASING_TERRACE pattern.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constraint</td>
<td>INCREASING_WIDTH_DECREASING_TERRACE(VARIABLES)</td>
</tr>
<tr>
<td>Argument</td>
<td>VARIABLES : collection(var−dvar)</td>
</tr>
<tr>
<td>Restriction</td>
<td>required(VARIABLES, var)</td>
</tr>
</tbody>
</table>

Succeeds if the values denoting the width of each occurrence of the DECREASING_TERRACE pattern in the time-series given by the VARIABLES collection are increasing.

### Purpose

An occurrence of the pattern DECREASING_TERRACE is the **maximal** subsequence which matches the regular expression \( \geq ^+ > \).

Assume that the occurrence of the pattern DECREASING_TERRACE starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

### Example

\[
(5, 2, 2, 1, 5, 4, 3, 3, 2, 4, 4, 6, 5, 5, 4)\]

Figure 3.431 provides an example where the INCREASING_WIDTH_DECREASING_TERRACE \((5, 2, 2, 1, 5, 4, 3, 3, 2, 4, 4, 6, 5, 5, 4)\) constraint holds.

### Typical

\[
|VARIABLES| > 3 \\
\text{range}(VARIABLES.var) > 2
\]
Figure 3.431: Illustrating the INCREASING_WIDTH_DECENDING_TERRACE constraint of the Example slot
Figure 3.432 depicts the automaton associated with the constraint INCREASING_WIDTH_DECREASING_TERRACE.

Automaton

Figure 3.432: Automaton for the INCREASING_WIDTH_DECREASING_TERRACE constraint obtained by applying decoration Table 2.36 to the seed transducer of the DECREASING_TERRACE pattern.
### 3.217 INCREASING_WIDTHGORGE

**DESCRIPTION AUTOMATON**

- **Origin**: Based on the GORGE pattern.
- **Constraint**: INCREASING_WIDTHGORGE(VARIABLES)
- **Argument**: VARIABLES : collection(var–dvar)
- **Restriction**: required(VARIABLES, var)

**Purpose**

Succeeds if the values denoting the width of each occurrence of the GORGE pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression 

\[
(> | > (= | >)^*) (> | < (= | <)^*) <)
\]

Assume that the occurrence of the pattern GORGE starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

**Example**

\[
((6, 2, 5, 1, 1, 5, 4, 2, 3, 6, 5, 4, 4, 3, 5, 5))
\]

Figure 3.433 provides an example where the INCREASING_WIDTHGORGE 

\([6, 2, 5, 1, 1, 5, 4, 2, 3, 6, 5, 4, 4, 3, 5, 5]\) constraint holds.

Figure 3.433: Illustrating the INCREASING_WIDTHGORGE constraint of the Example slot.
Typical

\[|\text{VARIABLES}| > 2\]

\[\text{range(VARIABLES.var)} > 1\]
Automaton Figure 3.434 depicts the automaton associated with the constraint INCREASING_WIDTH_GORGE.

Figure 3.434: Automaton for the INCREASING_WIDTH_GORGE constraint obtained by applying decoration Table 2.36 to the seed transducer of the GORGE pattern (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$)
INCREASING_WIDTH_GORGE

979
3.218  INCREASING_WIDTH_INCREASING_SEQUENCE

**Description**

Based on the `INCREASING_SEQUENCE` pattern.

**Constraint**

`INCREASING_WIDTH_INCREASING_SEQUENCE(VARIABLES)`

**Argument**

`VARIABLES : collection(var–dvar)`

**Restriction**

`required(VARIABLES, var)`

Succeeds if the values denoting the width of each occurrence of the `INCREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are increasing.

**Purpose**

An occurrence of the pattern `INCREASING_SEQUENCE` is the maximal subsequence which matches the regular expression `< (< | =)* < | <`. Assume that the occurrence of the pattern `INCREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `$j - i + 2$`.

**Example**

`((6, 2, 3, 6, 5, 5, 4, 5, 6, 1, 3, 3, 4, 5, 5))`

Figure 3.435 provides an example where the `INCREASING_WIDTH_INCREASING_SEQUENCE` `([6, 2, 3, 6, 5, 5, 4, 5, 6, 1, 3, 3, 4, 5, 5])` constraint holds.

**Typical**

`|VARIABLES| > 1`

`range(VARIABLES.var) > 1`
Figure 3.435: Illustrating the INCREASING_WIDTH_INCREASING_SEQUENCE constraint of the Example slot
Automaton  

Figure 3.436 depicts the automaton associated with the constraint INCREASING_WIDTH_INCREASING_SEQUENCE.

\[
\begin{align*}
C & \leftarrow -\infty \\
D & \leftarrow 0 \\
F & \leftarrow -\infty \\
R & \leftarrow 1
\end{align*}
\]

\[
\geq s
\]

\[
\begin{align*}
D & \leftarrow 0 \\
F & \leftarrow C \\
R & \leftarrow R \land (F \leq C)
\end{align*}
\]

\[
\leq t
\]

\[
\begin{align*}
C & \leftarrow D + 2 \\
D & \leftarrow 0
\end{align*}
\]

\[
\begin{align*}
D & \leftarrow D + 1 \\
C & \leftarrow C + D + 1
\end{align*}
\]

Figure 3.436: Automaton for the INCREASING_WIDTH_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the INCREASING_SEQUENCE pattern.
### 3.219 INCREASING_WIDTH_INCREASING_TERRACE

**Description**

Based on the INCREASING_TERRACE pattern.

**Argument**

\[ \text{VARIABLES} : \text{collection}(\text{var} \rightarrow \text{dvar}) \]

**Constraint**

\[ \text{INCREASING_WIDTH_INCREASING_TERRACE(VARIABLES)} \]

**Restriction**

\[ \text{required(VARIABLES, var)} \]

Succeeds if the values denoting the width of each occurrence of the INCREASING_TERRACE pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**

An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression \(<=+<\).

Assume that the occurrence of the pattern INCREASING_TERRACE starts at position \(i\) and ends at position \(j\). The feature \(\text{WIDTH}\) computes the value \(j - i\).

**Example**

\[ ((1, 2, 2, 4, 5, 5, 6, 6, 4, 4, 2, 5, 5, 7, 7)) \]

Figure 3.437 provides an example where the INCREASING_WIDTH_INCREASING_TERRACE \(([1, 2, 2, 4, 5, 5, 6, 6, 4, 4, 2, 5, 5, 7, 7])\) constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 3 \]

\[ \text{range(VARIABLES.var)} > 2 \]
Figure 3.437: Illustrating the `INCREASING_WIDTH_INCREASING_TERRACE` constraint of the `Example` slot
Figure 3.438 depicts the automaton associated with the constraint INCREASING_WIDTH_INCREASING_TERRACE.

\[
\begin{align*}
C & \leftarrow -\infty \\
D & \leftarrow 0 \\
F & \leftarrow -\infty \\
R & \leftarrow 1
\end{align*}
\]

\[
\begin{align*}
\{D \leftarrow D + 1\} \\
\{D \leftarrow 0\}
\end{align*}
\]

\[
\begin{align*}
R \land (F \leq C)
\end{align*}
\]

\[
\begin{align*}
\{D \leftarrow D + 1\} \\
\{D \leftarrow D + 1\}
\end{align*}
\]

\[
\begin{align*}
C & \leftarrow D + 1 \\
D & \leftarrow 0 \\
F & \leftarrow D + 1 \\
R & \leftarrow R \land (F \leq D + 1)
\end{align*}
\]

Figure 3.438: Automaton for the INCREASING_WIDTH_INCREASING_TERRACE constraint obtained by applying decoration Table 2.36 to the seed transducer of the INCREASING_TERRACE pattern.
3.220 INCREASING_WIDTH_INFLEXION

**DESCRIPTION AUTOMATON**

**Origin**
Based on the **INFLEXION** pattern.

**Constraint**
INCREASING_WIDTH_INFLEXION(VARIABLES)

**Argument**
VARIABLES : collection(var–dvar)

**Restriction**
required(VARIABLES, var)

Succeeds if the values denoting the width of each occurrence of the **INFLEXION** pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern **INFLEXION** is the **maximal** subsequence which matches the regular expression < (| =)^* | > (| =)^* <.

Assume that the occurrence of the pattern **INFLEXION** starts at position \( i \) and ends at position \( j \). The feature **WIDTH** computes the value \( j - i \).

**Example**
\( (1, 1, 2, 1, 3, 6, 4, 3, 3, 2, 3, 4, 5, 5, 6) \)

Figure 3.439 provides an example where the INCREASING_WIDTH_INFLEXION \( ([1, 1, 2, 1, 3, 6, 4, 3, 3, 2, 3, 4, 5, 5, 6]) \) constraint holds.
| Typical | \[|\text{VARIABLES}| > 2 \]
| | \[\text{range}(\text{VARIABLES.var}) > 1\]
Automaton

Figure 3.440 depicts the automaton associated with the constraint \textsc{increasing\_width\_inflexion}.

\[
\begin{align*}
C &\leftarrow -\infty \\
D &\leftarrow 0 \\
F &\leftarrow -\infty \\
R &\leftarrow 1
\end{align*}
\] =

\[
\begin{align*}
&> \\
&< \quad \text{if } R \wedge (F \leq C)
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow D + 1 \} &\quad \leq F \\
\{ D \leftarrow D + 1 \} &\quad \geq t
\end{align*}
\]

\[
\begin{align*}
C &\leftarrow D + 1 \\
D &\leftarrow 0 \\
F &\leftarrow D + 1 \\
R &\leftarrow R \wedge (F \leq D + 1)
\end{align*}
\]

Figure 3.440: Automaton for the \textsc{increasing\_width\_inflexion} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textsc{inflexion} pattern (transition \textit{r} \rightarrow \textit{t} has the same accumulators updates as transition \textit{t} \rightarrow \textit{r})
3.221 INCREASING_WIDTH_PEAK

**DESCRIPTION**

**Origin**
Based on the PEAK pattern.

**Constraint**
INCREASING_WIDTH_PEAK(VARIABLES)

**Argument**
VARIABLES : collection(var−dvar)

**Restriction**
required(VARIABLES,var)

**Purpose**
Succeeds if the values denoting the width of each occurrence of the PEAK pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression $< (= | <)^* (>| =)^* >$.

Assume that the occurrence of the pattern PEAK starts at position $i$ and ends at position $j$. The feature WIDTH computes the value $j - i$.

**Example**

$\langle 7, 5, 5, 1, 2, 3, 2, 2, 3, 4, 5, 2, 6, 6, 6, 1 \rangle$

Figure 3.441 provides an example where the INCREASING_WIDTH_PEAK ($\langle 7, 5, 5, 1, 2, 3, 2, 2, 3, 4, 5, 2, 6, 6, 6, 1 \rangle$) constraint holds.

**Typical**

$|\text{VARIABLES}| > 2$

$\text{range} (\text{VARIABLES}.\text{var}) > 1$
Figure 3.441: Illustrating the INCREASING_WIDTH_PEAK constraint of the Example slot
Automaton

Figure 3.442 depicts the automaton associated with the constraint *INCREASING_WIDTH_PEAK*.

![Automaton Diagram](image)

Figure 3.442: Automaton for the *INCREASING_WIDTH_PEAK* constraint obtained by applying decoration Table 2.36 to the seed transducer of the *PEAK* pattern.
INCREASING_WIDTH_PEAK 995
3.222  INCREASING_WIDTH_PLAIN

Description

Origin
Based on the PLAIN pattern.

Constraint
INCREASING_WIDTH_PLAIN(VARIABLES)

Argument
VARIABLES : collection(var−dvar)

Restriction
required(VARIABLES, var)

Purpose
Succeeds if the values denoting the width of each occurrence of the PLAIN pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression > = <. Assume that the occurrence of the pattern PLAIN starts at position i and ends at position j. The feature WIDTH computes the value j − i.

Example

((3, 6, 6, 3, 4, 5, 5, 4, 6, 6, 7, 5, 5, 6, 3, 2))

Figure 3.443 provides an example where the INCREASING_WIDTH_PLAIN ((3, 6, 6, 3, 4, 5, 5, 4, 6, 6, 7, 5, 5, 6, 3, 2)) constraint holds.

Figure 3.443: Illustrating the INCREASING_WIDTH_PLAIN constraint of the Example slot
| Typical | \(||\text{VARIABLES}\rangle > 2\)  |
|---------|---------------------------------|
|         | \(\text{range(VARIABLES.var)} > 1\) |
Automaton

Figure 3.444 depicts the automaton associated with the constraint INCREASING_WIDTH.PLAIN.

Figure 3.444: Automaton for the INCREASING_WIDTH.PLAIN constraint obtained by applying decoration Table 2.36 to the seed transducer of the PLAIN pattern.
### 3.223 INCREASING_WIDTH_PLATEAU

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the PLATEAU pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>INCREASING_WIDTH_PLATEAU(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : collection(var–dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td>required(VARIABLES, var)</td>
</tr>
</tbody>
</table>

**Purpose**
Succeeds if the values denoting the width of each occurrence of the PLATEAU pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression `<=*>.
Assume that the occurrence of the pattern PLATEAU starts at position $i$ and ends at position $j$. The feature WIDTH computes the value $j - i$.

**Example**

```
((7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5))
```

Figure 3.445 provides an example where the INCREASING_WIDTH_PLATEAU ($(7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5)$) constraint holds.

**Typical**

```
| VARIABLES | > 2
range(VARIABLES.var) | > 1
```
Figure 3.445: Illustrating the INCREASING_WIDTH_PLATEAU constraint of the Example slot
Figure 3.446 depicts the automaton associated with the constraint INCREASING_WIDTH_PLATEAU.

![Automaton Diagram]

Figure 3.446: Automaton for the INCREASING_WIDTH_PLATEAU constraint obtained by applying decoration Table 2.36 to the seed transducer of the PLATEAU pattern.
### 3.224 INCREASING_WIDTH_PROPER_PLAIN

**DESCRIPTION**

**Origin**

Based on the PROPER_PLAIN pattern.

**Constraint**

INCREASING_WIDTH_PROPER_PLAIN(VARIABLES)

**Argument**

VARIABLES : collection(var–dvar)

**Restriction**

required(VARIABLES, var)

**Purpose**

Succeeds if the values denoting the width of each occurrence of the PROPER_PLAIN pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern PROPER_PLAIN is the maximal subsequence which matches the regular expression \( > = + < \).

Assume that the occurrence of the pattern PROPER_PLAIN starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

**Example**

\([(5, 3, 3, 5, 6, 5, 4, 4, 7, 3, 6, 5, 5, 5, 7, 2)]\)

Figure 3.447 provides an example where the INCREASING_WIDTH_PROPER_PLAIN (\([5, 3, 3, 5, 6, 5, 4, 4, 7, 3, 6, 5, 5, 5, 7, 2]\)) constraint holds.

Figure 3.447: Illustrating the INCREASING_WIDTH_PROPER_PLAIN constraint of the Example slot
Typical

$|\text{VARIABLES}| > 3$

$\text{range}(\text{VARIABLES}.\text{var}) > 1$
Automaton

Figure 3.448 depicts the automaton associated with the constraint INCREASING_WIDTH_PROPER_PLAIN.

Figure 3.448: Automaton for the INCREASING_WIDTH_PROPER_PLAIN constraint obtained by applying decoration Table 2.36 to the seed transducer of the PROPER_PLAIN pattern
### 3.225 INCREASING_WIDTH_PROPER_PLATEAU

#### Description

- **Origin**: Based on the `PROPER_PLATEAU` pattern.

- **Constraint**: `INCREASING_WIDTH_PROPER_PLATEAU(VARIABLES)`

- **Argument**: `VARIABLES : collection(var−dvar)`

- **Restriction**: `required(VARIABLES,var)`

- **Purpose**: Succeeds if the values denoting the width of each occurrence of the `PROPER_PLATEAU` pattern in the time-series given by the `VARIABLES` collection are increasing. An occurrence of the pattern `PROPER_PLATEAU` is the *maximal* subsequence which matches the regular expression `< =+ >`. Assume that the occurrence of the pattern `PROPER_PLATEAU` starts at position $i$ and ends at position $j$. The feature `WIDTH` computes the value $j - i$.

- **Example**: `((7, 1, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3))`

  Figure 3.449 provides an example where the `INCREASING_WIDTH_PROPER_PLATEAU` `((7, 1, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3))` constraint holds.

- **Typical**: `|VARIABLES| > 3`
  `range(VARIABLES.var) > 1`
Figure 3.449: Illustrating the `INCREASING_WIDTH_PROPER_PLATEAU` constraint of the Example slot
Automaton

Figure 3.450 depicts the automaton associated with the constraint INCREASING_WIDTH_PROPER_PLATEAU.

\[
\begin{align*}
C &\leftarrow -\infty \\
D &\leftarrow 0 \\
F &\leftarrow -\infty \\
R &\leftarrow 1
\end{align*}
\]

Figure 3.450: Automaton for the INCREASING_WIDTH_PROPER_PLATEAU constraint obtained by applying decoration Table 2.36 to the seed transducer of the PROPER_PLATEAU pattern.
3.226 INCREASING\_WIDTH\_STEADY\_SEQUENCE

**DESCRIPTION**

**Origin**

Based on the STEADY\_SEQUENCE pattern.

**Constraint**

INCREASING\_WIDTH\_STEADY\_SEQUENCE(VARIABLES)

**Argument**

VARIABLES : collection(var−dvar)

**Restriction**

required(VARIABLES, var)

**Purpose**

Succeeds if the values denoting the width of each occurrence of the STEADY\_SEQUENCE pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern STEADY\_SEQUENCE is the maximal subsequence which matches the regular expression =+.

Assume that the occurrence of the pattern STEADY\_SEQUENCE starts at position $i$ and ends at position $j$. The feature WIDTH computes the value $j - i + 2$.

**Example**

\[
((6, 2, 2, 3, 5, 6, 4, 4, 2, 3, 5, 5, 2, 3, 4))
\]

Figure 3.451 provides an example where the INCREASING\_WIDTH\_STEADY\_SEQUENCE (\((6, 2, 2, 3, 5, 6, 4, 4, 2, 3, 5, 5, 2, 3, 4)\)) constraint holds.

**Typical**

$|\text{VARIABLES}| > 1$
Automaton. Figure 3.452 depicts the automaton associated with the constraint INCREASING_WIDTH_STEADY_SEQUENCE.

Figure 3.452: Automaton for the INCREASING_WIDTH_STEADY_SEQUENCE constraint obtained by applying decoration Table 2.36 to the seed transducer of the STEADY_SEQUENCE pattern.
### 3.227 INCREASING_WIDTH_STRICTLY_DECREASING_SEQUENCE

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th><strong>Automaton</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the <strong>STRICTLY_DECREASING_SEQUENCE</strong> pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td><strong>INCREASING_WIDTH_STRICTLY_DECREASING_SEQUENCE</strong>(VARIABLES)</td>
</tr>
<tr>
<td><strong>Argument</strong></td>
<td>VARIABLES : <em>collection</em>(var−dvar)</td>
</tr>
<tr>
<td><strong>Restriction</strong></td>
<td><strong>required</strong>(VARIABLES, var)</td>
</tr>
</tbody>
</table>

Succeeds if the values denoting the width of each occurrence of the **STRICTLY_DECREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection are increasing.

**Purpose**

An occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** is the *maximal* sub-sequence which matches the regular expression >+. Assume that the occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** starts at position *i* and ends at position *j*. The feature **WIDTH** computes the value *j* − *i* + 2.

**Example**

\[
(5, 7, 4, 4, 2, 3, 4, 3, 2, 2, 2, 4, 5, 6, 3, 2)
\]

Figure 3.453 provides an example where the **INCREASING_WIDTH_STRICTLY_DECREASING_SEQUENCE** (**\([5, 7, 4, 4, 2, 3, 4, 3, 2, 2, 2, 4, 5, 6, 3, 2]\)**) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
Figure 3.453: Illustrating the INCREASING_WIDTH STRICTLY DECREASING_SEQUENCE constraint of the Example slot
Figure 3.454: Automaton associated with the constraint \textsc{increasing\_width\_strictly\_decreasing\_sequence}.

\[
\begin{align*}
\{ & C \leftarrow -\infty \\
D & \leftarrow 0 \\
F & \leftarrow -\infty \\
R & \leftarrow 1 \\
\} \\
\downarrow \\
\leq s \\
\leq \\
\{ & D \leftarrow 0 \\
F & \leftarrow C \\
R & \leftarrow R \land (F \leq C) \\
\} \\
\forall \in S \\
\leq S \\
\{ & C \leftarrow D + 2 \\
D & \leftarrow 0 \\
\} \\
> r \\
\{ & C \leftarrow C + D + 1 \\
D & \leftarrow 0 \\
\}
\end{align*}
\]

Figure 3.454: Automaton for the \textsc{increasing\_width\_strictly\_decreasing\_sequence} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textsc{strictly\_decreasing\_sequence} pattern.
3.228 INCREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE

### Description

**Origin**
Based on the `STRICTLY_INCREASING_SEQUENCE` pattern.

**Constraint**
`INCREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE(VARIABLES)`

**Argument**
`VARIABLES : collection(var−dvar)`

**Restriction**
`required(VARIABLES, var)`

Succeeds if the values denoting the width of each occurrence of the `STRICTLY_INCREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection are increasing.

**Purpose**
An occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` is the maximal sub-sequence which matches the regular expression `<+`. Assume that the occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j − i + 2`.

**Example**
```
(⟨6, 2, 3, 6, 5, 5, 4, 5, 6, 1, 2, 3, 4, 5, 5⟩)
```

Figure 3.455 provides an example where the `INCREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE ([6, 2, 3, 6, 5, 5, 4, 5, 6, 1, 2, 3, 4, 5, 5])` constraint holds.

**Typical**

```
|VARIABLES| > 1
range(VARIABLES.var) > 1
```
Figure 3.455: Illustrating the `INCREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE` constraint of the `Example` slot
Automaton

Figure 3.456 depicts the automaton associated with the constraint \textsc{increasing\_width\_strictly\_increasing\_sequence}.

\begin{equation}
\begin{cases}
    C \leftarrow -\infty \\
    D \leftarrow 0 \\
    F \leftarrow -\infty \\
    R \leftarrow 1
\end{cases}
\end{equation}

\begin{equation}
\begin{cases}
    D \leftarrow 0 \\
    F \leftarrow C \\
    R \leftarrow R \land (F \leq C)
\end{cases}
\end{equation}

\begin{equation}
\begin{cases}
    C \leftarrow D + 2 \\
    D \leftarrow 0
\end{cases}
\end{equation}

\begin{equation}
\begin{cases}
    C \leftarrow C + D + 1 \\
    D \leftarrow 0
\end{cases}
\end{equation}

Figure 3.456: Automaton for the \textsc{increasing\_width\_strictly\_increasing\_sequence} constraint obtained by applying decoration Table 2.36 to the seed transducer of the \textsc{strictly\_increasing\_sequence} pattern.
3.229 INCREASING_WIDTH_SUMMIT

**DESCRIPTION**

**Origin**
Based on the SUMMIT pattern.

**Constraint**
INCREASING_WIDTH_SUMMIT(VARIABLES)

**Argument**
VARIABLES : collection(var--dvar)

**Restriction**
required(VARIABLES, var)

**Purpose**
Succeeds if the values denoting the width of each occurrence of the SUMMIT pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression (< | < (= | <)* <) (> | > (= | >)* >).

Assume that the occurrence of the pattern SUMMIT starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

**Example**

\((1, 5, 2, 1, 6, 6, 2, 3, 5, 4, 1, 4, 6, 4, 3, 2)\)

Figure 3.457 provides an example where the INCREASING_WIDTH_SUMMIT \((1, 5, 2, 1, 6, 6, 2, 3, 5, 4, 1, 4, 6, 4, 3, 2)\) constraint holds.

![Diagram showing example](image)

Figure 3.457: Illustrating the INCREASING_WIDTH_SUMMIT constraint of the Example slot
Typical

$|\text{VARIABLES}| > 2$

$\text{range} (\text{VARIABLES}. \text{var}) > 1$
Automaton

Figure 3.458 depicts the automaton associated with the constraint INCREASING_WIDTH_SUMMIT.

Figure 3.458: Automaton for the INCREASING_WIDTH_SUMMIT constraint obtained by applying decoration Table 2.36 to the seed transducer of the SUMMIT pattern (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$)
### 3.230 INCREASING_WIDTH_VALLEY

<table>
<thead>
<tr>
<th>Origin</th>
<th>Based on the VALLEY pattern.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constraint</td>
<td>INCREASING_WIDTH_VALLEY(VARIABLES)</td>
</tr>
<tr>
<td>Argument</td>
<td>VARIABLES : collection(var–dvar)</td>
</tr>
<tr>
<td>Restriction</td>
<td>required(VARIABLES,var)</td>
</tr>
</tbody>
</table>

**Purpose**

Succeeds if the values denoting the width of each occurrence of the VALLEY pattern in the time-series given by the VARIABLES collection are increasing.

An occurrence of the pattern VALLEY is the *maximal* subsequence which matches the regular expression \( > (= | >)^*(< | =)^* < \).

Assume that the occurrence of the pattern VALLEY starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

**Example**

\[
(7, 2, 2, 6, 3, 4, 5, 6, 6, 4, 4, 6, 7, 3, 3, 1)
\]

Figure 3.459 provides an example where the INCREASING_WIDTH_VALLEY \([(7, 2, 2, 6, 3, 4, 5, 6, 6, 4, 4, 6, 7, 3, 3, 1)]\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 2 \\
\text{range}(\text{VARIABLES.var}) > 1
\]
Figure 3.459: Illustrating the INCREASING_WIDTH_VALLEY constraint of the Example slot
Automaton

Figure 3.460 depicts the automaton associated with the constraint INCREASING_WIDTH_VALLEY.

\[
\begin{align*}
&C \leftarrow -\infty \\
&D \leftarrow 0 \\
&F \leftarrow -\infty \\
&R \leftarrow 1 \quad \{ D \leftarrow D + 1 \} \\
&\quad \{ C \leftarrow D + 1 \} \quad \{ D \leftarrow 0 \} \quad \{ C \leftarrow C + D + 1 \} \\
&\quad \{ F \leftarrow C \} \quad \{ R \leftarrow R \wedge (F \leq C) \} \\
\end{align*}
\]

Figure 3.460: Automaton for the INCREASING_WIDTH_VALLEY constraint obtained by applying decoration Table 2.36 to the seed transducer of the VALLEY pattern.
INCREASING_WIDTH_VALLEY
3.231  INCREASING_WIDTH_ZIGZAG

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
<tbody>
<tr>
<td>Origin</td>
<td>Based on the ZIGZAG pattern.</td>
</tr>
<tr>
<td>Constraint</td>
<td>INCREASING_WIDTH_ZIGZAG(VARIABLES)</td>
</tr>
<tr>
<td>Argument</td>
<td>VARIABLES : collection(var−dvar)</td>
</tr>
<tr>
<td>Restriction</td>
<td>required(VARIABLES, var)</td>
</tr>
</tbody>
</table>
| Purpose     | Succeeds if the values denoting the width of each occurrence of the ZIGZAG pattern in the time-series given by the VARIABLES collection are increasing.
An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression \((<>)^+ (< | <>) | (>><) (> | ><<)\).
Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\). |
| Example     | \(((7, 7, 2, 5, 3, 4, 6, 3, 6, 1, 1, 5, 4, 7, 5, 6))\) |
| Typical     | \(|\text{VARIABLES}| > 3\)
\(\text{range(VARIABLES.var)} > 1\) |

Figure 3.461 provides an example where the INCREASING_WIDTH_ZIGZAG ((7, 7, 2, 5, 3, 4, 6, 3, 6, 1, 1, 5, 4, 7, 5, 6)) constraint holds.
Figure 3.461: Illustrating the INCREASING_WIDTH_ZIGZAG constraint of the Example slot.
Automaton

Figure 3.462 depicts the automaton associated with the constraint INCREASING_WIDTH_ZIGZAG.
Figure 3.462: Automaton for the INCREASING_WIDTH_ZIGZAG constraint obtained by applying decoration Table 2.36 to the seed transducer of the ZIGZAG pattern; (1) missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=$; (2) on transitions from $b, c, e, f$ to $s$ the accumulator $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the accumulator $F$ is reset to $C$, and the accumulator $R$ is updated wrt $C$ and $F$. 
3.232 MAX_HEIGHT_DECREASING_TERRACE

DESCRIPTION

Based on the DECREASING_TERRACE pattern.

CONSTRAINT

MAX_HEIGHT_DECREASING_TERRACE(VALUE, VARIABLES)

ARGUMENTS

VALUE : dvar
VARIABLES : collection(var−dvar)

RESTRICTIONS

\( sv \leq 3 \lor rv \leq 2 \Rightarrow VALUE = -\infty \)
VALUE = \(-\infty \lor VALUE \geq \minv + 1 \)
VALUE \leq \maxv - 1@
required(VARIABLES, var)

where

\( \maxv = \text{maxval}(\text{VARIABLES}.\text{var}) \)
\( \minv = \text{minval}(\text{VARIABLES}.\text{var}) \)
\( sv = |\text{VARIABLES}| \)
\( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

VALUE is the maximum of all minimum values in each occurrence of the DECREASING_TERRACE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(-\infty\).

Assume that the occurrence of the pattern DECREASING_TERRACE starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\).

EXAMPLE

\((4, (6, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3))\)

Figure 3.463 provides an example where the MAX_HEIGHT_DECREASING_TERRACE (4, [6, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3]) constraint holds.

TYPICAL

| VARIABLES| > 3
range(VARIABLES.var) > 2

ARG. PROPERTIES

Functional dependency: VALUE determined by VARIABLES.
Figure 3.463: Illustrating the MAX_HEIGHT_DECREASING_TERRACE constraint of the Example slot
Figures 3.464 and 3.465 respectively depict the automaton associated with the constraint `MAX_HEIGHT_DECREASING_TERRACE` and its simplified form.

Figure 3.464: Automaton for the `MAX_HEIGHT_DECREASING_TERRACE` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `DECREASING_TERRACE` pattern where `default` is $-\infty$.

Figure 3.465: Automaton for the `MAX_HEIGHT_DECREASING_TERRACE` constraint obtained by applying decoration Table 2.37 to the seed transducer of the `DECREASING_TERRACE` pattern where `default` is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 3.1: Glue matrix for the MAX\_HEIGHT\_DECREASING\_TERRACE constraint defined as the composition of the DECREASING\_TERRACE pattern, the feature MIN, and the aggregator max: cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>max(C, C)</td>
<td>min(C, C)</td>
<td>max(C, C)</td>
</tr>
<tr>
<td>r</td>
<td>max(C, C)</td>
<td>max(C, C)</td>
<td>min(D, D, VAR_i)</td>
</tr>
<tr>
<td>t</td>
<td>max(C, C)</td>
<td>min(D, D, VAR_i)</td>
<td>min(D, D, VAR_i)</td>
</tr>
</tbody>
</table>
### 3.233 MAX_HEIGHT_INCREASING_TERRACE

#### DESCRIPTION

**Origin**

Based on the `INCREASING_TERRACE` pattern.

**Constraint**

`MAX_HEIGHT_INCREASING_TERRACE(VALUE, VARIABLES)`

**Arguments**

- `VALUE : dvar`
- `VARIABLES : collection(var−dvar)`

**Restrictions**

- `sv ≤ 3 ∨ rv ≤ 2 ⇒ VALUE = −∞`
- `VALUE = −∞ ∨ VALUE ≥ minv + 1`
- `VALUE ≤ maxv − 1`
- `required(VARIABLES, var)`

where

- `maxv = maxval(VARIABLES.var)`
- `minv = minval(VARIABLES.var)`
- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

**Purpose**

`VALUE` is the maximum of all minimum values in each occurrence of the `INCREASING_TERRACE` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, `VALUE` takes the default value $−∞$.

An occurrence of the pattern `INCREASING_TERRACE` is the maximal subsequence which matches the regular expression $< = ^+ <$.

Assume that the occurrence of the pattern `INCREASING_TERRACE` starts at position $i$ and ends at position $j$. The feature `MIN` computes the minimum of the values from index $i + 1$ to index $j$.

**Example**

$$\{(5, (1, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4))\}$$

Figure 3.466 provides an example where the `MAX_HEIGHT_INCREASING_TERRACE` ($\{(5, [1, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4])\}$) constraint holds.

**Typical**

- `|VARIABLES| > 3`
- `range(VARIABLES.var) > 2`

**Arg. properties**

Functional dependency: `VALUE` determined by `VARIABLES`.
Figure 3.466: Illustrating the MAX_HEIGHT_INCREASING_TERRACE constraint of the Example slot
Figures 3.467 and 3.468 respectively depict the automaton associated with the constraint MAX_HEIGHT_INCREASING_TERRACE and its simplified form.

Figure 3.467: Automaton for the MAX_HEIGHT_INCREASING_TERRACE constraint obtained by applying decoration Table 2.35 to the seed transducer of the INCREASING_TERRACE pattern where default is $-\infty$.

Figure 3.468: Automaton for the MAX_HEIGHT_INCREASING_TERRACE constraint obtained by applying decoration Table 2.37 to the seed transducer of the INCREASING_TERRACE pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
<table>
<thead>
<tr>
<th></th>
<th>( s )</th>
<th>( r )</th>
<th>( t )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>( \max(\vec{C}, \vec{C}) )</td>
<td>( \max(\vec{C}, \vec{C}) )</td>
<td>( \max(\vec{C}, \vec{C}) )</td>
</tr>
<tr>
<td>( r )</td>
<td>( \max(\vec{C}, \vec{C}) )</td>
<td>( \max(\vec{C}, \vec{C}) )</td>
<td>( \min(\vec{D}, \vec{D}, \text{VAR}_i) )</td>
</tr>
<tr>
<td>( t )</td>
<td>( \max(\vec{C}, \vec{C}) )</td>
<td>( \min(\vec{D}, \vec{D}, \text{VAR}_i) )</td>
<td>( \min(\vec{D}, \vec{D}, \text{VAR}_i) )</td>
</tr>
</tbody>
</table>

Table 3.2: Glue matrix for the `MAX_HEIGHT_INCREASING_TERRACE` constraint defined as the composition of the `INCREASING_TERRACE` pattern, the feature `MIN`, and the aggregator `max`: cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### 3.234 MAX_HEIGHT.PLAIN

**Description**

**Origin**

Based on the `PLAIN` pattern.

**Constraint**

`MAX_HEIGHT.PLAIN(VALUE, VARIABLES)`

**Arguments**

- `VALUE` : `dvar`
- `VARIABLES` : `collection(var−dvar)`

**Restrictions**

\[
sv \leq 2 \vee rv \leq 1 \Rightarrow VALUE = -\infty \\
VALUE = -\infty \vee VALUE \geq minv \\
VALUE \leq maxv - 1 \\
\text{required}(VARIABLES, var)
\]

where

- `maxv = \text{maxval}(VARIABLES.var)`
- `minv = \text{minval}(VARIABLES.var)`
- `sv = |VARIABLES|`
- `rv = \text{range}(VARIABLES.var)`

**Purpose**

An occurrence of the pattern `PLAIN` is the **maximal** subsequence which matches the regular expression `> = * <`.

Assume that the occurrence of the pattern `PLAIN` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i + 1` to index `j`.

**Example**

\[(5, (2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3))\]

Figure 3.469 provides an example where the `MAX_HEIGHT.PLAIN` \((5, [2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3])\) constraint holds.

**Typical**

- `|VARIABLES| > 2`
- `range(VARIABLES.var) > 1`

**Symmetry**

Items of `VARIABLES` can be reversed.

**Arg. properties**

Functional dependency: `VALUE` determined by `VARIABLES`. 
Figure 3.469: Illustrating the `MAX_HEIGHT_PLAIN` constraint of the `Example` slot
Figures 3.470 and 3.471 respectively depict the automaton associated with the constraint MAX_HEIGHT_PLAIN and its simplified form.

Figure 3.470: Automaton for the MAX_HEIGHT_PLAIN constraint obtained by applying decoration Table 2.35 to the seed transducer of the PLAIN pattern where default is −∞.

Figure 3.471: Automaton for the MAX_HEIGHT_PLAIN constraint obtained by applying decoration Table 2.37 to the seed transducer of the PLAIN pattern where default is −∞; \( R_i - R_{i-1} \geq 0 \) is a linear invariant.
Table 3.3: Glue matrix for the MAX_HEIGHTPLAIN constraint defined as the composition of the PLAIN pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.235  MAX_HEIGHT_PLATEAU

**Origin**
Based on the PLATEAU pattern.

**Constraint**
MAX_HEIGHT_PLATEAU(VALUE, VARIABLES)

**Arguments**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>: dvar</td>
</tr>
<tr>
<td>VARIABLES</td>
<td>: collection(var−dvar)</td>
</tr>
</tbody>
</table>

**Restrictions**

\[
sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = -\infty \\
VALUE = -\infty \lor VALUE \geq \text{minv} + 1 \\
VALUE \leq \text{maxv} \\
\text{required}(\text{VARIABLES}.\text{var})
\]

where

\[
\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \\
sv = |\text{VARIABLES}| \\
rv = \text{range}(\text{VARIABLES}.\text{var})
\]

VALUE is the maximum of all minimum values in each occurrence of the PLATEAU pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value $-\infty$.

**Purpose**

An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression $<*=^*>$.
Assume that the occurrence of the pattern PLATEAU starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i + 1$ to index $j$.

**Example**

(5, (7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5))

Figure 3.472 provides an example where the MAX_HEIGHT_PLATEAU (5, [7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5]) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 2 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.472: Illustrating the MAX_HEIGHT_PLATEAU constraint of the Example slot
Automaton

Figures 3.473 and 3.474 respectively depict the automaton associated with the constraint `MAX_HEIGHT_PLATEAU` and its simplified form.

Figure 3.473: Automaton for the `MAX_HEIGHT_PLATEAU` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `PLATEAU` pattern where `default` is $-\infty$.

Figure 3.474: Automaton for the `MAX_HEIGHT_PLATEAU` constraint obtained by applying decoration Table 2.37 to the seed transducer of the `PLATEAU` pattern where `default` is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 3.4: Glue matrix for the MAX_HEIGHT_PLATEAU constraint defined as the composition of the PLATEAU pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.236 MAX_HEIGHT_PROPER_PLAIN

### DESCRIPTION

**Origin**
Based on the PROPERPLAIN pattern.

**Constraint**
MAX_HEIGHT_PROPER_PLAIN(VALUE, VARIABLES)

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>dvar</td>
</tr>
<tr>
<td>VARIABLES</td>
<td>collection(var−dvar)</td>
</tr>
</tbody>
</table>

**Restrictions**

\[
sv \leq 3 \land rv \leq 1 \Rightarrow VALUE = -\infty \\
VALUE = -\infty \land VALUE \geq minv \\
VALUE \leq maxv - 1 \\
\text{required}(\text{VARIABLES}, \text{var})
\]

where

- \( maxv = \text{maxval}(\text{VARIABLES}.\text{var}) \)
- \( minv = \text{minval}(\text{VARIABLES}.\text{var}) \)
- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

**Purpose**

VALUE is the maximum of all minimum values in each occurrence of the PROPERPLAIN pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(-\infty\).

An occurrence of the pattern PROPERPLAIN is the maximal subsequence which matches the regular expression \( > = + < \).

Assume that the occurrence of the pattern PROPERPLAIN starts at position \( i \) and ends at position \( j \). The feature MIN computes the minimum of the values from index \( i + 1 \) to index \( j \).

**Example**

\((5, (2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5))\)

Figure 3.475 provides an example where the MAX_HEIGHT_PROPER_PLAIN (5, [2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5]) constraint holds.

**Typical**

\([|\text{VARIABLES}| > 3 \land \text{range}(\text{VARIABLES}.\text{var}) > 1\]

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.475: Illustrating the MAX_HEIGHT_PROPER_PLAIN constraint of the Example slot
Automaton

Figures 3.476 and 3.477 respectively depict the automaton associated with the constraint \textit{MAX\_HEIGHT\_PROPER\_PLAIN} and its simplified form.

\begin{align*}
\max(R,C) &\leq s \\
D &\leftarrow \max(R, \min(D, \text{VAR}_i)) \\
D &\leftarrow \min(D, \text{VAR}_i) \\
D &\leftarrow +\infty \\
\end{align*}

Figure 3.476: Automaton for the \textit{MAX\_HEIGHT\_PROPER\_PLAIN} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \textit{PROPER\_PLAIN} pattern where \texttt{default} is $-\infty$

\begin{align*}
\max(R, C) &\leq s \\
D &\leftarrow \max(R, \text{VAR}_i) \\
R &\leftarrow \text{default} \\
\end{align*}

Figure 3.477: Automaton for the \textit{MAX\_HEIGHT\_PROPER\_PLAIN} constraint obtained by applying decoration Table 2.37 to the seed transducer of the \textit{PROPER\_PLAIN} pattern where \texttt{default} is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 3.5: Glue matrix for the `MAX_HEIGHT_PROPER_PLAIN` constraint defined as the composition of the `PROPER_PLAIN` pattern, the feature `MIN`, and the aggregator `max`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.237 MAX_HEIGHT_PROPER_PLATEAU

**Origin**
Based on the PROPER_PLATEAU pattern.

**Constraint**
MAX_HEIGHT_PROPER_PLATEAU(VALUE, VARIABLES)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var−dvar)

**Restrictions**
- sv ≤ 3 ∨ rv ≤ 1 ⇒ VALUE = −∞
- VALUE = −∞ ∨ VALUE ≥ minv + 1
- VALUE ≤ maxv
  where
  - maxv = maxval(VARIABLES.var)
  - minv = minval(VARIABLES.var)
  - sv = |VARIABLES|
  - rv = range(VARIABLES.var)

**Purpose**
VALUE is the maximum of all minimum values in each occurrence of the PROPER_PLATEAU pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value −∞.

An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression < =+ >.

Assume that the occurrence of the pattern PROPER_PLATEAU starts at position i and ends at position j. The feature min computes the minimum of the values from index i+1 to index j.

**Example**
(5, (7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3))

Figure 3.478 provides an example where the MAX_HEIGHT_PROPER_PLATEAU (5, [7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3]) constraint holds.

**Typical**
- |VARIABLES| > 3
- range(VARIABLES.var) > 1

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.478: Illustrating the MAX_HEIGHT_PROPER_PLATEAU constraint of the Example slot
Figures 3.479 and 3.480 respectively depict the automaton associated with the constraint MAX\_HEIGHT\_PROPER\_PLATEAU and its simplified form.

Figure 3.479: Automaton for the MAX\_HEIGHT\_PROPER\_PLATEAU constraint obtained by applying decoration Table 2.35 to the seed transducer of the PROPER\_PLATEAU pattern where default is $-\infty$.

Figure 3.480: Automaton for the MAX\_HEIGHT\_PROPER\_PLATEAU constraint obtained by applying decoration Table 2.37 to the seed transducer of the PROPER\_PLATEAU pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 3.6: Glue matrix for the MAX_HEIGHT_PROPER_PLATEAU constraint defined as the composition of the PROPER_PLATEAU pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>max(\overrightarrow{C}, \overrightarrow{C})</td>
<td>max(\overrightarrow{C}, \overrightarrow{C})</td>
<td>max(\overrightarrow{C}, \overrightarrow{C})</td>
</tr>
<tr>
<td>r</td>
<td>max(\overrightarrow{C}, \overrightarrow{C})</td>
<td>max(\overrightarrow{C}, \overrightarrow{C})</td>
<td>min(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_i)</td>
</tr>
<tr>
<td>t</td>
<td>max(\overrightarrow{C}, \overrightarrow{C})</td>
<td>min(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_i)</td>
<td>min(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_i)</td>
</tr>
</tbody>
</table>
3.238 MAX_HEIGHT_STEADY

**DESCRIPTION**

**Origin**
Based on the STEADY pattern.

**Constraint**
MAX_HEIGHT_STEADY(VALUE, VARIABLES)

**Arguments**
VALUE : dvar
VARIABLES : collection(var−dvar)

**Restrictions**

\[
\begin{align*}
sv \leq 1 & \Rightarrow VALUE = -\infty \\
VALUE = -\infty \lor VALUE \geq \text{minv} \\
VALUE \leq \text{maxv} & \\
\text{required(VARIABLES, var)}
\end{align*}
\]

where

\[
\begin{align*}
\text{maxv} &= \text{maxval(VARIABLES, var)} \\
\text{minv} &= \text{minval(VARIABLES, var)} \\
sv &= |\text{VARIABLES}|
\end{align*}
\]

**Purpose**
An occurrence of the pattern STEADY is the subsequence which matches the regular expression \(=\).
Assume that the occurrence of the pattern STEADY starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i\) to index \(j + 1\).

**Example**

\[(6, (1, 1, 7, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6))\]

Figure 3.481 provides an example where the MAX_HEIGHT_STEADY (6, [1, 1, 7, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6]) constraint holds.

**Typical**

\(|\text{VARIABLES}| > 1\)

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.481: Illustrating the MAX_HEIGHT_STEADY constraint of the Example slot
Automaton

Figures 3.482 and 3.483 respectively depict the automaton associated with the constraint MAX_HEIGHT_STEADY and its simplified form.

$$\begin{cases} C \leftarrow \text{default} \\ D \leftarrow +\infty \\ R \leftarrow \text{default} \end{cases}$$

$$\{ \begin{array}{l} D \leftarrow +\infty \\ R \leftarrow \max(R, \min(D, \text{VAR}_i, \text{VAR}_{i+1})) \end{array} \}$$

Figure 3.482: Automaton for the MAX_HEIGHT_STEADY constraint obtained by applying decoration Table 2.35 to the seed transducer of the STEADY pattern where default is $-\infty$.

$$\{ R \leftarrow \text{default} \}$$

$$\{ R \leftarrow \max(R, \text{VAR}_i) \}$$

Figure 3.483: Automaton for the MAX_HEIGHT_STEADY constraint obtained by applying decoration Table 2.37 to the seed transducer of the STEADY pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.

Table 3.7: Glue matrix for the MAX_HEIGHT_STEADY constraint defined as the composition of the STEADY pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### 3.239 MAX_HEIGHT_STEADY_SEQUENCE

**Description**
- **Origin**: Based on the STEADY_SEQUENCE pattern.
- **Constraint**: MAX_HEIGHT_STEADY_SEQUENCE(VALUE, VARIABLES)
- **Arguments**
  - VALUE : dvar
  - VARIABLES : collection(var−dvar)
- **Restrictions**
  
  
  \[
  \begin{align*}
  sv \leq 1 & \Rightarrow VALUE = -\infty \\
  VALUE = -\infty & \lor VALUE \geq minv \\
  VALUE \leq maxv & \Rightarrow \text{required(VARIABLES, var)}
  \end{align*}
  
  \]

  where
  
  \[
  \begin{align*}
  \text{maxv} &= \maxval(VARIABLES, var) \\
  \text{minv} &= \minval(VARIABLES, var) \\
  sv &= |\text{VARIABLES}|
  \end{align*}
  
  **Purpose**
  
  VALUE is the maximum of all minimum values in each occurrence of the STEADY_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(-\infty\).

  An occurrence of the pattern STEADY_SEQUENCE is the maximal subsequence which matches the regular expression \(=+\).

  Assume that the occurrence of the pattern STEADY_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
(5, (3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1))
\]

Figure 3.484 provides an example where the MAX_HEIGHT_STEADY_SEQUENCE \((5, [3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1])\) constraint holds.

**Typical**

\[|\text{VARIABLES}| > 1\]

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.484: Illustrating the MAX_HEIGHT_STEADY_SEQUENCE constraint of the Example slot
Automaton

Figures 3.485 and 3.486 respectively depict the automaton associated with the constraint MAX_HEIGHT_STEADY_SEQUENCE and its simplified form.

Figure 3.485: Automaton for the MAX_HEIGHT_STEADY_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the STEADY_SEQUENCE pattern where default is $-\infty$.

Figure 3.486: Automaton for the MAX_HEIGHT_STEADY_SEQUENCE constraint obtained by applying decoration Table 2.37 to the seed transducer of the STEADY_SEQUENCE pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 3.8: Glue matrix for the MAX_HEIGHT_STEADY_SEQUENCE constraint defined as the composition of the STEADY_SEQUENCE pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}, \overrightarrow{VAR})$</td>
</tr>
</tbody>
</table>
### 3.240 MAX_MAX_BUMP_ON_DECREASING_SEQUENCE

#### Origin
Based on the `BUMP_ON_DECREASING_SEQUENCE` pattern.

#### Constraint
\[
\text{MAX}_\text{MAX}_\text{BUMP}_\text{ON}_\text{DECREASING}_\text{SEQUENCE}(\text{VALUE}, \text{VARIABLES})
\]

#### Arguments
- **VALUE**: `dvar`
- **VARIABLES**: `collection(var - dvar)`

#### Restrictions
\[
\begin{align*}
sv & \leq 5 \vee rv \leq 2 \Rightarrow VALUE = -\infty \\
VALUE & = -\infty \vee VALUE \geq \text{minv} + 2 \\
VALUE & \leq \text{maxv} \\
\text{required} & (\text{VARIABLES}, \text{var}) \\
\text{where} & \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}, \text{var}) \\
\text{minv} & = \text{minval}(\text{VARIABLES}, \text{var}) \\
sv & = |\text{VARIABLES}| \\
rv & = \text{range}(\text{VARIABLES}, \text{var})
\end{align*}
\]

**VALUE** is the maximum of all maximum values in each occurrence of the `BUMP_ON_DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, **VALUE** takes the default value $-\infty$.

#### Purpose
An occurrence of the pattern `BUMP_ON_DECREASING_SEQUENCE` is the subsequence which matches the regular expression `>><>`.
Assume that the occurrence of the pattern `BUMP_ON_DECREASING_SEQUENCE` starts at position $i$ and ends at position $j$. The feature `MAX` computes the maximum of the values from index $i + 2$ to index $j$.

#### Example
\[
(6, (7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3))
\]

Figure 3.487 provides an example where the `MAX_MAX_BUMP_ON_DECREASING_SEQUENCE` \((6, [7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3])\) constraint holds.

#### Typical
- $|\text{VARIABLES}| > 5$
- $\text{range}(\text{VARIABLES}, \text{var}) > 2$

#### Arg. properties
Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 3.487: Illustrating the MAX_MAX_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 3.488 and 3.489 respectively depict the automaton associated with the constraint **MAX_MAX_BUMP_ON_DECREASING_SEQUENCE** and its simplified form.

![Automaton Diagram](image)

Figure 3.488: Automaton for the **MAX_MAX_BUMP_ON_DECREASING_SEQUENCE** constraint obtained by applying decoration Table 2.35 to the seed transducer of the **BUMP_ON_DECREASING_SEQUENCE** pattern where **default** is $-\infty$.
Figure 3.489: Automaton for the MAX_MAX_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.27 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
### 3.241 MAX_MAX_DECREASING

**Description**

Based on the `DECREASING` pattern.

**Constraint**

$\text{MAX\_MAX\_DECREASING}(\text{VALUE}, \text{VARIABLES})$

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var-dvar)`

**Restrictions**

$sv \leq 1 \lor rv \leq 1 \Rightarrow \text{VALUE} = -\infty$

$\text{VALUE} = -\infty \lor \text{VALUE} \geq \text{minv} + 1$

$\text{VALUE} \leq \text{maxv}$

Required($\text{VARIABLES}, \text{var}$)

where

- $\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})$
- $\text{minv} = \text{minval}(\text{VARIABLES}.\text{var})$
- $sv = |\text{VARIABLES}|$
- $rv = \text{range}(\text{VARIABLES}.\text{var})$

**Purpose**

$\text{VALUE}$ is the maximum of all maximum values in each occurrence of the `DECREASING` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, $\text{VALUE}$ takes the default value $-\infty$.

An occurrence of the pattern `DECREASING` is the subsequence which matches the regular expression $>$. Assume that the occurrence of the pattern `DECREASING` starts at position $i$ and ends at position $j$. The feature `MAX` computes the maximum of the values from index $i$ to index $j + 1$.

**Example**

$\{(6, (3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))\}$

Figure 3.490 provides an example where the `MAX\_MAX\_DECREASING` ($\{6, [3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4]\}$) constraint holds.

**Typical**

- $|\text{VARIABLES}| > 1$
- $\text{range}(\text{VARIABLES}.\text{var}) > 1$

**Arg. properties**

Functional dependency: $\text{VALUE}$ determined by $\text{VARIABLES}$.
Figure 3.490: Illustrating the MAX_MAX_DECREASING constraint of the Example slot
Automaton

Figures 3.491 and 3.492 respectively depict the automaton associated with the constraint \texttt{MAX\_MAX\_DECREASING} and its simplified form.

\[
\begin{cases}
    C &\leftarrow \text{default} \\
    D &\leftarrow -\infty \\
    R &\leftarrow \text{default}
\end{cases}
\]

\[
\begin{cases}
    D &\leftarrow -\infty \\
    R &\leftarrow \max(R, \max(\max(D, \text{VAR}_i), \text{VAR}_{i+1}))
\end{cases}
\]

Figure 3.491: Automaton for the \texttt{MAX\_MAX\_DECREASING} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{DECREASING} pattern where \texttt{default} is \(-\infty\).

\[
\begin{cases}
    R &\leftarrow \text{default}
\end{cases}
\]

\[
\begin{cases}
    R &\leftarrow \max(R, \text{VAR}_i)
\end{cases}
\]

Figure 3.492: Automaton for the \texttt{MAX\_MAX\_DECREASING} constraint obtained by applying decoration Table 2.37 to the seed transducer of the \texttt{DECREASING} pattern where \texttt{default} is \(-\infty\); \(R_i - R_{i-1} \geq 0\) is a linear invariant.

Table 3.9: Glue matrix for the \texttt{MAX\_MAX\_DECREASING} constraint defined as the composition of the \texttt{DECREASING} pattern, the feature \texttt{MAX}, and the aggregator \texttt{max}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MAX_MAX_DECREASING

1073
### MAX_MAX_DECREASING_SEQUENCE

**Origin**
Based on the `DECREASING SEQUENCE` pattern.

**Constraint**

\[
\text{MAX_MAX_DECREASING_SEQUENCE}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE :</td>
<td><code>dvar</code></td>
</tr>
<tr>
<td>VARIABLES :</td>
<td><code>collection(var−dvar)</code></td>
</tr>
</tbody>
</table>

**Restrictions**

\[
sv \leq 1 \lor rv \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} = -\infty \lor \text{VALUE} \geq \minv + 1 \\
\text{VALUE} \leq \maxv \\
\text{required} (\text{VARIABLES}, \text{var}) \\
\text{where}
\]

\[
\maxv = \text{maxval}(\text{VARIABLES}.\text{var}) \\
\minv = \text{minval}(\text{VARIABLES}.\text{var}) \\
sv = |\text{VARIABLES}| \\
rv = \text{range}(\text{VARIABLES}.\text{var})
\]

**Purpose**

VALUE is the maximum of all maximum values in each occurrence of the `DECREASING SEQUENCE` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, VALUE takes the default value $-\infty$.

An occurrence of the pattern `DECREASING SEQUENCE` is the maximal subsequence which matches the regular expression $>(>|=)^*>|>$. Assume that the occurrence of the pattern `DECREASING SEQUENCE` starts at position $i$ and ends at position $j$. The feature MAX computes the maximum of the values from index $i$ to index $j + 1$.

**Example**

\[(6, (3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))\]

Figure 3.493 provides an example where the `MAX_MAX_DECREASING_SEQUENCE` (6, [3, 4, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4]) constraint holds.

**Typical**

\[|\text{VARIABLES}| > 1\]

\[\text{range}(\text{VARIABLES}.\text{var}) > 1\]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.493: Illustrating the MAX_MAX_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 3.494 and 3.495 respectively depict the automaton associated with the constraint \texttt{MAX\_MAX\_DECREASING\_SEQUENCE} and its simplified form.

\[
\begin{align*}
\{ & \quad C \leftarrow \text{default} \\
& \quad D \leftarrow -\infty \\
& \quad R \leftarrow \text{default} \}
\end{align*}
\]

Figure 3.494: Automaton for the \texttt{MAX\_MAX\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{DECREASING\_SEQUENCE} pattern where \texttt{default} is $-\infty$.

\[
\begin{align*}
C & \quad < \\
D & \quad \geq \max(R, C)
\end{align*}
\]

\[
\begin{align*}
\{ & \quad C \leftarrow \max(D, \text{VAR}_i, \text{VAR}_{i+1}) \\
& \quad D \leftarrow -\infty \}
\end{align*}
\]

\[
\begin{align*}
\{ & \quad C \leftarrow \max(C, \max(D, \text{VAR}_{i+1})) \\
& \quad D \leftarrow -\infty \}
\end{align*}
\]

\[
\begin{align*}
\{ & \quad D \leftarrow \max(D, \text{VAR}_{i+1}) \}
\end{align*}
\]

Figure 3.495: Automaton for the \texttt{MAX\_MAX\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.37 to the seed transducer of the \texttt{DECREASING\_SEQUENCE} pattern where \texttt{default} is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 3.10: Glue matrix for the MAX_MAX_DECREASING_SEQUENCE constraint defined as the composition of the DECREASING_SEQUENCE pattern, the feature MAX, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\max(\vec{C}, \vec{C})$</td>
<td>$\max(\vec{C}, \vec{C})$</td>
</tr>
<tr>
<td>t</td>
<td>$\max(\vec{C}, \vec{C})$</td>
<td>$\max(\vec{C}, \vec{C}, \vec{D}, \vec{D}, \text{VAR}_i)$</td>
</tr>
</tbody>
</table>
3.243 \textbf{MAX_MAX_DIP_ON_INCREASING_SEQUENCE}

**Description**

Based on the \textit{DIP_ON_INCREASING_SEQUENCE} pattern.

**Constraint**

\texttt{MAX_MAX_DIP_ON_INCREASING_SEQUENCE(\texttt{VALUE, VARIABLES})}

**Arguments**

\begin{itemize}
  \item \texttt{VALUE} : \texttt{dvar}
  \item \texttt{VARIABLES} : \texttt{collection(var-dvar)}
\end{itemize}

**Restrictions**

\begin{align*}
  & sv \leq 5 \lor rv \leq 2 \Rightarrow VALUE = -\infty \\
  & VALUE = -\infty \lor VALUE \geq \text{minv} + 2 \\
  & VALUE \leq \text{maxv} \\
  & \text{required}(\text{VARIABLES}, \text{var})
\end{align*}

where

\begin{itemize}
  \item \texttt{maxv} = \text{maxval}(\text{VARIABLES.var})
  \item \texttt{minv} = \text{minval}(\text{VARIABLES.var})
  \item \texttt{sv} = |\text{VARIABLES}|
  \item \texttt{rv} = \text{range}(\text{VARIABLES.var})
\end{itemize}

**Purpose**

\texttt{VALUE} is the maximum of all maximum values in each occurrence of the \textit{DIP_ON_INCREASING_SEQUENCE} pattern in the time-series given by the \texttt{VARIABLES} collection. If the pattern does not occur, \texttt{VALUE} takes the default value $-\infty$.

An occurrence of the pattern \textit{DIP_ON_INCREASING_SEQUENCE} is the subsequence which matches the regular expression \texttt{<<><<}. Assume that the occurrence of the pattern \textit{DIP_ON_INCREASING_SEQUENCE} starts at position $i$ and ends at position $j$. The feature \texttt{MAX} computes the maximum of the values from index $i + 2$ to index $j$.

**Example**

\begin{itemize}
  \item \texttt{(6, (1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4))}
\end{itemize}

Figure 3.496 provides an example where the \texttt{MAX_MAX_DIP_ON_INCREASING_SEQUENCE (6, [1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4])} constraint holds.

**Typical**

\begin{itemize}
  \item |\text{VARIABLES}| > 5
  \item range(\text{VARIABLES.var}) > 2
\end{itemize}

**Arg. properties**

Functional dependency: \texttt{VALUE} determined by \texttt{VARIABLES}.
Figure 3.496: Illustrating the MAX_MAX_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot
Figures 3.497 and 3.498 respectively depict the automaton associated with the constraint \texttt{MAX\_MAX\_DIP\_ON\_INCREASING\_SEQUENCE} and its simplified form.

\begin{itemize}
    \item $C \leftarrow \text{default}$
    \item $D \leftarrow -\infty$
    \item $R \leftarrow \text{default}$
\end{itemize}

Figure 3.497: Automaton for the \texttt{MAX\_MAX\_DIP\_ON\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.35 to the seed transducer of the DIP\_ON\_INCREASING\_SEQUENCE pattern where \texttt{default} is $-\infty$. 

Figure 3.498: Automaton for the MAX_MAX_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.25 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
### 3.244 MAX_MAX_INCREASING

**Description**

Based on the INCREASING pattern.

**Constraint**

\[
\text{MAX_MAX_INCREASING}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

<table>
<thead>
<tr>
<th>VALUE</th>
<th>: dvar</th>
</tr>
</thead>
<tbody>
<tr>
<td>VARIABLES</td>
<td>: collection(var–dvar)</td>
</tr>
</tbody>
</table>

**Restrictions**

\[
\begin{align*}
sv \leq 1 \lor rv \leq 1 & \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} = -\infty \lor \text{VALUE} \geq \text{minv} + 1 & \Rightarrow \text{VALUE} \leq \text{maxv} \\
\text{VALUE} \leq \text{maxv} & \quad \text{required(\text{VARIABLES.var})} \\
\text{where} & \\
\text{maxv} = & \maxval(\text{VARIABLES.var}) \\
\text{minv} = & \minval(\text{VARIABLES.var}) \\
sv = & |\text{VARIABLES}| \\
rv = & \text{range}(\text{VARIABLES.var})
\end{align*}
\]

**Purpose**

\[\text{VALUE}\] is the maximum of all maximum values in each occurrence of the INCREASING pattern in the time-series given by the \text{VARIABLES} collection. If the pattern does not occur, \text{VALUE} takes the default value \(-\infty\).

An occurrence of the pattern INCREASING is the subsequence which matches the regular expression <.

Assume that the occurrence of the pattern INCREASING starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
(6, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))
\]

Figure 3.499 provides an example where the MAX_MAX_INCREASING \((6, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3])\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES.var}) > 1
\]

**Arg. properties**

Functional dependency: \text{VALUE} determined by \text{VARIABLES}.
Figure 3.499: Illustrating the MAX\_MAX\_INCREASING constraint of the **Example** slot
Figures 3.500 and 3.501 respectively depict the automaton associated with the constraint MAX_MAX_INCREASING and its simplified form.

Figure 3.500: Automaton for the MAX_MAX_INCREASING constraint obtained by applying decoration Table 2.35 to the seed transducer of the INCREASING pattern where default is $-\infty$

Figure 3.501: Automaton for the MAX_MAX_INCREASING constraint obtained by applying decoration Table 2.38 to the seed transducer of the INCREASING pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.

Table 3.11: Glue matrix for the MAX_MAX_INCREASING constraint defined as the composition of the INCREASING pattern, the feature MAX, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.245 **MAX_MAX_INCREMENTING_SEQUENCE**

**DESCRIPTION**

**Origin**
Based on the `INCREMENTING_SEQUENCE` pattern.

**Constraint**

\[
\text{MAX_MAX_INCREMENTING_SEQUENCE}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td><code>dvar</code></td>
</tr>
<tr>
<td>VARIABLES</td>
<td><code>collection(var-dvar)</code></td>
</tr>
</tbody>
</table>

**Restrictions**

\[
\begin{align*}
sv \leq 1 \lor rv \leq 1 & \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} = -\infty \lor \text{VALUE} \geq \text{minv} + 1 & \\
\text{VALUE} \leq \text{maxv} & \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{where} & \\
\text{maxv} &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{minv} &= \text{minval}(\text{VARIABLES}.\text{var}) \\
sv &= |\text{VARIABLES}| \\
rv &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

VALUE is the maximum of all maximum values in each occurrence of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(-\infty\).

An occurrence of the pattern `INCREMENTING_SEQUENCE` is the maximal subsequence which matches the regular expression \(< (< | =)^* < | <\).

Assume that the occurrence of the pattern `INCREMENTING_SEQUENCE` starts at position \(i\) and ends at position \(j\). The feature `MAX` computes the maximum of the values from index \(i\) to index \(j + 1\).

**Example**

\[(6, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))\]

Figure 3.502 provides an example where the `MAX_MAX_INCREMENTING_SEQUENCE` \((6, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3])\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.502: Illustrating the MAX_MAX_INCREASING_SEQUENCE constraint of the Example slot
Figures 3.503 and 3.504 respectively depict the automaton associated with the constraint MAX_MAX_INCREASING_SEQUENCE and its simplified form.

Figure 3.503: Automaton for the MAX_MAX_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the INCREASING_SEQUENCE pattern where default is −∞.

Figure 3.504: Automaton for the MAX_MAX_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.38 to the seed transducer of the INCREASING_SEQUENCE pattern where default is −∞; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 3.12: Glue matrix for the \textsc{Max-Max-Increasing-Sequence} constraint defined as the composition of the \textsc{Increasing-Sequence} pattern, the feature \textsc{Max}, and the aggregator \textsc{max}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.246 MAX_MAX_INFLEXION

### Description

**Origin**
Based on the INFLEXION pattern.

**Constraint**
MAX_MAX_INFLEXION(VALUE, VARIABLES)

**Arguments**
- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**
- sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = −∞
- VALUE = −∞ ∨ VALUE ≥ minv
- VALUE ≤ maxv
  
  **required**(VARIABLES.var)

  where
  - maxv = maxval(VARIABLES.var)
  - minv = minval(VARIABLES.var)
  - sv = |VARIABLES|
  - rv = range(VARIABLES.var)

**Purpose**

VALUE is the maximum of all maximum values in each occurrence of the INFLEXION pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value −∞.

An occurrence of the pattern **INFLEXION** is the maximal subsequence which matches the regular expression < (| =)* > | > (| =)* <.

Assume that the occurrence of the pattern **INFLEXION** starts at position i and ends at position j. The feature **MAX** computes the maximum of the values from index i + 1 to index j.

**Example**

\[(6, (1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4))\]

Figure 3.505 provides an example where the MAX_MAX_INFLEXION (6, [1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4]) constraint holds.

**Typical**

\[|\text{VARIABLES}| > 2\]

\[\text{range}(\text{VARIABLES}.\text{var}) > 1\]

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.505: Illustrating the MAX_MAX_INFLEXION constraint of the Example slot
Automaton

Figures 3.506 and 3.507 respectively depict the automaton associated with the constraint MAX_MAX_INFLEXION and its simplified form.

Figure 3.506: Automaton for the MAX_MAX_INFLEXION constraint obtained by applying decoration Table 2.35 to the seed transducer of the INFLEXION pattern where default is $-\infty$ (transition $r \rightarrow t$ has the same accumulators updates as transition $t \rightarrow r$).

Figure 3.507: Automaton for the MAX_MAX_INFLEXION constraint obtained by applying decoration Table 2.25 to the seed transducer of the INFLEXION pattern where default is $-\infty$ (transition $r \rightarrow t$ has the same accumulators updates as transition $t \rightarrow r$); $R_i - R_{i-1} \geq 0$ is a linear invariant.
### 3.247 MAX_MAX_PEAK

**DESCRIPTION**

- **Origin**: Based on the PEAK pattern.

- **Constraint**: $\text{MAX\_MAX\_PEAK}(\text{VALUE}, \text{VARIABLES})$

- **Arguments**
  - $\text{VALUE} : \text{dvar}$
  - $\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})$

- **Restrictions**
  - $\text{sv} \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = -\infty$
  - $\text{VALUE} = -\infty \lor \text{VALUE} \geq \text{minv} + 1$
  - $\text{VALUE} \leq \text{maxv}$
  - $\text{required}(\text{VARIABLES}, \text{var})$
  - where
    - $\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})$
    - $\text{minv} = \text{minval}(\text{VARIABLES}.\text{var})$
    - $\text{sv} = |\text{VARIABLES}|$
    - $\text{rv} = \text{range}(\text{VARIABLES}.\text{var})$

**Purpose**

- $\text{VALUE}$ is the maximum of all maximum values in each occurrence of the PEAK pattern in the time-series given by the $\text{VARIABLES}$ collection. If the pattern does not occur, $\text{VALUE}$ takes the default value $-\infty$.

- An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression $< (= | <)^* (> | =)* >$.

- Assume that the occurrence of the pattern PEAK starts at position $i$ and ends at position $j$. The feature MAX computes the maximum of the values from index $i + 1$ to index $j$.

**Example**

- $(6, (7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 3, 1))$

Figure 3.508 provides an example where the MAX_MAX_PEAK $(6, [7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 3, 1])$ constraint holds.

- **Typical** $|\text{VARIABLES}| > 2$
  - $\text{range}(\text{VARIABLES}.\text{var}) > 1$

- **Symmetry** Items of $\text{VARIABLES}$ can be reversed.

- **Arg. properties** Functional dependency: $\text{VALUE}$ determined by $\text{VARIABLES}$. 
Figure 3.508: Illustrating the MAX_MAX_PEAK constraint of the Example slot
Figures 3.509 and 3.510 respectively depict the automaton associated with the constraint MAX_MAX_PEAK and its simplified form.

**Figure 3.509:** Automaton for the MAX_MAX_PEAK constraint obtained by applying decoration Table 2.35 to the seed transducer of the PEAK pattern where default is $-\infty$.

**Figure 3.510:** Automaton for the MAX_MAX_PEAK constraint obtained by applying decoration Table 2.37 to the seed transducer of the PEAK pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 3.13: Glue matrix for the MAX_MAX_PEAK constraint defined as the composition of the PEAK pattern, the feature MAX, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>r</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\max(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_i)$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}, \text{VAR}_i)$</td>
</tr>
<tr>
<td>t</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}, \text{VAR}_i)$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
</tbody>
</table>
3.248 MAX_MAX_STRICTLY_DECREASING_SEQUENCE

**Origin**
Based on the STRICTLY_DECREASING_SEQUENCE pattern.

**Constraint**
MAX_MAX_STRICTLY_DECREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var−dvar)

**Restrictions**
\[
\begin{align*}
sv &\leq 1 \lor rv \leq 1 \Rightarrow VALUE = -\infty \\
VALUE & = -\infty \lor VALUE \geq minv + 1 \\
VALUE & \leq maxv \\
\text{required}(VARIABLES, \text{var})
\end{align*}
\]
where
- \text{maxv} = \text{maxval}(VARIABLES, \text{var})
- \text{minv} = \text{minval}(VARIABLES, \text{var})
- \text{sv} = |VARIABLES|
- \text{rv} = \text{range}(VARIABLES, \text{var})

**Purpose**
VALUE is the maximum of all maximum values in each occurrence of the STRICTLY_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(-\infty\).

An occurrence of the pattern STRICTLY_DECREASING_SEQUENCE is the maximal sub-sequence which matches the regular expression \(>^+\).

Assume that the occurrence of the pattern STRICTLY_DECREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i\) to index \(j + 1\).

**Example**
\[(6, (4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3))\]

Figure 3.511 provides an example where the MAX_MAX_STRICTLY_DECREASING_SEQUENCE (6, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3]) constraint holds.

**Typical**
- \(|\text{VARIABLES}| > 1\)
- \(\text{range}(\text{VARIABLES, var}) > 1\)

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.511: Illustrating the MAX_MAX_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 3.512 and 3.513 respectively depict the automaton associated with the constraint MAX_MAX_STRICTLY_DECREASING_SEQUENCE and its simplified form.

Figure 3.512: Automaton for the MAX_MAX_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern where default is $-\infty$.

Figure 3.513: Automaton for the MAX_MAX_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.37 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 3.14: Glue matrix for the `MAX_MAX_STRICTLY_DECREASING_SEQUENCE` constraint defined as the composition of the `STRICTLY_DECREASING_SEQUENCE` pattern, the feature `MAX`, and the aggregator `max`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>( \max(\overrightarrow{C}, \overleftarrow{C}) )</td>
<td>( \max(\overrightarrow{C}, \overleftarrow{C}) )</td>
</tr>
<tr>
<td>r</td>
<td>( \max(\overrightarrow{C}, \overleftarrow{C}) )</td>
<td>( \max(\overrightarrow{C}, \overleftarrow{C}, \overrightarrow{D}, \overleftarrow{D}, \text{VAR}_i) )</td>
</tr>
</tbody>
</table>
3.249 MAX_MAX STRICTLY INCREASING SEQUENCE

DESCRIPTION

Origin
Based on the STRICTLY INCREASING SEQUENCE pattern.

Constraint
MAX_MAX STRICTLY INCREASING SEQUENCE(VALUE, VARIABLES)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>dvar</td>
</tr>
<tr>
<td>VARIABLES</td>
<td>collection(var−dvar)</td>
</tr>
</tbody>
</table>

Restrictions

\[
\begin{align*}
sv & \leq 1 \lor rv \leq 1 \Rightarrow VALUE = -\infty \\
VALUE &= -\infty \lor VALUE \geq minv + 1 \\
VALUE &\leq maxv \\
\text{required} &\text{(VARIABLES, var)} \\
\text{where} &
\begin{align*}
maxv &= \text{maxval} \text{(VARIABLES, var)} \\
minv &= \text{minval} \text{(VARIABLES, var)} \\
sv &= |\text{VARIABLES}| \\
rv &= \text{range} \text{(VARIABLES, var)}
\end{align*}
\]

Purpose

VALUE is the maximum of all maximum values in each occurrence of the STRICTLY INCREASING SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(-\infty\).

An occurrence of the pattern STRICTLY INCREASING SEQUENCE is the maximal sub-sequence which matches the regular expression <+. Assume that the occurrence of the pattern STRICTLY INCREASING SEQUENCE starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i\) to index \(j + 1\).

Example

\((6, (4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3))\)

Figure 3.514 provides an example where the MAX_MAX STRICTLY INCREASING SEQUENCE (6, [4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3]) constraint holds.

Typical

\(|\text{VARIABLES}| > 1 \\
\text{range} \text{(VARIABLES, var)} > 1\)

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 3.514: Illustrating the MAX_MAX.Strictly.Increasing.Sequence constraint of the Example slot
Figures 3.515 and 3.516 respectively depict the automaton associated with the constraint \textsc{max}\_\textsc{max}_\textsc{strictly}\_\textsc{increasing}\_\textsc{sequence} and its simplified form.

\begin{equation}
\begin{cases}
C \leftarrow \text{default} \\
D \leftarrow -\infty \\
R \leftarrow \text{default}
\end{cases}
\end{equation}

\begin{figure}
\centering
\begin{tikzpicture}
% Diagram code here
\end{tikzpicture}
\caption{Automaton for the \textsc{max}\_\textsc{max}_\textsc{strictly}\_\textsc{increasing}\_\textsc{sequence} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \textsc{strictly}\_\textsc{increasing}\_\textsc{sequence} pattern where \texttt{default} is $-\infty$.}
\end{figure}

\begin{equation}
\begin{cases}
C \leftarrow \max(D, \text{VAR}_i, \text{VAR}_{i+1}) \\
D \leftarrow -\infty
\end{cases}
\end{equation}

\begin{figure}
\centering
\begin{tikzpicture}
% Diagram code here
\end{tikzpicture}
\caption{Automaton for the \textsc{max}\_\textsc{max}_\textsc{strictly}\_\textsc{increasing}\_\textsc{sequence} constraint obtained by applying decoration Table 2.38 to the seed transducer of the \textsc{strictly}\_\textsc{increasing}\_\textsc{sequence} pattern where \texttt{default} is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.}
\end{figure}
Table 3.15: Glue matrix for the MAX_MAX_STRICTLY_INCREASING_SEQUENCE constraint defined as the composition of the STRICTLY_INCREASING_SEQUENCE pattern, the feature MAX, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}, \text{VAR}_i)$</td>
</tr>
</tbody>
</table>
### 3.250 MAX_MAX_SUMMIT

#### Description

**Origin**

Based on the SUMMIT pattern.

**Constraint**

\[
\text{MAX_MAX_SUMMIT}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**

\[
\begin{align*}
sv & \leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} & = -\infty \lor \text{VALUE} \geq \text{minv} + 1 \\
\text{VALUE} & \leq \text{maxv} \\
\text{required}(\text{VARIABLES}.\text{var})
\end{align*}
\]

where

- \(\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})\)
- \(\text{minv} = \text{minval}(\text{VARIABLES}.\text{var})\)
- \(sv = |\text{VARIABLES}|\)
- \(rv = \text{range}(\text{VARIABLES}.\text{var})\)

**Purpose**

**VALUE** is the maximum of all maximum values in each occurrence of the SUMMIT pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, **VALUE** takes the default value \(-\infty\).

An occurrence of the pattern **SUMMIT** is the *maximal* subsequence which matches the regular expression \((< | < (= | <)* < | > (= | >)* >)\).

Assume that the occurrence of the pattern **SUMMIT** starts at position \(i\) and ends at position \(j\). The feature **MAX** computes the maximum of the values from index \(i + 1\) to index \(j\).

#### Example

\((5, (7, 1, 5, 4, 3, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1))\)

*Figure 3.517* provides an example where the **MAX_MAX_SUMMIT** \((5, [7, 1, 5, 4, 3, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1])\) constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 2\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Symmetry**

Items of **VARIABLES** can be reversed.

**Arg. properties**

Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 3.517: Illustrating the MAX_MAX_SUMMIT constraint of the Example slot
Figures 3.518 and 3.519 respectively depict the automaton associated with the constraint MAX_MAX_SUMMIT and its simplified form.

Figure 3.518: Automaton for the MAX_MAX_SUMMIT constraint obtained by applying decoration Table 2.35 to the seed transducer of the SUMMIT pattern where default is $-\infty$ (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$).

Table 3.16: Glue matrix for the MAX_MAX_SUMMIT constraint defined as the composition of the SUMMIT pattern, the feature MAX, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 3.519: Automaton for the MAX_MAX_SUMMIT constraint obtained by applying decoration Table 2.37 to the seed transducer of the SUMMIT pattern where \( \text{default} \) is \(-\infty\) (transition \( u \rightarrow r \) has the same accumulator update as transition \( r \rightarrow u \)); \( R_i - R_{i-1} \geq 0 \) is a linear invariant.
3.251 MAX_MAX_ZIGZAG

**Description**

Based on the *ZIGZAG* pattern.

**Constraint**

\[ \text{MAX\_MAX\_ZIGZAG}(\text{VALUE}, \text{VARIABLES}) \]

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**

\[
\begin{align*}
\text{sv} \leq 3 & \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} = -\infty & \lor \text{VALUE} \geq \text{minv} + 1 \\
\text{VALUE} \leq \text{maxv} & \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{where} & \\
\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) & \\
\text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) & \\
\text{sv} = |\text{VARIABLES}| & \\
\text{rv} = \text{range}(\text{VARIABLES}.\text{var}) &
\end{align*}
\]

**Purpose**

An occurrence of the pattern *ZIGZAG* is the *maximal* subsequence which matches the regular expression \((<>)^+ (< | <>) | (<>)^+ (> | >>)\). Assume that the occurrence of the pattern *ZIGZAG* starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i + 1\) to index \(j\).

**Example**

\[(7, (4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1))\]

Figure 3.520 provides an example where the \text{MAX\_MAX\_ZIGZAG} constraint holds.

**Typical**

- \[|\text{VARIABLES}| > 3\]
- \[\text{range}(\text{VARIABLES}.\text{var}) > 1\]

**Symmetry**

Items of \text{VARIABLES} can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.520: Illustrating the MAX_MAX,ZIGZAG constraint of the Example slot
Automaton

Figures 3.521 and 3.522 respectively depict the automaton associated with the constraint MAX_MAX_ZIGZAG and its simplified form.
Figure 3.521: Automaton for the MAX_MAX_ZIGZAG constraint obtained by applying decoration Table 2.35 to the seed transducer of the ZIGZAG pattern where default is $-\infty$: (1) missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=$; (2) on transitions from $b, c, e, f$ to $s$ the accumulator $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the accumulator $R$ is updated wrt $C$ and the accumulator $C$ is reset to its initial value.
Figure 3.522: Automaton for the \texttt{MAX\_MAX\_ZIGZAG} constraint obtained by applying decoration Table 2.28 to the seed transducer of the \texttt{ZIGZAG} pattern where \texttt{default} is $-\infty$; (1) missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $\Rightarrow$; (2) on transitions from $b, c, e, f$ to $s$ the accumulator $D$ is reset to its initial value; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 3.17: Glue matrix for the MAX_MAX_ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature MAX, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.252 MAX_MIN_BUMP_ON_DECREASING_SEQUENCE

**Description**

Based on the BUMP_ON_DECREASING_SEQUENCE pattern.

**Constraint**

MAX_MIN_BUMP_ON_DECREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**

\[
sv \leq 5 \lor rv \leq 2 \Rightarrow VALUE = -\infty
\]

\[
VALUE = -\infty \lor VALUE \geq minv
\]

\[
VALUE \leq maxv - 2
\]

required(VARIABLES, var)

where

\[
maxv = \text{maxval}(\text{VARIABLES.var})
\]

\[
minv = \text{minval}(\text{VARIABLES.var})
\]

\[
sv = |\text{VARIABLES}|
\]

\[
rv = \text{range}(\text{VARIABLES.var})
\]

**Purpose**

VALUE is the maximum of all minimum values in each occurrence of the BUMP_ON_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value $-\infty$.

An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence which matches the regular expression $>>\langle\rangle$. Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i + 2$ to index $j$.

**Example**

\[(5, (7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3))\]

**Typical**

- $|\text{VARIABLES}| > 5$
- $\text{range(}\text{VARIABLES.var}) > 2$

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.523: Illustrating the MAX_MIN_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot
Figures 3.524 and 3.525 respectively depict the automaton associated with the constraint `MAX_MIN_BUMP_ON_DECREASING_SEQUENCE` and its simplified form.

Figure 3.524: Automaton for the `MAX_MIN_BUMP_ON_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `BUMP_ON_DECREASING_SEQUENCE` pattern where `default` is $-\infty$. 
Figure 3.525: Automaton for the MAX_MIN_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.25 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
3.253 MAX_MIN_DECREASING

DESCRIPTION

Origin
Based on the DECREASING pattern.

Constraint
MAX_MIN_DECREASING(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = −∞
VALUE = −∞ ∨ VALUE ≥ minv
VALUE ≤ maxv − 1
required(VARIABLES, var)

where
maxv = maxval(VARIABLES.var)
minv = minval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)

Purpose
VALUE is the maximum of all minimum values in each occurrence of the DECREASING pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value −∞.

An occurrence of the pattern DECREASING is the subsequence which matches the regular expression >.

Assume that the occurrence of the pattern DECREASING starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i to index j + 1.

Example
(4, [3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4])

Figure 3.526 provides an example where the MAX_MIN_DECREASING (4, [3, 4, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4]) constraint holds.

Typical
|VARIABLES| > 1
range(VARIABLES.var) > 1

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 3.526: Illustrating the MAX_MIN_DECREASING constraint of the Example slot
Figures 3.527 and 3.528 respectively depict the automaton associated with the constraint \texttt{MAX\_MIN\_DECREASING} and its simplified form.

\begin{align*}
\begin{cases}
C & \leftarrow \text{default} \\
D & \leftarrow +\infty \\
R & \leftarrow \text{default}
\end{cases}
\end{align*}

\begin{tikzpicture}
\node[state, initial] (s) {$s$};
\node[below] at (s.south) {$\max(R,C)$};
\draw[->] (s) -- node[above] {$\geq s$} (s);
\draw[->] (s) -- node[below] {$\leq$} (s);
\end{tikzpicture}

Figure 3.527: Automaton for the \texttt{MAX\_MIN\_DECREASING} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{DECREASING} pattern where \texttt{default} is $-\infty$.

\begin{align*}
\{ R & \leftarrow \text{default} \}
\end{align*}

\begin{tikzpicture}
\node[state, initial] (s) {$s$};
\node[below] at (s.south) {$R$};
\draw[->] (s) -- node[above] {$\geq s$} (s);
\draw[->] (s) -- node[below] {$\leq$} (s);
\end{tikzpicture}

Figure 3.528: Automaton for the \texttt{MAX\_MIN\_DECREASING} constraint obtained by applying decoration Table 2.38 to the seed transducer of the \texttt{DECREASING} pattern where \texttt{default} is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.

Table 3.18: Glue matrix for the \texttt{MAX\_MIN\_DECREASING} constraint defined as the composition of the \texttt{DECREASING} pattern, the feature \texttt{MIN}, and the aggregator \texttt{max}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MAX_MIN_DECREASING 1123
3.254 MAX_MIN_DECREASING_SEQUENCE

**DESCRIPTION**

**Origin**

Based on the `DECREASING_SEQUENCE` pattern.

**Constraint**

\[
\text{MAX_MIN_DECREASING_SEQUENCE}(\text{VALUE, VARIABLES})
\]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} - \text{dvar})
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
sv \leq 1 \lor rv \leq 1 & \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} = -\infty \lor \text{VALUE} \geq \min v & \\
\text{VALUE} \leq \max v - 1 & \\
\text{required} & (\text{VARIABLES, var})
\end{align*}
\]

Where

\[
\begin{align*}
\text{max v} & = \text{maxval}(\text{VARIABLES, var}) \\
\text{min v} & = \text{minval}(\text{VARIABLES, var}) \\
sv & = |\text{VARIABLES}| \\
rv & = \text{range}(\text{VARIABLES, var})
\end{align*}
\]

**Purpose**

\(\text{VALUE}\) is the maximum of all minimum values in each occurrence of the `DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, \(\text{VALUE}\) takes the default value \(-\infty\).

An occurrence of the pattern `DECREASING_SEQUENCE` is the maximal subsequence which matches the regular expression \((> | =) > | >\).

Assume that the occurrence of the pattern `DECREASING_SEQUENCE` starts at position \(i\) and ends at position \(j\). The feature `MIN` computes the minimum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
(4, (3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4))
\]

Figure 3.529 provides an example where the `MAX_MIN_DECREASING_SEQUENCE` constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES, var}) > 1
\]

**Arg. properties**

Functional dependency: `VALUE` determined by `VARIABLES`. 
Figure 3.529: Illustrating the MAX_MIN_DECREASING_SEQUENCE constraint of the Example slot
Figures 3.530 and 3.531 respectively depict the automaton associated with the constraint `MAX_MIN_DECREASING_SEQUENCE` and its simplified form.

```
{ C ← default 
  D ← +∞ 
  R ← default 
}

≤

<

{ C ← < 
  D ← +∞ 
  R ← max(R, C) 
}

≥

≤

≥

{ D ← min(D, VAR_i) 
}

= 

{ C ← default 
  R ← default 
}

≤

<

{ C ← default 
  R ← max(R, C) 
}

≥

= 

{ C ← default 
  R ← +∞ 
}
```

Figure 3.530: Automaton for the `MAX_MIN_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 2.35 to the seed transducer of the DECREASING_SEQUENCE pattern where `default` is $-\infty$.

```
{ C ← min(min(D, VAR_i), VAR_i+1) 
  D ← +∞ 
}
```

Figure 3.531: Automaton for the `MAX_MIN_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 2.24 to the seed transducer of the DECREASING_SEQUENCE pattern where `default` is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 3.19: Glue matrix for the `MAX_MIN_DECREASING_SEQUENCE` constraint defined as the composition of the `DECREASING_SEQUENCE` pattern, the `feature MIN`, and the `aggregator max`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>(s)</th>
<th>(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>(\max(\vec{C}, \vec{C}))</td>
<td>(\max(\vec{C}, \vec{C}))</td>
</tr>
<tr>
<td>(t)</td>
<td>(\max(\vec{C}, \vec{C}))</td>
<td>(\min(\vec{C}, \vec{C}, \vec{D}, \vec{D}, \text{VAR}_i))</td>
</tr>
</tbody>
</table>
3.255 **MAX\_MIN\_DIP\_ON\_INCREASING\_SEQUENCE**

**Origin**
Based on the **DIP\_ON\_INCREASING\_SEQUENCE** pattern.

**Constraint**

```
MAX\_MIN\_DIP\_ON\_INCREASING\_SEQUENCE(VALUE, VARIABLES)
```

**Arguments**

```
VALUE : dvar
VARIABLES : collection(var\_dvar)
```

**Restrictions**

```
sv ≤ 5 ∨ rv ≤ 2 ⇒ VALUE = −∞
VALUE = −∞ ∨ VALUE ≥ minv
VALUE ≤ maxv − 2 ⇒
required(VARIABLES.var)
where
maxv = maxval(VARIABLES.var)
minv = minval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)
```

**Purpose**
VALUE is the maximum of all minimum values in each occurrence of the **DIP\_ON\_INCREASING\_SEQUENCE** pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(-∞\).

An occurrence of the pattern **DIP\_ON\_INCREASING\_SEQUENCE** is the subsequence which matches the regular expression `<<><<`. Assume that the occurrence of the pattern **DIP\_ON\_INCREASING\_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **MIN** computes the minimum of the values from index \(i + 2\) to index \(j\).

**Example**

```
(2, (1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4))
```

Figure 3.532 provides an example where the **MAX\_MIN\_DIP\_ON\_INCREASING\_SEQUENCE** (2, [1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4]) constraint holds.

**Typical**

```
|VARIABLES| > 5
range(VARIABLES.var) > 2
```

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.532: Illustrating the MAX_MIN_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot
Figures 3.533 and 3.534 respectively depict the automaton associated with the constraint \texttt{MAX\_MIN\_DIP\_ON\_INCREASING\_SEQUENCE} and its simplified form.

\[
\begin{cases}
    C \leftarrow \text{default} \\
    D \leftarrow +\infty \\
    R \leftarrow \text{default}
\end{cases}
\]

\[
\begin{align*}
    \geq s & \overset{\geq}{\Rightarrow} < \overset{<}{\Rightarrow} r \overset{<}{\Rightarrow} < \overset{<}{\Rightarrow} t \\
    \{ D \leftarrow +\infty \} & \overset{\geq}{\Rightarrow} \{ D \leftarrow \min(D, R) \} & \overset{\geq}{\Rightarrow} \{ D \leftarrow +\infty \} & \overset{\geq}{\Rightarrow} \{ D \leftarrow \min(D, R) \} \\
    \{ D \leftarrow +\infty \} & \overset{\geq}{\Rightarrow} \{ D \leftarrow \min(D, R) \} & \overset{\geq}{\Rightarrow} \{ D \leftarrow +\infty \} & \overset{\geq}{\Rightarrow} \{ D \leftarrow \min(D, R) \} \\
\end{align*}
\]

Figure 3.533: Automaton for the \texttt{MAX\_MIN\_DIP\_ON\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{DIP\_ON\_INCREASING\_SEQUENCE} pattern where \texttt{default} is \(-\infty\).
Figure 3.534: Automaton for the MAX_MIN_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.27 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
### 3.256 MAX_MIN_GORGE

**DESCRIPTION**

*Origin*

Based on the GORGE pattern.

*Constraint*

\[ \text{MAX_MIN_GORGE(VALUE, VARIABLES)} \]

*Arguments*

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection(var-dvar)}
\end{align*}
\]

*Restrictions*

\[
\begin{align*}
\text{sv} \leq 2 \lor \text{rv} \leq 1 & \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} = -\infty & \lor \text{VALUE} \geq \text{minv} \\
\text{VALUE} \leq \text{maxv} - 1 & \Rightarrow \text{required}(\text{VARIABLES}.\text{var}) \\
\text{where} & \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{minv} & = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression \((> | > (= | >)^* >) (< | < (= | <)^* <)\). Assume that the occurrence of the pattern GORGE starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**

\[
(5, (1, 7, 3, 4, 4, 5, 5, 4, 2, 6, 5, 4, 6, 5, 7))
\]

Figure 3.535 provides an example where the MAX_MIN_GORGE \((5, [1, 7, 3, 4, 4, 5, 5, 4, 2, 6, 5, 4, 6, 5, 7])\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 2 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

- **Functional dependency**: VALUE determined by VARIABLES.
Figure 3.535: Illustrating the MAX_MIN_GORGE constraint of the Example slot
Figures 3.536 and 3.537 respectively depict the automaton associated with the constraint MAX_MIN_GORGE and its simplified form.

Figure 3.536: Automaton for the MAX_MIN_GORGE constraint obtained by applying decoration Table 2.35 to the seed transducer of the GORGE pattern where $\text{default}$ is $-\infty$ (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$)

Table 3.20: Glue matrix for the MAX_MIN_GORGE constraint defined as the composition of the GORGE pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 3.537: Automaton for the MAX_MIN_GORGE constraint obtained by applying decoration Table 2.37 to the seed transducer of the GORGE pattern where $\text{default}$ is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
### 3.257 MAX_MIN_INCREASING

<table>
<thead>
<tr>
<th>Description</th>
<th>AUTOMATON</th>
</tr>
</thead>
</table>

**Origin**

Based on the INCREASING pattern.

**Constraint**

\[
\text{MAX_MIN_INCREASING}(\text{VALUE, VARIABLES})
\]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection(var-dvar)}
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{sv} \leq 1 \lor \text{rv} \leq 1 & \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} = -\infty \lor \text{VALUE} \geq \text{minv} & \\
\text{VALUE} \leq \text{maxv} - 1 & \Rightarrow \text{required(VARIABLES, var)} \\
\text{where} \\
\text{maxv} &= \text{maxval(VARIABLES.var)} \\
\text{minv} &= \text{minval(VARIABLES.var)} \\
\text{sv} &= |\text{VARIABLES}| \\
\text{rv} &= \text{range(VARIABLES.var)}
\end{align*}
\]

**Purpose**

VALUE is the maximum of all minimum values in each occurrence of the INCREASING pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(-\infty\).

An occurrence of the pattern INCREASING is the subsequence which matches the regular expression \(<\).

Assume that the occurrence of the pattern INCREASING starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
\{4, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3)\}
\]

Figure 3.538 provides an example where the MAX_MIN_INCREASING \((4, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3])\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES.var}) > 1
\]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.538: Illustrating the MAX_MIN_INCREASING constraint of the Example slot
Figures 3.539 and 3.540 respectively depict the automaton associated with the constraint MAX_MIN_INCREASING and its simplified form.

\[
\begin{align*}
\{ & C \leftarrow \text{default} \cr & D \leftarrow +\infty \cr & R \leftarrow \text{default} \} \\
\{ & D \leftarrow +\infty \cr & R \leftarrow \max(R, \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1})) \}
\end{align*}
\]

Figure 3.539: Automaton for the MAX_MIN_INCREASING constraint obtained by applying decoration Table 2.35 to the seed transducer of the INCREASING pattern where default is $-\infty$

\[
\begin{align*}
\{ & R \leftarrow \text{default} \} \\
\{ & R \leftarrow \max(R, \text{VAR}_i) \}
\end{align*}
\]

Figure 3.540: Automaton for the MAX_MIN_INCREASING constraint obtained by applying decoration Table 2.37 to the seed transducer of the INCREASING pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.

Table 3.21: Glue matrix for the MAX_MIN_INCREASING constraint defined as the composition of the INCREASING pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MAX_MIN_INCREASING 1139
3.258 MAX_MIN_INCREASING_SEQUENCE

_origin_ Based on the INCREASING_SEQUENCE pattern.

**Constraint**

MAX_MIN_INCREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**

**VALUE** : dvar
**VARIABLES** : collection(var−dvar)

**Restrictions**

\[ sv \leq 1 \lor rv \leq 1 \Rightarrow \text{VALUE} = -\infty \]
\[ \text{VALUE} = -\infty \lor \text{VALUE} \geq \text{minv} \]
\[ \text{VALUE} \leq \text{maxv} - 1 \]
\[ \text{required}(\text{VARIABLES}, \text{var}) \]

where

\[ \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \]
\[ \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \]
\[ sv = |\text{VARIABLES}| \]
\[ rv = \text{range}(\text{VARIABLES}.\text{var}) \]

**Purpose**

VALUE is the maximum of all minimum values in each occurrence of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(-\infty\).

An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression \(< (< | =)^* < | <\). Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i\) to index \(j + 1\).

**Example**

\((3, (4, 3, 5, 5, 2, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))\)

Figure 3.541 provides an example where the MAX_MIN_INCREASING_SEQUENCE (3, [4, 3, 5, 5, 2, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3]) constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 1 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.541: Illustrating the MAX_MIN_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 3.542 and 3.543 respectively depict the automaton associated with the constraint MAX_MIN_INCREASING_SEQUENCE and its simplified form.

Figure 3.542: Automaton for the MAX_MIN_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the INCREASING_SEQUENCE pattern where default is \(-\infty\).

Figure 3.543: Automaton for the MAX_MIN_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.37 to the seed transducer of the INCREASING_SEQUENCE pattern where default is \(-\infty\); \(R_i - R_{i-1} \geq 0\) is a linear invariant.
Table 3.22: Glue matrix for the MAX_MIN_INCREASING_SEQUENCE constraint defined as the composition of the INCREASING_SEQUENCE pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>max(\overrightarrow{C}, \overrightarrow{C})</td>
<td>max(\overrightarrow{C}, \overrightarrow{C})</td>
</tr>
<tr>
<td>t</td>
<td>max(\overrightarrow{C}, \overrightarrow{C})</td>
<td>min(\overrightarrow{C}, \overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}, VAR_i)</td>
</tr>
</tbody>
</table>
## 3.259 MAX_MIN_INFLEXION

**DESCRIPTION**

### Origin

Based on the **INFLEXION** pattern.

### Constraint

\[ \text{MAX_MIN_INFLEXION}(\text{VALUE}, \text{VARIABLES}) \]

### Arguments

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var - dvar)`

### Restrictions

\[
\begin{align*}
sv & \leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} & = -\infty \lor \text{VALUE} \geq \text{minv} \\
\text{VALUE} & \leq \text{maxv} \\
\text{required} & (\text{VARIABLES}.\text{var}) \\
\text{where} & \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{minv} & = \text{minval}(\text{VARIABLES}.\text{var}) \\
sv & = |\text{VARIABLES}| \\
rv & = \text{range}(\text{VARIABLES}.\text{var}) \\
\end{align*}
\]

**Purpose**

\[ \text{VALUE} \] is the maximum of all minimum values in each occurrence of the **INFLEXION** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, **VALUE** takes the default value \(-\infty\).

An occurrence of the pattern **INFLEXION** is the maximal subsequence which matches the regular expression \(< (< | =) | > | (> | =) | ^* | <\).

Assume that the occurrence of the pattern **INFLEXION** starts at position \(i\) and ends at position \(j\). The feature **MIN** computes the minimum of the values from index \(i + 1\) to index \(j\).

### Example

\((5, (1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4))\)

Figure 3.544 provides an example where the **MAX_MIN_INFLEXION** \((5, [1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4])\) constraint holds.

### Typical

\[ |\text{VARIABLES}| > 2 \]

\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

### Symmetry

Items of **VARIABLES** can be reversed.

### Arg. properties

Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 3.544: Illustrating the MAX_MIN_INFLEXION constraint of the Example slot
Figures 3.545 and 3.546 respectively depict the automaton associated with the constraint \( \text{MAX}_{\text{MIN}}_{\text{INFLEXION}} \) and its simplified form.

Figure 3.545: Automaton for the \( \text{MAX}_{\text{MIN}}_{\text{INFLEXION}} \) constraint obtained by applying decoration Table 2.35 to the seed transducer of the \( \text{INFLEXION} \) pattern where \( \text{default} \) is \( -\infty \) (transition \( r \to t \) has the same accumulators updates as transition \( t \to r \)).

Figure 3.546: Automaton for the \( \text{MAX}_{\text{MIN}}_{\text{INFLEXION}} \) constraint obtained by applying decoration Table 2.25 to the seed transducer of the \( \text{INFLEXION} \) pattern where \( \text{default} \) is \( -\infty \) (transition \( r \to t \) has the same accumulators updates as transition \( t \to r \)); \( R_i - R_{i-1} \geq 0 \) is a linear invariant.
3.260  MAX_MIN_STRICTLY_DECREASING_SEQUENCE

DESCRIPTION

Origin
Based on the STRICTLY_DECREASING_SEQUENCE pattern.

Constraint
MAX_MIN_STRICTLY_DECREASING_SEQUENCE(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = −∞
VALUE = −∞ ∨ VALUE ≥ minv
VALUE ≤ maxv − 1
required(VARIABLES, var)

where
maxv = maxval(VARIABLES.var)
minv = minval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)

VALUE is the maximum of all minimum values in each occurrence of the
STRICTLY_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES
collection. If the pattern does not occur, VALUE takes the default value −∞.

Purpose
An occurrence of the pattern STRICTLY_DECREASING_SEQUENCE is the maximal sub-
sequence which matches the regular expression >+.
Assume that the occurrence of the pattern STRICTLY_DECREASING_SEQUENCE starts at
position i and ends at position j. The feature MIN computes the minimum of the values
from index i to index j + 1.

Example
(3, (4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3))

Figure 3.547 provides an example where the MAX_MIN_STRICTLY_DECREASING_SEQUENCE
(3, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3]) constraint holds.

Typical
|VARIABLES| > 1
range(VARIABLES.var) > 1

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 3.547: Illustrating the MAX_MIN_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Figures 3.548 and 3.549 respectively depict the automaton associated with the constraint MAX_MIN_STRICTLY_DECREASING_SEQUENCE and its simplified form.

Figure 3.548: Automaton for the MAX_MIN_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern where default is $-\infty$

Figure 3.549: Automaton for the MAX_MIN_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.24 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 3.23: Glue matrix for the `MAX_MIN_STRICTLY_DECREASING_SEQUENCE` constraint defined as the composition of the `STRICTLY_DECREASING_SEQUENCE` pattern, the feature `MIN`, and the aggregator `max`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>(s)</th>
<th>(r)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>(\max(\overleftarrow{C}, \overrightarrow{C}))</td>
<td>(\max(\overleftarrow{C}, \overrightarrow{C}))</td>
</tr>
<tr>
<td>(r)</td>
<td>(\max(\overleftarrow{C}, \overrightarrow{C}))</td>
<td>(\min(\overleftarrow{C}, \overrightarrow{C}, \overleftarrow{D}, \overrightarrow{D}, \text{VAR}_i))</td>
</tr>
</tbody>
</table>
3.261 MAX_MIN STRICTLY_INCREASING_SEQUENCE

**Description**

Based on the **STRICTLY_INCREASING_SEQUENCE** pattern.

**Constraint**

```
MAX_MIN STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES)
```

**Arguments**

```
VALUE : dvar
VARIABLES : collection(var−dvar)
```

**Restrictions**

```
sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = −∞
VALUE = −∞ ∨ VALUE ≥ minv
VALUE ≤ maxv − 1 ⇒
```

```
required(VARIABLES, var)
```

where

```
maxv = maxval(VARIABLES.var)
minv = minval(VARIABLES.var)
sv = |VARIABLES|
r = range(VARIABLES.var)
```

**Purpose**

VALUE is the maximum of all minimum values in each occurrence of the **STRICTLY_INCREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value $−\infty$.

An occurrence of the pattern **STRICTLY_INCREASING_SEQUENCE** is the maximal sub-sequence which matches the regular expression $<^+$. Assume that the occurrence of the pattern **STRICTLY_INCREASING_SEQUENCE** starts at position $i$ and ends at position $j$. The feature **MIN** computes the minimum of the values from index $i$ to index $j + 1$.

**Example**

```
(3, [4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3])
```

Figure 3.550 provides an example where the **MAX_MIN STRICTLY_INCREASING_SEQUENCE** (3, [4, 3, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3]) constraint holds.

**Typical**

```
|VARIABLES| > 1
range(VARIABLES.var) > 1
```

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.550: Illustrating the MAX_MIN.Strictly.Increasing.Sequence constraint of the Example slot
Automaton

Figures 3.551 and 3.552 respectively depict the automaton associated with the constraint MAX_MIN STRICTLY_INCREASING_SEQUENCE and its simplified form.

Figure 3.551: Automaton for the MAX_MIN STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern where default is $-\infty$.

Figure 3.552: Automaton for the MAX_MIN STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.37 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 3.24: Glue matrix for the `MAX_MIN_STRICTLY_INCREASING_SEQUENCE` constraint defined as the composition of the `STRICTLY_INCREASING_SEQUENCE` pattern, the feature `MIN`, and the aggregator `max`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
<tr>
<td>r</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>min((\overrightarrow{C}, \overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}, \text{VAR}_i))</td>
</tr>
</tbody>
</table>
3.262 MAX_MIN_VALLEY

**DESCRIPTION**

Origin: Based on the VALLEY pattern.

Constraint: \( \text{MAX_MIN_VALLEY}(\text{VALUE}, \text{VARIABLES}) \)

Arguments:
- \( \text{VALUE} : \text{dvar} \)
- \( \text{VARIABLES} : \text{collection}(\text{var} - \text{dvar}) \)

Restrictions:
- \( sv \leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = -\infty \)
- \( \text{VALUE} = -\infty \lor \text{VALUE} \geq \text{minv} \)
- \( \text{VALUE} \leq \text{maxv} - 1 \)
  
  required(\text{VARIABLES}, \text{var})
  
  where
  
  \( \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \)
  \( \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \)
  \( sv = |\text{VARIABLES}| \)
  \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

**Purpose**

VALUE is the maximum of all minimum values in each occurrence of the VALLEY pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \( -\infty \).

An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression \( \geq (\leq | \geq)^* |(\leq | |)^* \leq \).

Assume that the occurrence of the pattern VALLEY starts at position \( i \) and ends at position \( j \). The feature MIN computes the minimum of the values from index \( i + 1 \) to index \( j \).

**Example**

\( (5, (1, 3, 7, 4, 3, 6, 5, 3, 3, 2, 6, 5, 5, 5)) \)

Figure 3.553 provides an example where the \( \text{MAX_MIN_VALLEY} \)

\( (5, [1, 3, 7, 4, 3, 6, 5, 3, 3, 2, 6, 5, 5, 5]) \) constraint holds.

Typical:
- \( |\text{VARIABLES}| > 2 \)
- \( \text{range}(\text{VARIABLES}.\text{var}) > 1 \)

Symmetry: Items of VARIABLES can be reversed.

Arg. properties: Functional dependency: VALUE determined by VARIABLES.
Figure 3.553: Illustrating the MAX_MIN_VALLEY constraint of the **Example** slot
Automaton

Figures 3.554 and 3.555 respectively depict the automaton associated with the constraint \texttt{MAX\_MIN\_VALLEY} and its simplified form.

Figure 3.554: Automaton for the \texttt{MAX\_MIN\_VALLEY} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{VALLEY} pattern where \texttt{default} is $-\infty$.

Figure 3.555: Automaton for the \texttt{MAX\_MIN\_VALLEY} constraint obtained by applying decoration Table 2.37 to the seed transducer of the \texttt{VALLEY} pattern where \texttt{default} is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 3.25: Glue matrix for the MAX_MIN_VALLEY constraint defined as the composition of the VALLEY pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>max($\overrightarrow{C}$, $\overrightarrow{C}$)</td>
<td>max($\overrightarrow{C}$, $\overrightarrow{C}$)</td>
<td>max($\overrightarrow{C}$, $\overrightarrow{C}$)</td>
</tr>
<tr>
<td>r</td>
<td>max($\overrightarrow{C}$, $\overrightarrow{C}$)</td>
<td>min($\overrightarrow{D}$, $\overrightarrow{D}$, VAR$_i$)</td>
<td>min($\overrightarrow{C}$, $\overrightarrow{D}$, VAR$_i$)</td>
</tr>
<tr>
<td>t</td>
<td>max($\overrightarrow{C}$, $\overrightarrow{C}$)</td>
<td>min($\overrightarrow{C}$, $\overrightarrow{D}$, VAR$_i$)</td>
<td>max($\overrightarrow{C}$, $\overrightarrow{C}$)</td>
</tr>
</tbody>
</table>
3.263 MAX_MIN_ZIGZAG

DESCRIPTION

Origin
Based on the ZIGZAG pattern.

Constraint
MAX_MIN_ZIGZAG(VALUE, VARIABLES)

Arguments

| VALUE : dvar          |
| VARIABLES : collection(var−dvar) |

Restrictions

\[
sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = -\infty
\]
\[
VALUE = -\infty \lor VALUE \geq \text{minv}
\]
\[
VALUE \leq \text{maxv} - 1
\]

where

\[
\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})
\]
\[
\text{minv} = \text{minval}(\text{VARIABLES}.\text{var})
\]
\[
sv = |\text{VARIABLES}|
\]
\[
rv = \text{range}(\text{VARIABLES}.\text{var})
\]

VALUE is the maximum of all minimum values in each occurrence of the ZIGZAG pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value $-\infty$.

Purpose

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression $(<>)^+(< | <> | (<>)^+(> | >>))$. Assume that the occurrence of the pattern ZIGZAG starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i + 1$ to index $j$.

Example

\[
(1, (4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1))
\]

Figure 3.556 provides an example where the MAX_MIN_ZIGZAG (1, [4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1]) constraint holds.

Typical

| VARIABLES| > 3
| range(VARIABLES.var) > 1 |

Symmetry

Items of VARIABLES can be reversed.

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 3.556: Illustrating the MAX_MIN_ZIGZAG constraint of the **Example** slot
Automaton

Figures 3.557 and 3.558 respectively depict the automaton associated with the constraint MAX_MN_ZIGZAG and its simplified form.
Figure 3.557: Automaton for the MAX_MIN_ZIGZAG constraint obtained by applying decoration Table 2.35 to the seed transducer of the ZIGZAG pattern where default is $-\infty$: (1) missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=$; (2) on transitions from $b, c, e, f$ to $s$ the accumulator $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the accumulator $R$ is updated wrt $C$ and the accumulator $C$ is reset to its initial value.
Figure 3.558: Automaton for the MAX_MIN_ZIGZAG constraint obtained by applying decoration Table 2.23 to the seed transducer of the ZIGZAG pattern where default is $-\infty$; missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=\equiv$; (2) on transitions from $b, c, e, f$ to $s$ the accumulator $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the accumulator $R$ is updated wrt $C$ and the accumulator $C$ is reset to its initial value.; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 3.26: Glue matrix for the MAX\_MIN\_ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature MIN, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### 3.264 MAX_RANGE_DECREASING

**Description**

- **Origin**: Based on the **DECREASING** pattern.

**Constraint**

\[
\text{MAX\_RANGE\_DECREASING}(\text{VALUE, VARIABLES})
\]

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var−dvar)`

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & \geq 0 \\
\text{VALUE} & \leq \text{rv} - 1 \\
\text{required}(\text{VARIABLES, var}) & \Rightarrow \\
\text{where} & \\
\text{rv} & = \text{range}(\text{VARIABLES, var}) \\
\text{sv} & = |\text{VARIABLES}|
\end{align*}
\]

**Purpose**

- **VALUE**: The maximum value of the differences between the largest and smallest value in each occurrence of the **DECREASING** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, **VALUE** takes the default value 0.
- An occurrence of the pattern **DECREASING** is the subsequence which matches the regular expression `>`. Assume that the occurrence of the pattern **DECREASING** starts at position `i` and ends at position `j`. The feature **RANGE** computes the range of the values from index `i` to index `j + 1`.

**Example**

\[(2, (3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))\]

Figure 3.559 provides an example where the **MAX\_RANGE\_DECREASING** (2, \([3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4]\)) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1
\]

\[
\text{range}(\text{VARIABLES, var}) > 1
\]

**Symmetry**

One and the same constant can be added to the **var** attribute of all items of **VARIABLES**.

**Arg. properties**

- Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 3.559: Illustrating the MAX_RANGE_DECREASING constraint of the Example slot.
Figures 3.560 and 3.561 respectively depict the automaton associated with the constraint \textsc{MAX\_RANGE\_DECREASING} and its simplified form.

Figure 3.560: Automaton for the \textsc{MAX\_RANGE\_DECREASING} constraint obtained by applying decoration Table 2.46 to the seed transducer of the \textsc{DECREASING} pattern where \texttt{default} is 0.

Figure 3.561: Automaton for the \textsc{MAX\_RANGE\_DECREASING} constraint obtained by applying decoration Table 2.44 to the seed transducer of the \textsc{DECREASING} pattern where \texttt{default} is 0; $R_i - R_{i-1} \geq 0$ and $R_i + \text{VAR}_{i-1} - \text{VAR}_{i-2} \geq 0$ are linear invariants.
3.265 MAX_RANGE_DECREASING_SEQUENCE

**DESCRIPTION**

**Origin**

Based on the `DECREASING_SEQUENCE` pattern.

**Constraint**

`MAX_RANGE_DECREASING_SEQUENCE(VALUE, VARIABLES)`

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var−dvar)`

**Restrictions**

\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \]

\[ VALUE \geq 0 \]

\[ VALUE \leq rv - 1 \]

required(VARIABLES, var)

where

\[ rv = \text{range}(VARIABLES, var) \]

\[ sv = |VARIABLES| \]

**Purpose**

An occurrence of the pattern `DECREASING_SEQUENCE` is the *maximal* subsequence which matches the regular expression `>(>=)^*>|>`.

Assume that the occurrence of the pattern `DECREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `RANGE` computes the range of the values from index `i` to index `j + 1`.

**Example**

\[(5, [3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4])\]

Figure 3.562 provides an example where the `MAX_RANGE_DECREASING_SEQUENCE` constraint holds.

**Typical**

- `|VARIABLES| > 1`
- `range(VARIABLES, var) > 1`

**Symmetry**

One and the same constant can be added to the `var` attribute of all items of `VARIABLES`.

**Arg. properties**

Functional dependency: `VALUE` determined by `VARIABLES`. 
Figure 3.562: Illustrating the MAX_RANGE_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 3.563 and 3.564 respectively depict the automaton associated with the constraint MAX_RANGE_DECREASING_SEQUENCE and its simplified form.

Figure 3.563: Automaton for the MAX_RANGE_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.46 to the seed transducer of the DECREASING_SEQUENCE pattern where default is 0.

Figure 3.564: Automaton for the MAX_RANGE_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.42 to the seed transducer of the DECREASING_SEQUENCE pattern where default is 0; \( R_i - R_{i-1} \geq 0 \) is a linear invariant.
3.266 MAX_RANGE_INCREASING

DESCRIPTION

Origin
Based on the INCREASING pattern.

Constraint
MAX_RANGE_INCREASING(VALUE, VARIABLES)

Arguments
| VALUE : dvar |
| VARIABLES : collection(var−dvar) |

Restrictions
- sv ≤ 1 \lor rv ≤ 1 \Rightarrow VALUE = 0
- VALUE ≥ 0
- VALUE ≤ rv − 1
  \text{required}(VARIABLES, var)

where
- \text{rv = range}(VARIABLES.var)
- \text{sv = |VARIABLES|}

VALUE is the maximum value of the differences between the largest and smallest value in each occurrence of the INCREASING pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

Purpose
An occurrence of the pattern INCREASING is the subsequence which matches the regular expression <.
Assume that the occurrence of the pattern INCREASING starts at position \textit{i} and ends at position \textit{j}. The feature \texttt{RANGE} computes the range of the values from index \textit{i} to index \textit{j} + 1.

Example

```
(2, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))
```

Figure 3.565 provides an example where the MAX_RANGE_INCREASING (2, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3]) constraint holds.

Typical
- |VARIABLES| > 1
- range(VARIABLES.var) > 1

Symmetry
One and the same constant can be added to the \texttt{var} attribute of all items of VARIABLES.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 3.565: Illustrating the MAX_RANGE_INCREASING constraint of the Example slot
Figures 3.566 and 3.567 respectively depict the automaton associated with the constraint MAX_RANGE_INCREASING and its simplified form. 

Figure 3.566: Automaton for the MAX_RANGE_INCREASING constraint obtained by applying decoration Table 2.46 to the seed transducer of the INCREASING pattern where default is 0.

Figure 3.567: Automaton for the MAX_RANGE_INCREASING constraint obtained by applying decoration Table 2.45 to the seed transducer of the INCREASING pattern where default is 0; $R_i - R_{i-1} \geq 0$ and $R_i - \VAR_i - 1 + \VAR_{i-2} \geq 0$ are linear invariants.
### 3.267 MAX\_RANGE\_INCREASING\_SEQUENCE

- **Description**
  - **Origin**: Based on the `INCREASING\_SEQUENCE` pattern.
  - **Constraint**: `MAX\_RANGE\_INCREASING\_SEQUENCE(VALUE, VARIABLES)`
  - **Arguments**:
    - `VALUE` : `dvar`
    - `VARIABLES` : `collection(var - dvar)`
  - **Restrictions**:
    - \( sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \)
    - \( VALUE \geq 0 \)
    - \( VALUE \leq rv - 1 \)
    - required(VARIABLES, var)
    - where
      - \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)
      - \( sv = |\text{VARIABLES}| \)
  - **Purpose**: An occurrence of the pattern `INCREASING\_SEQUENCE` is the maximal subsequence which matches the regular expression `(<| =)^* < | <`.
  - Assume that the occurrence of the pattern `INCREASING\_SEQUENCE` starts at position `i` and ends at position `j`. The feature `RANGE` computes the range of the values from index `i` to index `j + 1`.

- **Example**: \( (5, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3)) \)

- **Figure 3.568** provides an example where the `MAX\_RANGE\_INCREASING\_SEQUENCE (5, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3])` constraint holds.

- **Typical**
  - \(|\text{VARIABLES}| > 1\)
  - \(\text{range(\text{VARIABLES}.\text{var})} > 1\)

- **Symmetry**: One and the same constant can be added to the `var` attribute of all items of `VARIABLES`.

- **Arg. properties**: Functional dependency: `VALUE` determined by `VARIABLES`. 
Figure 3.568: Illustrating the MAX\_RANGE\_INCREASING\_SEQUENCE constraint of the Example slot
Automaton

Figures 3.569 and 3.570 respectively depict the automaton associated with the constraint \texttt{MAX\_RANGE\_INCREASING\_SEQUENCE} and its simplified form.

\[
\begin{align*}
\{ \text{default} \} & \quad \geq \quad \{ \text{default} \} \\
H & \leftarrow \text{VAR}_i + 1 \\
R & \leftarrow \max(R, C)
\end{align*}
\]

\[
\begin{align*}
\{ \text{default} \} & \quad > \quad \{ \text{default} \} \\
H & \leftarrow \text{VAR}_i + 1 \\
R & \leftarrow \max(R, C)
\end{align*}
\]

\[
\begin{align*}
\{ C \leftarrow \text{default} \} & \quad \geq \quad \{ H \leftarrow \text{VAR}_{i+1} \} \\
\{ C \leftarrow \text{default} \} & \quad > \quad \{ C \leftarrow \text{default} \} \\
\{ C \leftarrow \text{default} \} & \quad \leq \quad \{ C \leftarrow \text{default} \} \\
\{ C \leftarrow \text{default} \} & \quad < \quad \{ C \leftarrow \text{default} \}
\end{align*}
\]

\[
\begin{align*}
\{ H \leftarrow \text{VAR}_i \} & \quad \geq \quad \{ H \leftarrow \text{VAR}_i \} \\
\{ H \leftarrow \text{VAR}_i \} & \quad > \quad \{ H \leftarrow \text{VAR}_i \} \\
\{ H \leftarrow \text{VAR}_i \} & \quad < \quad \{ H \leftarrow \text{VAR}_i \} \\
\{ H \leftarrow \text{VAR}_i \} & \quad = \quad \{ H \leftarrow \text{VAR}_i \}
\end{align*}
\]

Figure 3.569: Automaton for the \texttt{MAX\_RANGE\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.46 to the seed transducer of the \texttt{INCREASING\_SEQUENCE} pattern where \texttt{default} is 0

Figure 3.570: Automaton for the \texttt{MAX\_RANGE\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.43 to the seed transducer of the \texttt{INCREASING\_SEQUENCE} pattern where \texttt{default} is 0; \( R_i - R_{i-1} \geq 0 \) is a linear invariant.
3.26 MAX_RANGE_STRICTLY_DECREASING_SEQUENCE

**DESCRIPTION**

**Origin**
Based on the `STRICTLY_DECREASING_SEQUENCE` pattern.

**Constraint**

\[ \text{MAX\_RANGE\_STRICTLY\_DECREASING\_SEQUENCE}(\text{VALUE}, \text{VARIABLES}) \]

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var\_dvar)`

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & \geq 0 \\
\text{VALUE} & \leq \text{rv} - 1 \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{where} & \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\end{align*}
\]

**Purpose**

An occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` is the maximal subsequence which matches the regular expression `>+`. Assume that the occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` starts at position \(i\) and ends at position \(j\). The feature `RANGE` computes the range of the values from index \(i\) to index \(j + 1\).

**Example**

\[(5, (4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3))\]

Figure 3.57 provides an example where the `MAX\_RANGE\_STRICTLY\_DECREASING\_SEQUENCE` constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| & > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) & > 1 \\
\end{align*}
\]

**Symmetry**

One and the same constant can be added to the \texttt{var} attribute of all items of \texttt{VARIABLES}.

**Arg. properties**

Functional dependency: \texttt{VALUE} determined by \texttt{VARIABLES}. 

**AUTOMATON**
Figure 3.571: Illustrating the MAX_RANGE_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 3.572 and 3.573 respectively depict the automaton associated with the constraint MAX_RANGE_STRICTLY_DECREASING_SEQUENCE and its simplified form.

Figure 3.572: Automaton for the MAX_RANGE_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.46 to the seed transducer of the STRICTLY DECREASING_SEQUENCE pattern where default is 0;

\[ R_i - R_{i-1} \geq 0 \]

is a linear invariant.
3.269 MAX_RANGE STRICTLY_INCREASING_SEQUENCE

**DESCRIPTION**

- **Origin**: Based on the STRICTLY_INCREASING_SEQUENCE pattern.

**Constraint**

\[ \text{MAX\_RANGE\_STRICTLY\_INCREASING\_SEQUENCE}(\text{VALUE, VARIABLES}) \]

**Arguments**

- \text{VALUE} : dvar
- \text{VARIABLES} : collection(var–dvar)

**Restrictions**

\[ \begin{align*}
    & sv \leq 1 \vee rv \leq 1 \Rightarrow \text{VALUE} = 0 \\
    & \text{VALUE} \geq 0 \\
    & \text{VALUE} \leq rv - 1 \\
    & \text{required}(\text{VARIABLES}, \text{var})
\end{align*} \]

where

\[ rv = \text{range}(\text{VARIABLES}.\text{var}) \]

\[ sv = |\text{VARIABLES}| \]

**Purpose**

VALUE is the maximum value of the differences between the largest and smallest value in each occurrence of the STRICTLY_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, \text{VALUE} takes the default value 0.

An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal sub-sequence which matches the regular expression \(<^+\).

Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature RANGE computes the range of the values from index \(i\) to index \(j + 1\).

**Example**

\((5, (4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3))\)

Figure 3.574 provides an example where the MAX\_RANGE\_STRICTLY\_INCREASING\_SEQUENCE\((5, [4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3])\) constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 1 \]

\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

**Symmetry**

One and the same constant can be added to the \text{var} attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: \text{VALUE} determined by VARIABLES.
Figure 3.574: Illustrating the `MAX_RANGE STRICTLY_INCREASING_SEQUENCE` constraint of the Example slot.
Automaton

Figures 3.575 and 3.576 respectively depict the automaton associated with the constraint MAX_RANGE STRICTLY_INCREASING_SEQUENCE and its simplified form.

Figure 3.575: Automaton for the MAX_RANGE STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.46 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern where default is 0.

Figure 3.576: Automaton for the MAX_RANGE STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.43 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern where default is 0; $R_i - R_{i-1} \geq 0$ is a linear invariant.
3.270  MAX_SURF_BUMP_ON_DECREASING_SEQUENCE

**Origin**
Based on the BUMP_ON_DECREASING_SEQUENCE pattern.

**Constraint**
MAX_SURF_BUMP_ON_DECREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**
- **VALUE**: dvar
- **VARIABLES**: collection(var − dvar)

**Restrictions**
\[
\begin{align*}
sv \leq 5 \lor \text{rv} \leq 2 & \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} = -\infty \lor \text{VALUE} \geq 3 \ast \text{minv} + 3 & \\
\text{VALUE} \leq 3 \ast \text{maxv} - 3 & \\
\text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]
where
\[
\begin{align*}
\text{maxv} &= \maxval(\text{VARIABLES}, \text{var}) \\
\text{minv} &= \minval(\text{VARIABLES}, \text{var}) \\
sv &= |\text{VARIABLES}| \\
\text{rv} &= \text{range}(\text{VARIABLES}, \text{var})
\end{align*}
\]

**Purpose**
VALUE is the maximal surface of occurrences of the BUMP_ON_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value $-\infty$.

An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence which matches the regular expression $>>><>$. Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 2$ to index $j$.

**Example**

\[(16, \langle 7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3 \rangle)\]

Figure 3.577 provides an example where the MAX_SURF_BUMP_ON_DECREASING_SEQUENCE constraint holds.

**Typical**
- $|\text{VARIABLES}| > 5$
- $\text{range}(\text{VARIABLES}, \text{var}) > 2$

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.577: Illustrating the MAX_SURF_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot
Figures 3.578 and 3.579 respectively depict the automaton associated with the constraint MAX_SURF_BUMP_ON_DECREASING_SEQUENCE and its simplified form.

Figure 3.578: Automaton for the MAX_SURF_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern where default is $-\infty$. 
Figure 3.579: Automaton for the MAX_SURF_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.25 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
3.271 **MAX_SURF_DECREASING**

**DESCRIPTION**

- **Origin**: Based on the `DECREASING` pattern.

- **Constraint**: `MAX_SURF_DECREASING(VALUE, VARIABLES)`

- **Arguments**:
  - `VALUE`: `dvar`
  - `VARIABLES`: `collection(var−dvar)`

- **Restrictions**:
  
  
  \[
  \begin{align*}
  \text{sv} \leq 1 \lor \text{rv} \leq 1 & \Rightarrow \text{VALUE} = -\infty \\
  \text{VALUE} = -\infty \lor \text{VALUE} \geq 2 \times \text{minv} + 1 & \\
  \text{VALUE} \leq 2 \times \text{maxv} - 1 & \Rightarrow \text{required(VARIABLES, var)} \\
  \text{where} & \\
  \text{maxv} &= \text{maxval}(\text{VARIABLES.var}) \\
  \text{minv} &= \text{minval}(\text{VARIABLES.var}) \\
  \text{sv} &= |\text{VARIABLES}| \\
  \text{rv} &= \text{range}(\text{VARIABLES}.\text{var})
  \end{align*}
  \]

**Purpose**

- `VALUE` is the maximal surface of occurrences of the `DECREASING` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, `VALUE` takes the default value $-\infty$.

- An occurrence of the pattern `DECREASING` is the subsequence which matches the regular expression `>`.
- Assume that the occurrence of the pattern `DECREASING` starts at position $i$ and ends at position $j$. The feature `SURF` computes the sum of the values from index $i$ to index $j + 1$.

**Example**

```
(10, (3, 4, 2, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))
```

*Figure 3.580* provides an example where the `MAX_SURF_DECREASING (10, [3, 4, 2, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4])` constraint holds.

**Typical**

- `|\text{VARIABLES}| > 1`
- `\text{range}(\text{VARIABLES}.\text{var}) > 1`

**Arg. properties**

- Functional dependency: `VALUE` determined by `VARIABLES`. 
Figure 3.580: Illustrating the MAX_SURF_DECREASING constraint of the Example slot
Figures 3.581 and 3.582 respectively depict the automaton associated with the constraint \texttt{MAX\_SURF\_DECREASING} and its simplified form.

Figure 3.581: Automaton for the \texttt{MAX\_SURF\_DECREASING} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{DECREASING} pattern where \texttt{default} is $-\infty$.

Figure 3.582: Automaton for the \texttt{MAX\_SURF\_DECREASING} constraint obtained by applying decoration Table 2.38 to the seed transducer of the \texttt{DECREASING} pattern where \texttt{default} is $-\infty$; \( R_i - R_{i-1} \geq 0 \) is a linear invariant.

Table 3.27: Glue matrix for the \texttt{MAX\_SURF\_DECREASING} constraint defined as the composition of the \texttt{DECREASING} pattern, the feature \texttt{SURF}, and the aggregator \texttt{max}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.272 MAX_SURF_DECREASING_SEQUENCE

**DESCRIPTION**

Based on the DECREASING_SEQUENCE pattern.

**Constraint**

MAX_SURF_DECREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**

| VALUE | : dvar |
| VARIABLES | : collection(var−dvar) |

**Restrictions**

- $sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = -\infty$
- $rv = 2 \Rightarrow VALUE = -\infty \lor VALUE \geq 2 \times minv + 1$
- $rv \geq 3 \Rightarrow VALUE = -\infty \lor VALUE \geq \min(2 \times minv + 1, sv \times (minv + 1))$
- $rv = 2 \Rightarrow VALUE \leq 2 \times maxv - 1$
- $rv \geq 3 \Rightarrow VALUE \leq \max(2 \times maxv - 1, sv \times (maxv - 1)%)$

**Purpose**

VALUE is the maximal surface of occurrences of the DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value $-\infty$.

An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression $> (> | =)^* > | >$.

Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i$ to index $j + 1$.

**Example**

$$(18, [3, 4, 2, 5, 6, 4, 4, 4, 3, 1, 1, 4, 6, 4, 4])$$

Figure 3.583 provides an example where the MAX_SURF_DECREASING_SEQUENCE $(18, [3, 4, 2, 5, 6, 4, 4, 4, 3, 1, 1, 4, 6, 4, 4])$ constraint holds.

**Typical**

$|\text{VARIABLES}| > 1$

$\text{range}(\text{VARIABLES}.\text{var}) > 1$

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.583: Illustrating the MAX_SURF_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 3.584 depicts the automaton associated with the constraint MAX_SURF_DECREASING_SEQUENCE.

\[
\begin{align*}
\{ C \leftarrow \text{default} \} & \quad \{ D \leftarrow 0 \} & \quad \{ R \leftarrow \text{default} \} \\
\leq & \quad \leq & \\
\{ C \leftarrow \text{default} \} & \quad \{ D \leftarrow 0 \} & \quad \{ R \leftarrow \max(R, C) \} \\
< & \quad > & \quad \{ C \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \} & \quad \{ D \leftarrow 0 \} \\
\{ D \leftarrow D + \text{VAR}_{i+1} \} & \quad \{ C \leftarrow C + D + \text{VAR}_{i+1} \} & \quad \{ D \leftarrow 0 \}
\end{align*}
\]

Figure 3.584: Automaton for the MAX_SURF_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the DECREASING_SEQUENCE pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.

<table>
<thead>
<tr>
<th>s</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\overrightarrow{C} + \overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i$</td>
</tr>
</tbody>
</table>

Table 3.28: Glue matrix for the MAX_SURF_DECREASING_SEQUENCE constraint defined as the composition of the DECREASING_SEQUENCE pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### 3.273 MAX_SURF_DECREASING_TERRACE

**Description**

Based on the `DECREASING_TERRACE` pattern.

**Constraint**

`MAX_SURF_DECREASING_TERRACE(VALUE, VARIABLES)`

**Arguments**

`VALUE : dvar`

`VARIABLES : collection(var−dvar)`

**Restrictions**

\[
\begin{align*}
sv \leq 3 \lor rv \leq 2 \Rightarrow VALUE = -\infty \\
VALUE = -\infty \lor VALUE \geq \min(2*(\text{minv}+1), (sv-2) \cdot (\text{minv}+1)) \\
VALUE \leq \max(2*(\text{maxv}-1), (sv-2) \cdot (\text{maxv}-1)) \\
\text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where

\[
\begin{align*}
\text{maxv} &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
sv &= |\text{VARIABLES}| \\
\text{minv} &= \text{minval}(\text{VARIABLES}.\text{var}) \\
rv &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

VALUE is the maximal surface of occurrences of the `DECREASING_TERRACE` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, VALUE takes the default value \(-\infty\).

An occurrence of the pattern `DECREASING_TERRACE` is the maximal subsequence which matches the regular expression `\(\geq^+\)`.

Assume that the occurrence of the pattern `DECREASING_TERRACE` starts at position \(i\) and ends at position \(j\). The feature `SURF` computes the sum of the values from index \(i+1\) to index \(j\).

**Example**

\[(8, (6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3))\]

Figure 3.585 provides an example where the `MAX_SURF_DECREASING_TERRACE` (8, [6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3]) constraint holds.

**Typical**

\[|\text{VARIABLES}| > 3\]

\[\text{range}(\text{VARIABLES}.\text{var}) > 2\]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.585: Illustrating the **MAX_SURF_DECREASING_TERRACE** constraint of the **Example** slot
Automaton

Figures 3.586 and 3.587 respectively depict the automaton associated with the constraint \texttt{MAX\_SURF\_DECREASING\_TERRACE} and its simplified form.

\begin{align*}
\begin{cases}
C &\leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow \text{default}
\end{cases}
\end{align*}

Figure 3.586: Automaton for the \texttt{MAX\_SURF\_DECREASING\_TERRACE} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{DECREASING\_TERRACE} pattern where \texttt{default} is $-\infty$.

\begin{align*}
\begin{cases}
D &\leftarrow 0 \\
R &\leftarrow \text{default}
\end{cases}
\end{align*}

Figure 3.587: Automaton for the \texttt{MAX\_SURF\_DECREASING\_TERRACE} constraint obtained by applying decoration Table 2.25 to the seed transducer of the \texttt{DECREASING\_TERRACE} pattern where \texttt{default} is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 3.29: Glue matrix for the MAX_SURF_DECREASING_TERRACE constraint defined as the composition of the DECREASING_TERRACE pattern, the feature SURF, and the aggregator max: cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>max((\vec{C}, \vec{C}))</td>
<td>max((\vec{C}, \vec{C}))</td>
<td>max((\vec{C}, \vec{C}))</td>
</tr>
<tr>
<td>r</td>
<td>max((\vec{C}, \vec{C}))</td>
<td>max((\vec{C}, \vec{C}))</td>
<td>(\vec{D} + \vec{D} + \text{VAR}_i)</td>
</tr>
<tr>
<td>t</td>
<td>max((\vec{C}, \vec{C}))</td>
<td>(\vec{D} + \vec{D} + \text{VAR}_i)</td>
<td>(\vec{D} + \vec{D} + \text{VAR}_i)</td>
</tr>
</tbody>
</table>
3.274  MAX_SURF_DIP_ON_INCREASING_SEQUENCE

**Origin**  
Based on the DIP_ON_INCREASING_SEQUENCE pattern.

**Constraint**  
MAX_SURF_DIP_ON_INCREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**  
VALUE : dvar  
VARIABLES : collection(var−dvar)

**Restrictions**  
\[ sv \leq 5 \lor rv \leq 2 \Rightarrow VALUE = -\infty \]
\[ VALUE = -\infty \lor VALUE \geq 3 \times \minv + 3 \]
\[ VALUE \leq 3 \times \maxv - 3 \times \minv \]
\[ VALUE \leq 6 = 3 \times 3 - 3 \]

where
\[ \maxv = \maxval(\text{VARIABLES}.\text{var}) \]
\[ \minv = \minval(\text{VARIABLES}.\text{var}) \]
\[ sv = |\text{VARIABLES}| \]
\[ rv = \range(\text{VARIABLES}.\text{var}) \]

**Purpose**  
VALUE is the maximal surface of occurrences of the DIP_ON_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(-\infty\).

An occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence which matches the regular expression <<><<.

Assume that the occurrence of the pattern DIP_ON_INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 2\) to index \(j\).

**Example**  
\[(10, (1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4))\]

Figure 3.588 provides an example where the MAX_SURF_DIP_ON_INCREASING_SEQUENCE (10, [1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4]) constraint holds.

**Typical**  
\[ |\text{VARIABLES}| > 5 \]
\[ \range(\text{VARIABLES}.\text{var}) > 2 \]

**Arg. properties**  
Functional dependency: VALUE determined by VARIABLES.
Figure 3.588: Illustrating the MAX_SURF_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 3.589 and 3.590 respectively depict the automaton associated with the constraint $\text{MAX}_\text{SURF}_\text{DIP}_\text{ON}_\text{INCREASING}_\text{SEQUENCE}$ and its simplified form.

\[
\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow \text{default}
\end{align*}
\]

\[
\begin{align*}
\text{≥} &\quad \text{≠} \\
< &\quad \text{≤}
\end{align*}
\]

\[
\begin{align*}
\{D \leftarrow 0\} &\quad \{D \leftarrow D + \text{VAR}i\} \\
\{D \leftarrow 0\} &\quad \{D \leftarrow D + \text{VAR}i\}
\end{align*}
\]

Figure 3.589: Automaton for the $\text{MAX}_\text{SURF}_\text{DIP}_\text{ON}_\text{INCREASING}_\text{SEQUENCE}$ constraint obtained by applying decoration Table 2.35 to the seed transducer of the $\text{DIP}_\text{ON}_\text{INCREASING}_\text{SEQUENCE}$ pattern where $\text{default}$ is $-\infty$. 
Figure 3.590: Automaton for the \textsc{MAX$_{\text{SURF}}$DIP$_{\text{ON}}$INCREASING$_{\text{SEQUENCE}}$} constraint obtained by applying decoration Table 2.25 to the seed transducer of the \textsc{DIP$_{\text{ON}}$INCREASING$_{\text{SEQUENCE}}$} pattern where \texttt{default} is $-\infty$; $R_t - R_{t-1} \geq 0$ is a linear invariant.
### 3.275 MAX_SURF_GORGE

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the GORGE pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>MAX_SURF_GORGE(VALUE, VARIABLES)</td>
</tr>
</tbody>
</table>
| **Arguments** | VALUE : dvar  
| | VARIABLES : collection(var−dvar) |
| **Restrictions** | \(sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = -\infty\)  
| | \(rv = 2 \Rightarrow VALUE = -\infty \lor VALUE \geq \text{minv}\)  
| | \(rv \geq 3 \Rightarrow VALUE = -\infty \lor VALUE \geq \text{min}(\text{minv}, (sv - 2) \ast (\text{minv} + 1) - 1)\)  
| | \(rv = 2 \Rightarrow VALUE \leq \text{maxv} - 1\)  
| | \(rv \geq 3 \Rightarrow VALUE \leq \text{max}(\text{maxv} - 1\ast, (sv - 2) \ast (\text{maxv} - 1) - 1)\)  
| | required(VARIABLES, var)  
| where | \(\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})\)  
| | \(rv = \text{range}(\text{VARIABLES}.\text{var})\)  
| | \(sv = |\text{VARIABLES}|\)  
| | \(\text{minv} = \text{minval}(\text{VARIABLES}.\text{var})\) |

**Purpose**

An occurrence of the pattern GORGE is the **maximal** subsequence which matches the regular expression \((> | > (= | >)*)>(< | < (= | <)*)<\).

Assume that the occurrence of the pattern GORGE starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

\((11, 1, 7, 3, 4, 4, 5, 5, 4, 2, 6, 5, 4, 6, 5, 7))\)

Figure 3.591 provides an example where the MAX_SURF_GORGE \((11, [1, 7, 3, 4, 4, 5, 5, 4, 2, 6, 5, 4, 6, 5, 7])\) constraint holds.

**Typical**

\(|\text{VARIABLES}| > 2\)  
\(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.591: Illustrating the MAX_SURF_GORGE constraint of the Example slot
Automaton Figure 3.592 depicts the automaton associated with the constraint MAX_SURF_GORGE.

\[
\begin{align*}
\text{Figure 3.592: Automaton for the MAX_SURF_GORGE constraint obtained by applying} \\
\text{decoration Table 2.35 to the seed transducer of the GORGE pattern where default} \\
\text{is } -\infty \text{ (transition } u \rightarrow r \text{ has the same accumulator update as transition } r \rightarrow u); \\
R_i - R_{i-1} \geq 0 \text{ is a linear invariant.}
\end{align*}
\]
Table 3.30: Glue matrix for the MAX_SURF_GORGE constraint defined as the composition of the GORGE pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.276 MAX_SURF_INCREASING

**DESCRIPTION**

**Origin**

Based on the **INCREASING** pattern.

**Constraint**

\[ \text{MAX\_SURF\_INCREASING}(\text{VALUE, VARIABLES}) \]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} \rightarrow \text{dvar})
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{sv} \leq 1 & \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} & = -\infty \lor \text{VALUE} \geq 2 \times \minv + 1 \\
\text{VALUE} & \leq 2 \times \maxv - 1 \\
\text{required} & (\text{VARIABLES, var})
\end{align*}
\]

where

\[
\begin{align*}
\maxv & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
\minv & = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

An occurrence of the pattern **INCREASING** is the subsequence which matches the regular expression `<`. Assume that the occurrence of the pattern **INCREASING** starts at position \(i\) and ends at position \(j\). The feature **SURF** computes the sum of the values from index \(i\) to index \(j + 1\).

**Example**

\[(10, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))\]

Figure 3.593 provides an example where the MAX_SURF_INCREASING \((10, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3])\) constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| & > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) & > 1
\end{align*}
\]

**Arg. properties**

Functional dependency: \(\text{VALUE}\) determined by \(\text{VARIABLES}\).
Figure 3.593: Illustrating the MAX_SURF_INCREASING constraint of the Example slot
Automaton

Figures 3.594 and 3.595 respectively depict the automaton associated with the constraint `MAX_SURF_INCREASING` and its simplified form.

\[
\begin{cases}
  C \leftarrow \text{default} \\
  D \leftarrow 0 \\
  R \leftarrow \text{default}
\end{cases}
\]

\[
\begin{cases}
  D \leftarrow 0 \\
  R \leftarrow \max(R, D + \text{VAR}_i + \text{VAR}_{i+1})
\end{cases}
\]

Figure 3.594: Automaton for the `MAX_SURF_INCREASING` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `INCREASING` pattern where `default` is $-\infty$.

\[
\begin{cases}
  \{R \leftarrow \text{default}\}
\end{cases}
\]

\[
\begin{cases}
  \{R \leftarrow \max(R, \text{VAR}_i + \text{VAR}_{i+1})\}
\end{cases}
\]

Figure 3.595: Automaton for the `MAX_SURF_INCREASING` constraint obtained by applying decoration Table 2.38 to the seed transducer of the `INCREASING` pattern where `default` is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.

\[
\begin{array}{c}
  \text{s} \\
  \text{s}
\end{array}
\]

\[
\begin{array}{c}
  \text{s} \\
  \text{s}
\end{array}
\]

Table 3.31: Glue matrix for the `MAX_SURF_INCREASING` constraint defined as the composition of the `INCREASING` pattern, the feature `SURF`, and the aggregator `max`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MAX_SURF_INCREASING
### 3.277 MAX_SURF_INCREASING_SEQUENCE

**Description**

Based on the `INCREASING_SEQUENCE` pattern.

**Constraint**

\[
\text{MAX\_SURF\_INCREASING\_SEQUENCE}(\text{VALUE, VARIABLES})
\]

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**

\[
\begin{align*}
\text{sv} &\leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{rv} = 2 &\Rightarrow \text{VALUE} = -\infty \lor \text{VALUE} \geq 2 \ast \text{minv} + 1 \\
\text{rv} \geq 3 &\Rightarrow \text{VALUE} = -\infty \lor \text{VALUE} \geq \min(2 \ast \text{minv} + 1, \text{sv} \ast (\text{minv} + 1)) \\
\text{rv} = 2 &\Rightarrow \text{VALUE} \leq 2 \ast \text{maxv} - 1 \\
\text{rv} \geq 3 &\Rightarrow \text{VALUE} \leq \max(2 \ast \text{maxv} - 1, \text{sv} \ast (\text{maxv} - 1)\ast) \\
\text{required}(\text{VARIABLES, var}) &\end{align*}
\]

where

- \(\text{maxv} = \maxval(\text{VARIABLES, var})\)
- \(\text{rv} = \text{range}(\text{VARIABLES, var})\)
- \(\text{sv} = |\text{VARIABLES}|\)
- \(\text{minv} = \minval(\text{VARIABLES, var})\)

**Purpose**

VALUE is the maximal surface of occurrences of the `INCREASING_SEQUENCE` pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(-\infty\).

An occurrence of the pattern `INCREASING_SEQUENCE` is the maximal subsequence which matches the regular expression \(< (< | =)* < | <\).

Assume that the occurrence of the pattern `INCREASING_SEQUENCE` starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i\) to index \(j + 1\).

**Example**

\((17, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3])\)

Figure 3.596 provides an example where the `MAX_SURF_INCREASING_SEQUENCE (17, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3])` constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 1\)
- \(\text{range}(\text{VARIABLES, var}) > 1\)

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.596: Illustrating the MAX_SURF_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figure 3.597 depicts the automaton associated with the constraint MAX_SURF_INCREASING_SEQUENCE.

\[
\begin{align*}
\{ & C \leftarrow \text{default} \\
& D \leftarrow 0 \\
& R \leftarrow \text{default} \} \\
\xrightarrow{\ge s} \quad \ge \\
\{ & C \leftarrow \text{default} \\
& D \leftarrow 0 \\
& R \leftarrow \max(R, C) \} \\
\xrightarrow{\max(R, C)} \quad \le t \quad \xrightarrow{\le} \\
\{ & D \leftarrow D + \text{VAR}_{i+1} \} \\
\xrightarrow{=} \quad \le t \quad \xrightarrow{=} \\
\{ & C \leftarrow \text{default} \\
& D \leftarrow 0 \\
& R \leftarrow \max(R, C) \} \\
\end{align*}
\]

Figure 3.597: Automaton for the MAX_SURF_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the INCREASING_SEQUENCE pattern where default is $-\infty$; $R_i - R_{i-1} \ge 0$ is a linear invariant.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>(\max(C, \overline{C}))</td>
<td>(\max(C, \overline{C}))</td>
</tr>
<tr>
<td>$t$</td>
<td>(\max(C, \overline{C}))</td>
<td>(\overline{C} + C + \overline{D} + D + \text{VAR}_{i})</td>
</tr>
</tbody>
</table>

Table 3.32: Glue matrix for the MAX_SURF_INCREASING_SEQUENCE constraint defined as the composition of the INCREASING_SEQUENCE pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### 3.278 MAX_SURF_INCREASING_TERRACE

**Description**

Based on the INCREASING_TERRACE pattern.

**Constraint**

MAX_SURF_INCREASING_TERRACE(VALUE, VARIABLES)

**Arguments**

*VALUE*: `dvar`

*VARIABLES*: `collection(var−dvar)`

**Restrictions**

\[
sv \leq 3 \lor rv \leq 2 \Rightarrow VALUE = -\infty \\
VALUE = -\infty \lor VALUE \geq \min(2*\text{minv} + 1), (sv - 2) * (\text{minv} + 1) \\
VALUE \leq \max(2*\text{maxv} - 1), (sv - 2) * (\text{maxv} - 1) \\
\text{where}\]

\[
\text{maxv} = \maxval(VARIABLES.var) \\
sv = |VARIABLES| \\
\text{minv} = \minval(VARIABLES.var) \\
rv = \text{range}(VARIABLES.var)
\]

**Purpose**

VALUE is the maximal surface of occurrences of the INCREASING_TERRACE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(-\infty\).

An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression \(< = +<\).

Assume that the occurrence of the pattern INCREASING_TERRACE starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i+1\) to index \(j\).

**Example**

\[(10, (1, 3, 3, 2, 5, 6, 4, 4, 2, 3, 3, 4, 4))\]

Figure 3.598 provides an example where the MAX_SURF_INCREASING_TERRACE (10, [1, 3, 3, 2, 5, 6, 4, 4, 2, 3, 3, 4, 4]) constraint holds.

**Typical**

\[|\text{VARIABLES}| > 3\]

\[\text{range}(\text{VARIABLES.var}) > 2\]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.5.98: Illustrating the MAX_SURF_INCREASING_TERRACE constraint of the Example slot
Automaton

Figures 3.599 and 3.600 respectively depict the automaton associated with the constraint MAX_SURF_INCREASING_TERRACE and its simplified form.

Figure 3.599: Automaton for the MAX_SURF_INCREASING_TERRACE constraint obtained by applying decoration Table 2.35 to the seed transducer of the INCREAS-ING_TERRACE pattern where default is $-\infty$.

Figure 3.600: Automaton for the MAX_SURF_INCREASING_TERRACE constraint obtained by applying decoration Table 2.25 to the seed transducer of the INCREAS-ING_TERRACE pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 3.33: Glue matrix for the MAX_SURF_INCREASING_TERRACE constraint defined as the composition of the INCREASING_TERRACE pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.279  MAX_SURF_INFLEXION

**DESCRIPTION**

**Origin**
Based on the **INFLEXION** pattern.

**Constraint**

\[
\text{MAX\_SURF\_INFLEXION(VALUE, VARIABLES)}
\]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \ dvar \\
\text{VARIABLES} & : \ \text{collection}\ (\text{var} - \ dvar)
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} & = -\infty \lor \text{VALUE} \geq \min(\text{minv}, (\text{sv} - 2) \ast \text{minv}) \\
\text{VALUE} & \leq \max(\text{maxv}, (\text{sv} - 2) \ast \text{maxv}) \\
\text{VALUE} & = -\infty \lor \text{VALUE} \geq \min(\text{minv}, (\text{sv} - 2) \ast \text{minv}) \\
\text{VALUE} & \leq \max(\text{maxv}, (\text{sv} - 2) \ast \text{maxv})
\end{align*}
\]

where

\[
\begin{align*}
\text{maxv} & = \maxval(\text{VARIABLES} \cdot \text{var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{minv} & = \minval(\text{VARIABLES} \cdot \text{var}) \\
\text{rv} & = \text{range}(\text{VARIABLES} \cdot \text{var})
\end{align*}
\]

**Purpose**

**VALUE** is the maximal surface of occurrences of the **INFLEXION** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, **VALUE** takes the default value \(-\infty\).

An occurrence of the pattern **INFLEXION** is the maximal subsequence which matches the regular expression \(< (\leq | =)^* > | > (\geq | =)^* <\). Assume that the occurrence of the pattern **INFLEXION** starts at position \(i\) and ends at position \(j\). The feature **SURF** computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

\[(14, (1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4))\]

Figure 3.601 provides an example where the **MAX\_SURF\_INFLEXION** \((14, [1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4])\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 2 \\
\text{range}(\text{VARIABLES} \cdot \text{var}) > 1
\]

**Symmetry**

Items of **VARIABLES** can be reversed.

**Arg. properties**

Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 3.601: Illustrating the MAX_SURF_INFLEXION constraint of the Example slot
Figures 3.602 and 3.603 respectively depict the automaton associated with the constraint MAX_SURF_INFLEXION and its simplified form.

Figure 3.602: Automaton for the MAX_SURF_INFLEXION constraint obtained by applying decoration Table 2.35 to the seed transducer of the INFLEXION pattern where default is $-\infty$ (transition $r \rightarrow t$ has the same accumulators updates as transition $t \rightarrow r$).

Figure 3.603: Automaton for the MAX_SURF_INFLEXION constraint obtained by applying decoration Table 2.25 to the seed transducer of the INFLEXION pattern where default is $-\infty$ (transition $r \rightarrow t$ has the same accumulators updates as transition $t \rightarrow r$); $R_i - R_{i-1} \geq 0$ is a linear invariant.
3.280 **MAX_SURF_PEAK**

**Origin**
Based on the PEAK pattern.

**Constraint**
MAX_SURF_PEAK(VALUE, VARIABLES)

**Arguments**

| VALUE   | : dvar |
| VARIABLES | : collection(var–dvar) |

**Restrictions**

\[
sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = -\infty \\
VALUE = -\infty \lor VALUE \geq min(minv + 1, (sv - 2) \ast (minv + 1)) \\
VALUE \leq max(maxv, (sv - 2) + maxv) \\
required(VARIABLES, var) \\
\]

where

| minv = minval(VARIABLES.var) |
| maxv = maxval(VARIABLES.var) |
| sv = | VARIABLES |
| rv = range(VARIABLES.var) |

**Purpose**

An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression \(< (= | <)^* (> | =)^* >\).

Assume that the occurrence of the pattern PEAK starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

\((14, [7, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 3, 1])\)

Figure 3.604 provides an example where the MAX_SURF_PEAK \((14, [7, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 3, 1])\) constraint holds.

**Typical**

| VARIABLES \(>| >2\) |
| range(VARIABLES.var) \(>| >1\) |

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.604: Illustrating the MAX_SURF_PEAK constraint of the Example slot
Automaton

Figure 3.605 depicts the automaton associated with the constraint MAX_SURF_PEAK.

Figure 3.605: Automaton for the MAX_SURF_PEAK constraint obtained by applying decoration Table 2.35 to the seed transducer of the PEAK pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.

Table 3.34: Glue matrix for the MAX_SURF_PEAK constraint defined as the composition of the PEAK pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.281 MAX_SURF_PLAIN

**Origin**
Based on the PLAIN pattern.

**Constraint**
MAX_SURF_PLAIN(VALUE, VARIABLES)

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>dvar</td>
</tr>
<tr>
<td>VARIABLES</td>
<td>collection(var−dvar)</td>
</tr>
</tbody>
</table>

**Restrictions**

\[
sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = -\infty
\]
\[
VALUE = -\infty \lor VALUE \geq \min(minv, (sv - 2) \ast minv)
\]
\[
VALUE \leq \max(maxv - 1, (sv - 2) \ast (maxv - 1))
\]

required(VARIABLES, var)

where

\[
maxv = \maxval(VARIABLES.var)
\]
\[
sv = |VARIABLES|
\]
\[
minv = \minval(VARIABLES.var)
\]
\[
rv = \range(VARIABLES.var)
\]

**Purpose**

An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression \(\ast\).

Assume that the occurrence of the pattern PLAIN starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

(6, (2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3))

Figure 3.606 provides an example where the MAX_SURF_PLAIN (6, [2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3]) constraint holds.

**Typical**

\(|VARIABLES| > 2
\]
\(\text{range}(VARIABLES.var) > 1
\)

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.606: Illustrating the MAX_SURFPLAIN constraint of the Example slot
Automaton

Figures 3.607 and 3.608 respectively depict the automaton associated with the constraint `MAX_SURF_PLAIN` and its simplified form.

Figure 3.607: Automaton for the `MAX_SURF_PLAIN` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `PLAIN` pattern where `default` is $-\infty$.

Figure 3.608: Automaton for the `MAX_SURF_PLAIN` constraint obtained by applying decoration Table 2.25 to the seed transducer of the `PLAIN` pattern where `default` is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 3.35: Glue matrix for the MAX_SURF.PLAIN constraint defined as the composition of the PLAIN pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\max(C, \bar{C})$</td>
<td>$\max(C, \bar{C})$</td>
<td>$\max(C, \bar{C})$</td>
</tr>
<tr>
<td>r</td>
<td>$\max(\bar{C}, C)$</td>
<td>$\bar{D} + \bar{D} + \text{VAR}_i$</td>
<td>$\bar{D} + \bar{D} + \text{VAR}_i$</td>
</tr>
<tr>
<td>t</td>
<td>$\max(\bar{C}, C)$</td>
<td>$\bar{D} + \bar{D} + \text{VAR}_i$</td>
<td>$\bar{D} + \bar{D} + \text{VAR}_i$</td>
</tr>
</tbody>
</table>
3.282 \textbf{MAX_SURF_PLATEAU}

\begin{description}
\item[Origin] Based on the PLATEAU pattern.
\item[Constraint] \texttt{MAX\_SURF\_PLATEAU(VALUE, VARIABLES)}
\item[Arguments]
\begin{itemize}
\item \texttt{VALUE} : \texttt{dvar}
\item \texttt{VARIABLES} : \texttt{collection(var--dvar)}
\end{itemize}
\item[Restrictions]
\begin{align*}
sv & \leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} &= -\infty \lor \text{VALUE} \geq \min(\text{minv} + 1, (sv - 2) \times (\text{minv} + 1)) \\
\text{VALUE} &\leq \max(\text{maxv}, (sv - 2) \times \text{maxv}) \\
\text{required} &\left(\text{VARIABLES}, \text{var} \right)
\end{align*}
where
\begin{align*}
\text{maxv} &= \text{maxval}(\text{VARIABLES} \cdot \text{var}) \\
sv &= |\text{VARIABLES}| \\
\text{minv} &= \text{minval}(\text{VARIABLES} \cdot \text{var}) \\
rv &= \text{range}(\text{VARIABLES} \cdot \text{var})
\end{align*}
\item[Purpose] \texttt{VALUE} is the maximal surface of occurrences of the \texttt{PLATEAU} pattern in the time-series given by the \texttt{VARIABLES} collection. If the pattern does not occur, \texttt{VALUE} takes the default value $-\infty$.
An occurrence of the pattern \texttt{PLATEAU} is the maximal subsequence which matches the regular expression $<\ast>$.
Assume that the occurrence of the pattern \texttt{PLATEAU} starts at position $i$ and ends at position $j$. The feature \texttt{SURF} computes the sum of the values from index $i + 1$ to index $j$.
\item[Example] $(10, \langle 7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5 \rangle)$
\item[Typical]
\begin{itemize}
\item $|\text{VARIABLES}| > 2$
\item $\text{range} (\text{VARIABLES} \cdot \text{var}) > 1$
\end{itemize}
\item[Symmetry] Items of \texttt{VARIABLES} can be reversed.
\item[Arg. properties] Functional dependency: \texttt{VALUE} determined by \texttt{VARIABLES}.
\end{description}

Figure 3.609 provides an example where the \texttt{MAX\_SURF\_PLATEAU} $(10, [7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5])$ constraint holds.
Figure 3.609: Illustrating the MAX_SURF_PLATEAU constraint of the Example slot
Figures 3.610 and 3.611 respectively depict the automaton associated with the constraint MAX_SURF_PLATEAU and its simplified form.

Figure 3.610: Automaton for the MAX_SURF_PLATEAU constraint obtained by applying decoration Table 2.35 to the seed transducer of the PLATEAU pattern where default is $-\infty$.

Figure 3.611: Automaton for the MAX_SURF_PLATEAU constraint obtained by applying decoration Table 2.25 to the seed transducer of the PLATEAU pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 3.36: Glue matrix for the MAX_SURF_PLATEAU constraint defined as the composition of the PLATEAU pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.283  MAX_SURF_PROPER/plain

Origin
Based on the PROPER/plain pattern.

Constraint
MAX_SURF_PROPER/plain(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var − dvar)

Restrictions
sv ≤ 3 ∨ rv ≤ 1 ⇒ VALUE = −∞
VALUE = −∞ ∨ VALUE ≥ min(2 + minv.(sv − 2) * minv)
VALUE ≤ max(2 * (maxv − 1)ɔ, (sv − 2) * (maxv − 1)ɔ)
where
maxv =maxval(VARIABLES.var)
sv = |VARIABLES|
minv =minval(VARIABLES.var)
rv =range(VARIABLES.var)

Purpose
VALUE is the maximal surface of occurrences of the PROPER/plain pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value −∞.

An occurrence of the pattern PROPER/plain is the maximal subsequence which matches the regular expression > = < + .

Assume that the occurrence of the pattern PROPER/plain starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 1 to index j.

Example
(10, ⟨2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3⟩)

Figure 3.612 provides an example where the MAX_SURF_PROPER/plain (10, [2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3]) constraint holds.

Typical
|VARIABLES| > 3
range(VARIABLES.var) > 1

Symmetry
Items of VARIABLES can be reversed.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 3.612: Illustrating the **MAX_SURF_PROPER_PLAIN** constraint of the **Example** slot
Automaton

Figures 3.613 and 3.614 respectively depict the automaton associated with the constraint MAX_SURF_PROPER_plain and its simplified form.

Figure 3.613: Automaton for the MAX_SURF_PROPER_plain constraint obtained by applying decoration Table 2.35 to the seed transducer of the PROPER_plain pattern where default is $-\infty$.

Figure 3.614: Automaton for the MAX_SURF_PROPER_plain constraint obtained by applying decoration Table 2.25 to the seed transducer of the PROPER_plain pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 3.37: Glue matrix for the \texttt{MAX\_SURF\_PROPER\_PLAIN} constraint defined as the composition of the \texttt{PROPER\_PLAIN} pattern, the feature \texttt{SURF}, and the aggregator \texttt{max}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\max(\vec{C}, \vec{C})$</td>
<td>$\max(\vec{C}, \vec{C})$</td>
<td>$\max(\vec{C}, \vec{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\max(\vec{C}, \vec{C})$</td>
<td>$\max(\vec{C}, \vec{C})$</td>
<td>$\vec{D} + \vec{D} + \text{VAR}_i$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\max(\vec{C}, \vec{C})$</td>
<td>$\vec{D} + \vec{D} + \text{VAR}_i$</td>
<td>$\vec{D} + \vec{D} + \text{VAR}_i$</td>
</tr>
</tbody>
</table>
3.284 MAX_SURF_PROPER_PLATEAU

**Description**

Based on the PROPER_PLATEAU pattern.

**Constraint**

\[
\text{MAX_SURF_PROPER_PLATEAU}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

\[
\begin{align*}
\text{VALUE} : & \text{ dvar} \\
\text{VARIABLES} : & \text{ collection(var\text{-}dvar)}
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{sv} \leq 3 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} = -\infty \lor \text{VALUE} \geq \min(2 \ast (\text{minv} + 1), (\text{sv} - 2) \ast (\text{minv} + 1)) \\
\text{VALUE} \leq \max(2 \ast \text{maxv}, (\text{sv} - 2) \ast \text{maxv})
\end{align*}
\]

where

\[
\begin{align*}
\text{maxv} &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{sv} &= |\text{VARIABLES}| \\
\text{minv} &= \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{rv} &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

\[
\begin{align*}
\text{VALUE} \leq -2 = 2 - 1 \\
\text{VALUE} \leq 12 = (8 - 2) \ast 2
\end{align*}
\]

**Purpose**

VALUE is the maximal surface of occurrences of the PROPER_PLATEAU pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(-\infty\).

An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression \(<=+>\).

Assume that the occurrence of the pattern PROPER_PLATEAU starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

\((15, \{7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 5\})\)

Figure 3.615 provides an example where the MAX_SURF_PROPER_PLATEAU (15, [7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 5]) constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| & > 3 \\
\text{range}(\text{VARIABLES}.\text{var}) & > 1
\end{align*}
\]

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.615: Illustrating the MAX_SURF_PROPER_PLATEAU constraint of the Example slot
Figures 3.616 and 3.617 respectively depict the automaton associated with the constraint \texttt{MAX\_SURF\_PROPER\_PLATEAU} and its simplified form.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure3.616.png}
\caption{Automaton for the \texttt{MAX\_SURF\_PROPER\_PLATEAU} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{PROPER\_PLATEAU} pattern where \texttt{default} is $-\infty$.}
\end{figure}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure3.617.png}
\caption{Automaton for the \texttt{MAX\_SURF\_PROPER\_PLATEAU} constraint obtained by applying decoration Table 2.25 to the seed transducer of the \texttt{PROPER\_PLATEAU} pattern where \texttt{default} is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.}
\end{figure}
Table 3.38: Glue matrix for the **MAX_SURF_PROPER_PLATEAU** constraint defined as the composition of the **PROPER_PLATEAU** pattern, the feature **SURF**, and the aggregator **max**; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.285 MAX_SURF_STEADY

**Origin**
Based on the STEADY pattern.

**Constraint**
MAX_SURF_STEADY(VALUE, VARIABLES)

**Arguments**
VALUE : dvar
VARIABLES : collection(var−dvar)

**Restrictions**
\[ sv \leq 1 \Rightarrow VALUE = -\infty \]
\[ VALUE = -\infty \lor VALUE \geq 2 \times \text{minv} \]
\[ VALUE \leq 2 \times \text{maxv} \]
\[ \text{required}(\text{VARIABLES}, \text{var}) \]
where
\[ \text{maxv} = \text{maxval}(\text{VARIABLES} . \text{var}) \]
\[ \text{minv} = \text{minval}(\text{VARIABLES} . \text{var}) \]
\[ sv = |\text{VARIABLES}| \]

**Purpose**
An occurrence of the pattern STEADY is the subsequence which matches the regular expression `=`. Assume that the occurrence of the pattern STEADY starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i\) to index \(j + 1\).

**Example**

\[(12, (1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6))\]

Figure 3.618 provides an example where the MAX_SURF_STEADY (12, [1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6]) constraint holds.

**Typical**
\[ |\text{VARIABLES}| > 1 \]

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.618: Illustrating the MAX_SURF_STEADY constraint of the Example slot
Automaton

Figures 3.619 and 3.620 respectively depict the automaton associated with the constraint MAX_SURF_STEADY and its simplified form.

\[
\begin{align*}
\{ C \leftarrow \text{default} \\
D & \leftarrow 0 \\
R & \leftarrow \text{default} 
\}
\end{align*}
\]

Figure 3.619: Automaton for the MAX_SURF_STEADY constraint obtained by applying decoration Table 2.35 to the seed transducer of the STEADY pattern where default is $-\infty$

\[
\begin{align*}
\{ R \leftarrow \text{default} 
\}
\end{align*}
\]

Figure 3.620: Automaton for the MAX_SURF_STEADY constraint obtained by applying decoration Table 2.38 to the seed transducer of the STEADY pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.

Table 3.39: Glue matrix for the MAX_SURF_STEADY constraint defined as the composition of the STEADY pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MAX_SURF_STEADY

1253
### 3.286 MAX_SURF_STEADY_SEQUENCE

**DESCRIPTION**

*Origin*

Based on the `STEADY_SEQUENCE` pattern.

**Constraint**

```
MAX_SURF_STEADY_SEQUENCE(VALUE, VARIABLES)
```

**Arguments**

```
VALUE : dvar
VARIABLES : collection(var−dvar)
```

**Restrictions**

- \( sv \leq 1 \Rightarrow VALUE = -\infty \)
- \( rv = 1 \Rightarrow VALUE \geq sv + minv \)
- \( rv \geq 2 \Rightarrow VALUE = -\infty \lor VALUE \geq \min(2 \cdot minv, sv + minv) \)
- \( rv = 1 \Rightarrow VALUE \leq sv + maxv \)
- \( rv \geq 2 \Rightarrow VALUE \leq \max(2 \cdot maxv, sv + maxv) \)
- \( \text{required}(VARIABLES, var) \)

where

\[
\begin{align*}
maxv &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
rv &= \text{range}(\text{VARIABLES}.\text{var}) \\
sv &= |\text{VARIABLES}| \\
minv &= \text{minval}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

VALUE is the maximal surface of occurrences of the `STEADY_SEQUENCE` pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(-\infty\).

An occurrence of the pattern `STEADY_SEQUENCE` is the maximal subsequence which matches the regular expression \(=+\).

Assume that the occurrence of the pattern `STEADY_SEQUENCE` starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i\) to index \(j+1\).

**Example**

```
(15, [3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1])
```

Figure 3.621 provides an example where the `MAX_SURF_STEADY_SEQUENCE` constraint holds.

**Typical**

\(|\text{VARIABLES}| > 1\)

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.621: Illustrating the MAX_SURF_STEADY_SEQUENCE constraint of the Example slot
Automaton

Figures 3.622 and 3.623 respectively depict the automaton associated with the constraint MAX_SURF_STEADY_SEQUENCE and its simplified form.

\[
\begin{align*}
&\{ C \leftarrow \text{default} \}
\{ D \leftarrow 0 \}
\{ R \leftarrow \text{default} \}
\
&\{ C \leftarrow \text{default} \}
\{ D \leftarrow 0 \}
\{ R \leftarrow \max(R, C) \}
\
&\{ C \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \}
\{ D \leftarrow 0 \}
\
&\{ C \leftarrow C + D + \text{VAR}_{i+1} \}
\{ D \leftarrow 0 \}
\end{align*}
\]

Figure 3.622: Automaton for the MAX_SURF_STEADY_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the STEADY_SEQUENCE pattern where default is \(-\infty\).

\[
\begin{align*}
&\{ C \leftarrow \text{default} \}
\{ R \leftarrow \text{default} \}
\
&\{ C \leftarrow \text{default} \}
\{ R \leftarrow \max(R, C) \}
\
&\{ C \leftarrow \text{VAR}_i + \text{VAR}_{i+1} \}
\{ D \leftarrow 0 \}
\
&\{ C \leftarrow C + \text{VAR}_{i+1} \}
\end{align*}
\]

Figure 3.623: Automaton for the MAX_SURF_STEADY_SEQUENCE constraint obtained by applying decoration Table 2.24 to the seed transducer of the STEADY_SEQUENCE pattern where default is \(-\infty\); \(R_i - R_{i-1} \geq 0\) is a linear invariant.
Table 3.40: Glue matrix for the MAX_SURF_STEADY_SEQUENCE constraint defined as the composition of the STEADY_SEQUENCE pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.287 \textsc{maxsurf_strictly_decreasing_sequence}

\textbf{Description}

Origin
Based on the \textsc{strictly_decreasing_sequence} pattern.

Constraint
\textsc{maxsurf_strictly_decreasing_sequence}(\textsc{value}, \textsc{variables})

Arguments
\begin{itemize}
\item \textsc{value} : dvar
\item \textsc{variables} : collection(var-dvar)
\end{itemize}

Restrictions
\begin{itemize}
\item \( sv \leq 1 \lor rv \leq 1 \Rightarrow \textsc{value} = -\infty \)
\item \( \text{minv} < 0 \Rightarrow \textsc{value} = -\infty \lor \textsc{value} \geq \ell_1 \ast \text{minv} + \lceil \ell_1 \ast (\ell_1 - 1)/2 \rceil \)
\item \( \text{minv} \geq 0 \Rightarrow \textsc{value} = -\infty \lor \textsc{value} \geq 2 \ast \text{minv} + 1 \)
\item \( \text{maxv} > 0 \Rightarrow \textsc{value} \leq \ell_2 \ast \text{maxv} - \lceil \ell_2 \ast (\ell_2 - 1)/2 \rceil \)
\item \( \text{maxv} \leq 0 \Rightarrow \textsc{value} \leq 2 \ast \text{maxv} - 1 \)
\end{itemize}

\textsc{required}(\textsc{variables}.\textsc{var})

where
\begin{itemize}
\item \( \text{maxv} = \text{maxval}(\text{variables}.\text{var}) \)
\item \( \text{rv} = \text{range}(\text{variables}.\text{var}) \)
\item \( sv = |\text{variables}| \)
\item \( \ell_1 = \min(\min(sv, rv), |\text{minv}|) \)
\item \( \ell_2 = \min(\min(sv, rv), |\text{maxv}|) \)
\item \( \text{minv} = \text{minval}(\text{variables}.\text{var}) \)
\end{itemize}

\textsc{value} is the maximal surface of occurrences of the \textsc{strictly_decreasing_sequence} pattern in the time-series given by the \textsc{variables} collection. If the pattern does not occur, \textsc{value} takes the default value \(-\infty\).

An occurrence of the pattern \textsc{strictly_decreasing_sequence} is the maximal subsequence which matches the regular expression \( ^+ \).

Assume that the occurrence of the pattern \textsc{strictly_decreasing_sequence} starts at position \( i \) and ends at position \( j \). The feature \textsc{surf} computes the sum of the values from index \( i \) to index \( j + 1 \).

\textbf{Example}
\begin{itemize}
\item (13, ⟨4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3⟩)
\end{itemize}

Figure 3.624 provides an example where the \textsc{maxsurf_strictly_decreasing_sequence} (13, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3]) constraint holds.

Typical
\begin{itemize}
\item \(|\text{variables}| > 1\)
\item \(\text{range}(\text{variables}.\text{var}) > 1\)
\end{itemize}

\textbf{Arg. properties}
Functional dependency: \textsc{value} determined by \textsc{variables}.
Figure 3.624: Illustrating the MAX_SURF_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 3.625 and 3.626 respectively depict the automaton associated with the constraint MAX_SURF_STRICTLY_DECREASING_SEQUENCE and its simplified form.

Figure 3.625: Automaton for the MAX_SURF_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern where default is $-\infty$.

Figure 3.626: Automaton for the MAX_SURF_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.24 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 3.41: Glue matrix for the MAX_SURF_STRICTLY_DECREASING_SEQUENCE constraint defined as the composition of the STRICTLY_DECREASING_SEQUENCE pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\overrightarrow{C} + \overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + \text{VAR}$</td>
</tr>
</tbody>
</table>
3.288 MAX_SURF STRICTLY_INCREASING_SEQUENCE

DESCRIPTION AUTOMATON

Origin
Based on the STRICTLY_INCREASING_SEQUENCE pattern.

Constraint
MAX_SURF STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions

\[
\begin{align*}
sv &\leq 1 \lor rv \leq 1 \Rightarrow VALUE = -\infty \\
\text{minv} &< 0 \Rightarrow VALUE = -\infty \lor \text{VALUE} \geq \ell_1 * \text{minv} + \lfloor \ell_1 * (\ell_1 - 1)/2 \rfloor \\
\text{minv} &\geq 0 \Rightarrow VALUE = -\infty \lor \text{VALUE} \geq 2 * \text{minv} + 1 \\
\text{maxv} &> 0 \Rightarrow VALUE \leq \ell_2 * \text{maxv} - \lfloor \ell_2 * (\ell_2 - 1)/2 \rfloor + 3 \\
\text{maxv} &\leq 0 \Rightarrow VALUE \leq 2 * \text{maxv} - 1 \lor \text{VALUE} \geq \ell_1 * \text{minv} + \lfloor \ell_1 * (\ell_1 - 1)/2 \rfloor \\
\text{maxv} &\leq 0 \Rightarrow VALUE \leq 2 * \text{maxv} - 1 \lor \text{VALUE} \geq \ell_1 * \text{minv} + \lfloor \ell_1 * (\ell_1 - 1)/2 \rfloor \\
\end{align*}
\]

where

\[
\begin{align*}
\text{maxv} &= \text{maxval(VARIABLES.var)} \\
\text{rv} &= \text{range(VARIABLES.var)} \\
sv &= |\text{VARIABLES}| \\
\ell_1 &= \text{min}(\text{min}(sv, rv), |\text{minv}|) \\
\ell_2 &= \text{min}(\text{min}(sv, rv), |\text{maxv}|) \\
\text{minv} &= \text{minval(VARIABLES.var)}
\end{align*}
\]

VALUE is the maximal surface of occurrences of the STRICTLY_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value $-\infty$.

An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal sub-sequence which matches the regular expression $<^*$. Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i$ to index $j + 1$.

Purpose

Example

\[ (16, \langle 4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3 \rangle) \]

Figure 3.627 provides an example where the MAX_SURF STRICTLY_INCREASING_SEQUENCE (16, [4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3]) constraint holds.

Typical

\[
|\text{VARIABLES}| > 1 \\
\text{range(VARIABLES.var)} > 1
\]

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 3.627: Illustrating the MAX_SURF_STRICTLY_INCREASING_SEQUENCE constraint of the Example slot
Figures 3.628 and 3.629 respectively depict the automaton associated with the constraint `MAX_SURF_STRICTLY_INCREASING_SEQUENCE` and its simplified form.

Figure 3.628: Automaton for the `MAX_SURF_STRICTLY_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `STRICTLY_INCREASING_SEQUENCE` pattern where `default` is $-\infty$.

Figure 3.629: Automaton for the `MAX_SURF_STRICTLY_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 2.24 to the seed transducer of the `STRICTLY_INCREASING_SEQUENCE` pattern where `default` is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 3.42: Glue matrix for the MAX_SURF_STRICTLY_INCREASING_SEQUENCE constraint defined as the composition of the STRICTLY_INCREASING_SEQUENCE pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.289 MAX_SURF_SUMMIT

**Origin**
Based on the SUMMIT pattern.

**Constraint**
MAX_SURF_SUMMIT(VALUE, VARIABLES)

**Arguments**
VALUE : dvar
VARIABLES : collection(var−dvar)

**Restrictions**
\[ sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = -\infty \]
\[ rv = 2 \Rightarrow VALUE = -\infty \lor VALUE \geq \text{minv} + 1 \]
\[ rv \geq 3 \Rightarrow VALUE = -\infty \lor VALUE \geq \text{min}(\text{minv} + 1, (sv - 2) \times (\text{minv} + 1) + 1) \]
\[ rv = 2 \Rightarrow VALUE \leq \text{maxv} \]
\[ rv \geq 3 \Rightarrow VALUE \leq \text{max}(\text{maxv}, (sv - 2) \times (\text{maxv} - 1) + 1) \]

where
\[ \text{maxv} = \maxval(VARIABLES.var) \]
\[ rv = \text{range}(VARIABLES.var) \]
\[ sv = |VARIABLES| \]
\[ \text{minv} = \minval(VARIABLES.var) \]

**Purpose**
VALUE is the maximal surface of occurrences of the SUMMIT pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(-\infty\).

An occurrence of the pattern SUMMIT is the *maximal* subsequence which matches the regular expression \(<|<|=|<|*<|>|>=|>|*>|\). Assume that the occurrence of the pattern SUMMIT starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**
\[(13, (7, 1, 5, 4, 4, 3, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1))\]

Figure 3.630 provides an example where the MAX_SURF_SUMMIT (13, [7, 1, 5, 4, 4, 3, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1]) constraint holds.

**Typical**
\[ |\text{VARIABLES}| > 2 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.630: Illustrating the MAX_SURF_SUMMIT constraint of the Example slot
Automaton

Figure 3.631 depicts the automaton associated with the constraint MAX_SURF_SUMMIT.

Figure 3.631: Automaton for the MAX_SURF_SUMMIT constraint obtained by applying decoration Table 2.35 to the seed transducer of the SUMMIT pattern where default is $-\infty$ (transition $u \to r$ has the same accumulator update as transition $r \to u$); $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 3.43: Glue matrix for the MAX_SURF_SUMMIT constraint defined as the composition of the SUMMIT pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.290  MAX_SURF_VALLEY

DESCRIPTION

Origin
Based on the VALLEY pattern.

Constraint
MAX_SURF_VALLEY(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = −∞
VALUE = −∞ ∨ VALUE ≥ min(minv, (sv − 2) * minv)
VALUE ≤ max(maxv − 10, (sv − 2) * (maxv − 1))
required(VARIABLES, var)

where
maxv = maxval(VARIABLES.var)
sv = |VARIABLES|
minv = minval(VARIABLES.var)
rv = range(VARIABLES.var)

VALUE is the maximal surface of occurrences of the VALLEY pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value −∞.

Purpose
An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression > (= | >)*(< | | =)* <.
Assume that the occurrence of the pattern VALLEY starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 1 to index j.

Example
(15, (1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 5, 7))

Figure 3.632 provides an example where the MAX_SURF_VALLEY (15, [1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 7]) constraint holds.

Typical
|VARIABLES| > 2
range(VARIABLES.var) > 1

Symmetry
Items of VARIABLES can be reversed.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 3.632: Illustrating the MAX_SURF_VALLEY constraint of the Example slot
**MAX_SURF_VALLEY**

**Automaton**

Figure 3.633 depicts the automaton associated with the constraint MAX_SURF_VALLEY.

![Automaton Diagram]

Figure 3.633: Automaton for the MAX_SURF_VALLEY constraint obtained by applying decoration Table 2.35 to the seed transducer of the VALLEY pattern where default is $-\infty$; $R_i - R_{i-1} \geq 0$ is a linear invariant.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>max($\overline{C}$, $\overline{C}$)</td>
<td>max($\overline{C}$, $\overline{C}$)</td>
<td>max($\overline{C}$, $\overline{C}$)</td>
</tr>
<tr>
<td>r</td>
<td>max($\overline{C}$, $\overline{C}$)</td>
<td>$\overline{D} + \overline{D} + \text{VAR}_i$</td>
<td>$\overline{C} + \overline{D} + \overline{D} + \text{VAR}_i$</td>
</tr>
<tr>
<td>t</td>
<td>max($\overline{C}$, $\overline{C}$)</td>
<td>$\overline{C} + \overline{D} + \overline{D} + \text{VAR}_i$</td>
<td>max($\overline{C}$, $\overline{C}$)</td>
</tr>
</tbody>
</table>

Table 3.44: Glue matrix for the MAX_SURF_VALLEY constraint defined as the composition of the VALLEY pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.291 MAX_SURF_ZIGZAG

**Description**

**Origin**
Based on the ZIGZAG pattern.

**Constraint**
MAX_SURF_ZIGZAG(VALUE, VARIABLES)

**Arguments**
- VALUE : `dvar`
- VARIABLES : `collection(var−dvar)`

**Restrictions**

\[
\text{sv} \leq 3 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} \geq \min \left( \frac{2}{(\text{sv} - 1)/2 \times \text{minv} + \left| (\text{sv} - 2)/2 \right| \times (\text{minv} + 1)} \right) \\
\text{VALUE} \leq \max \left( \frac{2}{(\text{sv} - 1)/2 \times \text{maxv} + \left| (\text{sv} - 2)/2 \right| \times (\text{maxv} - 1)} \right)
\]

where

- \( \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \)
- \( \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \)
- \( \text{sv} = |\text{VARIABLES}| \)
- \( \text{rv} = \text{range}(\text{VARIABLES}.\text{var}) \)

**Purpose**

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression \((<>)^+((< | < >) | (> <>)^+ (> | > <))\). Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

\((21, (4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1))\)

Figure 3.634 provides an example where the MAX_SURF_ZIGZAG \((21, [4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1])\) constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 3\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**
- Functional dependency: VALUE determined by VARIABLES.
Figure 3.634: Illustrating the MAX_SURF_ZIGZAG constraint of the Example slot
Automaton

Figures 3.635 and 3.636 respectively depict the automaton associated with the constraint MAX\_SURF\_ZIGZAG and its simplified form.
Figure 3.635: Automaton for the MAX_SURF_ZIGZAG constraint obtained by applying decoration Table 2.35 to the seed transducer of the ZIGZAG pattern where default is $-\infty$: (1) missing transitions from $a$, $b$, $c$, $d$, $e$, $f$ to $s$ are labelled by $=$; (2) on transitions from $b$, $c$, $e$, $f$ to $s$ the accumulator $D$ is reset to its initial value; (3) on transitions from $c$, $f$ to $s$ the accumulator $R$ is updated wrt $C$ and the accumulator $C$ is reset to its initial value.
Figure 3.636: Automaton for the MAX_SURF_ZIGZAG constraint obtained by applying decoration Table 2.23 to the seed transducer of the ZIGZAG pattern where default is $-\infty$; missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=$; (2) on transitions from $b, c, e, f$ to $s$ the accumulator $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the accumulator $R$ is updated wrt $C$ and the accumulator $C$ is reset to its initial value.; $R_i - R_{i-1} \geq 0$ is a linear invariant.
Table 3.45: Glue matrix for the MAX-SURF-ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature SURF, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
<tr>
<td>a</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i)</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i)</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
<tr>
<td>b</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i)</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i)</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
<tr>
<td>c</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i)</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i)</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
<tr>
<td>d</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i)</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i)</td>
</tr>
<tr>
<td>e</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i)</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i)</td>
</tr>
<tr>
<td>f</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i)</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i)</td>
</tr>
</tbody>
</table>
3.292 **MAX_WIDTH_DECREASING_SEQUENCE**

**DESCRIPTION**

**Origin**

Based on the DECREASING_SEQUENCE pattern.

**Constraint**

\[
\text{MAX_WIDTH_DECREASING_SEQUENCE}(\text{VALUE, VARIABLES})
\]

**Arguments**

- VALUE : dvar
- VARIABLES : collection(var−dvar)

**Restrictions**

\[
sv \leq 1 \lor rv \leq 1 \Rightarrow \text{VALUE} = 0
\]

\[
\text{VALUE} = 0 \lor \text{VALUE} \geq 2
\]

\[
rv = 2 \Rightarrow \text{VALUE} \leq 2
\]

\[
rv \geq 3 \Rightarrow \text{VALUE} \leq sv
\]

\[
\text{required(VARIABLES, var)}
\]

where

\[
sv = |\text{VARIABLES}|
\]

\[
rv = \text{range(VARIABLES.var)}
\]

**Purpose**

An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression \( (> | =)^* > | > \). Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i + 2 \).

**Example**

\[
(5, (3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))
\]

Figure 3.637 provides an example where the MAX_WIDTH_DECREASING_SEQUENCE \((5, [3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4])\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1
\]

\[
r\text{ange(VARIABLES.var)} > 1
\]

**Symmetry**

One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.637: Illustrating the MAX_WIDTH_DECREASING_SEQUENCE constraint of the Example slot
Figures 3.638 and 3.639 respectively depict the automaton associated with the constraint \textsc{max\_width\_decreasing\_sequence} and its simplified form.

\[
\begin{align*}
&\begin{cases}
C \leftarrow \text{default} \\
D \leftarrow 0 \\
R \leftarrow \text{default}
\end{cases} \\
&\leq s \\
&\leq \\
&\begin{cases}
C \leftarrow \text{default} \\
D \leftarrow 0 \\
R \leftarrow \max(R, C)
\end{cases} \\
&> t \\
&\geq t
\end{align*}
\]

Figure 3.638: Automaton for the \textsc{max\_width\_decreasing\_sequence} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \textsc{decreasing\_sequence} pattern where \texttt{default} is 0

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$t$</td>
</tr>
<tr>
<td>(\max(\overline{C}, \overline{C}))</td>
<td>(\max(\overline{C}, \overline{C}))</td>
</tr>
<tr>
<td>(\max(\overline{C}, \overline{C}))</td>
<td>(\overline{C} + \overline{C} + \overline{D} + \overline{D} + 1)</td>
</tr>
</tbody>
</table>

Table 3.46: Glue matrix for the \textsc{max\_width\_decreasing\_sequence} constraint defined as the composition of the \textsc{decreasing\_sequence} pattern, the feature \textsc{width}, and the aggregator \texttt{max}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 3.639: Automaton for the MAX_WIDTH_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.30 to the seed transducer of the DECREASING_SEQUENCE pattern where default is 0; \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-1} + D_i \geq 0 \) are linear invariants.
3.293 MAX_WIDTH_DECREASING_TERRACE

**DESCRIPTION**

**Origin**
Based on the DECREASING_TERRACE pattern.

**Constraint**
MAX_WIDTH_DECREASING_TERRACE(VALUE, VARIABLES)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var−dvar)

**Restrictions**
- $sv \leq 3 \lor rv \leq 2 \Rightarrow VALUE = 0$
- $VALUE = 0 \lor VALUE \geq 2$
- $VALUE \leq \max(0, sv − 2x)$
- required(VARIABLES, var)

where
- $sv = |VARIABLES|$
- $rv = \text{range}(VARIABLES, \text{var})$

**Purpose**
An occurrence of the pattern DECREASING_TERRACE is the maximal subsequence which matches the regular expression $>\equiv ^{+} >$.
Assume that the occurrence of the pattern DECREASING_TERRACE starts at position $i$ and ends at position $j$. The feature WIDTH computes the value $j - i$.

**Example**

```
(2, (6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3))
```

Figure 3.640 provides an example where the MAX_WIDTH_DECREASING_TERRACE (2, [6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3]) constraint holds.

**Typical**
- $|\text{VARIABLES}| > 3$
- $\text{range}(\text{VARIABLES, var}) > 2$

**Symmetry**
One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.640: Illustrating the MAX_WIDTH_DECREASING_TERRACE constraint of the Example slot
Automaton

Figures 3.641 and 3.642 respectively depict the automaton associated with the constraint `MAX_WIDTH_DECREASING_TERRACE` and its simplified form.

\[
\begin{align*}
    &\{ C \leftarrow \text{default} \\
    &D \leftarrow 0 \\
    &R \leftarrow \text{default} \} \\
    \leq s \leq R &< \\
    \langle D \leftarrow 0 \rangle &> \\
    \{ D \leftarrow D + 1 \} &= \\
    \{ D \leftarrow D + 1 \} &= \\
    \{ D \leftarrow 0 \} &> > \\
    \{ D \leftarrow D + 1 \} &> > \\
    \{ D \leftarrow D + 1 \} &> > \\
\end{align*}
\]

Figure 3.641: Automaton for the `MAX_WIDTH_DECREASING_TERRACE` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `DECREASING_TERRACE` pattern where `default` is 0.

<table>
<thead>
<tr>
<th></th>
<th>(s)</th>
<th>(r)</th>
<th>(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>(\max(\overline{C}, \overline{C}))</td>
<td>(\max(\overline{C}, \overline{C}))</td>
<td>(\max(\overline{C}, \overline{C}))</td>
</tr>
<tr>
<td>(r)</td>
<td>(\max(\overline{C}, \overline{C}))</td>
<td>(\max(\overline{C}, \overline{C}))</td>
<td>(\overline{D} + \overline{D} + 1)</td>
</tr>
<tr>
<td>(t)</td>
<td>(\max(\overline{C}, \overline{C}))</td>
<td>(\overline{D} + \overline{D} + 1)</td>
<td>(\overline{D} + \overline{D} + 1)</td>
</tr>
</tbody>
</table>

Table 3.47: Glue matrix for the `MAX_WIDTH_DECREASING_TERRACE` constraint defined as the composition of the `DECREASING_TERRACE` pattern, the feature `WIDTH`, and the aggregator `max`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 3.642: Automaton for the MAX_WIDTH_DECREASING_TERRACE constraint obtained by applying decoration Table 2.25 to the seed transducer of the DECREASING_TERRACE pattern where default is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + D_i \geq 0$ are linear invariants.
3.294  MAX_WIDTH_GORGE

**Origin**
Based on the GORGE pattern.

**Constraint**
MAX_WIDTH_GORGE(VALUE, VARIABLES)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var−dvar)

**Restrictions**
- \( sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = 0 \)
- \( VALUE = 0 \lor VALUE \geq 1 \)
- \( rv = 2 \Rightarrow VALUE \leq 1 \)
- \( rv \geq 3 \Rightarrow VALUE \leq \max(0, sv - 2) \)
- \( required(VARIABLES, var) \)
  where
- \( sv = |VARIABLES| \)
- \( rv = \text{range}(VARIABLES.var) \)

VALUE is the maximal width of occurrences of the GORGE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

**Purpose**
An occurrence of the pattern GORGE is the **maximal** subsequence which matches the regular expression \((> | > (|=) > >/ < (=) < | < (=) <) <)\).
Assume that the occurrence of the pattern GORGE starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

**Example**
\((3, (1, 7, 3, 4, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7))\)

Figure 3.643 provides an example where the MAX_WIDTH_GORGE (3, [1, 7, 3, 4, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7]) constraint holds.

**Typical**
- \(|VARIABLES| > 2\)
- \(\text{range}(VARIABLES.var) > 1\)

**Symmetries**
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.643: Illustrating the MAX WIDTH GORGE constraint of the Example slot
Automaton

Figures 3.644 and 3.645 respectively depict the automaton associated with the constraint \textsc{MAX\_WIDTH\_GORGE} and its simplified form.

Figure 3.644: Automaton for the \textsc{MAX\_WIDTH\_GORGE} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \textsc{GORGE} pattern where \texttt{default} is 0 (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$)
Figure 3.645: Automaton for the MAX_WIDTHGORGE constraint obtained by applying decoration Table 2.26 to the seed transducer of the GORGE pattern where \texttt{default} is 0 (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$); $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + D_i \geq 0$ are linear invariants.
Table 3.48: Glue matrix for the `MAX_WIDTH_GORGE` constraint defined as the composition of the `GORGE` pattern, the feature `WIDTH`, and the aggregator `max`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
<th>u</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>max((\overrightarrow{C}, \overleftarrow{C}))</td>
<td>max((\overrightarrow{C}, \overleftarrow{C}))</td>
<td>max((\overrightarrow{C}, \overleftarrow{C}))</td>
<td>max((\overrightarrow{C}, \overleftarrow{C}))</td>
</tr>
<tr>
<td>r</td>
<td>max((\overrightarrow{C}, \overleftarrow{C}))</td>
<td>(\overrightarrow{D} + \overleftarrow{D} + 1)</td>
<td>(\overrightarrow{C} + \overrightarrow{D} + \overleftarrow{D} + 1)</td>
<td>max((\overrightarrow{C}, \overleftarrow{C}))</td>
</tr>
<tr>
<td>t</td>
<td>max((\overrightarrow{C}, \overleftarrow{C}))</td>
<td>(\overrightarrow{C} + \overrightarrow{D} + \overleftarrow{D} + 1)</td>
<td>max((\overrightarrow{C}, \overleftarrow{C}))</td>
<td>(\overrightarrow{C} + \overrightarrow{D} + \overleftarrow{D} + 1)</td>
</tr>
<tr>
<td>u</td>
<td>max((\overrightarrow{C}, \overleftarrow{C}))</td>
<td>max((\overrightarrow{C}, \overleftarrow{C}))</td>
<td>(\overrightarrow{C} + \overrightarrow{D} + \overleftarrow{D} + 1)</td>
<td>max((\overrightarrow{C}, \overleftarrow{C}))</td>
</tr>
</tbody>
</table>
3.295 MAX_WIDTH_INCREASING_SEQUENCE

**Origin**
Based on the **INCREASING_SEQUENCE** pattern.

**Constraint**
MAX_WIDTH_INCREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**
- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**
- \( sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \)
- \( VALUE = 0 \lor VALUE \geq 2 \)
- \( rv = 2 \Rightarrow VALUE \leq 2 \)
- \( rv \geq 3 \Rightarrow VALUE \leq sv \)

**Purpose**
VALUE is the maximal width of occurrences of the **INCREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern **INCREASING_SEQUENCE** is the maximal subsequence which matches the regular expression \(< (< | =)^* < | <\).

Assume that the occurrence of the pattern **INCREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **WIDTH** computes the value \(j - i + 2\).

**Example**
\((5, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))\)

Figure 3.646 provides an example where the MAX_WIDTH_INCREASING_SEQUENCE \((5, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3])\) constraint holds.

**Typical**
- \(|VARIABLES| > 1\)
- \(\text{range}(VARIABLES.var) > 1\)

**Symmetry**
One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.646: Illustrating the MAX_WIDTH_INCREASING_SEQUENCE constraint of the Example slot
Figures 3.647 and 3.648 respectively depict the automaton associated with the constraint \texttt{MAX_WIDTH_INCREASING_SEQUENCE} and its simplified form.

\[
\begin{align*}
\{ & C \leftarrow \text{default} \\
& D \leftarrow 0 \\
& R \leftarrow \text{default} \}\quad & \begin{array}{c}
\geq s \\
\geq \\
> \\
\leq t
\end{array}
\begin{array}{c}
\{ & C \leftarrow \text{default} \\
& D \leftarrow 0 \\
& R \leftarrow \max(R,C) \}\quad & \begin{array}{c}
C \leftarrow D + 2 \\
D \leftarrow 0
\end{array}
\end{array}
\{ & D \leftarrow D + 1 \}
\begin{array}{c}
\leq C + D + 1 \\
D \leftarrow 0
\end{array}
\end{align*}
\]

Figure 3.647: Automaton for the \texttt{MAX_WIDTH_INCREASING_SEQUENCE} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{INCREASING_SEQUENCE} pattern where default is 0

\[
\begin{array}{c|c|c}
& s & t \\
\hline
s & \max(\overrightarrow{C}, \overrightarrow{C}) & \max(\overrightarrow{C}, \overrightarrow{C}) \\
\hline
t & \max(\overrightarrow{C}, \overrightarrow{C}) & \overrightarrow{C} + \overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + 1
\end{array}
\]

Table 3.49: Glue matrix for the \texttt{MAX_WIDTH_INCREASING_SEQUENCE} constraint defined as the composition of the \texttt{INCREASING_SEQUENCE} pattern, the feature \texttt{WIDTH}, and the aggregator \texttt{max}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 3.648: Automaton for the MAX_WIDTH_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.30 to the seed transducer of the INCREASING_SEQUENCE pattern where default is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + D_i \geq 0$ are linear invariants.
3.296 MAX_WIDTH_INCREASING_TERRACE

DESCRIPTION

Origin

Based on the INCREASING_TERRACE pattern.

Constraint

MAX_WIDTH_INCREASING_TERRACE(VALUE, VARIABLES)

Arguments

VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions

\( sv \leq 3 \lor rv \leq 2 \Rightarrow VALUE = 0 \)
\( VALUE = 0 \lor VALUE \geq 2 \)
\( VALUE \leq \max(0, sv - 2k) \)
\( required(VARIABLES, var) \)

where

\( sv = |VARIABLES| \)
\( rv = range(VARIABLES.var) \)

VALUE is the maximal width of occurrences of the INCREASING_TERRACE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

Purpose

An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression \(<=+<\).

Assume that the occurrence of the pattern INCREASING_TERRACE starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

Example

\( (3, (1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4)) \)

Figure 3.649 provides an example where the MAX_WIDTH_INCREASING_TERRACE (3, [1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4]) constraint holds.

Typical

\( |VARIABLES| > 3 \)
\( range(VARIABLES.var) > 2 \)

Symmetry

One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 3.649: Illustrating the MAX_WIDTH_INCREASING_TERRACE constraint of the Example slot
Automaton

Figures 3.650 and 3.651 respectively depict the automaton associated with the constraint MAX_WIDTH_INCREASING_TERRACE and its simplified form.

\[
\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow \text{default}
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow D + 1 \} &\rightarrow \geq s \\
\{ D \leftarrow D + 1 \} &\rightarrow \gtrless t
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow 0 \} &\rightarrow < r \\
\{ D \leftarrow \max(R, D + 1) \} &\rightarrow < r
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow 0 \} &\rightarrow < r \\
\{ D \leftarrow \max(R, D + 1) \} &\rightarrow < r
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow D + 1 \} &\rightarrow \geq s \\
\{ D \leftarrow D + 1 \} &\rightarrow \gtrless t
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow 0 \} &\rightarrow < r \\
\{ D \leftarrow \max(R, D + 1) \} &\rightarrow < r
\end{align*}
\]

Figure 3.650: Automaton for the MAX_WIDTH_INCREASING_TERRACE constraint obtained by applying decoration Table 2.35 to the seed transducer of the INCREASING_TERRACE pattern where default is 0.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
<tr>
<td>r</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
</tr>
<tr>
<td>t</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
</tr>
</tbody>
</table>

Table 3.50: Glue matrix for the MAX_WIDTH_INCREASING_TERRACE constraint defined as the composition of the INCREASING_TERRACE pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 3.651: Automaton for the MAX_WIDTH_INCREASING_TERRACE constraint obtained by applying decoration Table 2.25 to the seed transducer of the INCREASING_TERRACE pattern where default is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + D_i \geq 0$ are linear invariants.
### 3.297 MAX_WIDTH_INFLEXION

**Description**

Based on the INFLEXION pattern.

**Constraint**

\[
\text{MAX_WIDTH_INFLEXION} (\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- \text{VALUE} : \text{dvar}
- \text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & \geq 0 \\
\text{VALUE} & \leq \max(0, \text{sv} - 2k) \\
\text{required} & (\text{VARIABLES, var}) \\
\text{where} & \\
\text{sv} &= |\text{VARIABLES}| \\
\text{rv} &= \text{range} (\text{VARIABLES, var})
\end{align*}
\]

**Purpose**

An occurrence of the pattern \text{INFLEXION} is the maximal subsequence which matches the regular expression \(< (< | =)^* | > | (> | =)^* <\). Assume that the occurrence of the pattern \text{INFLEXION} starts at position \(i\) and ends at position \(j\). The feature \text{WIDTH} computes the value \(j - i\).

**Example**

\[(3, (1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4))\]

Figure 3.652 provides an example where the \text{MAX_WIDTH_INFLEXION} \((3, [1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4])\) constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| & > 2 \\
\text{range} (\text{VARIABLES, var}) & > 1
\end{align*}
\]

**Symmetries**

- Items of \text{VARIABLES} can be reversed.
- One and the same constant can be added to the \text{var} attribute of all items of \text{VARIABLES}.

**Arg. properties**

Functional dependency: \text{VALUE} determined by \text{VARIABLES}.
Figure 3.652: Illustrating the MAX_WIDTH_INFLEXION constraint of the **Example** slot
Automaton

Figures 3.653 and 3.654 respectively depict the automaton associated with the constraint MAX_WIDTH_INFLEXION and its simplified form.

Figure 3.653: Automaton for the MAX_WIDTH_INFLEXION constraint obtained by applying decoration Table 2.35 to the seed transducer of the INFLEXION pattern where default is 0 (transition \( r \to t \) has the same accumulators updates as transition \( t \to r \)).

Figure 3.654: Automaton for the MAX_WIDTH_INFLEXION constraint obtained by applying decoration Table 2.25 to the seed transducer of the INFLEXION pattern where default is 0 (transition \( r \to t \) has the same accumulators updates as transition \( t \to r \)); \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-1} + D_i \geq 0 \) are linear invariants.
3.298  MAX_WIDTH_PEAK

**Origin**
Based on the PEAK pattern.

**Constraint**
MAX_WIDTH_PEAK(VALUE, VARIABLES)

**Arguments**
- **VALUE**: dvar
- **VARIABLES**: collection(var–dvar)

**Restrictions**
\[
sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = 0 \\
VALUE \geq 0 \\
VALUE \leq \max(0, sv - 2v) \\
\text{required}(\text{VARIABLES}, \text{var})
\]

where
\[
sv = |\text{VARIABLES}| \\
rv = \text{range}(\text{VARIABLES}.\text{var})
\]

**Purpose**
An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression \(< (= | <)^* (> | =)^* >\). Assume that the occurrence of the pattern PEAK starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

**Example**
\((3, [7, 5, 5, 1, 4, 5, 2, 3, 5, 6, 2, 3, 3, 3, 1])\)

Figure 3.655 provides an example where the MAX_WIDTH_PEAK (3, [7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 1]) constraint holds.

**Typical**
- \(|\text{VARIABLES}| > 2\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Symmetries**
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
- Functional dependency: VALUE determined by VARIABLES.
Figure 3.655: Illustrating the MAX_WIDTH_PEAK constraint of the Example slot
Automaton

Figures 3.656 and 3.657 respectively depict the automaton associated with the constraint MAX_WIDTH_PEAK and its simplified form.

Figure 3.656: Automaton for the MAX_WIDTH_PEAK constraint obtained by applying decoration Table 2.35 to the seed transducer of the PEAK pattern where default is 0.

Figure 3.657: Automaton for the MAX_WIDTH_PEAK constraint obtained by applying decoration Table 2.26 to the seed transducer of the PEAK pattern where default is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + D_i \geq 0$ are linear invariants.
Table 3.51: Glue matrix for the MAX WIDTH PEAK constraint defined as the composition of the PEAK pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\max(\vec{C}, \vec{C})$</td>
<td>$\max(\vec{C}, \vec{C})$</td>
<td>$\max(\vec{C}, \vec{C})$</td>
</tr>
<tr>
<td>r</td>
<td>$\max(\vec{C}, \vec{C})$</td>
<td>$\vec{D} + \vec{D} + 1$</td>
<td>$\vec{C} + \vec{D} + \vec{D} + 1$</td>
</tr>
<tr>
<td>t</td>
<td>$\max(\vec{C}, \vec{C})$</td>
<td>$\vec{C} + \vec{D} + \vec{D} + 1$</td>
<td>$\max(\vec{C}, \vec{C})$</td>
</tr>
</tbody>
</table>
3.299 \textbf{MAX\_WIDTH\_PLAIN}

\begin{tabular}{ll}
\multicolumn{2}{l}{\textbf{DESCRIPTION}} \\
\multicolumn{2}{l}{\textbf{AUTOMATON}} \\
\hline
\end{tabular}

\textbf{Origin} & Based on the \texttt{PLAIN} pattern. \\
\hline
\textbf{Constraint} & \texttt{MAX\_WIDTH\_PLAIN(VALUE, VARIABLES)} \\
\hline
\textbf{Arguments} & \\
VALUE & : \texttt{dvar} \smallskip \\
VARIABLES & : \texttt{collection(var-dvar)} \smallskip \\
\hline
\textbf{Restrictions} & \\
\texttt{sv} \leq 2 \lor \texttt{rv} \leq 1 \Rightarrow \texttt{VALUE} = 0 \\
\texttt{VALUE} & \geq 0 \\
\texttt{VALUE} & \leq \max(0, \texttt{sv} - 2k) \\
\texttt{required(VARIABLES, var)} & \smallskip \\
where & \\
\texttt{sv} & = |\texttt{VARIABLES}| \\
\texttt{rv} & = \texttt{range(VARIABLES.var)} \\
\hline
\textbf{Purpose} & \texttt{VALUE} is the maximal width of occurrences of the \texttt{PLAIN} pattern in the time-series given by the \texttt{VARIABLES} collection. If the pattern does not occur, \texttt{VALUE} takes the default value 0. \\
An occurrence of the pattern \texttt{PLAIN} is the \textit{maximal} subsequence which matches the regular expression \texttt{\^{>}\ast\textless}. \\
Assume that the occurrence of the pattern \texttt{PLAIN} starts at position \texttt{i} and ends at position \texttt{j}. The feature \texttt{WIDTH} computes the value \texttt{j} − \texttt{i}. \\
\hline
\textbf{Example} & (2, (2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3)) \\
\hline
\textbf{Typical} & \smallskip \\
\texttt{|VARIABLES|} & > 2 \\
\texttt{range(VARIABLES.var)} & > 1 \\
\hline
\textbf{Symmetries} & \\
\bullet & Items of \texttt{VARIABLES} can be reversed. \\
\bullet & One and the same constant can be added to the \texttt{var} attribute of all items of \texttt{VARIABLES}. \\
\hline
\textbf{Arg. properties} & Functional dependency: \texttt{VALUE} determined by \texttt{VARIABLES}. \\
\hline
\end{tabular}

Example Figure 3.658 provides an example where the \texttt{MAX\_WIDTH\_PLAIN} (2, [2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3]) constraint holds.
Figure 3.658: Illustrating the MAX_WIDTHPLAIN constraint of the Example slot
Automaton

Figures 3.659 and 3.660 respectively depict the automaton associated with the constraint MAX_WIDTH_PLAIN and its simplified form.

\[
\begin{align*}
\text{Figure 3.659: Automaton for the MAX_WIDTH_PLAIN constraint obtained by applying} \\
\text{decoration Table 2.35 to the seed transducer of the PLAIN pattern where default is 0.}
\end{align*}
\]

\[
\begin{align*}
\text{Figure 3.660: Automaton for the MAX_WIDTH_PLAIN constraint obtained by applying} \\
\text{decoration Table 2.25 to the seed transducer of the PLAIN pattern where default is 0;} \\
R_i - R_{i-1} \geq 0 \text{ and } -R_i + R_{i-1} + D_i \geq 0 \text{ are linear invariants.}
\end{align*}
\]
Table 3.52: Glue matrix for the MAX_WIDTHPLAIN constraint defined as the composition of the PLAIN pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.300  MAX_WIDTH_PLATEAU

**Description**

Origin

Based on the PLATEAU pattern.

Constraint

\[
\text{MAX_WIDTH_PLATEAU(value, variables)}
\]

Arguments

\[
\begin{align*}
\text{value} &: \text{dvar} \\
\text{variables} &: \text{collection(var − dvar)}
\end{align*}
\]

Restrictions

\[
\begin{align*}
sv &\leq 2 \land rv \leq 1 \Rightarrow \text{value} = 0 \\
\text{value} &\geq 0 \\
\text{value} &\leq \max(0, sv - 2v) \\
\text{required}(\text{variables}, \text{var})
\end{align*}
\]

where

\[
\begin{align*}
sv &= |\text{variables}| \\
rv &= \text{range}(\text{variables}.\text{var})
\end{align*}
\]

**Purpose**

\[
\text{value} \text{ is the maximal width of occurrences of the PLATEAU pattern in the time-series given by the variables collection. If the pattern does not occur, value takes the default value 0.}
\]

An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression \(<=\ast\>).

Assume that the occurrence of the pattern PLATEAU starts at position \(i\) and ends at position \(j\). The feature width computes the value \(j - i\).

Example

\[
(4, (1, 3, 3, 5, 5, 5, 2, 4, 4, 4, 3, 3, 1, 5, 5))
\]

Figure 3.661 provides an example where the MAX_WIDTH_PLATEAU (4, [1, 3, 3, 5, 5, 5, 2, 4, 4, 4, 3, 3, 1, 5, 5]) constraint holds.

**Typical**

\[
\begin{align*}
|\text{variables}| &> 2 \\
\text{range}(\text{variables}.\text{var}) &> 1
\end{align*}
\]

**Symmetries**

- Items of variables can be reversed.
- One and the same constant can be added to the var attribute of all items of variables.

**Arg. properties**

**Functional dependency:** \text{value} determined by \text{variables}.
Figure 3.661: Illustrating the MAX_WIDTH_PLATEAU constraint of the Example slot
Automaton

Figures 3.662 and 3.663 respectively depict the automaton associated with the constraint MAX_WIDTH_PLATEAU and its simplified form.

Figure 3.662: Automaton for the MAX_WIDTH_PLATEAU constraint obtained by applying decoration Table 2.35 to the seed transducer of the PLATEAU pattern where default is 0.

Figure 3.663: Automaton for the MAX_WIDTH_PLATEAU constraint obtained by applying decoration Table 2.25 to the seed transducer of the PLATEAU pattern where default is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + D_i \geq 0$ are linear invariants.
Table 3.53: Glue matrix for the MAX_WIDTH_PLATEAU constraint defined as the composition of the PLATEAU pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
<tr>
<td>$r$</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
</tr>
<tr>
<td>$t$</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + 1)</td>
</tr>
</tbody>
</table>
### 3.301 MAX_WIDTH_PROPERPLAIN

#### Description

**Origin**
Based on the PROPERPLAIN pattern.

**Constraint**
MAX_WIDTH_PROPERPLAIN(VALUE, VARIABLES)

**Arguments**
- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**

- \( sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = 0 \)
- \( VALUE = 0 \lor VALUE \geq 2 \)
- \( VALUE \leq \max(0, sv - 2x) \)
- \( \text{required}(\text{VARIABLES}, \text{var}) \)

where
- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

**Purpose**

An occurrence of the pattern PROPERPLAIN is the maximal subsequence which matches the regular expression \( > = + < \).

Assume that the occurrence of the pattern PROPERPLAIN starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

**Example**

\((3, (2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3, 5))\)

Figure 3.664 provides an example where the MAX_WIDTH_PROPERPLAIN \((3, [2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3, 5])\) constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 3\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Symmetries**
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
Figure 3.664: Illustrating the MAX_WIDTH_PROPER_PLAIN constraint of the Example slot
Figures 3.665 and 3.666 respectively depict the automaton associated with the constraint MAX_WIDTH_PROPER_PLAIN and its simplified form.

Figure 3.665: Automaton for the MAX_WIDTH_PROPER_PLAIN constraint obtained by applying decoration Table 2.35 to the seed transducer of the PROPER PLAIN pattern where default is 0.

Figure 3.666: Automaton for the MAX_WIDTH_PROPER_PLAIN constraint obtained by applying decoration Table 2.25 to the seed transducer of the PROPER PLAIN pattern where default is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + D_i \geq 0$ are linear invariants.
Table 3.54: Glue matrix for the MAX_WIDTH_PROPER_PLAIN constraint defined as the composition of the PROPER_PLAIN pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.302 MAX_WIDTH_PROPER_PLATEAU

**DESCRIPTION**

**Origin**
Based on the PROPER_PLATEAU pattern.

**Constraint**
MAX_WIDTH_PROPER_PLATEAU(VALUE, VARIABLES)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var − dvar)

**Restrictions**
- \( sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = 0 \)
- \( VALUE = 0 \lor VALUE \geq 2 \)
- \( VALUE \leq \max(0, sv - 2v) \)
- \( \text{required}(\text{VARIABLES} \var), \text{required}(\text{VARIABLES}) \var) \)

where
- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range}(\text{VARIABLES} \var) \)

**Purpose**
An occurrence of the pattern PROPER_PLATEAU is the *maximal* subsequence which matches the regular expression \( <\equiv^+> \).
Assume that the occurrence of the pattern PROPER_PLATEAU starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

**Example**
(3, [7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3])

Figure 3.667 provides an example where the MAX_WIDTH_PROPER_PLATEAU (3, [7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3]) constraint holds.

**Typical**
- \(|\text{VARIABLES}| > 3 \)
- \( \text{range}(\text{VARIABLES} \var) > 1 \)

**Symmetries**
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.667: Illustrating the MAX_WIDTH_PROPER_PLATEAU constraint of the Example slot
Automaton

Figures 3.668 and 3.669 respectively depict the automaton associated with the constraint MAX_WIDTH_PROPER_PLATEAU and its simplified form.

Figure 3.668: Automaton for the MAX_WIDTH_PROPER_PLATEAU constraint obtained by applying decoration Table 2.35 to the seed transducer of the PROPER_PLATEAU pattern where default is 0.

Figure 3.669: Automaton for the MAX_WIDTH_PROPER_PLATEAU constraint obtained by applying decoration Table 2.25 to the seed transducer of the PROPER_PLATEAU pattern where default is 0; \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-1} + D_i \geq 0 \) are linear invariants.
Table 3.55: Glue matrix for the MAX_WIDTH_PROPER_PLATEAU constraint defined as the composition of the PROPER_PLATEAU pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.303 **MAX_WIDTH_STEADY_SEQUENCE**

**DESCRIPTION**

**Origin**

Based on the `STEADY_SEQUENCE` pattern.

**Constraint**

```
MAX_WIDTH_STEADY_SEQUENCE(VALUE, VARIABLES)
```

**Arguments**

- `VALUE : dvar`
- `VARIABLES : collection(var−dvar)`

**Restrictions**

- `sv ≤ 1 ⇒ VALUE = 0`
- `VALUE = 0 ∨ VALUE ≥ 2`
- `rv = 1 ∧ sv ≥ 2 ⇒ VALUE ≥ sv`
- `rv ≥ 2 ∧ sv ≥ 2 ⇒ VALUE ≥ sv`
- `VALUE ≤ sv`
- `required(VARIABLES, var)`
  
  where

  - `sv = |VARIABLES|`
  - `rv = range(VARIABLES.var)`

**Purpose**

VALUE is the maximal width of occurrences of the `STEADY_SEQUENCE` pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern `STEADY_SEQUENCE` is the maximal subsequence which matches the regular expression `[^+].` Assume that the occurrence of the pattern `STEADY_SEQUENCE` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j − i + 2`.

**Example**

```
(3, (3, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1))
```

Figure 3.670 provides an example where the `MAX_WIDTH_STEADY_SEQUENCE (3, [3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1])` constraint holds.

**Typical**

```
|VARIABLES| > 1
```

**Symmetries**

- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.670: Illustrating the MAX_WIDTH_STEADY_SEQUENCE constraint of the Example slot
Automaton

Figures 3.671 and 3.672 respectively depict the automaton associated with the constraint `MAX_WIDTH_STEADY_SEQUENCE` and its simplified form.

\[
\begin{aligned}
&\{ C \leftarrow \text{default} \\
&\{ D \leftarrow 0 \\
&\{ R \leftarrow \text{default} \}
\end{aligned}
\]

Figure 3.671: Automaton for the `MAX_WIDTH_STEADY_SEQUENCE` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `STEADY_SEQUENCE` pattern where `default` is 0.

\[
\begin{aligned}
&\{ C \leftarrow \text{default} \\
&\{ D \leftarrow 0 \\
&\{ R \leftarrow \max(R, C) \}
\end{aligned}
\]

\[
\begin{aligned}
&\{ C \leftarrow \text{default} \\
&\{ D \leftarrow 0 \\
&\{ R \leftarrow \max(R, C) \}
\end{aligned}
\]

\[
\begin{aligned}
&\{ C \leftarrow \text{default} \\
&\{ D \leftarrow 0 \\
&\{ R \leftarrow \max(R, C) \}
\end{aligned}
\]

\[
\begin{aligned}
&\{ C \leftarrow \text{default} \\
&\{ D \leftarrow 0 \\
&\{ R \leftarrow \max(R, C) \}
\end{aligned}
\]

Table 3.56: Glue matrix for the `MAX_WIDTH_STEADY_SEQUENCE` constraint defined as the composition of the `STEADY_SEQUENCE` pattern, the feature `WIDTH`, and the aggregator `max`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 3.672: Automaton for the MAX_WIDTH_STEADY_SEQUENCE constraint obtained by applying decoration Table 2.30 to the seed transducer of the STEADY_SEQUENCE pattern where default is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + D_i \geq 0$ are linear invariants.
3.304 MAX_WIDTH_STRICTLY_DECREEASING_SEQUENCE

**Description**

**Origin**

Based on the `STRICTLY_DECREEASING_SEQUENCE` pattern.

**Constraint**

`MAX_WIDTH_STRICTLY_DECREEASING_SEQUENCE(VALUE, VARIABLES)`

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td><code>dvar</code></td>
</tr>
<tr>
<td>VARIABLES</td>
<td><code>collection(var − dvar)</code></td>
</tr>
</tbody>
</table>

**Restrictions**

```
sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = 0
VALUE = 0 ∨ VALUE ≥ 2
VALUE ≤ min(sv, rv)
required(VARIABLES, var)
```

where

```
sv = |VARIABLES|
rv = range(VARIABLES.var)
```

**Purpose**

An occurrence of the pattern `STRICTLY_DECREEASING_SEQUENCE` is the **maximal** subsequence which matches the regular expression `>`.

Assume that the occurrence of the pattern `STRICTLY_DECREEASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j − i + 2`.

**Example**

```
(3, (4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3))
```

Figure 3.673 provides an example where the `MAX_WIDTH_STRICTLY_DECREEASING_SEQUENCE (3, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3])` constraint holds.

**Typical**

```
|VARIABLES| > 1
range(VARIABLES.var) > 1
```

**Symmetry**

One and the same constant can be added to the `var` attribute of all items of `VARIABLES`.

**Arg. properties**

Functional dependency: `VALUE` determined by `VARIABLES`.
Figure 3.673: Illustrating the MAX_WIDTH_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Figures 3.674 and 3.675 respectively depict the automaton associated with the constraint \textsc{max width strictly decreasing sequence} and its simplified form.

\begin{align*}
\begin{cases}
    C &\leftarrow \text{default} \\
    D &\leftarrow 0 \\
    R &\leftarrow \text{default}
\end{cases}
\end{align*}

\begin{align*}
\leq \frac{s}{r} \max(\overrightarrow{C}, \overleftarrow{C}) &\begin{cases}
    C &\leftarrow D + 2 \\
    D &\leftarrow 0
\end{cases} \\
\leq \frac{\overrightarrow{C}}{\overrightarrow{D}} + \frac{\overrightarrow{D} + 1}{\overrightarrow{D}}
\end{align*}

Figure 3.674: Automaton for the \textsc{max width strictly decreasing sequence} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \textsc{strictly decreasing sequence} pattern where \texttt{default} is 0

<table>
<thead>
<tr>
<th>(s)</th>
<th>(r)</th>
</tr>
</thead>
<tbody>
<tr>
<td>max((\overrightarrow{C}, \overleftarrow{C}))</td>
<td>max((\overrightarrow{C}, \overleftarrow{C}))</td>
</tr>
<tr>
<td>max((\overrightarrow{C}, \overleftarrow{C}))</td>
<td>(\overrightarrow{C} + \overleftarrow{C} + \overrightarrow{D} + \overleftarrow{D} + 1)</td>
</tr>
</tbody>
</table>

Table 3.57: Glue matrix for the \textsc{max width strictly decreasing sequence} constraint defined as the composition of the \textsc{strictly decreasing sequence} pattern, the feature \textsc{width}, and the aggregator \textsc{max}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
\[
\begin{align*}
\{ & D \leftarrow 0 \\
& R \leftarrow \text{default} \} \\
& \leq \\
& \{ D \leftarrow 0 \} \\
& \leq \\
& \{ D \leftarrow D + 1 \} \\
& R \leftarrow \max(R, D + 1) \\
& > R \\
& \{ R \leftarrow \max(R, 2) \} \\
& \Rightarrow \\
& \{ D \leftarrow 0 \}
\end{align*}
\]

Figure 3.675: Automaton for the `MAX_WIDTH_STRICTLY_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 2.30 to the seed transducer of the `STRICTLY_DECREASING_SEQUENCE` pattern where `default` is 0; \( R_i - R_{i-1} \geq 0 \) and \(-R_i + R_{i-1} + 2 \geq 0\) are linear invariants.
3.305  MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE

**Description**

Based on the **STRICTLY_INCREASING_SEQUENCE** pattern.

**Constraint**

`MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES)`

**Arguments**

- `VALUE : dvar`
- `VARIABLES : collection(var−dvar)`

**Restrictions**

- `sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = 0`
- `VALUE = 0 ∨ VALUE ≥ 2`
- `VALUE ≤ min(sv, rv)`
- `required(VARIABLES, var)`

where

- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

**Purpose**

VALUE is the maximal width of occurrences of the **STRICTLY_INCREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern **STRICTLY_INCREASING_SEQUENCE** is the maximal subsequence which matches the regular expression `<+`. Assume that the occurrence of the pattern **STRICTLY_INCREASING_SEQUENCE** starts at position `i` and ends at position `j`. The feature WIDTH computes the value `j − i + 2`.

**Example**

```
(5, [4, 3, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3])
```

Figure 3.676 provides an example where the `MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE` `(5, [4, 3, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3])` constraint holds.

**Typical**

- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`

**Symmetry**

One and the same constant can be added to the `var` attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.676: Illustrating the \texttt{MAX\_WIDTH\_STRICTLY\_INCREASING\_SEQUENCE} constraint of the \texttt{Example} slot
Automaton

Figures 3.677 and 3.678 respectively depict the automaton associated with the constraint `MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE` and its simplified form.

\[
\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow \text{default}
\end{align*}
\]

Figure 3.677: Automaton for the `MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `STRICTLY_INCREASING_SEQUENCE` pattern where \( \text{default} = 0 \)

<table>
<thead>
<tr>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>( r )</td>
</tr>
<tr>
<td>( s )</td>
<td>( \max(\overline{C}, \overline{C}) )</td>
</tr>
<tr>
<td>( r )</td>
<td>( \max(\overline{C}, \overline{C}) )</td>
</tr>
</tbody>
</table>

Table 3.58: Glue matrix for the `MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE` constraint defined as the composition of the `STRICTLY_INCREASING_SEQUENCE` pattern, the `feature WIDTH`, and the `aggregator max`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 3.678: Automaton for the MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.30 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern where default is 0; \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-1} + 2 \geq 0 \) are linear invariants.
3.306 MAX_WIDTH_SUMMIT

**Origin**
Based on the SUMMIT pattern.

**Constraint**
MAX_WIDTH_SUMMIT(VALUE, VARIABLES)

**Arguments**
- **VALUE**: dvar
- **VARIABLES**: collection(var–dvar)

**Restrictions**
- \( sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = 0 \)
- \( VALUE \geq 0 \)
- \( rv = 2 \Rightarrow VALUE \leq 1 \)
- \( rv \geq 3 \Rightarrow VALUE \leq \max(0, sv - 2) \)
- **required**(VARIABLES, var)

where
- \( sv = |VARIABLES| \)
- \( rv = \text{range}(VARIABLES.var) \)

**Purpose**
An occurrence of the pattern SUMMIT is the *maximal* subsequence which matches the regular expression \((< | (= | <|^*<|>|=(>||^*)>|)\). Assume that the occurrence of the pattern SUMMIT starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

**Example**
\((3, (7, 1, 5, 4, 4, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1))\)

Figure 3.679 provides an example where the MAX_WIDTH_SUMMIT (3, [7, 1, 5, 4, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1]) constraint holds.

**Typical**
- \(|VARIABLES| > 2\)
- \(\text{range}(VARIABLES.var) > 1\)

**Symmetries**
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.679: Illustrating the MAX_WIDTH_SUMMIT constraint of the Example slot.
Automaton

Figures 3.680 and 3.681 respectively depict the automaton associated with the constraint

\[ \text{MAX_WIDTH_SUMMIT} \]

and its simplified form.

Figure 3.680: Automaton for the MAX_WIDTH_SUMMIT constraint obtained by applying decoration Table 2.35 to the seed transducer of the SUMMIT pattern where \( \text{default} \) is 0 (transition \( u \to r \) has the same accumulator update as transition \( r \to u \))
Figure 3.681: Automaton for the MAX_WIDTH_SUMMIT constraint obtained by applying decoration Table 2.26 to the seed transducer of the SUMMIT pattern where default is 0 (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$); $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + D_i \geq 0$ are linear invariants.
### Table 3.59: Glue matrix for the MAX_WIDTH_SUMMIT constraint defined as the composition of the SUMMIT pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
<th>u</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
<tr>
<td>r</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{C} + \overrightarrow{D} + 1)</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
<tr>
<td>t</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{C} + \overrightarrow{D} + 1)</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{C} + \overrightarrow{D} + 1)</td>
</tr>
<tr>
<td>u</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{C} + \overrightarrow{D} + 1)</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
</tbody>
</table>
3.307 MAX_WIDTH_VALLEY

**Description**

**Origin**
Based on the VALLEY pattern.

**Constraint**

\[
\text{MAX_WIDTH_VALLEY} (\text{VALUE, VARIABLES})
\]

**Arguments**

- VALUE : dvar
- VARIABLES : collection(var−dvar)

**Restrictions**

\[
\begin{align*}
sv & \leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & \geq 0 \\
\text{VALUE} & \leq \max (0, sv - 2k) \\
\text{required} & (\text{VARIABLES, var}) \\
\text{where} & \\
sv & = |\text{VARIABLES}| \\
rv & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression \(> (\geq >) (< \leq =)* <\).

Assume that the occurrence of the pattern VALLEY starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

**Example**

\[
(4, (1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 7))
\]

Figure 3.682 provides an example where the \text{MAX_WIDTH_VALLEY} constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 2 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Symmetries**
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.682: Illustrating the MAX_WIDTH VALLEY constraint of the Example slot
Figures 3.683 and 3.684 respectively depict the automaton associated with the constraint \textit{MAX_WIDTH_VALLEY} and its simplified form.

\begin{figure}
\centering
\begin{tikzpicture}
\node[state,initial] (init) {$p$};
\node[state,accepting] (accept) {$t$};
\node[state] (state1) {$s$};
\node[state] (state2) {$\max(R,C)$};
\node[state] (state3) {$\max(-R+D_i,D_i)\geq 0$};
\node[state] (state4) {$\max(-R+D_i,D_i)\geq 0$};

\path[->] (init) edge node {$\{D \leftarrow D + 1\}$} (state1);
\path[->] (state1) edge node {$\{C \leftarrow \text{default}\}$} (state2);
\path[->] (state2) edge node {$\{D \leftarrow 0\}$} (state3);
\path[->] (state3) edge node {$\{D \leftarrow 0\}$} (state4);
\path[->] (state4) edge node {$\{D \leftarrow D + 1\}$} (accept);
\end{tikzpicture}
\end{figure}

\begin{figure}
\centering
\begin{tikzpicture}
\node[state,initial] (init) {$p$};
\node[state,accepting] (accept) {$t$};
\node[state] (state1) {$s$};
\node[state] (state2) {$\max(R,C)$};
\node[state] (state3) {$\max(-R+D_i,D_i)\geq 0$};
\node[state] (state4) {$\max(-R+D_i,D_i)\geq 0$};

\path[->] (init) edge node {$\{D \leftarrow 0\}$} (state1);
\path[->] (state1) edge node {$\{R \leftarrow \text{default}\}$} (state2);
\path[->] (state2) edge node {$\{D \leftarrow \max(R,D_i)\}$} (state3);
\path[->] (state3) edge node {$\{D \leftarrow \max(R,D_i)\}$} (state4);
\path[->] (state4) edge node {$\{D \leftarrow D + 1\}$} (accept);
\end{tikzpicture}
\end{figure}

Figure 3.683: Automaton for the \textit{MAX_WIDTH_VALLEY} constraint obtained by applying decoration Table 2.35 to the seed transducer of the VALLEY pattern where \textit{default} is 0.

Figure 3.684: Automaton for the \textit{MAX_WIDTH_VALLEY} constraint obtained by applying decoration Table 2.26 to the seed transducer of the VALLEY pattern where \textit{default} is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + D_i \geq 0$ are linear invariants.
Table 3.60: Glue matrix for the \textsc{max\_width\_valley} constraint defined as the composition of the \textsc{valley} pattern, the feature \textsc{width}, and the aggregator \textsc{max}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.308 MAX_WIDTH_ZIGZAG

Description

Based on the ZIGZAG pattern.

Constraint

\[ \text{MAX_WIDTH_ZIGZAG}(\text{VALUE}, \text{VARIABLES}) \]

Arguments

- \text{VALUE} : \text{dvar}
- \text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})

Restrictions

- \( \text{sv} \leq 3 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \)
- \( \text{VALUE} = 0 \lor \text{VALUE} \geq 2 \)
- \( \text{VALUE} \leq \max(0, \text{sv} - 2\text{v}) \)
- \( \text{required}(\text{VARIABLES}, \text{var}) \)

where

\[ \text{sv} = |\text{VARIABLES}| \]
\[ \text{rv} = \text{range}(\text{VARIABLES}.\text{var}) \]

Purpose

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression \((<>)^+ (< | <>) | (<>)^+ (> | >>)\).

Assume that the occurrence of the pattern ZIGZAG starts at position \( i \) and ends at position \( j \). The feature \text{WIDTH} computes the value \( j - i \).

Example

\( (6, (4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1)) \)

Figure 3.685 provides an example where the MAX_WIDTH_ZIGZAG \((6, [4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1])\) constraint holds.

Typical

- \( |\text{VARIABLES}| > 3 \)
- \( \text{range}(\text{VARIABLES}.\text{var}) > 1 \)

Symmetries

- Items of \text{VARIABLES} can be reversed.
- One and the same constant can be added to the \text{var} attribute of all items of \text{VARIABLES}.

Arg. properties

Functional dependency: \text{VALUE} determined by \text{VARIABLES}. 

\[ (<>)^+ (< | <>) | (<>)^+ (> | >>) \]
Figure 3.685: Illustrating the MAX_WIDTH_ZIGZAG constraint of the Example slot
Figures 3.686 and 3.687 respectively depict the automaton associated with the constraint MAX_WIDTH_ZIGZAG and its simplified form.
Figure 3.686: Automaton for the MAX_WIDTH_ZIGZAG constraint obtained by applying decoration Table 2.35 to the seed transducer of the ZIGZAG pattern where \texttt{default} is 0; (1) missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=$; (2) on transitions from $b, c, e, f$ to $s$ the accumulator $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the accumulator $R$ is updated wrt $C$ and the accumulator $C$ is reset to its initial value.
Figure 3.687: Automaton for the MAX_WIDTH_ZIGZAG constraint obtained by applying decoration Table 2.26 to the seed transducer of the ZIGZAG pattern where $\text{default}$ is 0; (1) missing transitions from $a$, $b$, $c$, $d$, $e$, $f$ to $s$ are labelled by $=$; (2) on transitions from $b$, $c$, $e$, $f$ to $s$ the accumulator $D$ is reset to its initial value; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + 2 \geq 0$ are linear invariants.
Table 3.61: Glue matrix for the MAX-WIDTH-ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature WIDTH, and the aggregator max; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>max((\overline{a}, \overline{a}))</td>
</tr>
<tr>
<td>a</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>(\overline{a} + \overline{b} + \overline{b} + 1)</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>(\overline{b} + \overline{b} + 1)</td>
<td>max((\overline{a}, \overline{a}))</td>
</tr>
<tr>
<td>b</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>(\overline{a} + \overline{b} + 1)</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>(\overline{b} + 1)</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>(\overline{a} + \overline{b} + \overline{b} + 1)</td>
</tr>
<tr>
<td>c</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>(\overline{a} + \overline{b} + \overline{b} + 1)</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>(\overline{a} + \overline{b} + \overline{b} + 1)</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>max((\overline{a}, \overline{a}))</td>
</tr>
<tr>
<td>d</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>(\overline{a} + \overline{b} + 1)</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>(\overline{a} + \overline{b} + \overline{b} + 1)</td>
<td>max((\overline{a}, \overline{a}))</td>
</tr>
<tr>
<td>e</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>(\overline{a} + \overline{b} + 1)</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>(\overline{a} + \overline{b} + \overline{b} + 1)</td>
<td>max((\overline{a}, \overline{a}))</td>
</tr>
<tr>
<td>f</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>max((\overline{a}, \overline{a}))</td>
<td>max((\overline{a}, \overline{a}))</td>
</tr>
</tbody>
</table>
3.309 MIN_HEIGHT_DECREASING_TERRACE

**DESCRIPTION**

**Origin**

Based on the DECREASING_TERRACE pattern.

**Constraint**

MIN_HEIGHT_DECREASING_TERRACE(VALUE, VARIABLES)

**Arguments**

VALUE : dvar
VARIABLES : collection(var-dvar)

**Restrictions**

\[ sv \leq 3 \lor rv \leq 2 \Rightarrow VALUE = +\infty \]
\[ VALUE \geq minv + 1 \]
\[ VALUE = +\infty \lor VALUE \leq maxv - 1 \]
\[ required(VARIABLES, var) \]

where

\[ minv = minval(VARIABLES, var) \]
\[ maxv = maxval(VARIABLES, var) \]
\[ sv = |VARIABLES| \]
\[ rv = range(VARIABLES, var) \]

**Purpose**

VALUE is the minimum of all minimum values in each occurrence of the DECREASING_TERRACE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(+\infty\).

An occurrence of the pattern DECREASING_TERRACE is the maximal subsequence which matches the regular expression \(\geq^{+} >\).

Assume that the occurrence of the pattern DECREASING_TERRACE starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**

\[(2, (6, 4, 4, 4, 5, 2, 2, 1, 3, 5, 4, 4, 3, 3, 3))\]

Figure 3.688 provides an example where the MIN_HEIGHT_DECREASING_TERRACE (2, [6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3]) constraint holds.

**Typical**

\[|VARIABLES| > 3\]
\[range(VARIABLES, var) > 2\]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.688: Illustrating the MIN_HEIGHT_DECREASING_TERRACE constraint of the Example slot
Automaton Figures 3.689 and 3.690 respectively depict the automaton associated with the constraint \textsc{MIN\_HEIGHT\_DECREASING\_TERRACE} and its simplified form.

![Automaton Diagram](image)

Figure 3.689: Automaton for the \textsc{MIN\_HEIGHT\_DECREASING\_TERRACE} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \textsc{DECREASING\_TERRACE} pattern where \textsc{default} is $+\infty$.

![Automaton Diagram](image)

Figure 3.690: Automaton for the \textsc{MIN\_HEIGHT\_DECREASING\_TERRACE} constraint obtained by applying decoration Table 2.37 to the seed transducer of the \textsc{DECREASING\_TERRACE} pattern where \textsc{default} is $+\infty$; $-R_d + R_{d-1} \geq 0$ is a linear invariant.
Table 3.62: Glue matrix for the MIN_HEIGHT_DECENDING_TERRACE constraint defined as the composition of the DECREASING_TERRACE pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.310 MIN_HEIGHT_INCREASING_TERRACE

**Description**

Based on the INCREASING_TERRACE pattern.

**Constraint**

MIN_HEIGHT_INCREASING_TERRACE(VALUE, VARIABLES)

**Arguments**

- VALUE : dvar
- VARIABLES : collection(var ~ dvar)

**Restrictions**

\[
\begin{align*}
sv & \leq 3 \lor rv \leq 2 \Rightarrow VALUE = +\infty \\
VALUE & \geq \text{minv} + 1 \\
VALUE & = +\infty \lor VALUE \leq \text{maxv} - 1 \\
\text{required}(\text{VARIABLES}, \text{var}) & \text{ where} \\
\text{minv} & = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
sv & = |\text{VARIABLES}| \\
rv & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

VALUE is the minimum of all minimum values in each occurrence of the INCREASING_TERRACE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value $+\infty$.

An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression $<\equiv^+<$.

Assume that the occurrence of the pattern INCREASING_TERRACE starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i + 1$ to index $j$.

**Example**

(3, (1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4))

Figure 3.691 provides an example where the MIN_HEIGHT_INCREASING_TERRACE (3, [1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4]) constraint holds.

**Typical**

- $|\text{VARIABLES}| > 3$
- \text{range}(\text{VARIABLES}.\text{var}) > 2$

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.691: Illustrating the MIN\_HEIGHT\_INCREASING\_TERRACE constraint of the Example slot
Figures 3.692 and 3.693 respectively depict the automaton associated with the constraint MIN_HEIGHT_INCREASING_TERRACE and its simplified form.

Figure 3.692: Automaton for the MIN_HEIGHT_INCREASING_TERRACE constraint obtained by applying decoration Table 2.35 to the seed transducer of the INCREASING_TERRACE pattern where default is $+\infty$.

Figure 3.693: Automaton for the MIN_HEIGHT_INCREASING_TERRACE constraint obtained by applying decoration Table 2.37 to the seed transducer of the INCREASING_TERRACE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.63: Glue matrix for the MIN_HEIGHT_INCREASING_TERRACE constraint defined as the composition of the INCREASING_TERRACE pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>r</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_i)$</td>
</tr>
<tr>
<td>t</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_i)$</td>
<td>$\min(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_i)$</td>
</tr>
</tbody>
</table>
### MIN_HEIGHTPLAIN

**Origin**
Based on the **PLAIN** pattern.

**Constraint**
\[
\text{MIN\_HEIGHT\_PLAIN}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**
- \( \text{VALUE} : \text{dvar} \)
- \( \text{VARIABLES} : \text{collection(var-dvar)} \)

**Restrictions**
- \( \text{sv} \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty \)
- \( \text{VALUE} \geq \minv \)
- \( \text{VALUE} = +\infty \lor \text{VALUE} \leq \maxv - 1 \)

\[
\text{required}(\text{VARIABLES}, \text{var})
\]

where
- \( \minv = \text{minval}(\text{VARIABLES.var}) \)
- \( \maxv = \text{maxval}(\text{VARIABLES.var}) \)
- \( \text{sv} = |\text{VARIABLES}| \)
- \( \text{rv} = \text{range}(\text{VARIABLES.var}) \)

**Purpose**
An occurrence of the pattern **PLAIN** is the *maximal* subsequence which matches the regular expression \( > = * < \).
Assume that the occurrence of the pattern **PLAIN** starts at position \( i \) and ends at position \( j \). The feature **MIN** computes the minimum of the values from index \( i + 1 \) to index \( j \).

**Example**
\[
(3, (2, 3, 6, 5, 7, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3))
\]

Figure 3.694 provides an example where the **MIN\_HEIGHT\_PLAIN** constraint holds.

**Typical**
- \( |\text{VARIABLES}| > 2 \)
- \( \text{range}(\text{VARIABLES.var}) > 1 \)

**Symmetry**
Items of **VARIABLES** can be reversed.

**Arg. properties**
Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 3.694: Illustrating the MIN_HEIGHT_PLAIN constraint of the Example slot
Automaton

Figures 3.695 and 3.696 respectively depict the automaton associated with the constraint MIN_HEIGHTPLAIN and its simplified form.

Figure 3.695: Automaton for the MIN_HEIGHTPLAIN constraint obtained by applying decoration Table 2.35 to the seed transducer of the PLAIN pattern where default is $+\infty$.

Figure 3.696: Automaton for the MIN_HEIGHTPLAIN constraint obtained by applying decoration Table 2.37 to the seed transducer of the PLAIN pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.64: Glue matrix for the MIN_HEIGHTPLAIN constraint defined as the composition of the \texttt{PLAIN} pattern, the feature \texttt{MIN}, and the aggregator \texttt{min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\min(\widehat{C}, \widehat{C})$</td>
<td>$\min(\widehat{C}, \widehat{C})$</td>
<td>$\min(\widehat{C}, \widehat{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\min(\widehat{C}, \widehat{C})$</td>
<td>$\min(\widehat{D}, \widehat{D}, \text{VAR}_i)$</td>
<td>$\min(\widehat{D}, \widehat{D}, \text{VAR}_i)$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\min(\widehat{C}, \widehat{C})$</td>
<td>$\min(\widehat{D}, \widehat{D}, \text{VAR}_i)$</td>
<td>$\min(\widehat{D}, \widehat{D}, \text{VAR}_i)$</td>
</tr>
</tbody>
</table>
3.312  MIN_HEIGHT_PLATEAU

**DESCRIPTION AUTOMATON**

**Origin**
Based on the PLATEAU pattern.

**Constraint**
MIN_HEIGHT_PLATEAU(VALUE, VARIABLES)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var−dvar)

**Restrictions**
- \( sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = +\infty \)
- \( VALUE \geq \minv + 1 \)
- \( VALUE = +\infty \lor VALUE \leq \maxv \)

where
- \( \minv = \text{minval}(\text{VARIABLES}.\text{var}) \)
- \( \maxv = \text{maxval}(\text{VARIABLES}.\text{var}) \)
- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

**Purpose**
VALUE is the minimum of all minimum values in each occurrence of the PLATEAU pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(+\infty\).

An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression \(<=^*\).

Assume that the occurrence of the pattern PLATEAU starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i+1\) to index \(j\).

**Example**
\[
(3, (7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5))
\]

Figure 3.697 provides an example where the MIN_HEIGHT_PLATEAU \((3, [7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5])\) constraint holds.

**Typical**
- \(|\text{VARIABLES}| > 2\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.697: Illustrating the MIN_HEIGHT_PLATEAU constraint of the Example slot
Automaton

Figures 3.698 and 3.699 respectively depict the automaton associated with the constraint `MIN_HEIGHT_PLATEAU` and its simplified form.

Figure 3.698: Automaton for the `MIN_HEIGHT_PLATEAU` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `PLATEAU` pattern where `default` is $+\infty$

Figure 3.699: Automaton for the `MIN_HEIGHT_PLATEAU` constraint obtained by applying decoration Table 2.37 to the seed transducer of the `PLATEAU` pattern where `default` is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.65: Glue matrix for the MIN\_HEIGHT\_PLATEAU constraint defined as the composition of the PLATEAU pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_i)$</td>
<td>$\min(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_i)$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_i)$</td>
<td>$\min(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_i)$</td>
</tr>
</tbody>
</table>
3.313  MIN_HEIGHT_PROPER_PLAIN

**Origin**
Based on the PROPERPLAIN pattern.

**Constraint**
MIN_HEIGHT_PROPER_PLAIN(VALUE, VARIABLES)

**Arguments**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>: dvar</td>
</tr>
<tr>
<td>VARIABLES</td>
<td>: collection(var–dvar)</td>
</tr>
</tbody>
</table>

**Restrictions**

\[
sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = +\infty \\
VALUE \geq \text{minv} \\
VALUE = +\infty \lor VALUE \leq \text{maxv} - 1
\]

where

\[
\text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \\
sv = |\text{VARIABLES}| \\
rv = \text{range}(\text{VARIABLES}.\text{var})
\]

**Purpose**
VALUE is the minimum of all minimum values in each occurrence of the PROPERPLAIN pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +∞.

An occurrence of the pattern PROPERPLAIN is the maximal subsequence which matches the regular expression > =+ <.

Assume that the occurrence of the pattern PROPERPLAIN starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**

\((3, (2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3, 5))\)

Figure 3.700 provides an example where the MIN_HEIGHT_PROPER_PLAIN constraint holds.

**Typical**

\[|\text{VARIABLES}| > 3 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1\]

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.700: Illustrating the MIN_HEIGHT_PROPER_PLAIN constraint of the Example slot
Figures 3.701 and 3.702 respectively depict the automaton associated with the constraint MIN_HEIGHT_PROPER_PLAIN and its simplified form.

Figure 3.701: Automaton for the MIN_HEIGHT_PROPER_PLAIN constraint obtained by applying decoration Table 2.35 to the seed transducer of the PROPER_PLAIN pattern where default is $+\infty$.

Figure 3.702: Automaton for the MIN_HEIGHT_PROPER_PLAIN constraint obtained by applying decoration Table 2.37 to the seed transducer of the PROPER_PLAIN pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.66: Glue matrix for the MIN_HEIGHT_PROPER_PLAIN constraint defined as the composition of the PROPER_PLAIN pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### 3.314 MIN_HEIGHT_PROPER_PLATEAU

#### DESCRIPTION

**Origin**
Based on the `PROPER_PLATEAU` pattern.

**Constraint**

```
MIN_HEIGHT_PROPER_PLATEAU(VALUE, VARIABLES)
```

**Arguments**

```
VALUE : dvar
VARIABLES : collection(var−dvar)
```

**Restrictions**

```
sv ≤ 3 \lor rv ≤ 1 \Rightarrow VALUE = +\infty
VALUE ≥ \minv + 1 \Rightarrow VALUE = +\infty \lor VALUE ≤ \maxv
```

where

```
\minv = \minval(VARIABLES.var)
\maxv = \maxval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)
```

**Purpose**

`VALUE` is the minimum of all minimum values in each occurrence of the `PROPER_PLATEAU` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, `VALUE` takes the default value `+\infty`.

An occurrence of the pattern `PROPER_PLATEAU` is the *maximal* subsequence which matches the regular expression `<=+>`.

Assume that the occurrence of the pattern `PROPER_PLATEAU` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i + 1` to index `j`.

**Example**

```
(3, (7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3))
```

Figure 3.703 provides an example where the `MIN_HEIGHT_PROPER_PLATEAU (3, [7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3])` constraint holds.

**Typical**

```
|VARIABLES| > 3
range(VARIABLES.var) > 1
```

**Symmetry**
Items of `VARIABLES` can be reversed.

**Arg. properties**

Functional dependency: `VALUE` determined by `VARIABLES`. 
Figure 3.703: Illustrating the MIN_HEIGHT_PROPER_PLATEAU constraint of the Example slot
Figures 3.704 and 3.705 respectively depict the automaton associated with the constraint MIN_HEIGHT_PROPER_PLATEAU and its simplified form.

Figure 3.704: Automaton for the MIN_HEIGHT_PROPER_PLATEAU constraint obtained by applying decoration Table 2.35 to the seed transducer of the PROPER_PLATEAU pattern where default is $+\infty$.

Figure 3.705: Automaton for the MIN_HEIGHT_PROPER_PLATEAU constraint obtained by applying decoration Table 2.37 to the seed transducer of the PROPER_PLATEAU pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.67: Glue matrix for the \textsc{min\_height\_proper\_plateau} constraint defined as the composition of the \textsc{proper\_plateau} pattern, the feature \textsc{min}, and the aggregator \textsc{min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.315 MIN_HEIGHT_STEADY

DESCRIPTION

Origin
Based on the STEADY pattern.

Constraint
\[ MIN_{\text{HEIGHT\_STEADY}}(\text{VALUE}, \text{VARIABLES}) \]

Arguments
\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} - \text{dvar})
\end{align*}
\]

Restrictions
\[
\begin{align*}
\text{sv} \leq 1 & \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} & \geq \text{minv} \\
\text{VALUE} & = +\infty \lor \text{VALUE} \leq \text{maxv} \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{where} & \\
\text{minv} & = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{sv} & = |\text{VARIABLES}|
\end{align*}
\]

VALUE is the minimum of all minimum values in each occurrence of the STEADY pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +\infty.

Purpose
An occurrence of the pattern STEADY is the subsequence which matches the regular expression =.
Assume that the occurrence of the pattern STEADY starts at position \( i \) and ends at position \( j \). The feature MIN computes the minimum of the values from index \( i \) to index \( j + 1 \).

Example
\[ (1, (1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6)) \]

Figure 3.706 provides an example where the MIN_HEIGHT_STEADY \((1, [1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6])\) constraint holds.

Typical
\[ |\text{VARIABLES}| > 1 \]

Symmetry
Items of VARIABLES can be reversed.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 3.706: Illustrating the MIN_HEIGHT_STEADY constraint of the Example slot
Automaton

Figures 3.707 and 3.708 respectively depict the automaton associated with the constraint \( \text{MIN\_HEIGHT\_STEADY} \) and its simplified form.

\[
\begin{cases}
  C &\leftarrow \text{default} \\
  D &\leftarrow +\infty \\
  R &\leftarrow \text{default}
\end{cases}
\]

\[
\{ D \leftarrow +\infty \} \quad \{ R \leftarrow \min(R, \min(D, \text{VAR}_i, \text{VAR}_{i+1})) \}
\]

Figure 3.707: Automaton for the \( \text{MIN\_HEIGHT\_STEADY} \) constraint obtained by applying decoration Table 2.35 to the seed transducer of the \( \text{STEADY} \) pattern where \( \text{default} \) is \( +\infty \).

\[
\{ R \leftarrow \text{default} \}
\]

\[
\{ R \leftarrow \min(R, \text{VAR}_i) \}
\]

Figure 3.708: Automaton for the \( \text{MIN\_HEIGHT\_STEADY} \) constraint obtained by applying decoration Table 2.37 to the seed transducer of the \( \text{STEADY} \) pattern where \( \text{default} \) is \( +\infty \); \( -R_i + R_{i-1} \geq 0 \) is a linear invariant.

Table 3.68: Glue matrix for the \( \text{MIN\_HEIGHT\_STEADY} \) constraint defined as the composition of the \( \text{STEADY} \) pattern, the feature \( \text{MIN} \), and the aggregator \( \text{min} \); cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### 3.316 MIN_HEIGHT_STEADY_SEQUENCE

#### Description

**Origin**
Based on the `STEADY_SEQUENCE` pattern.

**Constraint**

\[ \text{MIN_HEIGHT_STEADY_SEQUENCE}(\text{VALUE}, \text{VARIABLES}) \]

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var-\text{dvar})`

**Restrictions**

- \( sv \leq 1 \Rightarrow \text{VALUE} = +\infty \)
- \( \text{VALUE} \geq \text{minv} \)
- \( \text{VALUE} = +\infty \lor \text{VALUE} \leq \text{maxv} \)

where

- \( \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \)
- \( \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \)
- \( sv = |\text{VARIABLES}| \)

**Purpose**

VALUE is the minimum of all minimum values in each occurrence of the `STEADY_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, VALUE takes the default value \(+\infty\).

An occurrence of the pattern `STEADY_SEQUENCE` is the maximal subsequence which matches the regular expression \( =^{+} \).

Assume that the occurrence of the pattern `STEADY_SEQUENCE` starts at position \( i \) and ends at position \( j \). The feature MIN computes the minimum of the values from index \( i \) to index \( j + 1 \).

**Example**

\[ (1, [3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1]) \]

Figure 3.709 provides an example where the `MIN_HEIGHT_STEADY_SEQUENCE` \((1, [3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1])\) constraint holds.

**Typical**

\( |\text{VARIABLES}| > 1 \)

**Symmetry**
Items of `VARIABLES` can be reversed.

**Arg. properties**

Functional dependency: `VALUE` determined by `VARIABLES`. 
Figure 3.709: Illustrating the MIN_HEIGHT_STEADY_SEQUENCE constraint of the Example slot
Figures 3.710 and 3.711 respectively depict the automaton associated with the constraint MIN_HEIGHT_STEADY_SEQUENCE and its simplified form.

\[
\begin{align*}
\{ & C \leftarrow \text{default} \\
D & \leftarrow +\infty \\
R & \leftarrow \text{default} \\
\} \\
\end{align*}
\]

\[
\begin{align*}
\{ & C \leftarrow \text{default} \\
D & \leftarrow +\infty \\
R & \leftarrow \text{min}(R, C) \\
\} \\
\end{align*}
\]
Table 3.69: Glue matrix for the MIN_HEIGHT_STEADY_SEQUENCE constraint defined as the composition of the STEADY_SEQUENCE pattern, the feature MIN, and the aggregator min: cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### 3.3.17 MIN_MAX_BUMP_ON_DECREASING_SEQUENCE

**Description**

Based on the `BUMP_ON_DECREASING_SEQUENCE` pattern.

**Constraint**

`MIN_MAX_BUMP_ON_DECREASING_SEQUENCE(VALUE, VARIABLES)`

**Arguments**

- VALUE : `dvar`
- VARIABLES : `collection(var−dvar)`

**Restrictions**

- `sv ≤ 5 ∨ rv ≤ 2 ⇒ VALUE = +∞`
- `VALUE ≥ minv + 2⇒`
- `VALUE = +∞ ∨ VALUE ≤ maxv`
- `required(VARIABLES, var)`
  
  where
  - `minv = minval(VARIABLES, var)`
  - `maxv = maxval(VARIABLES, var)`
  - `sv = |VARIABLES|`
  - `rv = range(VARIABLES, var)`

**Purpose**

VALUE is the minimum of all maximum values in each occurrence of the `BUMP_ON_DECREASING_SEQUENCE` pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value `+∞`. An occurrence of the pattern `BUMP_ON_DECREASING_SEQUENCE` is the subsequence which matches the regular expression `><>>`. Assume that the occurrence of the pattern `BUMP_ON_DECREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature MAX computes the maximum of the values from index `i + 2` to index `j`.

**Example**

```
(5, (7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3))
```

Figure 3.712 provides an example where the `MIN_MAX_BUMP_ON_DECREASING_SEQUENCE (5, [7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3])` constraint holds.

**Typical**

- `|VARIABLES| > 5`
- `range(VARIABLES, var) > 2`

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.712: Illustrating the `MIN_MAX_BUMP_ON_DECREASING_SEQUENCE` constraint of the Example slot
Figures 3.713 and 3.714 respectively depict the automaton associated with the constraint MIN_MAX_BUMP_ON_DECREASING_SEQUENCE and its simplified form.

**Figure 3.713:** Automaton for the MIN_MAX_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern where default is $+\infty$. 

---

**Automaton**
Figure 3.714: Automaton for the MIN_MAX_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.27 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
3.318 MIN_MAX_DECREASING

DESCRIPTION

Origin
Based on the DECREASING pattern.

Constraint

\[ \text{MIN\_MAX\_DECREASING}(\text{VALUE, VARIABLES}) \]

Arguments

\[
\begin{align*}
\text{VALUE} & : \ dvar \\
\text{VARIABLES} & : \ \text{collection}(\text{VAR} - \ dvar)
\end{align*}
\]

Restrictions

\[
\begin{align*}
sv & \leq 1 \lor rv \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} & \geq \text{min} + 1 \\
\text{VALUE} & = +\infty \lor \text{VALUE} \leq \text{maxv} \text{ required}(\text{VARIABLES}, \text{VAR}) \\
\text{where} & \\
\text{min} & = \text{minval}(\text{VARIABLES}, \text{VAR}) \\
\text{max} & = \text{maxval}(\text{VARIABLES}, \text{VAR}) \\
sv & = |\text{VARIABLES}| \\
rv & = \text{range}(\text{VARIABLES}, \text{VAR})
\end{align*}
\]

Purpose

\text{VALUE} is the minimum of all maximum values in each occurrence of the DECREASING pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, \text{VALUE} takes the default value +\infty.

An occurrence of the pattern DECREASING is the subsequence which matches the regular expression >.

Assume that the occurrence of the pattern DECREASING starts at position \( i \) and ends at position \( j \). The feature MAX computes the maximum of the values from index \( i \) to index \( j + 1 \).

Example

\[(3, (3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))\]

Figure 3.715 provides an example where the MIN\_MAX\_DECREASING \((3, [3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4])\) constraint holds.

Typical

\[ |\text{VARIABLES}| > 1 \]

\[ \text{range}(\text{VARIABLES}, \text{VAR}) > 1 \]

Arg. properties

Functional dependency: \text{VALUE} determined by \text{VARIABLES}.  

AUTOMATON
Figure 3.715: Illustrating the MIN_MAX_DECREASING constraint of the Example slot
Figures 3.716 and 3.717 respectively depict the automaton associated with the constraint \textsc{MIN\_MAX\_DECREASING} and its simplified form.

\[
\begin{cases}
    C \leftarrow \text{default} \\
    D \leftarrow -\infty \\
    R \leftarrow \text{default}
\end{cases}
\]

\[
\begin{cases}
    D \leftarrow -\infty \\
    R \leftarrow \min(R, \max(D, \mathit{VAR}_i, \mathit{VAR}_{i+1}))
\end{cases}
\]

Figure 3.716: Automaton for the \textsc{MIN\_MAX\_DECREASING} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \textsc{DECREASING} pattern where \textit{default} is \(+\infty\).

\[
\begin{cases}
    R \leftarrow \text{default}
\end{cases}
\]

\[
\begin{cases}
    R \leftarrow \min(R, \mathit{VAR}_i)
\end{cases}
\]

Figure 3.717: Automaton for the \textsc{MIN\_MAX\_DECREASING} constraint obtained by applying decoration Table 2.37 to the seed transducer of the \textsc{DECREASING} pattern where \textit{default} is \(+\infty\); \(-R_i + R_{i-1} \geq 0\) is a linear invariant.

Table 3.70: Glue matrix for the \textsc{MIN\_MAX\_DECREASING} constraint defined as the composition of the \textsc{DECREASING} pattern, the feature \textsc{MAX}, and the aggregator \textsc{min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### 3.319 MIN_MAX_DECREASING_SEQUENCE

**Description**

**Origin**

Based on the `DECREASING_SEQUENCE` pattern.

**Constraint**

\[
\text{MIN_MAX_DECREASING_SEQUENCE}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var-dvar)`

**Restrictions**

\[
\begin{align*}
sv & \leq 1 \lor rv \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} & \geq \text{minv} + 1 \\
\text{VALUE} & = +\infty \lor \text{VALUE} \leq \text{maxv} \\
\text{required} & (\text{VARIABLES}, \text{var})
\end{align*}
\]

where

- \( \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \)
- \( \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \)
- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

**Purpose**

**VALUE** is the minimum of all maximum values in each occurrence of the `DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, **VALUE** takes the default value `+\infty`.

An occurrence of the pattern `DECREASING_SEQUENCE` is the **maximal** subsequence which matches the regular expression `>(|=)*>|>`.

Assume that the occurrence of the pattern `DECREASING_SEQUENCE` starts at position \( i \) and ends at position \( j \). The feature **MAX** computes the maximum of the values from index \( i \) to index \( j + 1 \).

**Example**

\[
(4, \{3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4\})
\]

Figure 3.718 provides an example where the `MIN_MAX_DECREASING_SEQUENCE` \((4, [3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4])\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Arg. properties**

Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 3.718: Illustrating the MIN_MAX_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 3.719 and 3.720 respectively depict the automaton associated with the constraint MIN_MAX_DECREASING_SEQUENCE and its simplified form.

\[
C \leftarrow \text{default} \\
D \leftarrow -\infty \\
R \leftarrow \text{default}
\]

\[
\begin{cases}
  C \leftarrow \text{default} \quad D \leftarrow -\infty \\
  \min(R, C) \\
  D \leftarrow \max(D, \text{VAR}_i) \\
  \max(D, \text{VAR}_{i+1})
\end{cases}
\]

\[
\begin{cases}
  C \leftarrow \text{default} \\
  D \leftarrow -\infty \\
  R \leftarrow \text{default}
\end{cases}
\]

\[
\begin{cases}
  C \leftarrow \max(C, \text{VAR}_i + 1) \\
  D \leftarrow -\infty \\
  \max(D, \text{VAR}_{i+1})
\end{cases}
\]

Figure 3.719: Automaton for the MIN_MAX_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the DECREASING_SEQUENCE pattern where \text{default} is +\infty

\[
\begin{cases}
  R \leftarrow \text{default}
\end{cases}
\]

\[
\begin{cases}
  R \leftarrow \min(R, \text{VAR}_i) \\
  D \leftarrow -\infty \\
  C \leftarrow \max(D, \text{VAR}_{i+1})
\end{cases}
\]

\[
\begin{cases}
  R \leftarrow \min(R, \text{VAR}_i) \\
  D \leftarrow -\infty \\
  C \leftarrow \max(C, \text{VAR}_i + 1)
\end{cases}
\]

Figure 3.720: Automaton for the MIN_MAX_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.37 to the seed transducer of the DECREASING_SEQUENCE pattern where \text{default} is +\infty; \ -R_i + R_{i-1} \geq 0 \text{ is a linear invariant.}
Table 3.71: Glue matrix for the **MIN_MAX_DECREASING_SEQUENCE** constraint defined as the composition of the **DECREASING_SEQUENCE** pattern, the feature **MAX**, and the aggregator **min**; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
<tr>
<td>t</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}, \overrightarrow{VAR}))</td>
</tr>
</tbody>
</table>
### 3.320 MIN_MAX_DIP_ON_INCREASING_SEQUENCE

#### Description

- **Origin**: Based on the `DIP_ON_INCREASING_SEQUENCE` pattern.
- **Constraint**: `MIN_MAX_DIP_ON_INCREASING_SEQUENCE(VALUE, VARIABLES)`
- **Arguments**:
  - `VALUE : dvar`
  - `VARIABLES : collection(var−dvar)`
- **Restrictions**:
  
  \[
  \begin{align*}
  sv \leq 5 \lor rv \leq 2 & \Rightarrow VALUE = +\infty \\
  VALUE \geq minv + 2 & \\
  VALUE = +\infty \lor VALUE \leq maxv & \\
  required(VARIABLES, var) & \\
  \text{where} & \\
  minv &= \text{minval}(VARIABLES.var) \\
  \text{maxv} &= \text{maxval}(VARIABLES.var) \\
  sv &= |VARIABLES| \\
  rv &= \text{range}(VARIABLES.var)
  \end{align*}
  \]

- **Purpose**: `VALUE` is the minimum of all maximum values in each occurrence of the `DIP_ON_INCREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, `VALUE` takes the default value `+\infty`.

  An occurrence of the pattern `DIP_ON_INCREASING_SEQUENCE` is the subsequence which matches the regular expression `<<<><<`. Assume that the occurrence of the pattern `DIP_ON_INCREASING_SEQUENCE` starts at position $i$ and ends at position $j$. The feature `MAX` computes the maximum of the values from index $i + 2$ to index $j$.

- **Example**:
  
  \[(5, (1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4))\]

  Figure 3.721 provides an example where the `MIN_MAX_DIP_ON_INCREASING_SEQUENCE (5, [1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4])` constraint holds.

- **Typical**:
  
  \[|VARIABLES| > 5 \land \text{range}(VARIABLES.var) > 2\]

- **Arg. properties**: Functional dependency: `VALUE` determined by `VARIABLES`. 

Figure 3.721: Illustrating the MIN_MAX_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot
Figures 3.722 and 3.723 respectively depict the automaton associated with the constraint \texttt{MIN_MAX_DIP_ON_INCREASING_SEQUENCE} and its simplified form.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{automaton.png}
\caption{Automaton for the \texttt{MIN_MAX_DIP_ON_INCREASING_SEQUENCE} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{DIP_ON_INCREASING_SEQUENCE} pattern where \texttt{default} is $+\infty$}
\end{figure}
Figure 3.723: Automaton for the MIN_MAX_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.25 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
### 3.321 MIN_MAX_INCREASING

#### Description

**Origin**
Based on the `INCREASING` pattern.

**Constraint**

```
MIN_MAX_INCREASING(VALUE, VARIABLES)
```

**Arguments**

```
VALUE : dvar
VARIABLES : collection(var−dvar)
```

**Restrictions**

```
sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty
VALUE \geq minv + 1  
VALUE = +\infty \lor VALUE \leq maxv
```

where

```
minv = minval(VARIABLES.var)
maxv = maxval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)
```

**Purpose**

VALUE is the minimum of all maximum values in each occurrence of the `INCREASING` pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value $+\infty$.

An occurrence of the pattern `INCREASING` is the subsequence which matches the regular expression `<`. Assume that the occurrence of the pattern `INCREASING` starts at position $i$ and ends at position $j$. The feature `MAX` computes the maximum of the values from index $i$ to index $j + 1$.

**Example**

```
(3, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))
```

Figure 3.724 provides an example where the `MIN_MAX_INCREASING (3, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3])` constraint holds.

**Typical**

```
|VARIABLES| > 1
range(VARIABLES.var) > 1
```

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.724: Illustrating the MIN_MAX_INCREMENTING constraint of the Example slot
Automaton

Figures 3.725 and 3.726 respectively depict the automaton associated with the constraint MIN_MAX_INCREASING and its simplified form.

$$\begin{align*}
\{ C \leftarrow \text{default} \} \\
\{ D \leftarrow -\infty \} \\
\{ R \leftarrow \text{default} \}
\end{align*}$$

\[
\begin{cases}
D \leftarrow -\infty \\
R \leftarrow \min(R, \max(D, \text{VAR}_i, \text{VAR}_{i+1}))
\end{cases}
\]

Figure 3.725: Automaton for the MIN_MAX_INCREASING constraint obtained by applying decoration Table 2.35 to the seed transducer of the INCREASING pattern where default is \(+\infty\).

$$\begin{align*}
\{ R \leftarrow \text{default} \} \\
\{ R \leftarrow \min(R, \max(\text{VAR}_i, \text{VAR}_{i+1})) \}
\end{align*}$$

\[
\begin{cases}
R \leftarrow \min(R, \max(\text{VAR}_i, \text{VAR}_{i+1}))
\end{cases}
\]

Figure 3.726: Automaton for the MIN_MAX_INCREASING constraint obtained by applying decoration Table 2.38 to the seed transducer of the INCREASING pattern where default is \(+\infty\); \(-R_i + R_{i-1} \geq 0\) is a linear invariant.

Table 3.72: Glue matrix for the MIN_MAX_INCREASING constraint defined as the composition of the INCREASING pattern, the feature MAX, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.322  **MIN_MAX_INCREASING_SEQUENCE**

**Description**

Based on the `INCREASING_SEQUENCE` pattern.

**Constraint**

\[ \text{MIN_MAX_INCREASING_SEQUENCE} (\text{VALUE, VARIABLES}) \]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection(\text{var} - \text{dvar})}
\end{align*}
\]

**Restrictions**

\[
\text{sv} \leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} \geq \text{minv} + 1 \Rightarrow \text{VALUE} = +\infty \lor \text{VALUE} \leq \text{maxv} \\
\text{required}(\text{VARIABLES, var})
\]

where

\[
\begin{align*}
\text{minv} & = \text{minval}(\text{VARIABLES.var}) \\
\text{maxv} & = \text{maxval}(\text{VARIABLES.var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES.var})
\end{align*}
\]

**Purpose**

\[ \text{VALUE} \] is the minimum of all maximum values in each occurrence of the `INCREASING_SEQUENCE` pattern in the time-series given by the \text{VARIABLES} collection. If the pattern does not occur, \text{VALUE} takes the default value \(+\infty\).  

An occurrence of the pattern `INCREASING_SEQUENCE` is the maximal subsequence which matches the regular expression \(< (< | =)^* < | <\). 

Assume that the occurrence of the pattern `INCREASING_SEQUENCE` starts at position \(i\) and ends at position \(j\). The feature \text{MAX} computes the maximum of the values from index \(i\) to index \(j + 1\).

**Example**

\[(3, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))\]

Figure 3.727 provides an example where the `MIN_MAX_INCREASINGSEQUENCE` \(3, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3])\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES.var}) > 1
\]

**Arg. properties**

Functional dependency: \text{VALUE} determined by \text{VARIABLES}.
Figure 3.727: Illustrating the MIN_MAX_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 3.728 and 3.729 respectively depict the automaton associated with the constraint MIN_MAX_INCREASING_SEQUENCE and its simplified form.

Figure 3.728: Automaton for the MIN_MAX_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the INCREASING_SEQUENCE pattern where default is $+\infty$.

Figure 3.729: Automaton for the MIN_MAX_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.24 to the seed transducer of the INCREASING_SEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.73: Glue matrix for the MIN_MAX_INCREASING_SEQUENCE constraint defined as the composition of the INCREASING_SEQUENCE pattern, the feature MAX, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>min(C, C)</td>
<td></td>
</tr>
<tr>
<td>t</td>
<td>min(C, C)</td>
<td>max(C, C, D, D, VAR)</td>
</tr>
</tbody>
</table>
### 3.323 MIN_MAX_INFLEXION

**DESCRIPTION AUTOMATON**

- **Origin**: Based on the **INFLEXION** pattern.
- **Constraint**: 
  \[
  \text{MIN}\_\text{MAX}\_\text{INFLEXION}(\text{VALUE}, \text{VARIABLES})
  \]
- **Arguments**: 
  - \(\text{VALUE} : \text{dvar}\)
  - \(\text{VARIABLES} : \text{collection}(\text{var}−\text{dvar})\)
- **Restrictions**: 
  \[
  \begin{align*}
  sv \leq 2 & \lor rv \leq 1 \Rightarrow \text{VALUE} = +\infty \\
  \text{VALUE} \geq \text{minv} & \\
  \text{VALUE} = +\infty & \lor \text{VALUE} \leq \text{maxv} \\
  \text{required}(\text{VARIABLES}, \text{var}) & \\
  \text{where} & \\
  \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) & \\
  \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) & \\
  sv = |\text{VARIABLES}| & \\
  rv = \text{range}(\text{VARIABLES}.\text{var}) & \\
  \end{align*}
  \]
- **Purpose**: 
  - \(\text{VALUE}\) is the minimum of all maximum values in each occurrence of the **INFLEXION** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, \(\text{VALUE}\) takes the default value \(+\infty\).
  - An occurrence of the pattern **INFLEXION** is the maximal subsequence which matches the regular expression \(< (| =)\ast | | (>| =)\ast <\). Assume that the occurrence of the pattern **INFLEXION** starts at position \(i\) and ends at position \(j\). The feature **MAX** computes the maximum of the values from index \(i + 1\) to index \(j\).
- **Example**: 
  \[
  (1, (1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4))
  \]
  Figure 3.730 provides an example where the **MIN_MAX_INFLEXION** constraint holds.
- **Typical**: 
  \[
  |\text{VARIABLES}| > 2 \\
  \text{range}(\text{VARIABLES}.\text{var}) > 1
  \]
- **Symmetry**: Items of **VARIABLES** can be reversed.
- **Arg. properties**: Functional dependency: \(\text{VALUE}\) determined by **VARIABLES**.
Figure 3.730: Illustrating the MIN_MAX_INFLEXION constraint of the Example slot
Figures 3.731 and 3.732 respectively depict the automaton associated with the constraint MIN\_MAX\_INFLEXION and its simplified form.

**Figure 3.731:** Automaton for the MIN\_MAX\_INFLEXION constraint obtained by applying decoration Table 2.35 to the seed transducer of the INFLEXION pattern where \texttt{default} is $+\infty$ (transition $r \to t$ has the same accumulators updates as transition $t \to r$).

**Figure 3.732:** Automaton for the MIN\_MAX\_INFLEXION constraint obtained by applying decoration Table 2.25 to the seed transducer of the INFLEXION pattern where \texttt{default} is $+\infty$ (transition $r \to t$ has the same accumulators updates as transition $t \to r$); $-R_i + R_{i-1} \geq 0$ is a linear invariant.
MIN_MAX_PEAK

**DESCRIPTION**

**Origin**
Based on the PEAK pattern.

**Constraint**

\[
\text{MIN_MAX_PEAK} (\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection} (\text{var} - \text{dvar})
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{sv} \leq 2 \lor \text{rv} \leq 1 & \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} \geq \text{minv} + 1 & \\
\text{VALUE} = +\infty & \lor \text{VALUE} \leq \text{maxv} \\
\text{required} (\text{VARIABLES}, \text{var}) & \\
\text{where} & \\
\text{minv} & = \text{minval} (\text{VARIABLES}.\text{var}) \\
\text{maxv} & = \text{maxval} (\text{VARIABLES}.\text{var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range} (\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

VALUE is the minimum of all maximum values in each occurrence of the PEAK pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \( +\infty \).

An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression \(<(= | <)^* (>| =)^* >\). Assume that the occurrence of the pattern PEAK starts at position \(i\) and ends at position \(j\). The feature \text{MAX} computes the maximum of the values from index \(i + 1\) to index \(j\).

**Example**

\[(3, (7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 1))\]

Figure 3.733 provides an example where the MIN_MAX_PEAK \( (3, [7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 1]) \) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 2 \\
\text{range} (\text{VARIABLES}.\text{var}) > 1
\]

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.733: Illustrating the MIN_MAX_PEEK constraint of the Example slot
Figures 3.734 and 3.735 respectively depict the automaton associated with the constraint \textit{MIN\_MAX\_PEAK} and its simplified form.

Figure 3.734: Automaton for the \textit{MIN\_MAX\_PEAK} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \textit{PEAK} pattern where \textit{default} is $+\infty$.

Figure 3.735: Automaton for the \textit{MIN\_MAX\_PEAK} constraint obtained by applying decoration Table 2.37 to the seed transducer of the \textit{PEAK} pattern where \textit{default} is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.74: Glue matrix for the MIN\_MAX\_PEAK constraint defined as the composition of the PEAK pattern, the feature MAX, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>min(( \vec{C}, \vec{C} ))</td>
<td>min(( \vec{C}, \vec{C} ))</td>
<td>min(( \vec{C}, \vec{C} ))</td>
</tr>
<tr>
<td>r</td>
<td>min(( \vec{C}, \vec{C} ))</td>
<td>max(( \vec{D}, \vec{D}, \text{VAR}_i ))</td>
<td>max(( \vec{C}, \vec{D}, \vec{D}, \text{VAR}_i ))</td>
</tr>
<tr>
<td>t</td>
<td>min(( \vec{C}, \vec{C} ))</td>
<td>max(( \vec{C}, \vec{D}, \vec{D}, \text{VAR}_i ))</td>
<td>min(( \vec{C}, \vec{C} ))</td>
</tr>
</tbody>
</table>
**MIN_MAX.Strictly-Decreasing-Sequence**

**Description**
Based on the *Strictly-Decreasing-Sequence* pattern.

**Constraint**

\[
\text{MIN\_MAX\_Strictly-Decreasing-Sequence}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var−dvar)

**Restrictions**

\[
\begin{align*}
sv & \leq 1 \lor rv \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} & \geq \text{minv} + 1 \\
\text{VALUE} & = +\infty \lor \text{VALUE} \leq \text{maxv} \quad \text{required(\text{VARIABLES}, var)}
\end{align*}
\]

where

\[
\begin{align*}
\text{minv} & = \text{minval(\text{VARIABLES}.\text{var})} \\
\text{maxv} & = \text{maxval(\text{VARIABLES}.\text{var})} \\
sv & = |\text{VARIABLES}| \\
rv & = \text{range(\text{VARIABLES}.\text{var})}
\end{align*}
\]

**Purpose**

VALUE is the minimum of all maximum values in each occurrence of the *Strictly-Decreasing-Sequence* pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +\infty.

An occurrence of the pattern *Strictly-Decreasing-Sequence* is the maximal sub-sequence which matches the regular expression >+

Assume that the occurrence of the pattern *Strictly-Decreasing-Sequence* starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i\) to index \(j+1\).

**Example**

\[
(4, (4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3))
\]

Figure 3.736 provides an example where the MIN\_MAX\_Strictly-Decreasing-Sequence (4, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3]) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range(\text{VARIABLES}.\text{var})} > 1
\]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.736: Illustrating the `MIN_MAX_STRICTLY_DECREASING_SEQUENCE` constraint of the Example slot
Figures 3.737 and 3.738 respectively depict the automaton associated with the constraint MIN_MAX_STRICTLY_DECREASING_SEQUENCE and its simplified form.

\[
\begin{align*}
\{ & C \leftarrow \text{default} \\
& D \leftarrow -\infty \\
& R \leftarrow \text{default} \} \\
\end{align*}
\]

\[\leq\]

\[
\begin{align*}
\{ & C \leftarrow \text{default} \\
& D \leftarrow -\infty \\
& R \leftarrow \min(R, C) \} \\
\end{align*}
\]

\[\leq\]

\[
\begin{align*}
\{ & C \leftarrow \max(D, \text{VAR}\_i), \text{VAR}\_i + 1) \\
& D \leftarrow -\infty \} \\
\end{align*}
\]

\[\geq\]

\[
\begin{align*}
\{ & C \leftarrow \max(C, \max(D, \text{VAR}\_i + 1)) \\
& D \leftarrow -\infty \} \\
\end{align*}
\]

\[\leq\]

\[
\begin{align*}
\{ & R \leftarrow \text{default} \} \\
\end{align*}
\]

\[\leq\]

\[
\begin{align*}
\{ & R \leftarrow \min(R, \text{VAR}\_i) \} \\
\end{align*}
\]

\[\geq\]

\[
\begin{align*}
\{ & R \leftarrow \min(R, \text{VAR}\_i) \} \\
\end{align*}
\]

Figure 3.737: Automaton for the MIN_MAX_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern where default is $+\infty$.

\[
\begin{align*}
\{ & R \leftarrow \text{default} \} \\
\end{align*}
\]

\[\leq\]

\[
\begin{align*}
\{ & R \leftarrow \min(R, \text{VAR}\_i) \} \\
\end{align*}
\]

\[\geq\]

\[
\begin{align*}
\{ & R \leftarrow \min(R, \text{VAR}\_i) \} \\
\end{align*}
\]

Figure 3.738: Automaton for the MIN_MAX_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.37 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.75: Glue matrix for the MIN_MAX_STRICTLY_DECREASING_SEQUENCE constraint defined as the composition of the "STRICTLY_DECREASING_SEQUENCE pattern", the feature MAX, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>min((\overrightarrow{C}, \overrightarrow{C} ))</td>
<td>min((\overrightarrow{C}, \overrightarrow{C} ))</td>
</tr>
<tr>
<td>r</td>
<td>min((\overrightarrow{C}, \overrightarrow{C} ))</td>
<td>max((\overrightarrow{C}, \overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}, \text{VAR}_i ))</td>
</tr>
</tbody>
</table>
### 3.326 MIN_MAX_STRICTLY_INCREASING_SEQUENCE

**Origin**
Based on the `STRICTLY_INCREASING_SEQUENCE` pattern.

**Constraint**
`MIN_MAX_STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES)`

**Arguments**
- **VALUE**: `dvar`
- **VARIABLES**: `collection(var-dvar)`

**Restrictions**
- $sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty$
- $VALUE \geq minv + 1$
- $VALUE = +\infty \lor VALUE \leq maxv$
- $required(VARIABLES.var)$

where
- $minv = minval(VARIABLES.var)$
- $maxv = maxval(VARIABLES.var)$
- $sv = |VARIABLES|$
- $rv = range(VARIABLES.var)$

**Purpose**
VALUE is the minimum of all maximum values in each occurrence of the `STRICTLY_INCREASING_SEQUENCE` pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value $+\infty$.

An occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` is the **maximal** sub-sequence which matches the regular expression `<+`.

Assume that the occurrence of the pattern `STRICTLY_INCREASING_SEQUENCE` starts at position $i$ and ends at position $j$. The feature MAX computes the maximum of the values from index $i$ to index $j + 1$.

**Example**

```
(3, (4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3))
```

Figure 3.739 provides an example where the `MIN_MAX_STRICTLY_INCREASING_SEQUENCE` constraint holds.

**Typical**

```
|VARIABLES| > 1
range(VARIABLES.var) > 1
```

**Arg. properties**
- **Functional dependency**: VALUE determined by VARIABLES.
Figure 3.739: Illustrating the MIN_MAX_STRINGLY_INCREASING_SEQUENCE constraint of the Example slot.
Figures 3.740 and 3.741 respectively depict the automaton associated with the constraint MIN_MAX STRICTLY_INCREASING_SEQUENCE and its simplified form.

Figure 3.740: Automaton for the MIN_MAX STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern where default is $+\infty$.

Figure 3.741: Automaton for the MIN_MAX STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.24 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
<table>
<thead>
<tr>
<th></th>
<th>( s )</th>
<th>( r )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>( \min(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \min(\overrightarrow{C}, \overrightarrow{C}) )</td>
</tr>
<tr>
<td>( r )</td>
<td>( \min(\overrightarrow{C}, \overrightarrow{C}) )</td>
<td>( \max(\overrightarrow{C}, \overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}, \text{VAR}_i) )</td>
</tr>
</tbody>
</table>

Table 3.76: Glue matrix for the \textsc{min\_max\_strictly\_increasing\_sequence} constraint defined as the composition of the \textsc{strictly\_increasing\_sequence} pattern, the \textsc{feature \textsc{max}}, and the \textsc{aggregator \textsc{min}}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### MIN_MAX_SUMMIT

**Description**

Based on the SUMMIT pattern.

**Constraint**

\[
\text{MIN\_MAX\_SUMMIT}((\text{VALUE}, \text{VARIABLES}))
\]

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var–dvar)

**Restrictions**

\[
\begin{align*}
sv \leq 2 \lor rv \leq 1 & \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} \geq minv + 1 & \\
\text{VALUE} = +\infty \lor \text{VALUE} \leq maxv & \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{where} & \\
minv &= \text{minval}(\text{VARIABLES}.\text{var}) \\
maxv &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
sv &= |\text{VARIABLES}| \\
rv &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

VALUE is the minimum of all maximum values in each occurrence of the SUMMIT pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +∞.

An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression \((< | < (= | <)^* <)(> | > (= | >)^*>)\).

Assume that the occurrence of the pattern SUMMIT starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i + 1\) to index \(j\).

**Example**

\[(3, (7, 1, 5, 4, 4, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1))\]

Figure 3.742 provides an example where the MIN\_MAX\_SUMMIT\((3, [7, 1, 5, 4, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1])\) constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 2\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.742: Illustrating the MIN_MAX_SUMMIT constraint of the Example slot
Figures 3.743 and 3.744 respectively depict the automaton associated with the constraint MIN_MAX_SUMMIT and its simplified form.

Figure 3.743: Automaton for the MIN_MAX_SUMMIT constraint obtained by applying decoration Table 2.35 to the seed transducer of the SUMMIT pattern where default is +∞ (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$)

Table 3.77: Glue matrix for the MIN_MAX_SUMMIT constraint defined as the composition of the SUMMIT pattern, the feature MAX, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

```
<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
<th>u</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>min((\overline{C}, \overline{C}))</td>
</tr>
<tr>
<td>r</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>max((\overline{D}, \overline{D}, \text{VAR}_{i}))</td>
<td>max((\overline{C}, \overline{D}, \overline{D}, \text{VAR}_{i}))</td>
<td>min((\overline{C}, \overline{C}))</td>
</tr>
<tr>
<td>t</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>max((\overline{C}, \overline{D}, \overline{D}, \text{VAR}_{i}))</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>max((\overline{C}, \overline{D}, \overline{D}, \text{VAR}_{i}))</td>
</tr>
<tr>
<td>u</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>max((\overline{C}, \overline{D}, \overline{D}, \text{VAR}_{i}))</td>
<td>min((\overline{C}, \overline{C}))</td>
</tr>
</tbody>
</table>
```
Figure 3.744: Automaton for the MIN_MAX_SUMMIT constraint obtained by applying decoration Table 2.37 to the seed transducer of the SUMMIT pattern where default is $+\infty$ (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$); $-R_i + R_{i-1} \geq 0$ is a linear invariant.
3.328 MIN_MAX_ZIGZAG

DESCRIPTION AUTOMATON

Origin
Based on the ZIGZAG pattern.

Constraint
MIN_MAX_ZIGZAG(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var-dvar)

Restrictions
\[ sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = +\infty \]
VALUE \geq minv + 1
VALUE = +\infty \lor VALUE \leq maxv
\text{required} (VARIABLES, var)
where
minv = minval (VARIABLES.var)
maxv = maxval (VARIABLES.var)
sv = |VARIABLES|
rv = range (VARIABLES.var)

VALUE is the minimum of all maximum values in each occurrence of the ZIGZAG pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(+\infty\).

Purpose
An occurrence of the pattern ZIGZAG is the \textit{maximal} subsequence which matches the regular expression \((<>)^+ (<> | <>) | (>><>)^+ (> | >>)\).
Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i+1\) to index \(j\).

Example
\[(3, (4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1))\]

Figure 3.745 provides an example where the MIN_MAX_ZIGZAG (3, [4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1]) constraint holds.

Typical
|VARIABLES| > 3
range (VARIABLES.var) > 1

Symmetry
Items of VARIABLES can be reversed.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 3.745: Illustrating the MIN_MAX_ZIGZAG constraint of the Example slot
Automaton

Figures 3.746 and 3.747 respectively depict the automaton associated with the constraint MIN_MAX_ZIGZAG and its simplified form.
Figure 3.746: Automaton for the MIN_MAX_ZIGZAG constraint obtained by applying decoration Table 2.35 to the seed transducer of the ZIGZAG pattern where default is $+\infty$: (1) missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=$; (2) on transitions from $b, c, e, f$ to $s$ the accumulator $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the accumulator $R$ is updated wrt $C$ and the accumulator $C$ is reset to its initial value.
Figure 3.747: Automaton for the MIN_MAX_ZIGZAG constraint obtained by applying decoration Table 2.23 to the seed transducer of the ZIGZAG pattern where default is $+\infty$; missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=$; (2) on transitions from $b, c, e, f$ to $s$ the accumulator $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the accumulator $R$ is updated wrt $C$ and the accumulator $C$ is reset to its initial value.; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.78: Glue matrix for the MIN-MAX-ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature MAX, and the aggregator min. Cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>min((\vec{C}, \vec{C}))</td>
</tr>
<tr>
<td>a</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>max(\vec{C}, \vec{D}, \vec{D}, \text{VAR}_i)</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>max((\vec{D}, \vec{D}, \text{VAR}_i))</td>
<td>min((\vec{C}, \vec{C}))</td>
</tr>
<tr>
<td>b</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>max((\vec{D}, \vec{D}, \text{VAR}_i))</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>max((\vec{D}, \vec{D}, \text{VAR}_i))</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>max((\vec{C}, \vec{D}, \vec{D}, \text{VAR}_i))</td>
</tr>
<tr>
<td>c</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>max((\vec{C}, \vec{D}, \vec{D}, \text{VAR}_i))</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>max((\vec{C}, \vec{D}, \vec{D}, \text{VAR}_i))</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>max((\vec{C}, \vec{D}, \vec{D}, \text{VAR}_i))</td>
<td>min((\vec{C}, \vec{C}))</td>
</tr>
<tr>
<td>d</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>max((\vec{D}, \vec{D}, \text{VAR}_i))</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>max((\vec{C}, \vec{D}, \vec{D}, \text{VAR}_i))</td>
<td>min((\vec{C}, \vec{C}))</td>
</tr>
<tr>
<td>e</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>max((\vec{D}, \vec{D}, \text{VAR}_i))</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>max((\vec{D}, \vec{D}, \text{VAR}_i))</td>
<td>min((\vec{C}, \vec{C}))</td>
</tr>
<tr>
<td>f</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>max((\vec{D}, \vec{D}, \text{VAR}_i))</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>max((\vec{D}, \vec{D}, \text{VAR}_i))</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>min((\vec{C}, \vec{C}))</td>
</tr>
</tbody>
</table>
3.329 MIN_MIN_BUMP_ON_DECREASING_SEQUENCE

Origin
Based on the BUMP_ON_DECREASING_SEQUENCE pattern.

Constraint
MIN_MIN_BUMP_ON_DECREASING_SEQUENCE(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
\[ sv \leq 5 \lor rv \leq 2 \Rightarrow VALUE = +\infty \]
VALUE \geq minv
VALUE = +\infty \lor VALUE \leq maxv − 2
required(VARIABLES, var)

where
\[ minv = \text{minval}(\text{VARIABLES}.\text{var}) \]
\[ maxv = \text{maxval}(\text{VARIABLES}.\text{var}) \]
\[ sv = |\text{VARIABLES}| \]
\[ rv = \text{range}(\text{VARIABLES}.\text{var}) \]

VALUE is the minimum of all minimum values in each occurrence of the BUMP_ON_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +\infty.

Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature MIN computes the minimum of the values from index \( i + 2 \) to index \( j \).

Purpose

Example
\[ (2, (7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 3, 5, 4, 3, 3)) \]

Figure 3.748 provides an example where the MIN_MIN_BUMP_ON_DECREASING_SEQUENCE constraint holds.

Typical
\[ |\text{VARIABLES}| > 5 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 2 \]

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 3.748: Illustrating the MIN_MIN_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot
Figures 3.749 and 3.750 respectively depict the automaton associated with the constraint \texttt{MIN_MIN_BUMP_ON_DECREASING_SEQUENCE} and its simplified form.

Figure 3.749: Automaton for the \texttt{MIN_MIN_BUMP_ON_DECREASING_SEQUENCE} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{BUMP_ON_DECREASING_SEQUENCE} pattern where \texttt{default} is $+\infty$.
Figure 3.750: Automaton for the MIN_MIN_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.25 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
MIN_MIN_DECREASING

**Description**

**Origin**
Based on the **DECREASING** pattern.

**Constraint**
MIN_MIN_DECREASING(VALUE, VARIABLES)

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>dvar</td>
</tr>
<tr>
<td>VARIABLES</td>
<td>collection(var−dvar)</td>
</tr>
</tbody>
</table>

**Restrictions**

\[
sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty \\
VALUE \geq minv \\
VALUE = +\infty \lor VALUE \leq maxv - 1 \\
\text{required}(\text{VARIABLES}, \text{var})
\]

where

\[
\begin{align*}
\text{minv} &= \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{maxv} &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
sv &= |\text{VARIABLES}| \\
rv &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

VALUE is the minimum of all minimum values in each occurrence of the **DECREASING** pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value **+\infty**.

An occurrence of the pattern **DECREASING** is the subsequence which matches the regular expression >.

Assume that the occurrence of the pattern **DECREASING** starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i\) to index \(j + 1\).

**Example**

\[(1, (3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))\]

Figure 3.751 provides an example where the MIN_MIN_DECREASING (1, [3, 4, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4]) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
range(\text{VARIABLES}.\text{var}) > 1
\]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.751: Illustrating the MIN_MIN_DECREASING constraint of the Example slot
Automaton

Figures 3.752 and 3.753 respectively depict the automaton associated with the constraint MIN_MIN_DECREASING and its simplified form.

\[
\begin{align*}
\{ C & \leftarrow \text{default} \\
D & \leftarrow +\infty \\
R & \leftarrow \text{default} \} \\
\{ D & \leftarrow +\infty \\
R & \leftarrow \min(R, \min(min(D, \text{VAR}_i), \text{VAR}_{i+1})) \} \\
\end{align*}
\]

Figure 3.752: Automaton for the MIN_MIN_DECREASING constraint obtained by applying decoration Table 2.35 to the seed transducer of the DECREASING pattern where default is $+\infty$.

\[
\{ R \leftarrow \text{default} \} \\
\{ R \leftarrow \min(R, \min(\text{VAR}_i, \text{VAR}_{i+1})) \} \\
\]

Figure 3.753: Automaton for the MIN_MIN_DECREASING constraint obtained by applying decoration Table 2.38 to the seed transducer of the DECREASING pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.

Table 3.79: Glue matrix for the MIN_MIN_DECREASING constraint defined as the composition of the DECREASING pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### Description

**Origin**

Based on the `DECREASING_SEQUENCE` pattern.

**Constraint**

`MIN_MIN_DECREASING_SEQUENCE(VALUE, VARIABLES)`

**Arguments**

```
VALUE : dvar
VARIABLES : collection(var−dvar)
```

**Restrictions**

\[
sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty
\]

\[
VALUE \geq minv
\]

\[
VALUE = +\infty \lor VALUE \leq maxv - 1
\]

\[
\text{required}(\text{VARIABLES, var})
\]

where

\[
\text{minv} = \text{minval}(\text{VARIABLES}.\text{var})
\]

\[
\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})
\]

\[
sv = |\text{VARIABLES}|
\]

\[
rv = \text{range}(\text{VARIABLES}.\text{var})
\]

**Purpose**

VALUE is the minimum of all minimum values in each occurrence of the `DECREASING_SEQUENCE` pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(+\infty\).

An occurrence of the pattern `DECREASING_SEQUENCE` is the maximal subsequence which matches the regular expression \(>(>|=)^*>|\).

Assume that the occurrence of the pattern `DECREASING_SEQUENCE` starts at position \(i\) and ends at position \(j\). The feature `MIN` computes the minimum of the values from index \(i\) to index \(j + 1\).

**Example**

\[(1, (3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))\]

Figure 3.754 provides an example where the `MIN_MIN_DECREASING_SEQUENCE` \((1, [3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4])\) constraint holds.

**Typical**

\[|\text{VARIABLES}| > 1\]

\[range(\text{VARIABLES}.\text{var}) > 1\]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.754: Illustrating the MIN_MIN_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 3.755 and 3.756 respectively depict the automaton associated with the constraint MIN_MIN_DECREASING_SEQUENCE and its simplified form.

Figure 3.755: Automaton for the MIN_MIN_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the DECREASING_SEQUENCE pattern where default is $+\infty$.

Figure 3.756: Automaton for the MIN_MIN_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.38 to the seed transducer of the DECREASING_SEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.80: Glue matrix for the `MIN_MIN_DECREASING_SEQUENCE` constraint defined as the composition of the `DECREASING_SEQUENCE` pattern, the feature `MIN`, and the aggregator `min`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>( \min(\vec{C}, \vec{C}) )</td>
<td>( \min(\vec{C}, \vec{C}) )</td>
</tr>
<tr>
<td>t</td>
<td>( \min(\vec{C}, \vec{C}) )</td>
<td>( \min(\vec{C}, \vec{C}, \vec{D}, \vec{D}, \text{VAR}_i) )</td>
</tr>
</tbody>
</table>
3.332 MIN_MIN_DIP_ON_INCREASING_SEQUENCE

**DESCRIPTION**

**Origin**
Based on the DIP_ON_INCREASING_SEQUENCE pattern.

**Constraint**
MIN_MIN_DIP_ON_INCREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var−dvar)

**Restrictions**
- \( sv \leq 5 \lor rv \leq 2 \Rightarrow VALUE = +\infty \)
- \( VALUE \geq minv \)
- \( VALUE = +\infty \lor VALUE \leq maxv - 2 \)

\[
\text{required}(\text{VARIABLES}, \text{var})
\]

where
- \( minv = \minval(\text{VARIABLES}.\text{var}) \)
- \( maxv = \maxval(\text{VARIABLES}.\text{var}) \)
- \( sv = |\text{VARIABLES}| \)
- \( rv = \range(\text{VARIABLES}.\text{var}) \)

**Purpose**
VALUE is the minimum of all minimum values in each occurrence of the DIP_ON_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \( +\infty \).

An occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence which matches the regular expression \( <<><< \).

Assume that the occurrence of the pattern DIP_ON_INCREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature MIN computes the minimum of the values from index \( i + 2 \) to index \( j \).

**Example**
\[(1, (1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4))\]

Figure 3.757 provides an example where the MIN_MIN_DIP_ON_INCREASING_SEQUENCE (1, [1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4]) constraint holds.

**Typical**
- \( |\text{VARIABLES}| > 5 \)
- \( \range(\text{VARIABLES}.\text{var}) > 2 \)

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.757: Illustrating the MIN_MIN_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 3.758 and 3.759 respectively depict the automaton associated with the constraint MIN_MIN_DIP_ON_INCREASING_SEQUENCE and its simplified form.

Figure 3.758: Automaton for the MIN_MIN_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern where default is $+\infty$
Figure 3.759: Automaton for the MIN_MIN_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.27 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
3.333 MIN_MIN_GORGE

**DESCRIPTION**

- **Origin**: Based on the GORGE pattern.
- **Constraint**: MIN_MIN_GORGE(VALUE, VARIABLES)
- **Arguments**:
  - VALUE: dvar
  - VARIABLES: collection(var−dvar)
- **Restrictions**:
  - $sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = +\infty$
  - $VALUE \geq \min v$
  - $VALUE = +\infty \lor VALUE \leq \max v - 1$
  - required(VARIABLES, var)
  
  where
  - $\min v = \minval(VARIABLES.var)$
  - $\max v = \maxval(VARIABLES.var)$
  - $sv = |VARIABLES|$
  - $rv = \text{range}(VARIABLES.var)$

**Purpose**

An occurrence of the pattern GORGE is the *maximal* subsequence which matches the regular expression $(> | > (= | >)^* >)<|< (= | <)^* <)$. Assume that the occurrence of the pattern GORGE starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i + 1$ to index $j$.

**Example**

$$(3, (1, 7, 3, 4, 4, 5, 4, 2, 6, 5, 4, 6, 5, 7))$$

Figure 3.760 provides an example where the MIN_MIN_GORGE ($3, [1, 7, 3, 4, 4, 5, 4, 2, 6, 5, 4, 6, 5, 7]$) constraint holds.

**Typical**

- $|\text{VARIABLES}| > 2$
- $\text{range}(\text{VARIABLES}.\text{var}) > 1$

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.760: Illustrating the MIN_MIN_GORGE constraint of the Example slot
Figures 3.761 and 3.762 respectively depict the automaton associated with the constraint MIN_MIN_GORGE and its simplified form.

Figure 3.761: Automaton for the MIN_MIN_GORGE constraint obtained by applying decoration Table 2.35 to the seed transducer of the GORGE pattern where default is $+\infty$ (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$).

Table 3.81: Glue matrix for the MIN_MIN_GORGE constraint defined as the composition of the GORGE pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 3.762: Automaton for the MIN_MIN_GORGE constraint obtained by applying decoration Table 2.37 to the seed transducer of the GORGE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
### 3.334 MIN_MIN_INCREASING

**DESCRIPTION**

#### Origin
Based on the **INCREASING** pattern.

#### Constraint

<table>
<thead>
<tr>
<th>VALUE</th>
<th>VARIABLES</th>
</tr>
</thead>
<tbody>
<tr>
<td>: dvar</td>
<td>: collection(var−dvar)</td>
</tr>
</tbody>
</table>

#### Restrictions

\[
\begin{align*}
sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty \\
VALUE \geq minv \\
VALUE = +\infty \lor VALUE \leq maxv - 1 \\
\text{required(VARIABLES, var)}
\end{align*}
\]

where

\[
\begin{align*}
minv &= \text{minval(VARIABLES.var)} \\
maxv &= \text{maxval(VARIABLES.var)} \\
sv &= |VARIABLES| \\
rv &= \text{range(VARIABLES.var)}
\end{align*}
\]

**Purpose**

VALUE is the minimum of all minimum values in each occurrence of the INCREASING pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +∞.

An occurrence of the pattern **INCREASING** is the subsequence which matches the regular expression <.

Assume that the occurrence of the pattern **INCREASING** starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i\) to index \(j+1\).

#### Example

\[(1,(4,3,5,5,2,1,1,3,3,4,6,6,3,1,3,3))\]

Figure 3.763 provides an example where the MIN_MIN_INCREASING constraint holds.

#### Typical

\[
\begin{align*}
|VARIABLES| > 1 \\
\text{range(VARIABLES.var)} > 1
\end{align*}
\]

#### Arg. properties

**Functional dependency:** VALUE determined by VARIABLES.
Figure 3.763: Illustrating the \texttt{MIN\_MIN\_INCREASING} constraint of the \texttt{Example} slot
Figures 3.764 and 3.765 respectively depict the automaton associated with the constraint MIN_MIN_INCREASING and its simplified form.

\[
\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow +\infty \\
R &\leftarrow \text{default}
\end{align*}
\]

\[
\begin{align*}
\{ D &\leftarrow +\infty \} \\
\{ R &\leftarrow \min(R, \min(D, \text{VAR}_i), \text{VAR}_{i+1}) \}
\end{align*}
\]

Figure 3.764: Automaton for the MIN_MIN_INCREASING constraint obtained by applying decoration Table 2.35 to the seed transducer of the INCREASING pattern where \text{default} is \(+\infty\)

\[
\begin{align*}
\{ R &\leftarrow \text{default} \}
\end{align*}
\]

\[
\begin{align*}
\{ R &\leftarrow \min(R, \text{VAR}_i) \}
\end{align*}
\]

Figure 3.765: Automaton for the MIN_MIN_INCREASING constraint obtained by applying decoration Table 2.37 to the seed transducer of the INCREASING pattern where \text{default} is \(+\infty\); \(-R_i + R_{i-1} \geq 0\) is a linear invariant.

Table 3.82: Glue matrix for the MIN_MIN_INCREASING constraint defined as the composition of the INCREASING pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.335  MIN_MIN_INCREASING_SEQUENCE

**Origin**  
Based on the INCREASING_SEQUENCE pattern.

**Constraint**  
MIN_MIN_INCREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**  
VALUE : dvar  
VARIABLES : collection(var−dvar)

**Restrictions**  
\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty \]
\[ VALUE \geq minv \]
\[ VALUE = +\infty \lor VALUE \leq maxv - 1 \]
\[ \text{required}(\text{VARIABLES}, \text{var}) \]

where
\[ minv = \text{minval}(\text{VARIABLES}.\text{var}) \]
\[ maxv = \text{maxval}(\text{VARIABLES}.\text{var}) \]
\[ sv = |\text{VARIABLES}| \]
\[ rv = \text{range}(\text{VARIABLES}.\text{var}) \]

**Purpose**  
VALUE is the minimum of all minimum values in each occurrence of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +∞.  
An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression \(< (< | =)^* < | <\).  
Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i\) to index \(j + 1\).

**Example**  
\( (1, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3)) \)

Figure 3.766 provides an example where the MIN_MIN_INCREASING_SEQUENCE (\(1, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3)\)) constraint holds.

**Typical**  
\[ |\text{VARIABLES}| > 1 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

**Arg. properties**  
Functional dependency: VALUE determined by VARIABLES.
Figure 3.766: Illustrating the MIN_MIN_INCREASING_SEQUENCE constraint of the Example slot
Figures 3.767 and 3.768 respectively depict the automaton associated with the constraint MIN_MIN_INCREASING_SEQUENCE and its simplified form.

Figure 3.767: Automaton for the MIN_MIN_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the INCREASING_SEQUENCE pattern where default is $+\infty$.

Figure 3.768: Automaton for the MIN_MIN_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.37 to the seed transducer of the INCREASING_SEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.83: Glue matrix for the MIN\_MIN\_INCREASING\_SEQUENCE constraint defined as the composition of the INCREASING\_SEQUENCE pattern, the feature MIN, and the aggregator min: cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>\min(\overrightarrow{C}, \overrightarrow{C})</td>
<td>\min(\overrightarrow{C}, \overrightarrow{C})</td>
</tr>
<tr>
<td>t</td>
<td>\min(\overrightarrow{C}, \overrightarrow{C})</td>
<td>\min(\overrightarrow{C}, \overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}, \text{VAR})</td>
</tr>
</tbody>
</table>
### 3.336 MIN_MIN_INFLEXION

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Origin</strong></td>
<td>Based on the INFLEXION pattern.</td>
</tr>
<tr>
<td><strong>Constraint</strong></td>
<td>MIN_MIN_INFLEXION(VALUE, VARIABLES)</td>
</tr>
</tbody>
</table>
| **Arguments** | VALUE : dvar  
VARIABLES : collection(var−dvar) |
| **Restrictions** | \( sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = +\infty \)  
VALUE \( \geq \) minv  
VALUE \( = +\infty \lor VALUE \leq \) maxv  
required(VARIABLES, var)  
where  
minv = minval(VARIABLES.var)  
maxv = maxval(VARIABLES.var)  
sv = |VARIABLES|  
rv = range(VARIABLES.var) |
| **Purpose** | VALUE is the minimum of all minimum values in each occurrence of the INFLEXION pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(+\infty\).  
An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression \(< (< | =)* > | > | =)* <\).  
Assume that the occurrence of the pattern INFLEXION starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\). |
| **Example** | \((1, (1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4))\) |

Figure 3.769 provides an example where the MIN_MIN_INFLEXION \((1, [1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4])\) constraint holds.

| **Typical** | \(|\text{VARIABLES}| > 2\)  
\(\text{range}(\text{VARIABLES}.\text{var}) > 1\) |
| **Symmetry** | Items of VARIABLES can be reversed. |
| **Arg. properties** | Functional dependency: VALUE determined by VARIABLES. |
Figure 3.769: Illustrating the MIN_MIN_INFLEXION constraint of the Example slot
Figures 3.770 and 3.771 respectively depict the automaton associated with the constraint MIN_MIN_INFLEXION and its simplified form.

Figure 3.770: Automaton for the MIN_MIN_INFLEXION constraint obtained by applying decoration Table 2.35 to the seed transducer of the INFLEXION pattern where default is $+\infty$ (transition $r \rightarrow t$ has the same accumulators updates as transition $t \rightarrow r$).

Figure 3.771: Automaton for the MIN_MIN_INFLEXION constraint obtained by applying decoration Table 2.25 to the seed transducer of the INFLEXION pattern where default is $+\infty$ (transition $r \rightarrow t$ has the same accumulators updates as transition $t \rightarrow r$); $-R_i + R_{i-1} \geq 0$ is a linear invariant.
3.337  MIN_MIN_STRICKLY_DECREASING_SEQUENCE

DESCRIPTION

Based on the Strictly Decreasing Sequence pattern.

Constraint

\[
\text{MIN_MIN_STRICKLY_DECREASING_SEQUENCE}(\text{VALUE, VARIABLES})
\]

Arguments

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var}-\text{dvar})
\end{align*}
\]

Restrictions

\[
\begin{align*}
sv \leq 1 \land rv \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} \geq \minv \\
\text{VALUE} = +\infty \lor \text{VALUE} \leq \maxv - 1 \\
\text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where

\[
\begin{align*}
\minv & = \text{minval}(\text{VARIABLES}.\text{var}) \\
\maxv & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
sv & = |\text{VARIABLES}| \\
rv & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

VALUE is the minimum of all minimum values in each occurrence of the Strictly Decreasing Sequence pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(+\infty\).

Purpose

An occurrence of the pattern Strictly Decreasing Sequence is the maximal subsequence which matches the regular expression \(>^+\).

Assume that the occurrence of the pattern Strictly Decreasing Sequence starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i\) to index \(j + 1\).

Example

\[
(1, (4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 5, 2, 2, 4, 3))
\]

Figure 3.772 provides an example where the \(\text{MIN_MIN_STRICKLY_DECREASING_SEQUENCE}\) constraint holds.

Typical

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 3.772: Illustrating the \texttt{MIN_MIN\_STRICTLY\_DECREASING\_SEQUENCE} constraint of the \textbf{Example} slot
Automaton

Figures 3.773 and 3.774 respectively depict the automaton associated with the constraint MIN_MIN_STRICTLY_DECREASING_SEQUENCE and its simplified form.

\[
\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow +\infty \\
R &\leftarrow \text{default}
\end{align*}
\]

Figure 3.773: Automaton for the MIN_MIN_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern where default is +\infty

\[
\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow +\infty \\
R &\leftarrow \min(R, C)
\end{align*}
\]

\[
\begin{align*}
C &\leftarrow \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
D &\leftarrow +\infty
\end{align*}
\]

\[
\begin{align*}
C &\leftarrow \min(C, \min(D, \text{VAR}_{i+1})) \\
D &\leftarrow +\infty
\end{align*}
\]

\[
\begin{align*}
R &\leftarrow \text{default}
\end{align*}
\]

\[
\begin{align*}
R &\leftarrow \min(R, \text{VAR}_i, \text{VAR}_{i+1})
\end{align*}
\]

\[
\begin{align*}
R &\leftarrow \min(R, \text{VAR}_{i+1})
\end{align*}
\]

Figure 3.774: Automaton for the MIN_MIN_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.38 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern where default is +\infty; \(-R_i + R_{i-1} \geq 0\) is a linear invariant.
Table 3.84: Glue matrix for the MIN_MIN_STRICTLY_DECREASING_SEQUENCE constraint defined as the composition of the STRICTLY_DECREASING_SEQUENCE pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### 3.338 MIN_MIN_STRICTLY_INCREASING_SEQUENCE

**Description**

Based on the **STRICTLY_INCREASING_SEQUENCE** pattern.

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**

\[ sv ≤ 1 \lor rv ≤ 1 \Rightarrow VALUE = +\infty \]

VALUE \( \geq \minv \)

VALUE \( = +\infty \lor VALUE \leq \maxv - 1 \)

**Purpose**

- **VALUE** is the minimum of all minimum values in each occurrence of the **STRICTLY_INCREASING_SEQUENCE** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, **VALUE** takes the default value \(+\infty\).

An occurrence of the pattern **STRICTLY_INCREASING_SEQUENCE** is the **maximal** sub-sequence which matches the regular expression \(<^+\).

Assume that the occurrence of the pattern **STRICTLY_INCREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **MIN** computes the minimum of the values from index \(i\) to index \(j + 1\).

**Example**

\[ (1, (4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3)) \]

Figure 3.775 provides an example where the **MIN_MIN_STRICTLY_INCREASING_SEQUENCE** (1, [4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3]) constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 1\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Arg. properties**

**Functional dependency**: **VALUE** determined by **VARIABLES**.

**Automaton**

Diagram with states and transitions indicating the automaton's behavior.
Figure 3.775: Illustrating the MIN_MIN_STRICTLY_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 3.776 and 3.777 respectively depict the automaton associated with the constraint MIN_MIN.Strictly_Increasing_Sequence and its simplified form.

\[
\begin{align*}
\{ & C \leftarrow \text{default} \\
& D \leftarrow +\infty \\
& R \leftarrow \text{default} \\
\} \\
\begin{array}{c}
\geq s \\
\end{array}
\end{align*}
\]

\[
\begin{align*}
\{ & C \leftarrow \text{default} \\
& D \leftarrow +\infty \\
& R \leftarrow \min(R, C) \\
\} \\
\begin{array}{c}
\geq \\
\end{array}
\end{align*}
\]

\[
\begin{align*}
\begin{array}{c}
\geq s \\
\end{array}
\end{align*}
\]

\[
\begin{align*}
\{ & C \leftarrow \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}) \\
& D \leftarrow +\infty \\
\} \\
\begin{array}{c}
\leq \\
\end{array}
\end{align*}
\]

\[
\begin{align*}
\begin{array}{c}
\leq \\
\end{array}
\end{align*}
\]

\[
\begin{align*}
\{ & C \leftarrow \min(C, \min(D, \text{VAR}_{i+1})) \\
& D \leftarrow +\infty \\
\} \\
\begin{array}{c}
\geq \\
\leq \\
\end{array}
\end{align*}
\]

\[
\begin{align*}
\{ & R \leftarrow \text{default} \\
\} \\
\begin{array}{c}
\geq \\
\end{array}
\end{align*}
\]

\[
\begin{align*}
\begin{array}{c}
\geq s \\
\end{array}
\end{align*}
\]

\[
\begin{align*}
\{ & R \leftarrow \min(R, \text{VAR}_i) \\
\} \\
\begin{array}{c}
\leq \\
\end{array}
\end{align*}
\]

\[
\begin{align*}
\begin{array}{c}
\leq \\
\end{array}
\end{align*}
\]

\[
\begin{align*}
\begin{array}{c}
\leq \\
\end{array}
\end{align*}
\]

Figure 3.776: Automaton for the MIN_MIN.Strictly_Increasing_Sequence constraint obtained by applying decoration Table 2.35 to the seed transducer of the Strictly_Increasing_Sequence pattern where default is +\infty.

Figure 3.777: Automaton for the MIN_MIN.Strictly_Increasing_Sequence constraint obtained by applying decoration Table 2.37 to the seed transducer of the Strictly_Increasing_Sequence pattern where default is +\infty; \(-R_i + R_{i-1} \geq 0\) is a linear invariant.
Table 3.85: Glue matrix for the MIN_MIN_STRICTLY_INCREASING_SEQUENCE constraint defined as the composition of the STRICTLY_INCREASING_SEQUENCE pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\min(\overrightarrow{C}, \overleftarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overleftarrow{C})$</td>
</tr>
<tr>
<td>r</td>
<td>$\min(\overrightarrow{C}, \overleftarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overleftarrow{C}, \overrightarrow{D}, \overleftarrow{D}, \text{VAR}_i)$</td>
</tr>
</tbody>
</table>
### MIN_MIN_VALLEY

#### Description

**Origin**
Based on the **VALLEY** pattern.

**Constraint**

\[
\text{MIN\_MIN\_VALLEY}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var–dvar)

**Restrictions**

\[
\begin{align*}
\text{sv} &\leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} &\geq \text{minv} \\
\text{VALUE} &\geq +\infty \lor \text{VALUE} \leq \text{maxv} - 1 \\
\text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where

- \( \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \)
- \( \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \)
- \( \text{sv} = |\text{VARIABLES}| \)
- \( \text{rv} = \text{range}(\text{VARIABLES}.\text{var}) \)

**Purpose**

**VALUE** is the minimum of all minimum values in each occurrence of the **VALLEY** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, **VALUE** takes the default value +\(\infty\).

An occurrence of the pattern **VALLEY** is the maximal subsequence which matches the regular expression \( > (= | >) (| | =)^* < \).

Assume that the occurrence of the pattern **VALLEY** starts at position \( i \) and ends at position \( j \). The feature **MIN** computes the minimum of the values from index \( i + 1 \) to index \( j \).

**Example**

\((2, (1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 7))\)

Figure 3.778 provides an example where the **MIN\_MIN\_VALLEY** \((2, [1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 7])\) constraint holds.

**Typical**

\(|\text{VARIABLES}| > 2 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Symmetry**

Items of **VARIABLES** can be reversed.

**Arg. properties**

Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 3.778: Illustrating the MIN_MIN_VALLEY constraint of the Example slot
Automaton

Figures 3.779 and 3.780 respectively depict the automaton associated with the constraint MIN_MIN_VALLEY and its simplified form.

Figure 3.779: Automaton for the MIN_MIN_VALLEY constraint obtained by applying decoration Table 2.35 to the seed transducer of the VALLEY pattern where default is +∞.

Figure 3.780: Automaton for the MIN_MIN_VALLEY constraint obtained by applying decoration Table 2.37 to the seed transducer of the VALLEY pattern where default is +∞; \(-R_i + R_{i-1} \geq 0\) is a linear invariant.
Table 3.86: Glue matrix for the MIN_MIN_VALLEY constraint defined as the composition of the VALLEY pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\min(\overrightarrow{C}, \overleftarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overleftarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overleftarrow{C})$</td>
</tr>
<tr>
<td>r</td>
<td>$\min(\overrightarrow{C}, \overleftarrow{C})$</td>
<td>$\min(\overrightarrow{D}, \overleftarrow{D}, \text{VAR}_i)$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{D}, \overleftarrow{D}, \text{VAR}_i)$</td>
</tr>
<tr>
<td>t</td>
<td>$\min(\overrightarrow{C}, \overleftarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{D}, \overleftarrow{D}, \text{VAR}_i)$</td>
<td>$\min(\overrightarrow{C}, \overleftarrow{C})$</td>
</tr>
</tbody>
</table>
### 3.340 MIN_MIN_ZIGZAG

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
</table>

#### Origin
Based on the ZIGZAG pattern.

#### Constraint
\[
\text{MIN_MIN_ZIGZAG}(\text{VALUE}, \text{VARIABLES})
\]

#### Arguments
- \( \text{VALUE} : \text{dvar} \)
- \( \text{VARIABLES} : \text{collection}(\text{var} \rightarrow \text{dvar}) \)

#### Restrictions
\[
\begin{align*}
sv & \leq 3 \land rv \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} & \geq \minv \\
\text{VALUE} & = +\infty \lor \text{VALUE} \leq \maxv - 1 \\
\text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where
- \( \minv = \text{minval}(\text{VARIABLES}.\text{var}) \)
- \( \maxv = \text{maxval}(\text{VARIABLES}.\text{var}) \)
- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

#### Purpose
\( \text{VALUE} \) is the minimum of all minimum values in each occurrence of the ZIGZAG pattern in the time-series given by the \( \text{VARIABLES} \) collection. If the pattern does not occur, \( \text{VALUE} \) takes the default value \( +\infty \).

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression \((<>)^+(< | <>)(<>)^+(>|<>))\). Assume that the occurrence of the pattern ZIGZAG starts at position \( i \) and ends at position \( j \). The feature MIN computes the minimum of the values from index \( i + 1 \) to index \( j \).

#### Example
\[
(1, (4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1))
\]

Figure 3.781 provides an example where the MIN_MIN_ZIGZAG \((1, [4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1])\) constraint holds.

#### Typical
- \( |\text{VARIABLES}| > 3 \)
- \( \text{range}(\text{VARIABLES}.\text{var}) > 1 \)

#### Symmetry
Items of \( \text{VARIABLES} \) can be reversed.

#### Arg. properties
- Functional dependency: \( \text{VALUE} \) determined by \( \text{VARIABLES} \).
Figure 3.781: Illustrating the MIN_MIN_ZIGZAG constraint of the Example slot
Automaton

Figures 3.782 and 3.783 respectively depict the automaton associated with the constraint MIN_MIN_ZIGZAG and its simplified form.
Figure 3.782: Automaton for the MIN_MIN_ZIGZAG constraint obtained by applying decoration Table 2.35 to the seed transducer of the ZIGZAG pattern where default is $+\infty$; (1) missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=$; (2) on transitions from $b, c, e, f$ to $s$ the accumulator $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the accumulator $R$ is updated wrt $C$ and the accumulator $C$ is reset to its initial value.
Figure 3.783: Automaton for the MIN_MIN_ZIGZAG constraint obtained by applying decoration Table 2.28 to the seed transducer of the ZIGZAG pattern where default is $+\infty$; (1) missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=; (2)$ on transitions from $b, c, e, f$ to $s$ the accumulator $D$ is reset to its initial value; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.87: Glue matrix for the MIN-MIN-ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature MIN, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td></td>
</tr>
<tr>
<td>d</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td></td>
</tr>
<tr>
<td>f</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td>( \text{min}(\bar{C}, \bar{C}) )</td>
<td></td>
</tr>
</tbody>
</table>
3.341 MIN_RANGE_DECREASING

**Description**

 Based on the DECREASING pattern.

**Constraint**

 MIN_RANGE_DECREASING(VALUE, VARIABLES)

**Arguments**

 VALUE : dvar
 VARIABLES : collection(var−dvar)

**Restrictions**

 sv ≤ 1 \lor rv ≤ 1 \Rightarrow VALUE = +\infty
 VALUE ≥ 1∀
 required(VARIABLES, var)
 where
 sv = |VARIABLES|
 rv = range(VARIABLES.var)

**Purpose**

 VALUE is the minimum value of the differences between the largest and smallest value in each occurrence of the DECREASING pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +\infty.

 An occurrence of the pattern DECREASING is the subsequence which matches the regular expression >.

 Assume that the occurrence of the pattern DECREASING starts at position i and ends at position j. The feature RANGE computes the range of the values from index i to index j + 1.

**Example**

 (1, ⟨3, 4, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4⟩)

 Figure 3.784 provides an example where the MIN_RANGE_DECREASING (1, [3, 4, 2, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4]) constraint holds.

**Typical**

 |VARIABLES| > 1
 range(VARIABLES.var) > 1

**Symmetry**

 One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**

 Functional dependency: VALUE determined by VARIABLES.
Figure 3.784: Illustrating the MIN_RANGE_DECREASING constraint of the Example slot
Automaton

Figures 3.785 and 3.786 respectively depict the automaton associated with the constraint MIN\_RANGE\_DECREASING and its simplified form.

Figure 3.785: Automaton for the MIN\_RANGE\_DECREASING constraint obtained by applying decoration Table 2.46 to the seed transducer of the DECREASING pattern where default is +\infty

Figure 3.786: Automaton for the MIN\_RANGE\_DECREASING constraint obtained by applying decoration Table 2.44 to the seed transducer of the DECREASING pattern where default is +\infty; \(-R_i + R_{i-1} \geq 0\) is a linear invariant.
### 3.342 MIN_RANGE_DECREASING_SEQUENCE

**DESCRIPTION**

**Origin**
Based on the DECREASING_SEQUENCE pattern.

**Constraint**
MIN_RANGE_DECREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**
- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**

\[
sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty \\
VALUE \geq 1 \Rightarrow \\
\text{required}(\text{VARIABLES}, \text{var})
\]

where

\[
sv = |\text{VARIABLES}|
\]

\[
rv = \text{range}(\text{VARIABLES}.\text{var})
\]

**Purpose**
VALUE is the minimum value of the differences between the largest and smallest value in each occurrence of the DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +∞. An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression \((> | =)^* > | >\).

Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature RANGE computes the range of the values from index \(i\) to index \(j + 1\).

**Example**

\[(2, (3, 4, 2, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))\]

Figure 3.787 provides an example where the MIN_RANGE_DECREASING_SEQUENCE constraint holds.

**Typical**

\[|\text{VARIABLES}| > 1\]

\[\text{range}(\text{VARIABLES}.\text{var}) > 1\]

**Symmetry**
One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.787: Illustrating the MIN_RANGE_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 3.788 and 3.789 respectively depict the automaton associated with the constraint MIN\_RANGE\_DECREASING\_SEQUENCE and its simplified form.

\[
\begin{align*}
C &\leftarrow \text{default} \\
H &\leftarrow \text{VAR}_i \\
R &\leftarrow \text{default}
\end{align*}
\]

\[
\leq s \quad \{ H \leftarrow \text{VAR}_{i+1} \}
\]

\[
\begin{align*}
C &\leftarrow \text{default} \\
H &\leftarrow \text{VAR}_{i+1} \\
R &\leftarrow \min(R, C)
\end{align*}
\]

\[
\begin{array}{c}
\leq \{ H \leftarrow \text{VAR}_{i+1} \} \\
\geq t \quad \{ C \leftarrow |H - \text{VAR}_{i+1}| \}
\end{array}
\]

Figure 3.788: Automaton for the MIN\_RANGE\_DECREASING\_SEQUENCE constraint obtained by applying decoration Table 2.46 to the seed transducer of the DECREASING\_SEQUENCE pattern where default is $+\infty$.

\[
\begin{align*}
C &\leftarrow \text{default} \\
H &\leftarrow \text{VAR}_i \\
R &\leftarrow \text{default}
\end{align*}
\]

\[
\leq s \quad \{ H \leftarrow \text{VAR}_{i+1} \}
\]

\[
\begin{align*}
C &\leftarrow \text{default} \\
H &\leftarrow \text{VAR}_{i+1} \\
R &\leftarrow \min(R, C)
\end{align*}
\]

\[
\begin{array}{c}
\leq \{ H \leftarrow \text{VAR}_{i+1} \} \\
\geq t \quad \{ C \leftarrow |H - \text{VAR}_{i+1}| \}
\end{array}
\]

Figure 3.789: Automaton for the MIN\_RANGE\_DECREASING\_SEQUENCE constraint obtained by applying decoration Table 2.40 to the seed transducer of the DECREASING\_SEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
3.343 MIN_RANGE_INCREASING

**DESCRIPTION**

**Origin**
Based on the `INCREASING` pattern.

**Constraint**

\[
\text{MIN\_\text{\_}\text{\_}INCREASING}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection(\text{var} - \text{dvar})}
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{sv} \leq 1 & \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} & \geq 1
\end{align*}
\]

**Purpose**

VALUE is the minimum value of the differences between the largest and smallest value in each occurrence of the `INCREASING` pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +∞.

An occurrence of the pattern `INCREASING` is the subsequence which matches the regular expression `<`. Assume that the occurrence of the pattern `INCREASING` starts at position \(i\) and ends at position \(j\). The feature `RANGE` computes the range of the values from index \(i\) to index \(j + 1\).

**Example**

\[
(1, \langle 4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3 \rangle)
\]

Figure 3.790 provides an example where the MIN\_\text{\_}\text{\_}INCREASING (1, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3]) constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| & > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) & > 1
\end{align*}
\]

**Symmetry**

One and the same constant can be added to the \text{var} attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.790: Illustrating the MIN_RANGE_INCREASING constraint of the Example slot
Figures 3.791 and 3.792 respectively depict the automaton associated with the constraint MIN\_RANGE\_INCREASING and its simplified form.

Figure 3.791: Automaton for the MIN\_RANGE\_INCREASING constraint obtained by applying decoration Table 2.46 to the seed transducer of the INCREASING pattern where default is $+\infty$.

Figure 3.792: Automaton for the MIN\_RANGE\_INCREASING constraint obtained by applying decoration Table 2.45 to the seed transducer of the INCREASING pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
**MIN_RANGE_INCREASING_SEQUENCE**

### Description

**Origin**
Based on the INCREASING_SEQUENCE pattern.

**Constraint**

\[
\text{MIN\_RANGE\_INCREASING\_SEQUENCE}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var−dvar)`

**Restrictions**

\[
\begin{align*}
    \text{sv} \leq 1 & \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty \\
    \text{VALUE} \geq 1 & \\
    \text{required}(\text{VARIABLES}, \text{var}) & \\
    \text{where} & \\
    \text{sv} & = |\text{VARIABLES}| \\
    \text{rv} & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

VALUE is the minimum value of the differences between the largest and smallest value in each occurrence of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(+\infty\).

An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression \(<(\leq | =)* \leq | <\).

Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature RANGE computes the range of the values from index \(i\) to index \(j + 1\).

**Example**

\[\begin{align*}
    2, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3)
\end{align*}\]

Figure 3.793 provides an example where the MIN_RANGE_INCREASING_SEQUENCE constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 1\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Symmetry**
One and the same constant can be added to the \(\text{var}\) attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.793: Illustrating the MIN_RANGE_INCREASING_SEQUENCE constraint of the Example slot
Figures 3.794 and 3.795 respectively depict the automaton associated with the constraint \texttt{MIN\_RANGE\_INCREASING\_SEQUENCE} and its simplified form.

Figure 3.794: Automaton for the \texttt{MIN\_RANGE\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.46 to the seed transducer of the \texttt{INCREASING\_SEQUENCE} pattern where \texttt{default} is $+\infty$.

Figure 3.795: Automaton for the \texttt{MIN\_RANGE\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.41 to the seed transducer of the \texttt{INCREASING\_SEQUENCE} pattern where \texttt{default} is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
3.345  MIN_RANGE_STRICTLY_DECREASING_SEQUENCE

**Description**

Based on the `STRICTLY_DECREASING_SEQUENCE` pattern.

**Constraint**

`MIN_RANGE_STRICTLY_DECREASING_SEQUENCE(VALUE, VARIABLES)`

**Arguments**

- `VALUE`: `dvar`
- `VARIABLES`: `collection(var−dvar)`

**Restrictions**

\[
sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty \\
VALUE \geq 1 \\
required(VARIABLES, var)
\]

where

\[
sv = |VARIABLES| \\
rv = range(VARIABLES.var)
\]

**Purpose**

An occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` is the maximal subsequence which matches the regular expression `>^`. Assume that the occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `RANGE` computes the range of the values from index `i` to index `j + 1`.

**Example**

\[(1, (4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3))\]

Figure 3.796 provides an example where the `MIN_RANGE_STRICTLY_DECREASING_SEQUENCE (1, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3])` constraint holds.

**Typical**

\[
|VARIABLES| > 1 \\
range(VARIABLES.var) > 1
\]

**Symmetry**

One and the same constant can be added to the `var` attribute of all items of `VARIABLES`.

**Arg. properties**

Functional dependency: `VALUE` determined by `VARIABLES`.
Figure 3.796: Illustrating the `MIN_RANGE_STRICTLY_DECREASING_SEQUENCE` constraint of the `Example` slot
Figures 3.797 and 3.798 respectively depict the automaton associated with the constraint MIN\_RANGE\_STRICLY\_DECREASING\_SEQUENCE and its simplified form.

Figure 3.797: Automaton for the MIN\_RANGE\_STRICLY\_DECREASING\_SEQUENCE constraint obtained by applying decoration Table 2.46 to the seed transducer of the STRICTLY\_DECREASING\_SEQUENCE pattern where default is $+\infty$. 
Figure 3.798: Automaton for the MIN_RANGE_STRICTLY_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.40 to the seed transducer of the STRICTLY_DECREASING_SEQUENCE pattern where default is $+\infty$; $R_i + R_{i-1} \geq 0$ is a linear invariant.
3.346  MIN_RANGE STRICTLY_INCREASING_SEQUENCE

DESCRIPTION

Origin
Based on the STRICTLY_INCREASING_SEQUENCE pattern.

Constraint
MIN_RANGE STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = +∞
VALUE ≥ 1
required(VARIABLES, var)
where
sv = |VARIABLES|
rv = range(VARIABLES.var)

VALUE is the minimum value of the differences between the largest and smallest value in each occurrence of the STRICTLY_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +∞.

Purpose
An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression <+. Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at position i and ends at position j. The feature RANGE computes the range of the values from index i to index j + 1.

Example
(2, (4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3))

Figure 3.799 provides an example where the MIN_RANGE STRICTLY_INCREASING_SEQUENCE (2, [4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3]) constraint holds.

Typical
|VARIABLES| > 1
range(VARIABLES.var) > 1

Symmetry
One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 3.799: Illustrating the MIN_RANGE_STRICLY_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 3.800 and 3.801 respectively depict the automaton associated with the constraint \textbf{MIN\_RANGE\_STRICTLY\_INCREASING\_SEQUENCE} and its simplified form.

\[
\begin{align*}
\{ & C \leftarrow \text{default} \\
& H \leftarrow \text{VAR}_i \\
& R \leftarrow \text{default} \} \\
\{ & C \leftarrow \text{default} \\
& H \leftarrow \text{VAR}_{i+1} \\
& R \leftarrow \min(R, C) \}
\end{align*}
\]

Figure 3.800: Automaton for the \textbf{MIN\_RANGE\_STRICTLY\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.46 to the seed transducer of the \textbf{STRICTLY\_INCREASING\_SEQUENCE} pattern where \texttt{default} is $+\infty$. 
Figure 3.801: Automaton for the `MIN_RANGE_STRICTLY_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 2.41 to the seed transducer of the `STRICTLY_INCREASING_SEQUENCE` pattern where `default` is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
MIN_SURF_BUMP_ON_DECREASING_SEQUENCE

3.347 MIN_SURF_BUMP_ON_DECREASING_SEQUENCE

**Description**

Based on the BUMP_ON_DECREASING_SEQUENCE pattern.

**Constraint**

MIN_SURF_BUMP_ON_DECREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**

| VALUE  | : dvar         |
| VARIABLES | : collection(var–dvar) |

**Restrictions**

\[
\begin{align*}
sv & \leq 5 \lor rv \leq 2 \Rightarrow VALUE = +\infty \\
VALUE & \geq 3 \ast \text{minv} + 3 \\
VALUE & = +\infty \lor VALUE \leq 3 \ast \text{maxv} - 3 \\
\text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where

\[
\begin{align*}
\text{minv} & = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var}) \\
sv & = |\text{VARIABLES}| \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

VALUE is the minimal surface of occurrences of the BUMP_ON_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +\infty.

An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence which matches the regular expression >>><>.

Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 2\) to index \(j\).

**Example**

\( (11, [7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3]) \)

Figure 3.802 provides an example where the MIN_SURF_BUMP_ON_DECREASING_SEQUENCE (11, [7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3]) constraint holds.

**Typical**

\( |\text{VARIABLES}| > 5 \)

\( \text{range}(\text{VARIABLES}.\text{var}) > 2 \)

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.802: Illustrating the `MIN_SURF_BUMP_ON_DECREASING_SEQUENCE` constraint of the Example slot
Figures 3.803 and 3.804 respectively depict the automaton associated with the constraint \textsc{min\_surf\_bump\_on\_decreasing\_sequence} and its simplified form.

\[
\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow \text{default}
\end{align*}
\]

\[
\begin{align*}
\leq & \quad > \\
\leq & \quad > \\
< & \quad > \\
\min(R, C) &
\end{align*}
\]

\[
\begin{align*}
(D &\leftarrow D + \text{VAR}_i) \\
(D &\leftarrow 0) \\
(D &\leftarrow D + \text{VAR}_i) \\
(D &\leftarrow 0)
\end{align*}
\]

Figure 3.803: Automaton for the \textsc{min\_surf\_bump\_on\_decreasing\_sequence} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \textsc{bump\_on\_decreasing\_sequence} pattern where \text{default} is $+\infty$. 

\[
\begin{align*}
\leq & \quad > \\
\leq & \quad > \\
< & \quad > \\
\min(R, C) &
\end{align*}
\]

\[
\begin{align*}
(D &\leftarrow D + \text{VAR}_i) \\
(D &\leftarrow 0) \\
(D &\leftarrow D + \text{VAR}_i) \\
(D &\leftarrow 0)
\end{align*}
\]
Figure 3.804: Automaton for the MIN_SURF_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.25 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
3.348  MIN_SURF_DECREASING

**Description**

**Origin**
Based on the **DECREASING** pattern.

**Constraint**
MIN_SURF_DECREASING(VALUE, VARIABLES)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var−dvar)

**Restrictions**

\[
\begin{align*}
sv \leq 1 \lor rv \leq 1 & \Rightarrow VALUE = +\infty \\
VALUE & \geq 2 \ast \text{minv} + 1 \\
VALUE & = +\infty \lor VALUE \leq 2 \ast \text{maxv} - 1 \\
\text{required}(\text{VARIABLES}, \text{var}) & \text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where
- \text{minv} = \text{minval}($\text{VARIABLES}.\text{var}$)
- \text{maxv} = \text{maxval}($\text{VARIABLES}.\text{var}$)
- \text{sv} = |\text{VARIABLES}|
- \text{rv} = \text{range}(\text{VARIABLES}.\text{var})

**Purpose**
An occurrence of the pattern **DECREASING** is the subsequence which matches the regular expression $>$. Assume that the occurrence of the pattern **DECREASING** starts at position $i$ and ends at position $j$. The feature **SURF** computes the sum of the values from index $i$ to index $j + 1$.

**Example**

\[(4, (3, 4, 2, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))\]

Figure 3.805 provides an example where the **MIN_SURF_DECREASING** (4, [3, 4, 2, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4]) constraint holds.

**Typical**
- $|\text{VARIABLES}| > 1$
- $\text{range}(\text{VARIABLES}.\text{var}) > 1$

**Arg. properties**
Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 3.805: Illustrating the MIN_SURF_DECREASING constraint of the Example slot
Automaton

Figures 3.806 and 3.807 respectively depict the automaton associated with the constraint MIN_SURF_DECREASING and its simplified form.

\[
\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow \text{default}
\end{align*}
\]

\[
\begin{align*}
D &\leftarrow 0 \\
R &\leftarrow \min(R, D + \text{VAR}_i + \text{VAR}_{i+1})
\end{align*}
\]

\[
\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow \min(R, \text{VAR}_i + \text{VAR}_{i+1})
\end{align*}
\]

Figure 3.806: Automaton for the MIN_SURF_DECREASING constraint obtained by applying decoration Table 2.35 to the seed transducer of the DECREASING pattern where default is $+\infty$.

\[
\begin{align*}
\{R &\leftarrow \text{default}\} \\
\{R &\leftarrow \min(R, \text{VAR}_i + \text{VAR}_{i+1})\}
\end{align*}
\]

Figure 3.807: Automaton for the MIN_SURF_DECREASING constraint obtained by applying decoration Table 2.38 to the seed transducer of the DECREASING pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.

Table 3.88: Glue matrix for the MIN_SURF_DECREASING constraint defined as the composition of the DECREASING pattern, the feature SURF, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.349 MIN_SURF_DECREASING_SEQUENCE

**DESCRIPTION**

Based on the DECREASING_SEQUENCE pattern.

**Constraint**

\[
\text{MIN\_SURF\_DECREASING\_SEQUENCE}(\text{VALUE, VARIABLES})
\]

**Arguments**

\[
\begin{align*}
\text{VALUE} &: \text{ dvar} \\
\text{VARIABLES} &: \text{ collection(var–dvar)}
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{sv} &\leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{rv} &\geq 2 \Rightarrow \text{VALUE} \geq 2 * \text{minv} + 1 \\
\text{rv} &\geq 3 \Rightarrow \text{VALUE} \geq \min(2 * \text{minv} + 1, \text{sv} * (\text{minv} + 1)) \\
\text{rv} &\geq 2 \Rightarrow \text{VALUE} = +\infty \lor \text{VALUE} \leq 2 * \text{maxv} - 1 \\
\text{rv} &\geq 3 \Rightarrow \text{VALUE} = +\infty \lor \text{VALUE} \leq \max(2 * \text{maxv} - 1, \text{sv} * (\text{maxv} - 1))
\end{align*}
\]

\[\text{required} (\text{VARIABLES, var})\]

\[
\begin{align*}
\text{minv} &= \text{minval} (\text{VARIABLES.var}) \\
\text{rv} &= \text{range} (\text{VARIABLES.var}) \\
\text{sv} &= |\text{VARIABLES}| \\
\text{maxv} &= \text{maxval} (\text{VARIABLES.var})
\end{align*}
\]

**Purpose**

VALUE is the minimal surface of occurrences of the DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(+\infty\).

An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression \(> (>|=)^*|>\).

Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i\) to index \(j + 1\).

**Example**

\[(6, [3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4])\]

Figure 3.808 provides an example where the MIN_SURF_DECREASING_SEQUENCE (6, [3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4]) constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| &> 1 \\
\text{range} (\text{VARIABLES.var}) &> 1
\end{align*}
\]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.808: Illustrating the MIN_SURF_DECREASING_SEQUENCE constraint of the Example slot
Figure 3.809 depicts the automaton associated with the constraint MIN_SURF_DECREASING_SEQUENCE.

\[
\begin{align*}
\{ C \leftarrow \text{default} \\
D & \leftarrow 0 \\
R & \leftarrow \text{default} \\
\leq \ s
\end{align*}
\]

\[
\begin{align*}
\{ C \leftarrow \text{default} \\
D & \leftarrow 0 \\
R & \leftarrow \min(R, C) \\
< \ C
\end{align*}
\]

\[
\begin{align*}
\{ C \leftarrow \text{default} \\
D & \leftarrow 0 \\
R & \leftarrow \min(R, C) \\
\geq \ t
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow D + \text{VAR}_{i+1} \}
\end{align*}
\]

Figure 3.809: Automaton for the MIN_SURF_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the DECREASING_SEQUENCE pattern where \texttt{default} is \(+\infty\); \(-R_i + R_{i-1} \geq 0\) is a linear invariant.

<table>
<thead>
<tr>
<th>s</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\min(\overline{C}, \overline{C}))</td>
</tr>
<tr>
<td>t</td>
<td>(\min(\overline{C}, \overline{C}))</td>
</tr>
</tbody>
</table>

Table 3.89: Glue matrix for the MIN_SURF_DECREASING_SEQUENCE constraint defined as the composition of the DECREASING_SEQUENCE pattern, the feature SURF, and the aggregator \(\min\); cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MIN_SURF_DECREASING_SEQUENCE 1521
3.350 MIN_SURF_DECREASING_TERRACE

DESCRIPTION

Origin
Based on the DECREASING_TERRACE pattern.

Constraint
MIN_SURF_DECREASING_TERRACE(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
\[ sv \leq 3 \lor rv \leq 2 \Rightarrow VALUE = +\infty \]
\[ VALUE \geq \min(2 \ast (\text{minv} + 1) \lor (sv - 2) \ast (\text{minv} + 1) \lor ) \]
\[ VALUE = +\infty \lor VALUE \leq \max(2 \ast (\text{maxv} - 1) \lor (sv - 2) \ast (\text{maxv} - 1)) \]
where
\[ \text{minv} = \minval(VARIABLES.var) \]
\[ sv = |\text{VARIABLES}| \]
\[ \text{maxv} = \maxval(VARIABLES.var) \]
\[ rv = \text{range}(VARIABLES.var) \]

Purpose
VALUE is the minimal surface of occurrences of the DECREASING_TERRACE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +\infty.

An occurrence of the pattern DECREASING_TERRACE is the maximal subsequence which matches the regular expression >\rightarrow.

Assume that the occurrence of the pattern DECREASING_TERRACE starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

Example
\[(4, (6, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3))\]

Figure 3.810 provides an example where the MIN_SURF_DECREASING_TERRACE (4, [6, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3]) constraint holds.

Typical
\[|\text{VARIABLES}| > 3\]
\[\text{range}(\text{VARIABLES}.var) > 2\]

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 3.810: Illustrating the MIN_SURF_DECREASING_TERRACE constraint of the Example slot
Automaton

Figures 3.811 and 3.812 respectively depict the automaton associated with the constraint MIN_SURF_DECREASING_TERRACE and its simplified form.

Figure 3.811: Automaton for the MIN_SURF_DECREASING_TERRACE constraint obtained by applying decoration Table 2.35 to the seed transducer of the DECREASING_TERRACE pattern where default is $+\infty$.

Figure 3.812: Automaton for the MIN_SURF_DECREASING_TERRACE constraint obtained by applying decoration Table 2.25 to the seed transducer of the DECREASING_TERRACE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.90: Glue matrix for the MIN_SURF_DECREASING_TERRACE constraint defined as the composition of the DECREASING_TERRACE pattern, the feature SURF, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>min($\overrightarrow{C}$, $\overrightarrow{C}$)</td>
<td>min($\overrightarrow{C}$, $\overrightarrow{C}$)</td>
<td>min($\overrightarrow{C}$, $\overrightarrow{C}$)</td>
</tr>
<tr>
<td>r</td>
<td>min($\overrightarrow{C}$, $\overrightarrow{C}$)</td>
<td>min($\overrightarrow{C}$, $\overrightarrow{C}$)</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i$</td>
</tr>
<tr>
<td>t</td>
<td>min($\overrightarrow{C}$, $\overrightarrow{C}$)</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i$</td>
</tr>
</tbody>
</table>
3.351 MIN_SURF_DIP_ON_INCREASING_SEQUENCE

### Description
Based on the `DIP_ON_INCREASING_SEQUENCE` pattern.

### Automaton
![Automaton Diagram]

### Origin

**Constraint**

`MIN_SURF_DIP_ON_INCREASING_SEQUENCE(VALUE, VARIABLES)`

**Arguments**

- `VALUE` : `dvar`
- `VARIABLES` : `collection(var−dvar)`

**Restrictions**

- `sv ≤ 5 ∨ rv ≤ 2 ⇒ VALUE = +∞`
- `VALUE ≥ 3 * minv + 3`
- `VALUE = +∞ ∨ VALUE ≤ 3 * maxv - 3`

where

- `minv = minval(VARIABLES.var)`
- `maxv = maxval(VARIABLES.var)`
- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

### Purpose

**VALUE** is the minimal surface of occurrences of the `DIP_ON_INCREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, **VALUE** takes the default value `+∞`.

An occurrence of the pattern `DIP_ON_INCREASING_SEQUENCE` is the subsequence which matches the regular expression `<<><<`. Assume that the occurrence of the pattern `DIP_ON_INCREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature **SURF** computes the sum of the values from index `i + 2` to index `j`.

### Example

```
(9, (1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4))
```

Figure 3.813 provides an example where the `MIN_SURF_DIP_ON_INCREASING_SEQUENCE` (9, [1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4]) constraint holds.

**Typical**

- `|VARIABLES| > 5`
- `range(VARIABLES.var) > 2`

**Arg. properties**

Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 3.813: Illustrating the MIN_SURF_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 3.814 and 3.815 respectively depict the automaton associated with the constraint \texttt{MIN_SURF\_DIP\_ON\_INCREASING\_SEQUENCE} and its simplified form.

\[
\begin{align*}
C & \leftarrow \text{default} \\
D & \leftarrow 0 \\
R & \leftarrow \text{default} \\
\{ D \geq 0 \} & \Rightarrow \min(R, C) < \\
\{ D < 0 \} & \Rightarrow D \rightarrow D + \text{VAR} \\
\{ D \geq 0 \} & \Rightarrow D = D + \text{VAR} \\
\{ D < 0 \} & \Rightarrow D = D + \text{VAR} \\
\end{align*}
\]

Figure 3.814: Automaton for the \texttt{MIN_SURF\_DIP\_ON\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{DIP\_ON\_INCREASING\_SEQUENCE} pattern where \texttt{default} is $\oplus \infty$
Figure 3.815: Automaton for the MIN_SURF_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.25 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
3.352 MIN_SURF_GORGE

### Description

**Origin**

Based on the GORGE pattern.

**Constraint**

\[
\text{MIN\_SURF\_GORGE}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- \text{VALUE} : dvar
- \text{VARIABLES} : collection(var\_dvar)

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{rv} = 2 & \Rightarrow \text{VALUE} \geq \text{minv} \\
\text{rv} \geq 3 & \Rightarrow \text{VALUE} \geq \min(\text{minv}, (\text{sv} - 2) \times (\text{minv} + 1) - 1) \\
\text{rv} = 2 & \Rightarrow \text{VALUE} = +\infty \lor \text{VALUE} \leq \text{maxv} - 1 \\
\text{rv} \geq 3 & \Rightarrow \text{VALUE} = +\infty \lor \text{VALUE} \leq \max(\text{maxv} - 1, (\text{sv} - 2) \times (\text{maxv} - 1) - 1) \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{where} & \\
\text{minv} & = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

\[\text{VALUE} \text{ is the minimal surface of occurrences of the GORGE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value } +\infty.\]

An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression \((\text{>|} \lor \text{|=})\text{>}\times(\text{<|} \lor \text{|=})\text{<}\).

Assume that the occurrence of the pattern GORGE starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

\((5, (1, 7, 3, 4, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7))\)

Figure 3.816 provides an example where the MIN\_SURF\_GORGE \((5, [1, 7, 3, 4, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7])\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 2 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.816: Illustrating the MIN_SURF_GORGE constraint of the Example slot
Figure 3.817 depicts the automaton associated with the constraint MIN_SURF_GORGE.

Figure 3.817: Automaton for the MIN_SURF_GORGE constraint obtained by applying decoration Table 2.35 to the seed transducer of the GORGE pattern where default is $+\infty$ (transition $u \to r$ has the same accumulator update as transition $r \to u$); $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.91: Glue matrix for the MIN_SURF_GORGE constraint defined as the composition of the GORGE pattern, the feature SURF, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### 3.353 MIN_SURF_INCREASING

#### DESCRIPTION

**Origin**
Based on the INCREASING pattern.

**Constraint**
MIN_SURF_INCREASING(VALUE, VARIABLES)

**Arguments**
- **VALUE**: dvar
- **VARIABLES**: collection(var–dvar)

**Restrictions**

\[
\begin{align*}
sv & \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty \\
VALUE & \geq 2 \cdot \text{minv} + 1 \\
VALUE & = +\infty \lor VALUE \leq 2 \cdot \text{maxv} - 1 \\
\text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where

\[
\begin{align*}
\text{minv} &= \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{maxv} &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
sv &= |\text{VARIABLES}| \\
rv &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**
An occurrence of the pattern INCREASING is the subsequence which matches the regular expression `<`. Assume that the occurrence of the pattern INCREASING starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i\) to index \(j + 1\).

**Example**

\[(4, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))\]

Figure 3.818 provides an example where the MIN_SURF_INCREASING \((4, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3])\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.818: Illustrating the MIN_SURF_INCREASING constraint of the Example slot
Figures 3.819 and 3.820 respectively depict the automaton associated with the constraint MIN_SURF_INCREASING and its simplified form.

Figure 3.819: Automaton for the MIN_SURF_INCREASING constraint obtained by applying decoration Table 2.35 to the seed transducer of the INCREASING pattern where default is $+\infty$

Figure 3.820: Automaton for the MIN_SURF_INCREASING constraint obtained by applying decoration Table 2.38 to the seed transducer of the INCREASING pattern where default is $+\infty$; $R_i + R_{i-1} \geq 0$ is a linear invariant.

Table 3.92: Glue matrix for the MIN_SURF_INCREASING constraint defined as the composition of the INCREASING pattern, the feature SURF, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.354  **MIN_SURF_INCREASING_SEQUENCE**

**Origin**
Based on the **INCREASING_SEQUENCE** pattern.

**Constraint**
```
MIN_SURF_INCREASING_SEQUENCE(VALUE, VARIABLES)
```

**Arguments**
```
VALUE : dvar
VARIABLES : collection(var − dvar)
```

**Restrictions**
```
sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = +∞
rv = 2 ⇒ VALUE ≥ 2 * minv + 1
rv ≥ 3 ⇒ VALUE ≥ min(2 * minv + 1, sv * (minv + 1)*)
rv = 2 ⇒ VALUE = +∞ ∨ VALUE ≤ 2 * maxv - 1
rv ≥ 3 ⇒ VALUE = +∞ ∨ VALUE ≤ max(2 * maxv - 1, sv * (maxv - 1))
```

where
```
minv = minval(VARIABLES.var)
rv = range(VARIABLES.var)
sv = |VARIABLES|
maxv = maxval(VARIABLES.var)
```

**Purpose**
VALUE is the minimal surface of occurrences of the **INCREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +∞.

An occurrence of the pattern **INCREASING_SEQUENCE** is the **maximal** subsequence which matches the regular expression `< (< | =)* < | <`

Assume that the occurrence of the pattern **INCREASING_SEQUENCE** starts at position i and ends at position j. The feature SURF computes the sum of the values from index i to index j + 1.

**Example**
```
(4, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3])
```

Figure 3.821 provides an example where the **MIN_SURF_INCREASING_SEQUENCE** (4, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3]) constraint holds.

**Typical**
```
|VARIABLES| > 1
range(VARIABLES.var) > 1
```

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.821: Illustrating the MIN_SURF_INCREASING_SEQUENCE constraint of the Example slot
Automaton Figure 3.822 depicts the automaton associated with the constraint MIN_SURF_INCREASING_SEQUENCE.

\[
\begin{align*}
\{ C \leftarrow \text{default} \\
D \leftarrow 0 \\
R \leftarrow \text{default} \}
\end{align*}
\]

\[
\begin{align*}
\geq s & \geq \\
\{ C \leftarrow \text{default} \\
D \leftarrow 0 \\
R \leftarrow \min(R,C) \}
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow D + \text{VAR}_i \} \\
\{ C \leftarrow \text{default} \\
D \leftarrow 0 \\
R \leftarrow \min(R,C) \}
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow D + \text{VAR}_{i+1} \} \\
\{ C \leftarrow C + D + \text{VAR}_{i+1} \}
\end{align*}
\]

Figure 3.822: Automaton for the MIN_SURF_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the INCREASING_SEQUENCE pattern where \text{default} is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\min(C, \overrightarrow{C})$</td>
<td>$\min(C, \overrightarrow{C})$</td>
</tr>
<tr>
<td>t</td>
<td>$\min(C, \overrightarrow{C})$</td>
<td>$C + \overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i$</td>
</tr>
</tbody>
</table>

Table 3.93: Glue matrix for the MIN_SURF_INCREASING_SEQUENCE constraint defined as the composition of the INCREASING_SEQUENCE pattern, the feature SURF, and the aggregator $\min$; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.355 MIN_SURF_INCREASING_TERRACE

**DESCRIPTION**

**Origin**
Based on the INCREASING_TERRACE pattern.

**Constraint**
MIN_SURF_INCREASING_TERRACE(VALUE, VARIABLES)

**Arguments**
VALUE : dvar
VARIABLES : collection(var−dvar)

**Restrictions**
\[ sv \leq 3 \lor rv \leq 2 \Rightarrow VALUE = +\infty \]
\[ VALUE \geq \min(2 \ast (\text{minv} + 1), (sv - 2) \ast (\text{minv} + 1)) \]
\[ VALUE = +\infty \lor VALUE \leq \max(2 \ast (\maxv - 1), (sv - 2) \ast (\maxv - 1)) \]

\[ \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \]
\[ sv = |\text{VARIABLES}| \]
\[ \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \]
\[ \text{rv} = \text{range}(\text{VARIABLES}.\text{var}) \]

**Purpose**
VALUE is the minimal surface of occurrences of the INCREASING_TERRACE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +\infty.

An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression \(<=+=<\).

Assume that the occurrence of the pattern INCREASING_TERRACE starts at position \(i\) and ends at position \(j\). The feature \text{SURF} computes the sum of the values from index \(i+1\) to index \(j\).

**Example**
\[(9, (1, 3, 3, 3, 2, 5, 5, 6, 4, 2, 3, 3, 3, 4, 4))\]

Figure 3.823 provides an example where the MIN_SURF_INCREASING_TERRACE (9, [1, 3, 3, 3, 2, 5, 5, 6, 4, 2, 3, 3, 3, 4, 4]) constraint holds.

**Typical**
\[ |\text{VARIABLES}| > 3 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 2 \]

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.823: Illustrating the MIN_SURF_INCREASING_TERRACE constraint of the Example slot
Figures 3.824 and 3.825 respectively depict the automaton associated with the constraint MIN_SURF_INCREASING_TERRACE and its simplified form.

**Figure 3.824:** Automaton for the MIN_SURF_INCREASING_TERRACE constraint obtained by applying decoration Table 2.35 to the seed transducer of the INCREASING_TERRACE pattern where default is $+\infty$.

**Figure 3.825:** Automaton for the MIN_SURF_INCREASING_TERRACE constraint obtained by applying decoration Table 2.25 to the seed transducer of the INCREASING_TERRACE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.94: Glue matrix for the MIN_SURF_INCREASING_TERRACE constraint defined as the composition of the INCREASING_TERRACE pattern, the feature SURF, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>min(\overrightarrow{C}, \overrightarrow{C})</td>
<td>min(\overrightarrow{C}, \overrightarrow{C})</td>
<td>min(\overrightarrow{C}, \overrightarrow{C})</td>
</tr>
<tr>
<td>r</td>
<td>min(\overrightarrow{C}, \overrightarrow{C})</td>
<td>min(\overrightarrow{C}, \overrightarrow{C})</td>
<td>\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i</td>
</tr>
<tr>
<td>t</td>
<td>min(\overrightarrow{C}, \overrightarrow{C})</td>
<td>\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i</td>
<td>\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i</td>
</tr>
</tbody>
</table>
3.356 MIN_SURF_INFLEXION

**DESCRIPTION**

**Origin**
Based on the INFLEXION pattern.

**Constraint**

\[
\text{MIN\_SURF\_INFLEXION}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- \text{VALUE} : \text{dvar}
- \text{VARIABLES} : \text{collection(\text{var} - \text{dvar})}

**Restrictions**

\[
\text{sv} \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty
\]

\[
\text{VALUE} \geq \min(\text{minv,} \, (\text{sv} - 2) + \text{minv})
\]

\[
\text{VALUE} = +\infty \lor \text{VALUE} \leq \max(\text{maxv,} \, (\text{sv} - 2) + \text{maxv})
\]

\[
\text{required}(\text{VARIABLES}, \text{var})
\]

where

- \text{minv} = \text{minval}(\text{VARIABLES}.\text{var})
- \text{sv} = |\text{VARIABLES}|
- \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})
- \text{rv} = \text{range}(\text{VARIABLES}.\text{var})

VALUE is the minimal surface of occurrences of the INFLEXION pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +\infty.

**Purpose**

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression \(\langle \langle \mid = \rangle^* \rangle \mid > \mid > \mid = \rangle^* \langle\).

Assume that the occurrence of the pattern INFLEXION starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

\[(1, (1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4))\]

Figure 3.826 provides an example where the MIN\_SURF\_INFLEXION \((1, [1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4])\) constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 2\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.826: Illustrating the MIN_SURF_INFLEXION constraint of the Example slot
Automaton

Figures 3.827 and 3.828 respectively depict the automaton associated with the constraint MIN_SURF_INFLEXION and its simplified form.

\[
\begin{align*}
\{ C \leftarrow \text{default} \} \\
\{ D \leftarrow 0 \} \\
\{ R \leftarrow \text{default} \}
\end{align*}
\]

\[
\leq R \\
\leq D \xrightarrow{D \leftarrow D + \text{VAR}_i} \leq t
\]

\[
\{ D \leftarrow \text{default} \} \\
\leq R \\
\leq D \xrightarrow{D \leftarrow D + \text{VAR}_i} \leq t
\]

\[
\{ D \leftarrow 0 \} \\
\{ R \leftarrow \min(R, D + \text{VAR}_i) \}
\]

\[
\geq R \\
\geq D \xrightarrow{D \leftarrow D + \text{VAR}_i} \geq t
\]

\[
\{ D \leftarrow \text{default} \} \\
\geq R \\
\geq D \xrightarrow{D \leftarrow D + \text{VAR}_i} \geq t
\]

\[
\{ D \leftarrow 0 \} \\
\{ R \leftarrow \min(R, D + \text{VAR}_i) \}
\]

Figure 3.827: Automaton for the MIN_SURF_INFLEXION constraint obtained by applying decoration Table 2.35 to the seed transducer of the INFLEXION pattern where \textbf{default} is \(+\infty\) (transition \(r \rightarrow t\) has the same accumulators updates as transition \(t \rightarrow r\)).

Figure 3.828: Automaton for the MIN_SURF_INFLEXION constraint obtained by applying decoration Table 2.25 to the seed transducer of the INFLEXION pattern where \textbf{default} is \(+\infty\) (transition \(r \rightarrow t\) has the same accumulators updates as transition \(t \rightarrow r\)); \(-R_i + R_{i-1} \geq 0\) is a linear invariant.
3.357  **MIN_SURF_PEAK**

**DESCRIPTION**

**Origin**
Based on the PEAK pattern.

**Constraint**

\[ \text{MIN\_SURF\_PEAK}(\text{VALUE, VARIABLES}) \]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection(var − dvar)}
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} & \geq \min(\text{minv} + 1, (\text{sv} - 2) \ast (\text{minv} + 1)\%)
\end{align*}
\]

**Purpose**

An occurrence of the pattern PEAK is the **maximal** subsequence which matches the regular expression \(< (\text{=} | <) \ast (> | =) \ast >\).

Assume that the occurrence of the pattern PEAK starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

\[ (9, (7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 3, 1)) \]

Figure 3.829 provides an example where the MIN\_SURF\_PEAK (9, [7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 3, 1]) constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 2 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.829: Illustrating the MIN_SURF_PEAK constraint of the Example slot
Automaton

Figure 3.830 depicts the automaton associated with the constraint MIN\_SURF\_PEAK.

\[
\begin{align*}
\{ C &\leftarrow \text{default} \} \quad \geq s \quad \geq \min(R, C) \\
\{ D &\leftarrow 0 \} \quad < \quad \{ C &\leftarrow D + \text{VAR}_i \} \\
\{ R &\leftarrow \text{default} \} \\
\{ D &\leftarrow D + \text{VAR}_i \} \\
\{ \{ C &\leftarrow \text{default} \} \quad D &\leftarrow 0 \} \\
\{ R &\leftarrow \min(R, C) \} \\
\{ D &\leftarrow D + \text{VAR}_i \} \\
\end{align*}
\]

Figure 3.830: Automaton for the MIN\_SURF\_PEAK constraint obtained by applying decoration Table 2.35 to the seed transducer of the PEAK pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i$</td>
<td>$\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
</tbody>
</table>

Table 3.95: Glue matrix for the MIN\_SURF\_PEAK constraint defined as the composition of the PEAK pattern, the feature SURF, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
MIN_SURF_PEAK  1553
### 3.358 MIN_SURF.PLAIN

**Origin**
Based on the PLAIN pattern.

**Constraint**
MIN_SURF.PLAIN(VALUE, VARIABLES)

**Arguments**
- **VALUE**: dvar
- **VARIABLES**: collection(var\(\rightarrow\)dvar)

**Restrictions**
\[
\begin{align*}
sv & \leq 2 \lor rv \leq 1 \Rightarrow VALUE = +\infty \\
VALUE & \geq \min(\minv, (sv - 2) + \minv) \\
VALUE & = +\infty \lor VALUE \leq \max(\maxv - 1, (sv - 2) * (\maxv - 1)) \\
\text{required}(\text{VARIABLES}.\text{var})
\end{align*}
\]
where
- \(\minv = \minval(\text{VARIABLES}.\text{var})\)
- \(sv = |\text{VARIABLES}|\)
- \(\maxv = \maxval(\text{VARIABLES}.\text{var})\)
- \(rv = \text{range}(\text{VARIABLES}.\text{var})\)

**Purpose**
An occurrence of the pattern PLAIN is the *maximal* subsequence which matches the regular expression \(>^*<\).
Assume that the occurrence of the pattern PLAIN starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**
\[(4, (2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3))\]

Figure 3.831 provides an example where the MIN_SURF.PLAIN \((4, [2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3])\) constraint holds.

**Typical**
- \(|\text{VARIABLES}| > 2\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.831: Illustrating the MIN_SURFPLAIN constraint of the Example slot
Figures 3.832 and 3.833 respectively depict the automaton associated with the constraint `MIN_SURF_PLAIN` and its simplified form.

Figure 3.832: Automaton for the `MIN_SURF_PLAIN` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `PLAIN` pattern where `default` is $+\infty$.

Figure 3.833: Automaton for the `MIN_SURF_PLAIN` constraint obtained by applying decoration Table 2.25 to the seed transducer of the `PLAIN` pattern where `default` is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.96: Glue matrix for the MIN_SURF_PLAIN constraint defined as the composition of the PLAIN pattern, the feature SURF, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.359 MIN_SURF_PLATEAU

DESCRIPTION

Origin
Based on the PLATEAU pattern.

Constraint
MIN_SURF_PLATEAU(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = +∞
VALUE ≥ min(minv + 10, (sv − 2) * (minv + 1))
VALUE = +∞ ∨ VALUE ≤ max(maxv, (sv − 2) * maxv)
required(VARIABLES, var)

where
minv = minval(VARIABLES.var)
sv = |VARIABLES|
maxv = maxval(VARIABLES.var)
rv = range(VARIABLES.var)

VALUE is the minimal surface of occurrences of the PLATEAU pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +∞.

An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression <∗>.

Assume that the occurrence of the pattern PLATEAU starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 1 to index j.

Example
(3, ⟨7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5⟩)

Figure 3.834 provides an example where the MIN_SURF_PLATEAU (3, [7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5]) constraint holds.

Typical
|VARIABLES| > 2
range(VARIABLES.var) > 1

Symmetry
Items of VARIABLES can be reversed.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 3.834: Illustrating the MIN_SURF_PLATEAU constraint of the Example slot
Figures 3.835 and 3.836 respectively depict the automaton associated with the constraint MIN_SURF_PLATEAU and its simplified form.

Figure 3.835: Automaton for the MIN_SURF_PLATEAU constraint obtained by applying decoration Table 2.35 to the seed transducer of the PLATEAU pattern where default is $+\infty$.

Figure 3.836: Automaton for the MIN_SURF_PLATEAU constraint obtained by applying decoration Table 2.25 to the seed transducer of the PLATEAU pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.97: Glue matrix for the MIN_SURF_PLATEAU constraint defined as the composition of the PLATEAU pattern, the feature SURF, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>min((\vec{C}, \vec{C}))</td>
</tr>
<tr>
<td>r</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>(\vec{D} + \vec{D} + \text{VAR}_i)</td>
<td>(\vec{D} + \vec{D} + \text{VAR}_i)</td>
</tr>
<tr>
<td>t</td>
<td>min((\vec{C}, \vec{C}))</td>
<td>(\vec{D} + \vec{D} + \text{VAR}_i)</td>
<td>(\vec{D} + \vec{D} + \text{VAR}_i)</td>
</tr>
</tbody>
</table>
3.360 MIN_SURF_PROPER.PLAIN

DESCRIPTION

Origin

Based on the PROPER.PLAIN pattern.

Constraint

MIN_SURF_PROPER.PLAIN(VALUE, VARIABLES)

Arguments

VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions

\[
sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = +\infty \\
VALUE \geq \min(2 + \min v, (sv - 2) + \min v) \\
VALUE = +\infty \lor VALUE \leq \max(2 + (max v - 1), (sv - 2) + (max v - 1))
\]

where

\[
\begin{align*}
\min v &= \min val(VARIABLES.var) \\
sv &= |VARIABLES| \\
max v &= \max val(VARIABLES.var) \\
rv &= \text{range}(VARIABLES.var)
\end{align*}
\]

VALUE is the minimal surface of occurrences of the PROPER.PLAIN pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value $+ \infty$.

Purpose

An occurrence of the pattern PROPER.PLAIN is the maximal subsequence which matches the regular expression $> = + <$.

Assume that the occurrence of the pattern PROPER.PLAIN starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

Example

\[ (8, (2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5)) \]

Figure 3.837 provides an example where the MIN_SURF_PROPER.PLAIN (8, [2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5]) constraint holds.

Typical

\[ |VARIABLES| > 3 \]

\[ \text{range}(VARIABLES.var) > 1 \]

Symmetry

Items of VARIABLES can be reversed.

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 3.837: Illustrating the \textsc{MIN\_SURF\_PROPER\_PLAIN} constraint of the \textbf{Example} slot
Figures 3.838 and 3.839 respectively depict the automaton associated with the constraint `MIN_SURF_PROPERPlain` and its simplified form.

Figure 3.838: Automaton for the `MIN_SURF_PROPERPlain` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `PROPERPlain` pattern where `default` is $+\infty$.

Figure 3.839: Automaton for the `MIN_SURF_PROPERPlain` constraint obtained by applying decoration Table 2.25 to the seed transducer of the `PROPERPlain` pattern where `default` is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.98: Glue matrix for the MIN_SURF_PROPER_PLAIN constraint defined as the composition of the PROPER_PLAIN pattern, the feature SURF, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>min((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>min((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>min((\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
<tr>
<td>r</td>
<td>min((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>min((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i)</td>
</tr>
<tr>
<td>t</td>
<td>min((\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i)</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i)</td>
</tr>
</tbody>
</table>
### MIN_SURF_PROPER_PLATEAU

#### Description

**Origin**
Based on the `PROPER_PLATEAU` pattern.

**Constraint**

\[
\text{MIN\_SURF\_PROPER\_PLATEAU}(\text{VALUE, VARIABLES})
\]

**Arguments**

- \( \text{VALUE} : \text{dvar} \)
- \( \text{VARIABLES} : \text{collection}(\text{var} - \text{dvar}) \)

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 3 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} & \geq \min(2 \cdot (\text{minv} + 1)^\delta, (\text{sv} - 2) \cdot (\text{minv} + 1)^\delta) \\
\text{VALUE} & = +\infty \lor \text{VALUE} \leq \max(2 \cdot \text{maxv}, (\text{sv} - 2) \cdot \text{maxv})
\end{align*}
\]

where

- \( \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \)
- \( \text{sv} = |\text{VARIABLES}| \)
- \( \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \)
- \( \text{rv} = \text{range}(\text{VARIABLES}.\text{var}) \)

**Purpose**

`VALUE` is the minimal surface of occurrences of the `PROPER_PLATEAU` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, `VALUE` takes the default value \(+\infty\).

An occurrence of the pattern `PROPER_PLATEAU` is the maximal subsequence which matches the regular expression `\(< = + >\)`.

Assume that the occurrence of the pattern `PROPER_PLATEAU` starts at position \(i\) and ends at position \(j\). The feature `SURF` computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

\[
(6, (7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3))
\]

Figure 3.840 provides an example where the `MIN\_SURF\_PROPER\_PLATEAU` constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| & > 3 \\
\text{range}(\text{VARIABLES}.\text{var}) & > 1
\end{align*}
\]

**Symmetry**

Items of `VARIABLES` can be reversed.

**Arg. properties**

Functional dependency: `VALUE` determined by `VARIABLES`. 
Figure 3.840: Illustrating the MIN_SURF_PROPER_PLATEAU constraint of the Example slot
Automaton

Figures 3.841 and 3.842 respectively depict the automaton associated with the constraint MIN_SURF_PROPER_PLATEAU and its simplified form.

Figure 3.841: Automaton for the MIN_SURF_PROPER_PLATEAU constraint obtained by applying decoration Table 2.35 to the seed transducer of the PROPER_PLATEAU pattern where default is $+\infty$.

Figure 3.842: Automaton for the MIN_SURF_PROPER_PLATEAU constraint obtained by applying decoration Table 2.25 to the seed transducer of the PROPER_PLATEAU pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.99: Glue matrix for the MIN_SURF_PROPER_PLATEAU constraint defined as the composition of the PROPER_PLATEAU pattern, the feature SURF, and the aggregator \text{min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>(s)</th>
<th>(r)</th>
<th>(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>(\text{min}(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\text{min}(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\text{min}(\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
<tr>
<td>(r)</td>
<td>(\text{min}(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\text{min}(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i)</td>
</tr>
<tr>
<td>(t)</td>
<td>(\text{min}(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i)</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i)</td>
</tr>
</tbody>
</table>
### 3.362 MIN_SURF_STEADY

**DESCRIPTION**

- **Origin**: Based on the `STEADY` pattern.
- **Constraint**: `MIN_SURF_STEADY(VALUE, VARIABLES)`
- **Arguments**:
  - `VALUE : dvar`
  - `VARIABLES : collection(var-dvar)`
- **Restrictions**:
  - `sv ≤ 1 ⇒ VALUE = +∞`
  - `VALUE ≥ 2 * minv`
  - `VALUE = +∞ ∨ VALUE ≤ 2 * maxv`
  - `required(VARIABLES, var)`
  - where
    - `minv = minval(VARIABLES.var)`
    - `maxv = maxval(VARIABLES.var)`
    - `sv = |VARIABLES|`
- **Purpose**: An occurrence of the pattern `STEADY` is the subsequence which matches the regular expression `=`. Assume that the occurrence of the pattern `STEADY` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i` to index `j + 1`.
- **Example**: `(2, (1, 1, 7, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6))`

Figure 3.843 provides an example where the `MIN_SURF_STEADY (2, [1, 1, 7, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6])` constraint holds.

**Typical**

- `|VARIABLES| > 1`

**Symmetry**

- Items of `VARIABLES` can be reversed.

**Arg. properties**

- Functional dependency: `VALUE` determined by `VARIABLES`.
Figure 3.843: Illustrating the MIN_SURF_STEADY constraint of the Example slot
Automaton

Figures 3.844 and 3.845 respectively depict the automaton associated with the constraint MIN_SURF_STEADY and its simplified form.

\[
C \leftarrow \text{default} \\
D \leftarrow 0 \\
R \leftarrow \text{default}
\]

\[
\begin{cases}
D \leftarrow 0 \\
R \leftarrow \min(R, D + \text{VAR}_i + \text{VAR}_{i+1})
\end{cases}
\]

\[
\min(R, C)
\]

Figure 3.844: Automaton for the MIN_SURF_STEADY constraint obtained by applying decoration Table 2.35 to the seed transducer of the STEADY pattern where \text{default} is \(+\infty\).

\[
\{R \leftarrow \text{default}\}
\]

\[
\{R \leftarrow \min(R, \text{VAR}_i + \text{VAR}_{i+1})\}
\]

\[
R
\]

Figure 3.845: Automaton for the MIN_SURF_STEADY constraint obtained by applying decoration Table 2.38 to the seed transducer of the STEADY pattern where \text{default} is \(+\infty\); \(-R_i + R_{i-1} \geq 0\) is a linear invariant.

\[
\begin{array}{|c|}
\hline
s & \min(\frac{\overline{C}}{C}, \frac{\overline{C}}{C}) \\
\hline
\end{array}
\]

Table 3.100: Glue matrix for the MIN_SURF_STEADY constraint defined as the composition of the STEADY pattern, the feature SURF, and the aggregator \text{min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### 3.363 MIN_SURF_STEADY_SEQUENCE

**Description**

Based on the `STEADY_SEQUENCE` pattern.

**Constraint**

`MIN_SURF_STEADY_SEQUENCE(VALUE, VARIABLES)`

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**

\[
\begin{align*}
sv & \leq 1 \Rightarrow VALUE = +\infty \\
rv & = 1 \Rightarrow VALUE \geq sv \cdot \minv \& \\
rv & \geq 2 \Rightarrow VALUE \geq \min(2 \cdot \minv, sv \cdot \minv) \\
rv & = 1 \Rightarrow VALUE = +\infty \lor VALUE \leq sv \cdot \maxv \\
rv & \geq 2 \Rightarrow VALUE = +\infty \lor VALUE \leq \max(2 \cdot \maxv, sv \cdot \maxv)
\end{align*}
\]

**Purpose**

VALUE is the minimal surface of occurrences of the `STEADY_SEQUENCE` pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value $+\infty$.

An occurrence of the pattern `STEADY_SEQUENCE` is the maximal subsequence which matches the regular expression $=\infty$.

Assume that the occurrence of the pattern `STEADY_SEQUENCE` starts at position $i$ and ends at position $j$. The feature `SURF` computes the sum of the values from index $i$ to index $j + 1$.

**Example**

\[(2, (3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1))\]

Figure 3.846 provides an example where the `MIN_SURF_STEADY_SEQUENCE (2, [3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1])` constraint holds.

**Typical**

$|\text{VARIABLES}| > 1$

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.846: Illustrating the MIN_SURF_STEADY_SEQUENCE constraint of the Example slot
Figures 3.847 and 3.848 respectively depict the automaton associated with the constraint MIN_SURF_STEADY_SEQUENCE and its simplified form.

Figure 3.847: Automaton for the MIN_SURF_STEADY_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the STEADY_SEQUENCE pattern where default is $+\infty$.

Figure 3.848: Automaton for the MIN_SURF_STEADY_SEQUENCE constraint obtained by applying decoration Table 2.24 to the seed transducer of the STEADY_SEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.101: Glue matrix for the `MIN_SURF_STEADY_SEQUENCE` constraint defined as the composition of the `STEADY_SEQUENCE` pattern, the feature `SURF`, and the aggregator `min`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>(s)</th>
<th>(r)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
<tr>
<td>(r)</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{C} + \overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i)</td>
</tr>
</tbody>
</table>
3.364 MIN_SURF_STRICTLY_DECREASING_SEQUENCE

DESCRIPTION

Origin

Based on the STRICTLY_DECREASING_SEQUENCE pattern.

Constraint

MIN_SURF_STRICTLY_DECREASING_SEQUENCE(VALUE, VARIABLES)

Arguments

VALUE : dvar
VARIABLES : collection(var–dvar)

Restrictions

sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = +∞
minv < 0 ⇒ VALUE ≥ ℓ₁ * minv + [ℓ₁ * (ℓ₁ − 1)]/2
minv ≥ 0 ⇒ VALUE ≥ 2 * minv + 1
maxv > 0 ⇒ VALUE = +∞ ∨ VALUE ≤ ℓ₂ * maxv − [ℓ₂ * (ℓ₂ − 1)]/2
maxv ≤ 0 ⇒ VALUE = +∞ ∨ VALUE ≤ 2 * maxv − 1
where

minv = minval(VARIABLES.var)
rv = range(VARIABLES.var)
sv = |VARIABLES|
ℓ₁ = min(min(sv, rv), |minv|)
ℓ₂ = min(min(sv, rv), |maxv|)
maxv = maxval(VARIABLES.var)

VALUE is the minimal surface of occurrences of the STRICTLY_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +∞.

An occurrence of the pattern STRICTLY_DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression >+.

Assume that the occurrence of the pattern STRICTLY_DECREASING_SEQUENCE starts at position i and ends at position j. The feature SURF computes the sum of the values from index i to index j + 1.

Example

(7, ⟨4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3⟩)

Figure 3.849 provides an example where the MIN_SURF_STRICTLY_DECREASING_SEQUENCE (7, ⟨4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3⟩) constraint holds.

Typical

|VARIABLES| > 1
range(VARIABLES.var) > 1

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 3.849: Illustrating the \texttt{MIN\_SURF\_STRICTLY\_DECREASING\_SEQUENCE} constraint of the \texttt{Example} slot
Automaton

Figures 3.850 and 3.851 respectively depict the automaton associated with the constraint `MIN_SURF_STRICTLY_DECREASING_SEQUENCE` and its simplified form.

\[
\begin{align*}
\{ & C \leftarrow \text{default} \\
& D \leftarrow 0 \\
& R \leftarrow \text{default} \\
\} \\
\leq & s \\
\leq & \{ & C \leftarrow \text{default} \\
& D \leftarrow 0 \\
& R \leftarrow \min(R, C) \\
\} \quad \{ & C \leftarrow D + \text{VAR}_i + \text{VAR}_{i+1} \\
& D \leftarrow 0 \\
\} \\
> & p \\
> & \{ & C \leftarrow C + D + \text{VAR}_{i+1} \\
& D \leftarrow 0 \\
\} \\
\leq & \{ & C \leftarrow \text{default} \\
& R \leftarrow \min(R, C) \\
\} \\
\leq & s \\
\end{align*}
\]

Figure 3.850: Automaton for the `MIN_SURF_STRICTLY_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `STRICTLY_DECREASING_SEQUENCE` pattern where `default` is `+\infty`.

\[
\begin{align*}
\{ & C \leftarrow \text{default} \\
& R \leftarrow \text{default} \\
\} \\
\leq & s \\
\leq & \{ & C \leftarrow \text{default} \\
& R \leftarrow \min(R, C) \\
\} \quad \{ & C \leftarrow \text{VAR}_i + \text{VAR}_{i+1} \\
\} \\
> & p \\
> & \{ & C \leftarrow C + \text{VAR}_{i+1} \\
\} \\
\leq & \{ & C \leftarrow \text{default} \\
& R \leftarrow \min(R, C) \\
\} \\
\leq & s \\
\end{align*}
\]

Figure 3.851: Automaton for the `MIN_SURF_STRICTLY_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 2.24 to the seed transducer of the `STRICTLY_DECREASING_SEQUENCE` pattern where `default` is `+\infty`; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.102: Glue matrix for the MIN_SURF_STRICTLY_DECREASING_SEQUENCE constraint defined as the composition of the STRICTLY_DECREASING_SEQUENCE pattern, the feature SURF, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.365 MIN_SURF_STRICTLY_INCREASING_SEQUENCE

**DESCRIPTION**

**Origin**
Based on the STRICTLY_INCREASING_SEQUENCE pattern.

**Constraint**
MIN_SURF_STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**

<table>
<thead>
<tr>
<th>VALUE</th>
<th>dvar</th>
</tr>
</thead>
<tbody>
<tr>
<td>VARIABLES</td>
<td>collection(var–dvar)</td>
</tr>
</tbody>
</table>

**Restrictions**

\[
\begin{align*}
sv \leq 1 \lor rv \leq 1 & \Rightarrow VALUE = +\infty \\
minv < 0 & \Rightarrow VALUE \geq \ell_1 + minv + [\ell_1 \times (\ell_1 - 1)/2] \\
minv \geq 0 & \Rightarrow VALUE \geq 2 \times minv + 1 \\
maxv > 0 & \Rightarrow VALUE = +\infty \lor VALUE \leq \ell_2 + maxv - [\ell_2 \times (\ell_2 - 1)/2] \\
maxv \leq 0 & \Rightarrow VALUE = +\infty \lor VALUE \leq 2 \times maxv - 1
\end{align*}
\]

where

\[
\begin{align*}
minv &= \text{minval}(\text{VARIABLES}.\text{var}) \\
rv &= \text{range}(\text{VARIABLES}.\text{var}) \\
sv &= |\text{VARIABLES}| \\
\ell_1 &= \min(\min(sv, rv), |minv|) \\
\ell_2 &= \min(\min(sv, rv), |maxv|) \\
maxv &= \text{maxval}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

VALUE is the minimal surface of occurrences of the STRICTLY_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value +\infty.

An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression \(<^{+}\).

Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i\) to index \(j + 1\).

**Example**

\((6, 4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3)\)

Figure 3.852 provides an example where the MIN_SURF_STRICTLY_INCREASING_SEQUENCE \((6, [4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3])\) constraint holds.

**Typical**

\[|\text{VARIABLES}| > 1\]
\[\text{range}(\text{VARIABLES}.\text{var}) > 1\]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.852: Illustrating the `MIN_SURF_STRICTLY_INCREASING_SEQUENCE` constraint of the **Example** slot
Figures 3.853 and 3.854 respectively depict the automaton associated with the constraint MIN_SURF STRICTLY INCREASING SEQUENCE and its simplified form.

Figure 3.853: Automaton for the MIN_SURF STRICTLY INCREASING SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the STRICTLY INCREASING SEQUENCE pattern where default is $+\infty$.

Figure 3.854: Automaton for the MIN_SURF STRICTLY INCREASING SEQUENCE constraint obtained by applying decoration Table 2.24 to the seed transducer of the STRICTLY INCREASING SEQUENCE pattern where default is $+\infty$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.103: Glue matrix for the MIN_SURF_STRICTLY_INCREASING_SEQUENCE constraint defined as the composition of the STRICTLY_INCREASING_SEQUENCE pattern, the feature SURF, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.366 MIN_SURF_SUMMIT

**DESCRIPTION**

- **Origin**: Based on the SUMMIT pattern.

- **Constraint**: MIN_SURF_SUMMIT(VALUE, VARIABLES)

- **Arguments**
  - VALUE : dvar
  - VARIABLES : collection(var−dvar)

- **Restrictions**
  - $sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = +\infty$
  - $rv = 2 \Rightarrow VALUE \geq minv + 1$
  - $rv \geq 3 \Rightarrow VALUE \geq \min(minv + 1, (sv - 2) \times (minv + 1) + 12)$
  - $rv = 2 \Rightarrow VALUE = +\infty \lor VALUE \leq maxv$
  - $rv \geq 3 \Rightarrow VALUE = +\infty \lor VALUE \leq \max(maxv, (sv - 2) \times (maxv - 1) + 1)$

- **Purpose**
  - $VALUE$ is the minimal surface of occurrences of the SUMMIT pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, $VALUE$ takes the default value $+\infty$.
  - An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression $(< | < (= | <) <) (> | > (= | >))$.
  - Assume that the occurrence of the pattern SUMMIT starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 1$ to index $j$.

- **Example**
  - $(3, (7, 1, 5, 4, 3, 3, 4, 6, 2, 3, 4, 2, 3, 1))$

- **Typical**
  - $|VARIABLES| > 2$
  - range(VARIABLES.var) > 1

- **Symmetry**
  - Items of VARIABLES can be reversed.

- **Arg. properties**
  - Functional dependency: VALUE determined by VARIABLES.

**AUTOMATON**

The automaton describes the transitions based on the specified patterns and constraints. The pattern $(< | < (= | <) <) (> | > (= | >))$ is shown with arrows indicating the transitions between states based on the values of $VALUE$.
Figure 3.855: Illustrating the MIN_SURF_SUMMIT constraint of the Example slot
Automaton

Figure 3.856 depicts the automaton associated with the constraint MIN_SURF_SUMMIT.

\begin{align*}
\{ C \leftarrow \text{default} \} \\
\{ D \leftarrow 0 \} \\
\{ R \leftarrow \text{default} \}
\end{align*}

Figure 3.856: Automaton for the MIN_SURF_SUMMIT constraint obtained by applying decoration Table 2.35 to the seed transducer of the SUMMIT pattern where default is $+\infty$ (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$); $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.104: Glue matrix for the `MIN_SURF_SUMMIT` constraint defined as the composition of the `SUMMIT` pattern, the feature `SURF`, and the aggregator `min`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.367 MIN_SURF_VALLEY

**DESCRIPTION AUTOMATON**

**Origin**
Based on the VALLEY pattern.

**Constraint**
MIN_SURF_VALLEY(VALUE, VARIABLES)

**Arguments**
VALUE : dvar
VARIABLES : collection(var−dvar)

**Restrictions**

\[
\text{sv} \leq 2 \vee \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} \geq \text{min}(\text{sv}, (\text{sv} - 2) + \text{minv}) \\
\text{VALUE} = +\infty \vee \text{VALUE} \leq \text{max}(\text{sv} - 2, (\text{sv} - 2) + (\text{maxv} - 1))
\]

required(VARIABLES, var)

where

\[
\text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{sv} = |\text{VARIABLES}| \\
\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{rv} = \text{range}(\text{VARIABLES}.\text{var})
\]

**Purpose**
An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression \( (> | >)^* (< | =)^* < \).
Assume that the occurrence of the pattern VALLEY starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i + 1 \) to index \( j \).

**Example**

\((7, (1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 5, 7))\)

Figure 3.857 provides an example where the MIN_SURF_VALLEY (7, [1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 5, 7]) constraint holds.

**Typical**

\(|\text{VARIABLES}| > 2 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.857: Illustrating the MIN_SURF_VALLEY constraint of the Example slot
Figure 3.858 depicts the automaton associated with the constraint MIN_SURF_VALLEY.

![Automaton Diagram]

Figure 3.858: Automaton for the MIN_SURF_VALLEY constraint obtained by applying decoration Table 2.35 to the seed transducer of the VALLEY pattern where *default* is \(+\infty\); \(-R_i + R_{i-1} \geq 0\) is a linear invariant.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
<tr>
<td>r</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i)</td>
<td>(\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i)</td>
</tr>
<tr>
<td>t</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i)</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
</tbody>
</table>

Table 3.105: Glue matrix for the MIN_SURF_VALLEY constraint defined as the composition of the VALLEY pattern, the feature SURF, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.368 MIN_SURF_ZIGZAG

### DESCRIPTION AUTOMATON

<table>
<thead>
<tr>
<th>Origin</th>
<th>Based on the ZIGZAG pattern.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constraint</td>
<td>MIN_SURF_ZIGZAG(VALUE, VARIABLES)</td>
</tr>
<tr>
<td>Arguments</td>
<td>VALUE : dvar</td>
</tr>
<tr>
<td>VARIABLES : collection(var−dvar)</td>
<td></td>
</tr>
</tbody>
</table>

#### Restrictions

\[
\begin{align*}
sv & \leq 3 \lor rv \leq 1 \Rightarrow VALUE = +\infty \\
VALUE & \geq \min \left( 2 \times \text{minv} + 10, \frac{2}{3} \times (sv - 1) \times \text{minv} + \frac{2}{3} \times (sv - 2) \times \text{minv} + 1 \right) \\
VALUE & \leq \max \left( 2 \times \text{maxv} - 1, \frac{2}{3} \times (sv - 1) \times \text{maxv} + \frac{2}{3} \times (sv - 2) \times \text{maxv} - 1 \right) \\
\end{align*}
\]

where

\[
\begin{align*}
\text{minv} &= \minval(VARIABLES.var) \\
sv &= |VARIABLES| \\
\text{maxv} &= \maxval(VARIABLES.var) \\
rv &= \range(VARIABLES.var) \\
\end{align*}
\]

#### Purpose

The occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression \((<>)^+(< | <>) | (>><)^+ (> | >>)\). Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

#### Example

\((5, (4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1))\)

Figure 3.859 provides an example where the MIN_SURF_ZIGZAG \((5, [4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1])\) constraint holds.

#### Typical

\[
\begin{align*}
|VARIABLES| & > 3 \\
\range(VARIABLES.var) & > 1 \\
\end{align*}
\]

#### Symmetry

Items of VARIABLES can be reversed.

#### Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 3.859: Illustrating the MIN_SURF_ZIGZAG constraint of the Example slot
Automaton

Figures 3.860 and 3.861 respectively depict the automaton associated with the constraint MIN_SURF_ZIGZAG and its simplified form.
Figure 3.860: Automaton for the MIN_SURF_ZIGZAG constraint obtained by applying decoration Table 2.35 to the seed transducer of the ZIGZAG pattern where default is $+\infty$; (1) missing transitions from $a$, $b$, $c$, $d$, $e$, $f$ to $s$ are labelled by $=$; (2) on transitions from $b$, $c$, $e$, $f$ to $s$ the accumulator $D$ is reset to its initial value; (3) on transitions from $c$, $f$ to $s$ the accumulator $R$ is updated wrt $C$ and the accumulator $C$ is reset to its initial value.
Figure 3.861: Automaton for the MIN_SURF_ZIGZAG constraint obtained by applying decoration Table 2.23 to the seed transducer of the ZIGZAG pattern where default is $+\infty$; missing transitions from $a$, $b$, $c$, $d$, $e$, $f$ to $s$ are labelled by $=$; (2) on transitions from $b$, $c$, $e$, $f$ to $s$ the accumulator $D$ is reset to its initial value; (3) on transitions from $c$, $f$ to $s$ the accumulator $R$ is updated wrt $C$ and the accumulator $C$ is reset to its initial value; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.106: Glue matrix for the MIN-SURF-ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature SURF, and the aggregator min.; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>min((\overline{C}, \overline{C}))</td>
</tr>
<tr>
<td>a</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>(\overline{D} + \overline{D} + \text{VAR}_i)</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>(\overline{D} + \overline{D} + \text{VAR}_i)</td>
<td>min((\overline{C}, \overline{C}))</td>
</tr>
<tr>
<td>b</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>(\overline{D} + \overline{D} + \text{VAR}_i)</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>(\overline{D} + \overline{D} + \text{VAR}_i)</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>(\overline{C} + \overline{D} + \overline{D} + \text{VAR}_i)</td>
</tr>
<tr>
<td>c</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>(\overline{C} + \overline{D} + \overline{D} + \text{VAR}_i)</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>(\overline{C} + \overline{D} + \overline{D} + \text{VAR}_i)</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>(\overline{C} + \overline{D} + \overline{D} + \text{VAR}_i)</td>
<td>min((\overline{C}, \overline{C}))</td>
</tr>
<tr>
<td>d</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>(\overline{D} + \overline{D} + \text{VAR}_i)</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>(\overline{C} + \overline{D} + \overline{D} + \text{VAR}_i)</td>
<td>min((\overline{C}, \overline{C}))</td>
</tr>
<tr>
<td>e</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>(\overline{D} + \overline{D} + \text{VAR}_i)</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>(\overline{D} + \overline{D} + \text{VAR}_i)</td>
</tr>
<tr>
<td>f</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>(\overline{C} + \overline{D} + \overline{D} + \text{VAR}_i)</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>(\overline{C} + \overline{D} + \overline{D} + \text{VAR}_i)</td>
<td>min((\overline{C}, \overline{C}))</td>
<td>(\overline{C} + \overline{C} + \overline{D} + \overline{D} + \text{VAR}_i)</td>
</tr>
</tbody>
</table>
3.369 MIN_WIDTH_DECREASING_SEQUENCE

DESCRIPTION AUTOMATON

Origin
Based on the DECREASING_SEQUENCE pattern.

Constraint
MIN_WIDTH_DECREASING_SEQUENCE(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = sv + 1
VALUE ≥ 2
rv = 2 ⇒ VALUE = sv + 1 ∨ VALUE ≤ 2 ∧
rv ≥ 3 ⇒ VALUE = sv + 1 ∨ VALUE ≤ sv ∧
required(VARIABLES, var)
where
sv = |VARIABLES|
rv = range(VARIABLES, var)

Purpose
VALUE is the minimal width of occurrences of the DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value |VARIABLES| + 1.

An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression > (> |=)* > | >.
Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i + 2 \).

Example
\[(2, (3, 4, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))\]

Figure 3.862 provides an example where the MIN_WIDTH_DECREASING_SEQUENCE (2, [3, 4, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4]) constraint holds.

Typical
|VARIABLES| > 1
range(VARIABLES, var) > 1

Symmetry
One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 3.862: Illustrating the MIN_WIDTH_DECREASING_SEQUENCE constraint of the Example slot
Automaton  Figure 3.863 depicts the automaton associated with the constraint MIN_WIDTH_DECREASING_SEQUENCE.

\[
\begin{align*}
\text{MIN_WIDTH_DECREASING_SEQUENCE} & \quad \text{MIN_WIDTH_DECREASING_SEQUENCE} \\
\leq s & \quad \leq \\
\text{Figure 3.863: Automaton for the MIN_WIDTH_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the DECREASING_SEQUENCE pattern where default is } |\text{VARIABLES}| + 1; -R_i + R_{i-1} \geq 0 \text{ is a linear invariant.} \\
\end{align*}
\]

Table 3.107: Glue matrix for the MIN_WIDTH_DECREASING_SEQUENCE constraint defined as the composition of the DECREASING_SEQUENCE pattern, the feature WIDTH, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.370  MIN_WIDTH_DECREASING_TERRACE

**DESCRIPTION**

**Origin**  
Based on the `DECREASING_TERRACE` pattern.

**Constraint**  
`MIN_WIDTH_DECREASING_TERRACE(VALUE, VARIABLES)`

**Arguments**  
- `VALUE`: `dvar`
- `VARIABLES`: `collection(var−dvar)`

**Restrictions**  
- `sv ≤ 3 ∨ rv ≤ 2` \(⇒\) `VALUE = sv + 1`
- `VALUE ≥ 2`
- `VALUE = sv + 1 ∨ VALUE ≤ sv − 2 \&\& required(VARIABLES, var)`

where
- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

**Purpose**  
An occurrence of the pattern `DECREASING_TERRACE` is the maximal subsequence which matches the regular expression `>\=+>`. Assume that the occurrence of the pattern `DECREASING_TERRACE` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j − i`.

**Example**  

```
(2, (6, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3))
```

Figure 3.864 provides an example where the `MIN_WIDTH_DECREASING_TERRACE` (2, [6, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3]) constraint holds.

**Typical**  
- `|VARIABLES| > 3`
- `range(VARIABLES.var) > 2`

**Symmetry**  
One and the same constant can be added to the `var` attribute of all items of `VARIABLES`.

**Arg. properties**  
Functional dependency: `VALUE` determined by `VARIABLES`. 
Figure 3.864: Illustrating the MIN_WIDTH_DECREASING_TERRACE constraint of the Example slot
Figures 3.865 and 3.866 respectively depict the automaton associated with the constraint MIN_WIDTH_DECREASING_TERRACE and its simplified form.

Figure 3.865: Automaton for the MIN_WIDTH_DECREASING_TERRACE constraint obtained by applying decoration Table 2.35 to the seed transducer of the DECREASING_TERRACE pattern where default is $\abs{\text{VARIABLES}} + 1$

<table>
<thead>
<tr>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
</tr>
</tbody>
</table>

Table 3.108: Glue matrix for the MIN_WIDTH_DECREASING_TERRACE constraint defined as the composition of the DECREASING_TERRACE pattern, the feature WIDTH, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 3.866: Automaton for the MIN_WIDTH_DECREASING_TERRACE constraint obtained by applying decoration Table 2.25 to the seed transducer of the DECREASING_TERRACE pattern where default is \(|\text{VARIABLES}| + 1; -R_i + R_{i-1} \geq 0\) is a linear invariant.
3.371 MIN_WIDTH_GORGE

**Origin**
Based on the GORGE pattern.

**Constraint**
MIN_WIDTH_GORGE(VALUE, VARIABLES)

**Arguments**
- VALUE : `dvar`
- VARIABLES : `collection(var−dvar)`

**Restrictions**
\[
sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = sv + 1 \\
VALUE \geq 1 \\
rv = 2 \Rightarrow VALUE = sv + 1 \lor VALUE \leq 1 \\
rv \geq 3 \Rightarrow VALUE = sv + 1 \lor VALUE \leq sv - 2 \\
\text{required}(VARIABLES, var) \\
\text{where} \\
sv = |VARIABLES| \\
rv = \text{range}(VARIABLES, var)
\]

**Purpose**
VALUE is the minimal width of occurrences of the GORGE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value $|VARIABLES| + 1$.

An occurrence of the pattern GORGE is the *maximal* subsequence which matches the regular expression $(>|=(|>)*)<(>|<(|=)<)$. Assume that the occurrence of the pattern GORGE starts at position $i$ and ends at position $j$. The feature WIDTH computes the value $j - i$.

**Example**
\[
(1, (1, 7, 3, 4, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7))
\]

Figure 3.867 provides an example where the MIN_WIDTH_GORGE (1, [1, 7, 3, 4, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7]) constraint holds.

**Typical**
- $|VARIABLES| > 2$
- $\text{range}(VARIABLES, var) > 1$

**Symmetries**
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.867: Illustrating the MIN_WIDTH_GORGE constraint of the Example slot
Automaton

Figure 3.868 depicts the automaton associated with the constraint MIN_WIDTH_GORGE.

Figure 3.868: Automaton for the MIN_WIDTH_GORGE constraint obtained by applying decoration Table 2.35 to the seed transducer of the GORGE pattern where default is |VARIABLES| + 1 (transition u → r has the same accumulator update as transition r → u); \(-R_i + R_{i-1} \geq 0\) is a linear invariant.
Table 3.109: Glue matrix for the MIN_WIDTH_GORGE constraint defined as the composition of the GORGE pattern, the feature WIDTH, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.372 **MIN\_WIDTH\_INCREASING\_SEQUENCE**

**DESCRIPTION**

**Origin**

Based on the **INCREASING\_SEQUENCE** pattern.

**Constraint**

\[
\text{MIN\_WIDTH\_INCREASING\_SEQUENCE}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var}\,-\text{dvar})
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{sv} \leq 1 & \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = \text{sv} + 1 \\
\text{VALUE} \geq 2 & \\
\text{sv} \leq 1 & \lor (\text{rv} = 2 \Rightarrow \text{VALUE} = \text{sv} + 1 \lor \text{VALUE} \leq 2) \\
\text{rv} \geq 3 & \Rightarrow \text{VALUE} = \text{sv} + 1 \lor \text{VALUE} \leq \text{sv} \\
\text{required}(&\text{VARIABLES, var})
\end{align*}
\]

where

\[
\begin{align*}
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

VALUE is the minimal width of occurrences of the **INCREASING\_SEQUENCE** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, VALUE takes the default value \(|\text{VARIABLES}| + 1\).

An occurrence of the pattern **INCREASING\_SEQUENCE** is the maximal subsequence which matches the regular expression \(< (\text{< }=)^{+} \text{< }<\).

Assume that the occurrence of the pattern **INCREASING\_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **WIDTH** computes the value \(j - i + 2\).

**Example**

\[
(2, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))
\]

Figure 3.869 provides an example where the **MIN\_WIDTH\_INCREASING\_SEQUENCE** \((2, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3])\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Symmetry**

One and the same constant can be added to the \text{var} attribute of all items of **VARIABLES**.

**Arg. properties**

Functional dependency: VALUE determined by **VARIABLES**.
Figure 3.869: Illustrating the MIN_WIDTH_INCREASING_SEQUENCE constraint of the **Example** slot
Figure 3.870 depicts the automaton associated with the constraint \texttt{MIN\_WIDTH\_INCREASING\_SEQUENCE}.

\begin{verbatim}
\{ C \leftarrow \text{default} \\
\quad D \leftarrow 0 \\
\quad R \leftarrow \text{default}
\}

\end{verbatim}

\begin{verbatim}
\geq s \\
\geq
\end{verbatim}

\begin{verbatim}
\{ C \leftarrow \text{default} \\
\quad D \leftarrow 0 \\
\quad R \leftarrow \min(R,C)
\}

\min(R,C)

\begin{verbatim}
\leq t
\end{verbatim}

\begin{verbatim}
\{ D \leftarrow D + 1
\}
\end{verbatim}

\begin{verbatim}
\{ C \leftarrow \text{default} \\
\quad D \leftarrow 0 \\
\quad R \leftarrow \min(R,C)
\}

\min(R,C)

\begin{verbatim}
\leq \{ C \leftarrow D + 2 \\
\quad D \leftarrow 0
\}
\end{verbatim}

\begin{verbatim}
\geq \{ C \leftarrow D + 2 \\
\quad D \leftarrow 0
\}
\end{verbatim}

\begin{verbatim}
\leq \{ C \leftarrow C + D + 1 \\
\quad D \leftarrow 0
\}
\end{verbatim}

\begin{verbatim}
\geq \{ D \leftarrow D + 1
\}
\end{verbatim}

Figure 3.870: Automaton for the \texttt{MIN\_WIDTH\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{INCREASING\_SEQUENCE} pattern where \texttt{default} is \(|\text{VARIABLES}| + 1; -R_i + R_{i-1} \geq 0\) is a linear invariant.

\begin{verbatim}
\begin{tabular}{|c|c|}
\hline
\text{\texttt{s}} & \text{\texttt{t}} \\
\hline
\min(\overrightarrow{C}, \overrightarrow{C}) & \min(\overrightarrow{C}, \overrightarrow{C}) \\
\min(\overrightarrow{C}, \overrightarrow{C}) & \overrightarrow{C} + \overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + 1 \\
\hline
\end{tabular}
\end{verbatim}

Table 3.110: Glue matrix for the \texttt{MIN\_WIDTH\_INCREASING\_SEQUENCE} constraint defined as the composition of the \texttt{INCREASING\_SEQUENCE} pattern, the feature \texttt{WIDTH}, and the \texttt{aggregator min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### 3.373 MIN_WIDTH_INCREASING_TERRACE

**Origin** Based on the INCREASING_TERRACE pattern.

**Constraint**

\[
\text{MIN_WIDTH_INCREASING_TERRACE}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**

\[
\begin{align*}
\text{sv} &\leq 3 \vee \text{rv} \leq 2 \Rightarrow \text{VALUE} = \text{sv} + 1 \\
\text{VALUE} &\geq 2 \\
\text{VALUE} = \text{sv} + 1 \vee \text{VALUE} \leq \text{sv} - 2 &\text{ required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where

\[
\begin{align*}
\text{sv} &= |\text{VARIABLES}| \\
\text{rv} &= \text{range}(\text{VARIABLES}, \text{var})
\end{align*}
\]

**Purpose**

An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression \(<=^+<\).

Assume that the occurrence of the pattern INCREASING_TERRACE starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

**Example**

\[(2, (1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 4, 4))\]

Figure 3.871 provides an example where the MIN_WIDTH_INCREASING_TERRACE (2, [1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 4, 4]) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 3 \\
\text{range}(\text{VARIABLES}, \text{var}) > 2
\]

**Symmetry**

One and the same constant can be added to the \text{var} attribute of all items of \text{VARIABLES}.

**Arg. properties** Functional dependency: VALUE determined by VARIABLES.
Figure 3.871: Illustrating the MIN_WIDTH_INCREASING_TERRACE constraint of the Example slot
Automaton

Figures 3.872 and 3.873 respectively depict the automaton associated with the constraint MIN\_WIDTH\_INCREASING\_TERRACE and its simplified form.

\[
\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow \text{default}
\end{align*}
\]

\[
\begin{align*}
\geq s &< r = t \leftarrow \text{default} \\
\geq &\begin{cases}
\{D \leftarrow D + 1\} \\
\{D \leftarrow 0\} \\
\{R \leftarrow \min(R, D + 1)\}
\end{cases}
\end{align*}
\]

\[
\begin{align*}
\min(R, C) &\leftarrow D + 1 \\
\min(R, C) &\leftarrow D + 1 \\
\min(R, C) &\leftarrow D + 1
\end{align*}
\]

Figure 3.872: Automaton for the MIN\_WIDTH\_INCREASING\_TERRACE constraint obtained by applying decoration Table 2.35 to the seed transducer of the INCREASING\_TERRACE pattern where default is $|\text{VARIABLES}| + 1$

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\min(C, \overline{C})$</td>
<td>$\min(C, \overline{C})$</td>
<td>$\min(C, \overline{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\min(C, \overline{C})$</td>
<td>$\min(C, \overline{C})$</td>
<td>$\overline{D} + D + 1$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\min(C, \overline{C})$</td>
<td>$\overline{D} + D + 1$</td>
<td>$\overline{D} + D + 1$</td>
</tr>
</tbody>
</table>

Table 3.111: Glue matrix for the MIN\_WIDTH\_INCREASING\_TERRACE constraint defined as the composition of the INCREASING\_TERRACE pattern, the feature WIDTH, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 3.873: Automaton for the MIN_WIDTH_INCREASING_TERRACE constraint obtained by applying decoration Table 2.25 to the seed transducer of the INCREASING_TERRACE pattern where default is |VARIABLES| + 1; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
3.374 MIN_WIDTH_INFLEXION

**DESCRIPTION**

**Origin**

Based on the INFLEXION pattern.

**Constraint**

\[
\text{MIN_WIDTH_INFLEXION}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

\[
\begin{align*}
\text{VALUE} &: \text{dvar} \\
\text{VARIABLES} &: \text{collection}(\text{var} - \text{dvar})
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
sv &\leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = sv + 1 \\
\text{VALUE} &\geq 1 \\
\text{VALUE} &= sv + 1 \lor \text{VALUE} \leq sv - 2 \\
\text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where

\[
\begin{align*}
sv &= |\text{VARIABLES}| \\
rv &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

An occurrence of the pattern INFLEXION is the \textit{maximal} subsequence which matches the regular expression \(< (< | =)^* > | > (> | =)^* <\). Assume that the occurrence of the pattern INFLEXION starts at position \(i\) and ends at position \(j\). The feature \textit{WIDTH} computes the value \(j - i\).

**Example**

\[(1, (1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4))\]

Figure 3.874 provides an example where the MIN_WIDTH_INFLEXION \((1, [1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4])\) constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| &> 2 \\
\text{range}(\text{VARIABLES}.\text{var}) &> 1
\end{align*}
\]

**Symmetries**

- Items of \text{VARIABLES} can be reversed.
- One and the same constant can be added to the \text{var} attribute of all items of \text{VARIABLES}.

**Arg. properties**

Functional dependency: \text{VALUE} determined by \text{VARIABLES}.
Figure 3.874: Illustrating the MIN_WIDTH_INFLEXION constraint of the Example slot
Automaton

Figures 3.875 and 3.876 respectively depict the automaton associated with the constraint MIN_WIDTH_INFLEXION and its simplified form.

Figure 3.875: Automaton for the MIN_WIDTH_INFLEXION constraint obtained by applying decoration Table 2.35 to the seed transducer of the INFLEXION pattern where default is $|\text{VARIABLES}| + 1$ (transition $r \rightarrow t$ has the same accumulators updates as transition $t \rightarrow r$).

Figure 3.876: Automaton for the MIN_WIDTH_INFLEXION constraint obtained by applying decoration Table 2.25 to the seed transducer of the INFLEXION pattern where default is $|\text{VARIABLES}| + 1$ (transition $r \rightarrow t$ has the same accumulators updates as transition $t \rightarrow r$); $-R_i + R_{i-1} \geq 0$ is a linear invariant.
3.375 MIN_WIDTH_PEAK

**Origin**
Based on the PEAK pattern.

**Constraint**
MIN_WIDTH_PEAK(VALUE, VARIABLES)

**Arguments**
- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**
- \( sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = sv + 1 \)
- \( VALUE \geq 1 \)
- \( VALUE = sv + 1 \lor VALUE \leq sv - 2sv\) required(VARIABLES, var)

where
- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

**Purpose**
VALUE is the minimal width of occurrences of the PEAK pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(|\text{VARIABLES}| + 1\).

An occurrence of the pattern **PEAK** is the maximal subsequence which matches the regular expression \(< (= | <)* (> | =)* >\).
Assume that the occurrence of the pattern **PEAK** starts at position \(i\) and ends at position \(j\). The feature **WIDTH** computes the value \(j - i\).

**Example**
\( (2, \{7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 3, 1\}) \)

Figure 3.877 provides an example where the MIN_WIDTH_PEAK \((2, \{7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 3, 1\})\) constraint holds.

**Typical**
- \(|\text{VARIABLES}| > 2\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Symmetries**
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.877: Illustrating the MIN_WIDTH_PEAK constraint of the Example slot
Automaton

Figure 3.878 depicts the automaton associated with the constraint MIN_WIDTH_PEAK.

\[
\begin{align*}
\{ C \leftarrow \text{default} \\
\{ D \leftarrow 0 \\
\{ R \leftarrow \text{default} \}
\end{align*}
\]

\[
\begin{align*}
\{ C \leftarrow \text{default} \\
\{ D \leftarrow 0 \\
\{ R \leftarrow \text{min}(R, C) \}
\end{align*}
\]

\[
\begin{align*}
\{ C \leftarrow D + 1 \\
\{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow D + 1 \}
\end{align*}
\]

\[
\begin{align*}
\{ C \leftarrow C + D + 1 \}
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow D + 1 \}
\end{align*}
\]

Figure 3.878: Automaton for the MIN_WIDTH_PEAK constraint obtained by applying decoration Table 2.35 to the seed transducer of the PEAK pattern where default is $|\text{VARIABLES}| + 1$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\overrightarrow{B} + \overrightarrow{D} + 1$</td>
<td>$\overrightarrow{C} + \overrightarrow{B} + \overrightarrow{D} + 1$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\overrightarrow{C} + \overrightarrow{B} + \overrightarrow{D} + 1$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
</tbody>
</table>

Table 3.112: Glue matrix for the MIN_WIDTH_PEAK constraint defined as the composition of the PEAK pattern, the feature WIDTH, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.376  MIN_WIDTH_PLAIN

**Description**

**Origin**  
Based on the **PLAIN** pattern.

**Constraint**  
MIN_WIDTH_PLAIN(VALUE, VARIABLES)

**Arguments**  
VALUE : dvar  
VARIABLES : collection(var−dvar)

**Restrictions**  
\[
sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = sv + 1  
\]

\[
VALUE \geq 1  
\]

\[
VALUE = sv + 1 \land VALUE \leq sv - 2  
\]

\[
\text{required}(\text{VARIABLES}, \text{var})  
\]

where  
\[
sv = |\text{VARIABLES}|  
\]

\[
rv = \text{range}(\text{VARIABLES.var})  
\]

**Purpose**  
An occurrence of the pattern **PLAIN** is the **maximal** subsequence which matches the regular expression \( >=^* < \).

Assume that the occurrence of the pattern **PLAIN** starts at position \( i \) and ends at position \( j \). The feature **WIDTH** computes the value \( j - i \).

**Example**  
\[ (1, (2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3)) \]

Figure 3.879 provides an example where the MIN_WIDTH_PLAIN \((1, [2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3])\) constraint holds.

**Typical**  
\[
|\text{VARIABLES}| > 2  
\]

\[
\text{range}(\text{VARIABLES.var}) > 1  
\]

**Symmetries**  
- Items of **VARIABLES** can be **reversed**.
- One and the same constant can be **added** to the **var** attribute of all items of **VARIABLES**.

**Arg. properties**  
Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 3.879: Illustrating the MIN_WIDTH.PLAIN constraint of the Example slot
Figures 3.880 and 3.881 respectively depict the automaton associated with the constraint MIN_WIDTHPLAIN and its simplified form.

Figure 3.880: Automaton for the MIN_WIDTHPLAIN constraint obtained by applying decoration Table 2.35 to the seed transducer of the PLAIN pattern where default is $|\text{VARIABLES}| + 1$

Figure 3.881: Automaton for the MIN_WIDTHPLAIN constraint obtained by applying decoration Table 2.25 to the seed transducer of the PLAIN pattern where default is $|\text{VARIABLES}| + 1$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.113: Glue matrix for the \texttt{MIN\_WIDTH\_PLAIN} constraint defined as the composition of the \texttt{PLAIN} pattern, the feature \texttt{WIDTH}, and the aggregator \texttt{min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### 3.377 MIN_WIDTH_PLATEAU

#### Description

**Origin**
Based on the PLATEAU pattern.

**Constraint**

\[
\text{MIN_WIDTH_PLATEAU}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var-dvar)`

**Restrictions**

\[
\begin{align*}
sv & \leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = sv + 1 \\
\text{VALUE} & \geq 1 \\
\text{VALUE} & = sv + 1 \lor \text{VALUE} \leq sv - 2 \\
\text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where

\[
sv = |\text{VARIABLES}|
\]

\[
rv = \text{range(}\text{VARIABLES}.\text{var})
\]

**Purpose**
An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression `<=*`.
Assume that the occurrence of the pattern PLATEAU starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

**Example**

\[
(3, (1, 3, 3, 5, 5, 5, 2, 4, 4, 4, 3, 3, 1, 5, 5))
\]

Figure 3.882 provides an example where the MIN_WIDTH_PLATEAU (3, [1, 3, 3, 5, 5, 5, 2, 4, 4, 4, 3, 3, 1, 5, 5]) constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| & > 2 \\
\text{range(}\text{VARIABLES}.\text{var}) & > 1
\end{align*}
\]

**Symmetries**

- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.882: Illustrating the MIN\_WIDTH\_PLATEAU constraint of the Example slot
Figures 3.883 and 3.884 respectively depict the automaton associated with the constraint MIN_WIDTH PLATEAU and its simplified form.

Figure 3.883: Automaton for the MIN_WIDTH PLATEAU constraint obtained by applying decoration Table 2.35 to the seed transducer of the PLATEAU pattern where default is |VARIABLES| + 1.

Figure 3.884: Automaton for the MIN_WIDTH PLATEAU constraint obtained by applying decoration Table 2.25 to the seed transducer of the PLATEAU pattern where default is |VARIABLES| + 1; \(-R_i + R_{i-1} \geq 0\) is a linear invariant.
Table 3.114: Glue matrix for the \textsc{min\_width\_plateau} constraint defined as the composition of the \textsc{plateau} pattern, the feature \textsc{width}, and the aggregator \texttt{min}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.378 MIN_WIDTH_PROPER_PLAIN

**Description**

Based on the PROPERPlain pattern.

**Constraint**

MIN_WIDTH_PROPER_PLAIN(VALUE, VARIABLES)

**Arguments**

| VALUE  | : dvar |
| VARIABLES | : collection(var–dvar) |

**Restrictions**

\[ sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = sv + 1 \]

\[ VALUE \geq 2 \]

\[ VALUE = sv + 1 \lor VALUE \leq sv - 2 \]

required(VARIABLES, var)

where

\[ sv = |VARIABLES| \]

\[ rv = \text{range}(\text{VARIABLES}.\text{var}) \]

**Purpose**

An occurrence of the pattern PROPERPlain is the maximal subsequence which matches the regular expression > = + <.

Assume that the occurrence of the pattern PROPERPlain starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

**Example**

\[(2, (2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3, 5))\]

Figure 3.885 provides an example where the MIN_WIDTH_PROPER_PLAIN (2, [2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3, 5]) constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 3 \]

\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

**Symmetries**

- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.885: Illustrating the MIN_WIDTH_PROPER_PLAIN constraint of the Example slot
Automaton

Figures 3.886 and 3.887 respectively depict the automaton associated with the constraint MIN_WIDTH_PROPER_PLAIN and its simplified form.

Figure 3.886: Automaton for the MIN_WIDTH_PROPER_PLAIN constraint obtained by applying decoration Table 2.35 to the seed transducer of the PROPER_PLAIN pattern where default is |VARIABLES| + 1

Figure 3.887: Automaton for the MIN_WIDTH_PROPER_PLAIN constraint obtained by applying decoration Table 2.25 to the seed transducer of the PROPER_PLAIN pattern where default is |VARIABLES| + 1; \(-R_i + R_{i-1} \geq 0\) is a linear invariant.
Table 3.115: Glue matrix for the MIN_WIDTH_PROPER_PLAIN constraint defined as the composition of the PROPER_PLAIN pattern, the feature WIDTH, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### 3.379 MIN_WIDTH_PROPER_PLATEAU

#### DESCRIPTION

**Origin**
Based on the PROPER_PLATEAU pattern.

**Constraint**
MIN_WIDTH_PROPER_PLATEAU(VALUE, VARIABLES)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var−dvar)

**Restrictions**
- \( sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = sv + 1 \)
- \( VALUE \geq 2 \)
- \( VALUE = sv + 1 \lor VALUE \leq sv - 2 \)

**Purpose**
An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression \(<\equiv+>\).

Assume that the occurrence of the pattern PROPER_PLATEAU starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

**Example**

\( (2, \{7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3\}) \)

Figure 3.888 provides an example where the MIN_WIDTH_PROPER_PLATEAU (2, \([7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3]\)) constraint holds.

**Typical**
- \(|\text{VARIABLES}| > 3\)
- range(VARIABLES.var) > 1

**Symmetries**
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.888: Illustrating the MIN_WIDTH_PROPER_PLATEAU constraint of the Example slot
Automaton

Figures 3.889 and 3.890 respectively depict the automaton associated with the constraint MIN_WIDTH_PROPER_PLATEAU and its simplified form.

Figure 3.889: Automaton for the MIN_WIDTH_PROPER_PLATEAU constraint obtained by applying decoration Table 2.35 to the seed transducer of the PROPER_PLATEAU pattern where default is |VARIABLES| + 1.

Figure 3.890: Automaton for the MIN_WIDTH_PROPER_PLATEAU constraint obtained by applying decoration Table 2.25 to the seed transducer of the PROPER_PLATEAU pattern where default is |VARIABLES| + 1; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
</tr>
</tbody>
</table>

Table 3.116: Glue matrix for the `MIN_WIDTH_PROPER_PLATEAU` constraint defined as the composition of the `PROPER_PLATEAU` pattern, the feature `WIDTH`, and the aggregator `min`. Cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### MIN_WIDTH_STEADY_SEQUENCE

**Description**

Based on the `STEADY_SEQUENCE` pattern.

**Constraint**

```
MIN_WIDTH_STEADY_SEQUENCE(VALUE, VARIABLES)
```

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var − dvar)`

**Restrictions**

- `sv ≤ 1 ⇒ VALUE = sv + 1`
- `rv = 1 ⇒ VALUE ≥ sv`
- `rv ≥ 2 ⇒ VALUE ≥ 2`
- `VALUE = sv + 1 ∨ VALUE ≤ sv`

where
- `rv = range(VARIABLES.var)`
- `sv = |VARIABLES|`

**Purpose**

VALUE is the minimal width of occurrences of the `STEADY_SEQUENCE` pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value `|VARIABLES| + 1`.

An occurrence of the pattern `STEADY_SEQUENCE` is the maximal subsequence which matches the regular expression `=+`. Assume that the occurrence of the pattern `STEADY_SEQUENCE` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j − i + 2`.

**Example**

```
(2, (3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 3, 2, 1, 1))
```

Figure 3.891 provides an example where the `MIN_WIDTH_STEADY_SEQUENCE` (2, [3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 3, 2, 1, 1]) constraint holds.

**Typical**

```
|VARIABLES| > 1
```

**Symmetries**

- Items of VARIABLES can be reversed.
- One and the same constant can be added to the `var` attribute of all items of VARIABLES.

**Arg. properties**

Functional dependence: VALUE determined by VARIABLES.
Figure 3.891: Illustrating the MIN_WIDTH_STEADY_SEQUENCE constraint of the Example slot
Figures 3.892 and 3.893 respectively depict the automaton associated with the constraint \texttt{MIN\_WIDTH\_STEADY\_SEQUENCE} and its simplified form.

\begin{align*}
&\{ \begin{array}{l}
C \leftarrow \text{default} \\
D \leftarrow 0 \\
R \leftarrow \text{default}
\end{array} \} \\
&\{ \begin{array}{l}
C \leftarrow \text{default} \\
D \leftarrow 0 \\
R \leftarrow \min(R, C)
\end{array} \} \quad \{ \begin{array}{l}
C \leftarrow D + 2 \\
D \leftarrow 0
\end{array} \}
\end{align*}

Figure 3.892: Automaton for the \texttt{MIN\_WIDTH\_STEADY\_SEQUENCE} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{STEADY\_SEQUENCE} pattern where \texttt{default} is $|\text{VARIABLES}| + 1$

\begin{align*}
&\{ \begin{array}{l}
C \leftarrow \text{default} \\
R \leftarrow \text{default}
\end{array} \} \\
&\{ \begin{array}{l}
C \leftarrow \text{default} \\
R \leftarrow \min(R, C)
\end{array} \} \quad \{ C \leftarrow 2 \}
\end{align*}

Figure 3.893: Automaton for the \texttt{MIN\_WIDTH\_STEADY\_SEQUENCE} constraint obtained by applying decoration Table 2.24 to the seed transducer of the \texttt{STEADY\_SEQUENCE} pattern where \texttt{default} is $|\text{VARIABLES}| + 1$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.117: Glue matrix for the `MIN_WIDTH_STEADY_SEQUENCE` constraint defined as the composition of the `STEADY_SEQUENCE` pattern, the feature `WIDTH`, and the aggregator `min`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C})$</td>
<td>$\overrightarrow{C} + \overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + 1$</td>
</tr>
</tbody>
</table>
3.381 MIN_WIDTH.Strictly.DECREASING.SEQUENCE

DESCRIPTION

Origin
Based on the STRICTLY.DECREASING.SEQUENCE pattern.

Constraint
MIN_WIDTH.Strictly.DECREASING.SEQUENCE(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var - dvar)

Restrictions
\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = sv + 1 \]
\[ VALUE \geq 2 \]
\[ VALUE = sv + 1 \lor VALUE \leq \min(sv, rv) \]
where
\[ sv = |VARIABLES| \]
\[ rv = \text{range}(\text{VARIABLES}.\text{var}) \]

Purpose
An occurrence of the pattern STRICTLY.DECREASING.SEQUENCE is the maximal sub-sequence which matches the regular expression \( >^+ \).
Assume that the occurrence of the pattern STRICTLY.DECREASING.SEQUENCE starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i + 2 \).

Example
\[ (2, (4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3)) \]

Figure 3.894 provides an example where the MIN_WIDTH.Strictly.DECREASING.SEQUENCE (2, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3]) constraint holds.

Typical
\[ |\text{VARIABLES}| > 1 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

Symmetry
One and the same constant can be added to the \text{var} attribute of all items of VARIABLES.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 3.894: Illustrating the MIN_WIDTH_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 3.895 and 3.896 respectively depict the automaton associated with the constraint \texttt{MIN\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE} and its simplified form.

Figure 3.895: Automaton for the \texttt{MIN\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{STRICTLY\_DECREASING\_SEQUENCE} pattern where \texttt{default} is $|VARIABLES| + 1$.

Figure 3.896: Automaton for the \texttt{MIN\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.24 to the seed transducer of the \texttt{STRICTLY\_DECREASING\_SEQUENCE} pattern where \texttt{default} is $|VARIABLES| + 1$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.118: Glue matrix for the MIN_WIDTH STRICTLY_DECREASING_SEQUENCE constraint defined as the composition of the STRICTLY_DECREASING_SEQUENCE pattern, the feature WIDTH, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\min(\overrightarrow{C}, \overleftarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overleftarrow{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\min(\overrightarrow{C}, \overleftarrow{C})$</td>
<td>$\overrightarrow{C} + \overleftarrow{C} + \overrightarrow{D} + \overleftarrow{D} + 1$</td>
</tr>
</tbody>
</table>
3.382  MIN_WIDTH STRICTLY_INCREASING_SEQUENCE

**Origin**
Based on the STRICTLY_INCREASING_SEQUENCE pattern.

**Constraint**
MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**
VALUE : dvar
VARIABLES : collection(var−dvar)

**Restrictions**
sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = sv + 1
VALUE ≥ 2
VALUE = sv + 1 ∨ VALUE ≤ min(sv, rv)
required(VARIABLES, var)
where
sv = |VARIABLES|
rv = range(VARIABLES.var)

**Value** is the minimal width of occurrences of the STRICTLY_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value |VARIABLES| + 1.

**Purpose**
An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression <+. Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at position i and ends at position j. The feature WIDTH computes the value j − i + 2.

**Example**
(2, (4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3))

Figure 3.897 provides an example where the MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE (2, [4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3]) constraint holds.

**Typical**
|VARIABLES| > 1
range(VARIABLES.var) > 1

**Symmetry**
One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.897: Illustrating the MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE constraint of the Example slot
Figures 3.898 and 3.899 respectively depict the automaton associated with the constraint MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE and its simplified form.

Figure 3.898: Automaton for the MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern where default is $|\text{VARIABLES}| + 1$.

Figure 3.899: Automaton for the MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.24 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern where default is $|\text{VARIABLES}| + 1$; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.119: Glue matrix for the `MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE` constraint defined as the composition of the `STRICTLY_INCREASING_SEQUENCE` pattern, the feature `WIDTH`, and the aggregator `min`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\min(\overrightarrow{C}, \overleftarrow{C})$</td>
<td>$\min(\overrightarrow{C}, \overleftarrow{C})$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\min(\overrightarrow{C}, \overleftarrow{C})$</td>
<td>$\overrightarrow{C} + \overleftarrow{C} + \overrightarrow{D} + \overleftarrow{D} + 1$</td>
</tr>
</tbody>
</table>
3.383 MIN_WIDTH_SUMMIT

**Origin**
Based on the SUMMIT pattern.

**Constraint**
MIN_WIDTH_SUMMIT(VALUE, VARIABLES)

**Arguments**
VALUE : dvar
VARIABLES : collection(var−dvar)

**Restrictions**
\[ sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = sv + 1 \]
\[ VALUE \geq 1 \]
\[ rv = 2 \Rightarrow VALUE = sv + 1 \lor VALUE \leq 1 \]
\[ rv \geq 3 \Rightarrow VALUE = sv + 1 \lor VALUE \leq sv - 2 \]
\[ \text{required}(\text{VARIABLES}, \text{var}) \]
where
\[ sv = |\text{VARIABLES}| \]
\[ rv = \text{range}(\text{VARIABLES}.\text{var}) \]

**Purpose**
VALUE is the minimal width of occurrences of the SUMMIT pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value \(|\text{VARIABLES}| + 1\).

An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression \((< | < (= | <)^* <(| > | > (= | >)^* >)\).
Assume that the occurrence of the pattern SUMMIT starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

**Example**
\[ (1, (7, 1, 5, 4, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1)) \]

Figure 3.900 provides an example where the MIN_WIDTH_SUMMIT (1, [7, 1, 5, 4, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1]) constraint holds.

**Typical**
\[ |\text{VARIABLES}| > 2 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

**Symmetries**
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.900: Illustrating the MIN_WIDTH_SUMMIT constraint of the Example slot
Figure 3.901 depicts the automaton associated with the constraint MIN_WIDTH_SUMMIT.

Figure 3.901: Automaton for the MIN_WIDTH_SUMMIT constraint obtained by applying decoration Table 2.35 to the seed transducer of the SUMMIT pattern where default is |VARIABLES| + 1 (transition u → r has the same accumulator update as transition r → u); \(-R_i + R_{i-1} \geq 0\) is a linear invariant.
Table 3.120: Glue matrix for the MIN_WIDTH_SUMMIT constraint defined as the composition of the SUMMIT pattern, the feature WIDTH, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.384 **MIN_WIDTH_VALLEY**

**Origin**

Based on the **VALLEY** pattern.

**Constraint**

\[ \text{MIN_WIDTH_VALLEY}(\text{VALUE}, \text{VARIABLES}) \]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} - \text{dvar})
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{sv} \leq 2 \lor \text{rv} \leq 1 & \Rightarrow \text{VALUE} = \text{sv} + 1 \\
\text{VALUE} \geq 1 & \\
\text{VALUE} = \text{sv} + 1 \lor \text{VALUE} \leq \text{sv} - 2 \Rightarrow \\
\text{required}(\text{VARIABLES}, \text{var}) & \text{ where} \\
\text{sv} = |\text{VARIABLES}| & \\
\text{rv} = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

\[\text{VALUE}\] is the minimal width of occurrences of the **VALLEY** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, \[\text{VALUE}\] takes the default value \[|\text{VARIABLES}| + 1\].

**Purpose**

An occurrence of the pattern **VALLEY** is the maximal subsequence which matches the regular expression \[>(= | >)^*(< | =)^*<\].

Assume that the occurrence of the pattern **VALLEY** starts at position \(i\) and ends at position \(j\). The feature \text{WIDTH} computes the value \(j - i\).

**Example**

\[(2, (1, 3, 7, 4, 3, 6, 5, 3, 3, 2, 6, 5, 5, 7))\]

Figure 3.902 provides an example where the \text{MIN_WIDTH_VALLEY} \((2, [1, 3, 7, 4, 3, 6, 5, 3, 3, 2, 6, 5, 5, 7])\) constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| & > 2 \\
\text{range}(\text{VARIABLES}.\text{var}) & > 1
\end{align*}
\]

**Symmetries**

- Items of **VARIABLES** can be reversed.
- One and the same constant can be added to the \text{var} attribute of all items of **VARIABLES**.

**Arg. properties**

Functional dependency: \text{VALUE} determined by **VARIABLES**.
Figure 3.902: Illustrating the MIN_WIDTH_VALLEY constraint of the Example slot
Automaton

Figure 3.903 depicts the automaton associated with the constraint MIN_WIDTH_VALLEY.

\[
\begin{align*}
& \{ \begin{align*}
C & \leftarrow \text{default} \\
D & \leftarrow 0 \\
R & \leftarrow \text{default}
\end{align*} \} \quad \leq s \\
& \{ \begin{align*}
C & \leftarrow D + 1 \\
D & \leftarrow 0
\end{align*} \} \quad \leq \min(R, C) \\
& \{ \begin{align*}
C & \leftarrow \text{default} \\
D & \leftarrow 0 \\
R & \leftarrow \min(R, C)
\end{align*} \} \quad > \{ D \leftarrow D + 1 \}
\end{align*}
\]

\[
\begin{align*}
& \{ D \leftarrow D + 1 \} \\
& \{ C \leftarrow \text{default} \\
& \min(R, C) \}
\end{align*}
\]

\[
\begin{align*}
& \{ D \leftarrow D + 1 \} \\
& \{ C \leftarrow C + D + 1 \}
\end{align*}
\]

Figure 3.903: Automaton for the MIN_WIDTH_VALLEY constraint obtained by applying decoration Table 2.35 to the seed transducer of the VALLEY pattern where default is \(|\text{VARIABLES}| + 1\); \(-R_i + R_{i-1} \geq 0\) is a linear invariant.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
<tr>
<td>r</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{B} + \overrightarrow{D} + 1)</td>
<td>(\overrightarrow{C} + \overrightarrow{B} + \overrightarrow{D} + 1)</td>
</tr>
<tr>
<td>t</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
<td>(\overrightarrow{C} + \overrightarrow{B} + \overrightarrow{D} + 1)</td>
<td>(\min(\overrightarrow{C}, \overrightarrow{C}))</td>
</tr>
</tbody>
</table>

Table 3.121: Glue matrix for the MIN_WIDTH_VALLEY constraint defined as the composition of the VALLEY pattern, the feature WIDTH, and the aggregator \(\min\); cells of the glue matrix are coloured with the colour of the constituent to which they are related.
**3.385 MIN_WIDTH_ZIGZAG**

**DESCRIPTION**

- **Origin**: Based on the ZIGZAG pattern.

- **Constraint**: MIN_WIDTH_ZIGZAG(VALUE, VARIABLES)

- **Arguments**:
  - VALUE : dvar
  - VARIABLES : collection(var–dvar)

- **Restrictions**:
  - \( sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = sv + 1 \)
  - \( VALUE \geq 2 \)
  - \( VALUE = sv + 1 \lor VALUE \leq sv - 2 \)
  - required(VARIABLES, var)

  where
  - \( sv = |VARIABLES| \)
  - \( rv = range(VARIABLES.var) \)

**Purpose**

An occurrence of the pattern ZIGZAG is the *maximal* subsequence which matches the regular expression \((<>)^+ (< | <>) | (<>)^+ (> | >>)\). Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

**Example**

\((2, (4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1))\)

**Typical**

- \(|VARIABLES| > 3\)
- \(range(VARIABLES.var) > 1\)

**Symmetries**

- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.904: Illustrating the MIN_WIDTH_ZIGZAG constraint of the Example slot
Automaton

Figures 3.905 and 3.906 respectively depict the automaton associated with the constraint MIN_WIDTH_ZIGZAG and its simplified form.
Figure 3.905: Automaton for the MIN_WIDTH_ZIGZAG constraint obtained by applying decoration Table 2.35 to the seed transducer of the ZIGZAG pattern where default is $|VARIABLES| + 1$; (1) missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=; (2)$ on transitions from $b, c, e, f$ to $s$ the accumulator $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the accumulator $R$ is updated wrt $C$ and the accumulator $C$ is reset to its initial value.
Figure 3.906: Automaton for the \textsc{MIN\_WIDTH\_ZIGZAG} constraint obtained by applying decoration Table 2.23 to the seed transducer of the \textsc{ZIGZAG} pattern where default is $|\text{VARIABLES}| + 1$; missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=$; (2) on transitions from $b, c, e, f$ to $s$ the accumulator $D$ is reset to its initial value; (3) on transitions from $c, f$ to $s$ the accumulator $R$ is updated wrt $C$ and the accumulator $C$ is reset to its initial value; $-R_i + R_{i-1} \geq 0$ is a linear invariant.
Table 3.122: Glue matrix for the MIN_WIDTH_ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature WIDTH, and the aggregator min; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
**3.386  **NB_BUMP_ON_DECREASING_SEQUENCE

**Origin**
Based on the BUMP_ON_DECREASING_SEQUENCE pattern.

**Constraint**
NB_BUMP_ON_DECREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**
VALUE : dvar
VARIABLES : collection(var−dvar)

**Restrictions**
\[ sv \leq 5 \lor rv \leq 2 \Rightarrow VALUE = 0 \]
VALUE \geq 0
VALUE \leq \max(0, \lceil (sv - 3)/3 \rceil) \Rightarrow VALUE \leq 4
\text{required}(\text{VARIABLES}, \text{var})
\text{where}
sv = |\text{VARIABLES}|
rv = \text{range}(\text{VARIABLES}.\text{var})

**Purpose**
VALUE is the number of occurrences of the BUMP_ON_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.
An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence which matches the regular expression >>>>.

**Example**
\[(2, \langle 7,6,5,6,5,4,1,4,7,5,4,2,5,4,3,3 \rangle)\]

Figure 3.907 provides an example where the NB_BUMP_ON_DECREASING_SEQUENCE (2, \[7,6,5,6,5,4,1,4,7,5,4,2,5,4,3,3\]) constraint holds.

**Typical**
|VARIABLES| \( \geq 5 \)
range(VARIABLES.var) \( \geq 2 \)

**Symmetry**
One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.907: Illustrating the \texttt{NB\_BUMP\_ON\_DECREASING\_SEQUENCE} constraint of the \texttt{Example} slot
Figures 3.908 and 3.909 respectively depict the automaton associated with the constraint \texttt{NB\_BUMP\_ON\_DECREASING\_SEQUENCE} and its simplified form.

Figure 3.908: Automaton for the \texttt{NB\_BUMP\_ON\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{BUMP\_ON\_DECREASING\_SEQUENCE} pattern where \texttt{default} is 0
Figure 3.909: Automaton for the \textit{NB\_BUMP\_ON\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.37 to the seed transducer of the \textit{BUMP\_ON\_DECREASING\_SEQUENCE} pattern where default is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-3} + 1 \geq 0$ are linear invariants.
3.387 NB_DECREASING

**DESCRIPTION**

**Origin**
Based on the DECREASING pattern.

**Constraint**

\[
\text{NB\_DECREASING}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- VALUE : dvar
- VARIABLES : collection(var−dvar)

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & \geq 0 \\
\text{VALUE} & \leq \max(0, (\ell - 1) \ast \lfloor \text{sv}/\ell \rfloor + \max(0, \text{sv} \mod \ell - 1)) \\
\text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where

\[
\begin{align*}
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var}) \\
\ell & = \min(\text{sv}, \text{rv})
\end{align*}
\]

**Purpose**

VALUE is the number of occurrences of the DECREASING pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern DECREASING is the subsequence which matches the regular expression \(>\).

**Example**

\[(5, (3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))\]

Figure 3.910 provides an example where the NB\_DECREASING (5, [3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4]) constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| & > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) & > 1
\end{align*}
\]

**Symmetry**

One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.910: Illustrating the NB\_DECREASING constraint of the Example slot
Automaton

Figures 3.911 and 3.912 respectively depict the automaton associated with the constraint \( \text{NB\_DECREASING} \) and its simplified form.

\[
\begin{aligned}
C &\leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow \text{default}
\end{aligned}
\]

\[
\begin{cases}
D &\leftarrow 0 \\
R &\leftarrow R + \max(D, 1)
\end{cases}
\]

Figure 3.911: Automaton for the \( \text{NB\_DECREASING} \) constraint obtained by applying decoration Table 2.35 to the seed transducer of the \( \text{DECREASING} \) pattern where \( \text{default} \) is 0.

\[
\begin{cases}
R &\leftarrow \text{default}
\end{cases}
\]

\[
\begin{cases}
R &\leftarrow R + 1
\end{cases}
\]

Figure 3.912: Automaton for the \( \text{NB\_DECREASING} \) constraint obtained by applying decoration Table 2.37 to the seed transducer of the \( \text{DECREASING} \) pattern where \( \text{default} \) is 0; \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-1} + 1 \geq 0 \) are linear invariants.

Table 3.123: Glue matrix for the \( \text{NB\_DECREASING} \) constraint defined as the composition of the \( \text{DECREASING} \) pattern, the feature \( \text{ONE} \), and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.388  NB_DECREASING_SEQUENCE

**DESCRIPTION**

**Origin**
Based on the DECREASING_SEQUENCE pattern.

**Constraint**

\[ \text{NB_DECREASING_SEQUENCE}(\text{VALUE}, \text{VARIABLES}) \]

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var − dvar)

**Restrictions**

\[
\begin{align*}
\text{sv} &\leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} &\geq 0 \\
\text{VALUE} &\leq \lfloor \text{sv}/2 \rfloor ! \text{required}(\text{VARIABLES}, \text{var}) \\
\text{where} &\text{sv} = |\text{VARIABLES}| \\
\text{rv} &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**
VALUE is the number of occurrences of the DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression \( (> | =)^* > | > \).

**Example**

\( (3, (3, 4, 2, 2, 5, 6, 4, 3, 1, 1, 4, 6, 4, 4)) \)

Figure 3.913 provides an example where the NB_DECREASING_SEQUENCE (3, [3, 4, 2, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4]) constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 1 \]

\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

**Symmetry**
One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.913: Illustrating the NB_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 3.914 and 3.915 respectively depict the automaton associated with the constraint \texttt{NB\_DECREASING\_SEQUENCE} and its simplified form.

\begin{align*}
\begin{cases}
C &\leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow \text{default}
\end{cases}
\end{align*}

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{automaton1}
\caption{Automaton for the \texttt{NB\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{DECREASING\_SEQUENCE} pattern where \texttt{default} is 0}
\end{figure}

\begin{align*}
\begin{cases}
C &\leftarrow \max(D, 1) \\
D &\leftarrow 0
\end{cases}
\end{align*}

\begin{align*}
\begin{cases}
C &\leftarrow \max(C, \max(D, 1)) \\
D &\leftarrow 0
\end{cases}
\end{align*}

\begin{align*}
\begin{cases}
D &\leftarrow \max(D, 1)
\end{cases}
\end{align*}

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{automaton2}
\caption{Automaton for the \texttt{NB\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.37 to the seed transducer of the \texttt{DECREASING\_SEQUENCE} pattern where \texttt{default} is 0; \(R_i - R_{i-1} \geq 0\) and \(-R_i + R_{i-2} + 1 \geq 0\) are linear invariants.}
\end{figure}
Table 3.124: Glue matrix for the `NB_DECREASING_SEQUENCE` constraint defined as the composition of the `DECREASING_SEQUENCE` pattern, the feature `ONE`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>( s )</th>
<th>( t )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>( \overrightarrow{C} + \overleftarrow{C} )</td>
<td>( \overrightarrow{C} + \overleftarrow{C} )</td>
</tr>
<tr>
<td>( t )</td>
<td>( \overrightarrow{C} + \overleftarrow{C} )</td>
<td>1</td>
</tr>
</tbody>
</table>
3.389 NB DECREASING_TERRACE

**Origin**
Based on the DECREASING_TERRACE pattern.

**Constraint**
NB DECREASING_TERRACE(VALUE, VARIABLES)

**Arguments**
VALUE : dvar  
VARIABLES : collection(var–dvar)

**Restrictions**
\[ sv \leq 3 \lor rv \leq 2 \Rightarrow VALUE = 0 \]
\[ VALUE \geq 0 \]
\[ VALUE \leq \max(0, \lfloor (\ell - 2)/2 \rfloor \ast \lfloor sv/\ell \rfloor + \max(0, \lfloor sv mod (\ell - 2)/2 \rfloor)) \]
\[ \text{required}(\text{VARIABLES}, \text{var}) \]
where
\[ sv = |\text{VARIABLES}| \]
\[ rv = \text{range}(\text{VARIABLES}.\text{var}) \]
\[ \ell = \text{max}(1, \text{min}(sv, 2 * rv - 2)) \]

**Purpose**
VALUE is the number of occurrences of the DECREASING_TERRACE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern DECREASING_TERRACE is the maximal subsequence which matches the regular expression \( >\equiv^> \).

**Example**
\( (2, (6, 4, 4, 5, 2, 2, 1, 3, 5, 4, 4, 3, 3)) \)

Figure 3.916 provides an example where the NB DECREASING_TERRACE \( (2, [6, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3]) \) constraint holds.

**Typical**
\[ |\text{VARIABLES}| > 3 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 2 \]

**Symmetry**
One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.916: Illustrating the $\text{NB}_\text{DECREASING}_\text{TERRACE}$ constraint of the Example slot
Automaton

Figures 3.917 and 3.918 respectively depict the automaton associated with the constraint \textsc{nb\_decreasing\_terrace} and its simplified form.

$\leq s \begin{cases} C \leftarrow \text{default} \\ D \leftarrow 0 \\ R \leftarrow \text{default} \end{cases} \leq \begin{cases} D \leftarrow \varnothing \end{cases}$

$R + C$

$> r = \begin{cases} D \leftarrow \max(D, 1) \end{cases}$

$= \begin{cases} D \leftarrow \max(D, 1) \end{cases}$

$< \begin{cases} D \leftarrow \max(D, 1) \end{cases}$

$< \begin{cases} D \leftarrow 0 \\ R \leftarrow R + \max(D, 1) \end{cases}$

$< \begin{cases} D \leftarrow 0 \\ R \leftarrow R + \max(D, 1) \end{cases}$

Figure 3.917: Automaton for the \textsc{nb\_decreasing\_terrace} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \textsc{decreasing\_terrace} pattern where \text{default} is 0.

$\leq s \begin{cases} R \leftarrow \text{default} \end{cases} \leq \begin{cases} R \end{cases}$

$> r = \begin{cases} R \leftarrow R + 1 \end{cases}$

$= \begin{cases} R \end{cases}$

$= \begin{cases} R \end{cases}$

$> r \begin{cases} R \end{cases}$

$> r \begin{cases} R \leftarrow R + 1 \end{cases}$

Figure 3.918: Automaton for the \textsc{nb\_decreasing\_terrace} constraint obtained by applying decoration Table 2.37 to the seed transducer of the \textsc{decreasing\_terrace} pattern where \text{default} is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-2} + 1 \geq 0$ are linear invariants.
Table 3.125: Glue matrix for the `NB_DECREASING_TERRACE` constraint defined as the composition of the `DECREASING_TERRACE` pattern, the feature `ONE`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{C}$</td>
</tr>
<tr>
<td>r</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{C}$</td>
<td>1</td>
</tr>
<tr>
<td>t</td>
<td>$\overline{C} + \overline{C}$</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
### 3.390 NB_DIP_ON_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th><strong>DESCRIPTION</strong></th>
<th><strong>AUTOMATON</strong></th>
</tr>
</thead>
</table>

**Origin**
Based on the `DIP_ON_INCREASING_SEQUENCE` pattern.

**Constraint**
\[
\text{NB_DIP_ON_INCREASING_SEQUENCE}(\text{VALUE, VARIABLES})
\]

**Arguments**
- `VALUE`: dvar
- `VARIABLES`: collection(var–dvar)

**Restrictions**
\[
\begin{align*}
sv & \leq 5 \lor rv \leq 2 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & \geq 0 \\
\text{VALUE} & \leq \max(0, \left\lfloor (sv - 3)/3 \right\rfloor) \\
& \text{required}(\text{VARIABLES, var})
\end{align*}
\]
where
\[
sv = |\text{VARIABLES}|
\]
\[
rv = \text{range}(\text{VARIABLES.var})
\]

**Purpose**
VALUE is the number of occurrences of the `DIP_ON_INCREASING_SEQUENCE` pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern `DIP_ON_INCREASING_SEQUENCE` is the subsequence `<<><<<`.

**Example**

\[
(2, \langle 1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4 \rangle)
\]

Figure 3.919 provides an example where the `NB_DIP_ON_INCREASING_SEQUENCE` constraint holds.

**Typical**
- `|\text{VARIABLES}| > 5`
- `\text{range}(\text{VARIABLES.var}) > 2`

**Symmetry**
One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.919: Illustrating the NB_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot
Figures 3.920 and 3.921 respectively depict the automaton associated with the constraint \texttt{NB\_DIP\_ON\_INCREASING\_SEQUENCE} and its simplified form.

\begin{verbatim}
\{ C ← \texttt{default} \\
D ← 0 \\
R ← \texttt{default} \}
\geq s
\rightarrow
\begin{array}{ll}
\leq r \\
\leq t \\
\end{array}
\begin{array}{ll}
\{ D ← \max(D, 1) \} \\
\{ D ← 0 \} \\
\{ R ← R + \max(D, 1) \} \\
\{ D ← \min(D, 1) \} \\
\{ D ← \max(D, 1) \} \\
\{ D ← \min(D, 1) \} \\
\end{array}
\end{verbatim}

Figure 3.920: Automaton for the \texttt{NB\_DIP\_ON\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{DIP\_ON\_INCREASING\_SEQUENCE} pattern where \texttt{default} is 0
Figure 3.921: Automaton for the `NB_DIP_ON_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 2.37 to the seed transducer of the `DIP_ON_INCREASING_SEQUENCE` pattern where `default` is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-3} + 1 \geq 0$ are linear invariants.
3.391 **NB_GORGE**

<table>
<thead>
<tr>
<th>Origin</th>
<th>Based on the <strong>GORGE</strong> pattern.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constraint</td>
<td><strong>NB_GORGE</strong>((\text{VALUE, VARIABLES}))</td>
</tr>
</tbody>
</table>
| Arguments | \(\text{VALUE} : \text{dvar}\)  
\(\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})\) |
| Restrictions | \(\text{sv} \leq 2 \vee \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0\)  
\(\text{VALUE} \geq 0\)  
\(\text{VALUE} \leq \max(0, \lceil (\text{sv} - 1)/2 \rceil)\)  
\(\text{required}(\text{VARIABLES}, \text{var})\)  
where  
\(\text{sv} = |\text{VARIABLES}|\)  
\(\text{rv} = \text{range}(\text{VARIABLES}..\text{var})\) |
| Purpose | **VALUE** is the number of occurrences of the **GORGE** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, **VALUE** takes the default value 0. An occurrence of the pattern **GORGE** is the maximal subsequence which matches the regular expression \((> | > (= | >)* >)(< | < (= | <)* <)\). |
| Example | \((3, (1, 7, 3, 4, 4, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7))\) |

Figure 3.922 provides an example where the **NB_GORGE** \((3, [1, 7, 3, 4, 4, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7])\) constraint holds.

| Typical | \(|\text{VARIABLES}| > 2\)  
\(\text{range}(\text{VARIABLES}..\text{var}) > 1\) |
| Symmetries | - Items of **VARIABLES** can be reversed.  
- One and the same constant can be added to the **var** attribute of all items of **VARIABLES**. |
| Arg. properties | Functional dependency: **VALUE** determined by **VARIABLES**. |
Figure 3.922: Illustrating the NB_GORGE constraint of the Example slot
Figures 3.923 and 3.924 respectively depict the automaton associated with the constraint `
NB_GORGE` and its simplified form.

**Automaton**

Figure 3.923: Automaton for the NB_GORGE constraint obtained by applying decora-
tion Table 2.35 to the seed transducer of the GORGE pattern where default is 0 (transi-
tion $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$)
Figure 3.924: Automaton for the NB_GORGE constraint obtained by applying decoration Table 2.37 to the seed transducer of the GORGE pattern where default is 0; \( R_i - R_{i-1} \geq 0 \) and \(-R_i + R_{i-2} + 1 \geq 0\) are linear invariants.

Table 3.126: Glue matrix for the NB_GORGE constraint defined as the composition of the GORGE pattern, the feature ONE, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.392  **NB_INCREASING**

**DESCRIPTION**

**Origin**
Based on the INCREASING pattern.

**Constraint**

\[
\text{NB\_INCREASING}(\text{VALUE, VARIABLES})
\]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} - \text{dvar})
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & \geq 0 \\
\text{VALUE} & \leq \max(0, (\ell - 1) * \lfloor \text{sv}/\ell \rfloor + \max(0, \text{sv} \mod \ell - 1)) \\
\text{required}(\text{VARIABLES, var})
\end{align*}
\]

where

\[
\begin{align*}
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var}) \\
\ell & = \min(\text{sv, rv})
\end{align*}
\]

**Purpose**

\text{VALUE} is the number of occurrences of the INCREASING pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, \text{VALUE} takes the default value 0.

An occurrence of the pattern INCREASING is the subsequence which matches the regular expression $<\text{.}$

**Example**

\[(5, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))\]

Figure 3.925 provides an example where the \text{NB\_INCREASING} (5, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3]) constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| & > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) & > 1
\end{align*}
\]

**Symmetry**

One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: \text{VALUE} determined by VARIABLES.

---

**AUTOMATON**
Figure 3.925: Illustrating the **NB_INCREASING** constraint of the **Example** slot
Automaton

Figures 3.926 and 3.927 respectively depict the automaton associated with the constraint \texttt{NB\_INCREASING} and its simplified form.

\[
\begin{align*}
\{ & C \leftarrow \text{default} \\
 & D \leftarrow 0 \\
 & R \leftarrow \text{default} \\
\} \\
\{ & D \leftarrow 0 \\
 & R \leftarrow R + \max(D, 1) \\
\} \\
\{ & R \geq < \\
\} \\
\{ & R \geq R + \max(D, 1) \\
\} \\
\{ & R \geq \\
\} \\
\{ & R \geq \}
\end{align*}
\]

Figure 3.926: Automaton for the \texttt{NB\_INCREASING} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{INCREASING} pattern where \texttt{default} is 0

\[
\begin{align*}
\{ & R \leftarrow \text{default} \\
\} \\
\{ & R \leftarrow R + 1 \\
\} \\
\{ & R \geq \\
\} \\
\{ & R \geq \}
\end{align*}
\]

Figure 3.927: Automaton for the \texttt{NB\_INCREASING} constraint obtained by applying decoration Table 2.37 to the seed transducer of the \texttt{INCREASING} pattern where \texttt{default} is 0; \( R_i - R_{i-1} \geq 0 \) and \(-R_i + R_{i-1} + 1 \geq 0 \) are linear invariants.

Table 3.127: Glue matrix for the \texttt{NB\_INCREASING} constraint defined as the composition of the \texttt{INCREASING} pattern, the \texttt{feature ONE}, and the \texttt{aggregator sum}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
INCREASING

1697
NB_INCREASINGSEQUENCE

3.393 NB_INCREASINGSEQUENCE

**DESCRIPTION**

**Origin**

Based on the INCREASING_SEQUENCE pattern.

**Constraint**

NB_INCREASINGSEQUENCE(VALUE, VARIABLES)

**Arguments**

VALUE : dvar  
VARIABLES : collection(var−dvar)

**Restrictions**

sv ≤ 1 \lor rv ≤ 1 \Rightarrow VALUE = 0  
VALUE ≥ 0 
VALUE ≤ \lfloor sv/2 \rfloor  
required(VARIABLES, var) 
where 
sv = |VARIABLES|  
rv = range(VARIABLES.var)

**Purpose**

VALUE is the number of occurrences of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression \(< (< | | =)^* | < | <\).

**Example**

(3, (4, 3, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))

Figure 3.928 provides an example where the NB_INCREASINGSEQUENCE (3, [4, 3, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3]) constraint holds.

**Typical**

|VARIABLES| > 1  
range(VARIABLES.var) > 1

**Symmetry**

One and the same constant can be added to the VAR attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.928: Illustrating the NB_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 3.929 and 3.930 respectively depict the automaton associated with the constraint \texttt{NB\_INCREASING\_SEQUENCE} and its simplified form.

\[
\begin{cases}
C &\leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow \text{default}
\end{cases}
\]

\[
\begin{cases}
C &\leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow R + C
\end{cases}
\]

\[
\begin{cases}
D &\leftarrow \max(D, 1)
\end{cases}
\]

\[
\begin{cases}
C &\leftarrow \max(C, \max(D, 1)) \\
D &\leftarrow 0
\end{cases}
\]

Figure 3.929: Automaton for the \texttt{NB\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{INCREASING\_SEQUENCE} pattern where default is 0.

\[
\begin{cases}
R &\leftarrow \text{default}
\end{cases}
\]

\[
\begin{cases}
R &\leftarrow R + 1
\end{cases}
\]

\[
\begin{cases}
R &\leftarrow \max(R, -R + 1)
\end{cases}
\]

Figure 3.930: Automaton for the \texttt{NB\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.37 to the seed transducer of the \texttt{INCREASING\_SEQUENCE} pattern where default is 0; \(R_i - R_{i-1} \geq 0\) and \(-R_{i-2} + R_{i-1} \geq 0\) are linear invariants.
Table 3.128: Glue matrix for the \texttt{NB\_INCREASING\_SEQUENCE} constraint defined as the composition of the \texttt{INCREASING\_SEQUENCE} pattern, the feature \texttt{ONE}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.394 **NB_INCREASING_TERRACE**

**Description**

*Origin*

Based on the INCREASING_TERRACE pattern.

*Constraint*

\[ \text{NB_INCREASING_TERRACE}(\text{VALUE, VARIABLES}) \]

*Arguments*

| VALUE    | : dvar          |
| VARIABLES| : collection(var−dvar) |

*Restrictions*

\[
\begin{align*}
sv \leq 3 \lor rv \leq 2 & \Rightarrow VALUE = 0 \\
VALUE & \geq 0 \\
VALUE & \leq \max(0, \lfloor (\ell - 2)/2 \rfloor + \max(0, \lfloor (sv \mod \ell - 2)/2 \rfloor)) \overset{\circ}{\circ} \\
& \text{required(VARIABLES, var)} \\
sv & = |\text{VARIABLES}| \\
rv & = \text{range(VARIABLES.var)} \\
\ell & = \max(1, \min(sv, 2 \times rv - 2))
\end{align*}
\]

**Purpose**

VALUE is the number of occurrences of the INCREASING_TERRACE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0. An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression \(<=^+=<\).

**Example**

\[(2, (1, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4))\]

Figure 3.931 provides an example where the **NB_INCREASING_TERRACE** (2, [1, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4]) constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| & > 3 \\
\text{range(VARIABLES.var)} & > 2
\end{align*}
\]

**Symmetry**

One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.931: Illustrating the `NB_INCREASING_TERRACE` constraint of the `Example` slot.
Automaton

Figures 3.932 and 3.933 respectively depict the automaton associated with the constraint \texttt{NB_INCREASING_TERRACE} and its simplified form.

\[
\begin{align*}
\{ C \leftarrow \text{default} \\
D \leftarrow 0 \\
R \leftarrow \text{default} \}\\
\{ D \leftarrow \max(D, 1) \}
\end{align*}
\]

Figure 3.932: Automaton for the \texttt{NB_INCREASING_TERRACE} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{INCREASING_TERRACE} pattern where \texttt{default} is 0.

\[
\begin{align*}
\{ R \leftarrow \text{default} \}\\
\{ D \leftarrow 0 \\
R \leftarrow R + \max(D, 1) \}\\
\{ R \leftarrow R + 1 \}
\end{align*}
\]

Figure 3.933: Automaton for the \texttt{NB_INCREASING_TERRACE} constraint obtained by applying decoration Table 2.37 to the seed transducer of the \texttt{INCREASING_TERRACE} pattern where \texttt{default} is 0; \( R_i - R_{i-1} \geq 0 \) and \(-R_i + R_{i-2} + 1 \geq 0 \) are linear invariants.
Table 3.129: Glue matrix for the NB_INCREASING_TERRACE constraint defined as the composition of the INCREASING_TERRACE pattern, the feature ONE, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\vec{C} + \vec{C}$</td>
<td>$\vec{C} + \vec{C}$</td>
<td>$\vec{C} + \vec{C}$</td>
</tr>
<tr>
<td>r</td>
<td>$\vec{C} + \vec{C}$</td>
<td>$\vec{C} + \vec{C}$</td>
<td>1</td>
</tr>
<tr>
<td>t</td>
<td>$\vec{C} + \vec{C}$</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
3.395  NB_INFLExION

**Origin**
Based on the INFLEXION pattern.

**Constraint**

\[ \text{NB\_INFLExION}(\text{VALUE, VARIABLES}) \]

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var - dvar)`

**Restrictions**

\[
\begin{align*}
sv & \leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & \geq 0 \\
\text{VALUE} & \leq \max(0, sv - 2k) \\
\text{required}(\text{VARIABLES, var}) & \text{required}(\text{VARIABLES, var}) \\
\text{where} \\
sv & = |\text{VARIABLES}| \\
rv & = \text{range}(\text{VARIABLES, var})
\end{align*}
\]

**Purpose**

**Example**

\[(8, (1, 2, 6, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4))\]

Figure 3.934 provides an example where the **NB\_INFLExION** (8, [1, 2, 6, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4]) constraint holds.

**Typical**

\[|\text{VARIABLES}| > 2 \]
\[\text{range}(\text{VARIABLES, var}) > 1\]

**Symmetries**

- Items of **VARIABLES** can be reversed.
- One and the same constant can be added to the var attribute of all items of **VARIABLES**.

**Arg. properties**

Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 3.934: Illustrating the NB_INFLEXION constraint of the **Example** slot
Automaton

Figures 3.935 and 3.936 respectively depict the automaton associated with the constraint NB_INFLEXION and its simplified form.

Figure 3.935: Automaton for the NB_INFLEXION constraint obtained by applying decoration Table 2.35 to the seed transducer of the INFLEXION pattern where default is 0 (transition r → t has the same accumulators updates as transition t → r)

Figure 3.936: Automaton for the NB_INFLEXION constraint obtained by applying decoration Table 2.37 to the seed transducer of the INFLEXION pattern where default is 0 (transition r → t has the same accumulator update as transition t → r); $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + 1 \geq 0$ are linear invariants.
### 3.396 NB_Peak

#### Description

Based on the PEAK pattern.

#### Constraint

\[ \text{NB}_\text{PEAK}(\text{VALUE, VARIABLES}) \]

#### Arguments

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

#### Restrictions

\[
\begin{align*}
\text{sv} &\leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} &\geq 0 \\
\text{VALUE} &\leq \max(0, \lfloor (\text{sv} - 1)/2 \rfloor) \& \text{required}(\text{VARIABLES, var})
\end{align*}
\]

where
\[
\begin{align*}
\text{sv} &= |\text{VARIABLES}| \\
\text{rv} &= \text{range}(\text{VARIABLES.var})
\end{align*}
\]

#### Purpose

VALUE is the number of occurrences of the PEAK pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0. An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression \(< (= \mid <)^* (> \mid =)^* >\).

#### Example

\((3, (7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 1))\)

Figure 3.937 provides an example where the \(\text{NB}_\text{PEAK}(3, [7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 1])\) constraint holds.

#### Typical

\[
\begin{align*}
|\text{VARIABLES}| &> 2 \\
\text{range}(\text{VARIABLES.var}) &> 1
\end{align*}
\]

#### Symmetries

- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

#### Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 3.937: Illustrating the NB_PEAK constraint of the Example slot
Automaton

Figures 3.938 and 3.939 respectively depict the automaton associated with the constraint \texttt{NB\_PEAK} and its simplified form.

Figure 3.938: Automaton for the \texttt{NB\_PEAK} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{PEAK} pattern where \texttt{default} is 0

Figure 3.939: Automaton for the \texttt{NB\_PEAK} constraint obtained by applying decoration Table 2.37 to the seed transducer of the \texttt{PEAK} pattern where \texttt{default} is 0; \(R_i - R_{i-1} \geq 0\) and \(-R_i + R_{i-2} + 1 \geq 0\) are linear invariants.
Table 3.130: Glue matrix for the **NB**\_**PEAK** constraint defined as the composition of the **PEAK** pattern, the feature **ONE**, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
</tr>
<tr>
<td>r</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>t</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>1</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
</tr>
</tbody>
</table>
3.397  NB_PLAIN

Description

Origin
Based on the PLAIN pattern.

Constraint
NB_PLAIN(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions

\[ sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = 0 \]
\[ VALUE \geq 0 \]
\[ VALUE \leq \max(0, \lfloor (sv - 1)/2 \rfloor) \]

where

\[ sv = |VARIABLES| \]
\[ rv = \text{range}(VARIABLES,var) \]

Purpose
VALUE is the number of occurrences of the PLAIN pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0. An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression \( >^*< \).

Example

\((3, (2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3))\)

Figure 3.940 provides an example where the NB_PLAIN constraint holds.

Typical

\[ |VARIABLES| > 2 \]
\[ \text{range}(VARIABLES,var) > 1 \]

Symmetries

- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 3.940: Illustrating the NB_PLAIN constraint of the Example slot
Automaton

Figures 3.941 and 3.942 respectively depict the automaton associated with the constraint \textit{NB\_PLAIN} and its simplified form.

\begin{align*}
D & = 0 \\
R & = R + \max(D, 1) \quad \{ D \leftarrow \max(D, 1) \} \\
\{ D \leftarrow \max(D, 1) \} & \quad \{ D \leftarrow 0 \ \& \ R \leftarrow \max(0, R + \max(D, 1)) \} \\
\{ D \leftarrow 0 \ \& \ R \leftarrow \max(0, R + \max(D, 1)) \} & \quad \{ D \leftarrow \max(D, 1) \ \& \ R \leftarrow \max(0, R + \max(D, 1)) \}
\end{align*}

Figure 3.941: Automaton for the \textit{NB\_PLAIN} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \textit{PLAIN} pattern where \textit{default} is 0.

\begin{align*}
{ R } & = { R } + 1 \quad \{ R \leftarrow { R } + 1 \} \\
\{ R \leftarrow { R } + 1 \} & \quad \{ R \leftarrow \max(0, { R } + 1) \} \\
\{ R \leftarrow \max(0, { R } + 1) \} & \quad \{ R \leftarrow \max(0, { R } + 1) \}
\end{align*}

Figure 3.942: Automaton for the \textit{NB\_PLAIN} constraint obtained by applying decoration Table 2.37 to the seed transducer of the \textit{PLAIN} pattern where \textit{default} is 0; \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-2} + 1 \geq 0 \) are linear invariants.
Table 3.131: Glue matrix for the NB.PLAIN constraint defined as the composition of the PLAIN pattern, the feature ONE, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.398 NB_PLATEAU

**Origin**
Based on the PLATEAU pattern.

**Constraint**

```plaintext
NB_PLATEAU(VALUE, VARIABLES)
```

**Arguments**

```plaintext
VALUE : dvar
VARIABLES : collection(var−dvar)
```

**Restrictions**

```plaintext
sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = 0
VALUE ≥ 0
VALUE ≤ max(0, ⌊(sv − 1)/2⌋)

where
sv = |VARIABLES|
rv = range(VARIABLES.var)
```

**Purpose**
VALUE is the number of occurrences of the PLATEAU pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression `<=*`.

**Example**

```plaintext
(3, ⟨7, 5, 2, 3, 1, 2, 2, 4, 3, 4, 5, 5, 2, 2, 5⟩)
```

Figure 3.943 provides an example where the NB_PLATEAU (3, ⟨7, 5, 2, 3, 1, 2, 2, 4, 3, 4, 5, 5, 2, 2, 5⟩) constraint holds.

**Typical**

```plaintext
|VARIABLES| > 2
range(VARIABLES.var) > 1
```

**Symmetries**

- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.943: Illustrating the **NB_PLATEAU** constraint of the **Example** slot
Automaton Figures 3.944 and 3.945 respectively depict the automaton associated with the constraint NB_PLATEAU and its simplified form.

\[
\begin{align*}
\text{Fig. 3.944: Automaton for the NB_PLATEAU constraint obtained by applying decoration Table 2.35 to the seed transducer of the PLATEAU pattern where default is 0.}
\end{align*}
\]

\[
\begin{align*}
\text{Fig. 3.945: Automaton for the NB_PLATEAU constraint obtained by applying decoration Table 2.37 to the seed transducer of the PLATEAU pattern where default is 0; } R_i - R_{i-1} \geq 0 \text{ and } -R_i + R_{i-2} + 1 \geq 0 \text{ are linear invariants.}
\end{align*}
\]
Table 3.132: Glue matrix for the `NB.PLATEAU` constraint defined as the composition of the `PLATEAU` pattern, the feature `ONE`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.399  NB PROPER PLAIN

**Description**

Based on the **PROPER PLAIN** pattern.

**Constraint**

\[ \text{NB PROPER PLAIN}(\text{VALUE, VARIABLES}) \]

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var, dvar)

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 3 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & \geq 0 \\
\text{VALUE} & \leq \max(0, \left(\text{sv} - 1\right)/3) \times \text{required}(\text{VARIABLES, var})
\end{align*}
\]

where

- \( \text{sv} = |\text{VARIABLES}| \)
- \( \text{rv} = \text{range}(\text{VARIABLES}.\text{var}) \)

**Purpose**

**VALUE** is the number of occurrences of the **PROPER PLAIN** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, **VALUE** takes the default value 0.

An occurrence of the pattern **PROPER PLAIN** is the **maximal** subsequence which matches the regular expression \( >\geq+< \).

**Example**

\((3, (2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3, 5))\)

Figure 3.946 provides an example where the **NB PROPER PLAIN** \((3, (2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3, 5))\) constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| & > 3 \\
\text{range}(\text{VARIABLES}.\text{var}) & > 1
\end{align*}
\]

**Symmetries**

- Items of **VARIABLES** can be reversed.
- One and the same constant can be added to the var attribute of all items of **VARIABLES**.

**Arg. properties**

- **Functional dependency**: **VALUE** determined by **VARIABLES**.
Figure 3.946: Illustrating the NB_PROPER_PLAIN constraint of the Example slot
Automaton

Figures 3.947 and 3.948 respectively depict the automaton associated with the constraint \( \text{NB\_PROPER\_PLAIN} \) and its simplified form.

\[
\begin{align*}
\{ D \leftarrow \max(D, 1) \} & \quad \leq \quad \{ R \leftarrow R + 1 \} \\
\{ D \leftarrow \max(D, 1) \} & \quad > \quad \{ D \leftarrow 0 \}
\end{align*}
\]

\[
\begin{align*}
\{ C \leftarrow \text{default} \} & \quad \leq \quad \{ D \leftarrow 0 \} \\
\{ R \leftarrow \text{default} \} & \quad > \quad \{ D \leftarrow \text{default} \}
\end{align*}
\]

Figure 3.947: Automaton for the \( \text{NB\_PROPER\_PLAIN} \) constraint obtained by applying decoration Table 2.35 to the seed transducer of the \( \text{PROPER\_PLAIN} \) pattern where \( \text{default} \) is 0.

\[
\begin{align*}
\{ R \leftarrow \text{default} \} & \quad \leq \quad \{ R \leftarrow \text{default} \} \\
\{ R \leftarrow \text{default} \} & \quad > \quad \{ R \leftarrow \text{default} \}
\end{align*}
\]

\[
\begin{align*}
\{ R \leftarrow R + 1 \} & \quad > \quad \{ R \leftarrow R + 1 \} \\
\{ R \leftarrow \text{default} \} & \quad < \quad \{ R \leftarrow \text{default} \}
\end{align*}
\]

Figure 3.948: Automaton for the \( \text{NB\_PROPER\_PLAIN} \) constraint obtained by applying decoration Table 2.37 to the seed transducer of the \( \text{PROPER\_PLAIN} \) pattern where \( \text{default} \) is 0; \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-3} + 1 \geq 0 \) are linear invariants.
Table 3.133: Glue matrix for the NB_PROPER.PLAIN constraint defined as the composition of the PROPER.PLAIN pattern, the feature ONE, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{C}$</td>
</tr>
<tr>
<td>r</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{C}$</td>
<td>1</td>
</tr>
<tr>
<td>t</td>
<td>$\overline{C} + \overline{C}$</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
### 3.400 NB_PROPER_PLATEAU

#### Description

**Origin**

Based on the PROPER_PLATEAU pattern.

**Constraint**

\[ \text{NB_PROPER_PLATEAU}(\text{VALUE}, \text{VARIABLES}) \]

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>dvar</td>
</tr>
<tr>
<td>VARIABLES</td>
<td>collection(var − dvar)</td>
</tr>
</tbody>
</table>

**Restrictions**

\[
\begin{align*}
sv & \leq 3 \lor rv \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & \geq 0 \\
\text{VALUE} & \leq \max(0, \left\lfloor (sv - 1)/3 \right\rfloor) \\
\text{required(} & \text{VARIABLES, var})
\end{align*}
\]

where

\[
\begin{align*}
sv &= |\text{VARIABLES}| \\
rv &= \text{range(} \text{VARIABLES}.\text{var})
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{VALUE} & \leq 4
\end{align*}
\]

**Purpose**

VALUE is the number of occurrences of the PROPER_PLATEAU pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression \(< = + >\).

**Example**

\[(3, \langle 7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3 \rangle)\]

Figure 3.949 provides an example where the NB_PROPER_PLATEAU \((3, \langle 7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3 \rangle)\) constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| & > 3 \\
\text{range(} & \text{VARIABLES}.\text{var}) > 1
\end{align*}
\]

**Symmetries**

- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.949: Illustrating the NB_PROPER_PLATEAU constraint of the Example slot
Automaton

Figures 3.950 and 3.951 respectively depict the automaton associated with the constraint `NB_PROPER_PLATEAU` and its simplified form.

Figure 3.950: Automaton for the `NB_PROPER_PLATEAU` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `PROPER_PLATEAU` pattern where default is 0.

Figure 3.951: Automaton for the `NB_PROPER_PLATEAU` constraint obtained by applying decoration Table 2.37 to the seed transducer of the `PROPER_PLATEAU` pattern where default is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-3} + 1 \geq 0$ are linear invariants.
Table 3.134: Glue matrix for the `NB.PROPER.PLATEAU` constraint defined as the composition of the `PROPER.PLATEAU` pattern, the `feature.ONE`, and the `aggregator.sum`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.401 NB_STEADY

**DESCRIPTION**

**Origin**
Based on the STEADY pattern.

**Constraint**
NB_STEADY(VALUE, VARIABLES)

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**

<table>
<thead>
<tr>
<th>Condition</th>
<th>Constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>sv ≤ 1</td>
<td>VALUE = 0</td>
</tr>
<tr>
<td>rv = 1</td>
<td>VALUE ≥ sv − 1</td>
</tr>
<tr>
<td>rv ≥ 2</td>
<td>VALUE ≥ 0</td>
</tr>
<tr>
<td>VALUE ≤ max(0, sv − 1)</td>
<td>required(VARIABLES, var)</td>
</tr>
</tbody>
</table>

where

- sv = |VARIABLES|
- rv = range(VARIABLES.var)

**Purpose**

VALUE is the number of occurrences of the STEADY pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern STEADY is the subsequence which matches the regular expression =.

**Example**

(7, (1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 5, 7, 2, 6, 6))

Figure 3.952 provides an example where the NB_STEADY (7, [1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 5, 7, 2, 6, 6]) constraint holds.

**Typical**

| VARIABLES | > 1 |

**Symmetries**

- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.952: Illustrating the **NB_STEADY** constraint of the **Example** slot
Automaton

Figures 3.953 and 3.954 respectively depict the automaton associated with the constraint \texttt{NB\_STEADY} and its simplified form.

\[
\begin{align*}
\{ C \leftarrow \text{default} \\ D \leftarrow 0 \\ R \leftarrow \text{default} \} \\
\{ D \leftarrow 0 \\ R \leftarrow R + \max(D, 1) \}
\end{align*}
\]

Figure 3.953: Automaton for the \texttt{NB\_STEADY} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{STEADY} pattern where \texttt{default} is 0

\[
\{ R \leftarrow \text{default} \} \\
\{ R \leftarrow R + 1 \}
\]

Figure 3.954: Automaton for the \texttt{NB\_STEADY} constraint obtained by applying decoration Table 2.37 to the seed transducer of the \texttt{STEADY} pattern where \texttt{default} is 0; \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-1} + 1 \geq 0 \) are linear invariants.

Table 3.135: Glue matrix for the \texttt{NB\_STEADY} constraint defined as the composition of the \texttt{STEADY} pattern, the feature \texttt{ONE}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.402  **NB_STEADY_SEQUENCE**

**Description**

Origin

Based on the `STEADY_SEQUENCE` pattern.

Constraint

```
NB_STEADY_SEQUENCE(VALUE, VARIABLES)
```

Arguments

```
VALUE : dvar
VARIABLES : collection(var - dvar)
```

Restrictions

```
sv \leq 1 \Rightarrow VALUE = 0
rv = 1 \land sv \geq 2 \Rightarrow VALUE \geq 1
rv \geq 2 \Rightarrow VALUE \geq 0
rv = 1 \land sv \geq 2 \Rightarrow VALUE \leq 1
rv \geq 2 \land sv \geq 2 \Rightarrow VALUE \leq \lfloor sv/2 \rfloor
```

where

```
sv = |VARIABLES|
rv = range(VARIABLES.var)
```

Purpose

VALUE is the number of occurrences of the `STEADY_SEQUENCE` pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern `STEADY_SEQUENCE` is the maximal subsequence which matches the regular expression `\=+`. 

Example

```
(5, (3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1))
```

Figure 3.955 provides an example where the `NB_STEADY_SEQUENCE (5, [3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1])` constraint holds.

Typical

```
|VARIABLES| > 1
```

Symmetries

- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties

**Functional dependency:** VALUE determined by VARIABLES.
Figure 3.955: Illustrating the NB_STEADY_SEQUENCE constraint of the Example slot
Automaton

Figures 3.956 and 3.957 respectively depict the automaton associated with the constraint \( \text{NB\_STEADY\_SEQUENCE} \) and its simplified form.

\[
\begin{align*}
\{ C \leftarrow \text{default} \} \\
D \leftarrow 0 \\
R \leftarrow \text{default}
\end{align*}
\]

\[
\begin{align*}
\{ C \leftarrow \text{default} \} \\
D \leftarrow 0 \\
R \leftarrow R + C
\end{align*}
\]

\[
\begin{align*}
C \leftarrow \max(D, 1) \\
D \leftarrow 0 \\
C \leftarrow \max(C, \max(D, 1)) \\
D \leftarrow 0
\end{align*}
\]

Figure 3.956: Automaton for the \( \text{NB\_STEADY\_SEQUENCE} \) constraint obtained by applying decoration Table 2.35 to the seed transducer of the \( \text{STEADY\_SEQUENCE} \) pattern where default is 0.

\[
\begin{align*}
\{ R \leftarrow \text{default} \}
\end{align*}
\]

\[
\begin{align*}
\{ R \leftarrow R + 1 \}
\end{align*}
\]

Figure 3.957: Automaton for the \( \text{NB\_STEADY\_SEQUENCE} \) constraint obtained by applying decoration Table 2.37 to the seed transducer of the \( \text{STEADY\_SEQUENCE} \) pattern where default is 0; \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-2} + 1 \geq 0 \) are linear invariants.
Table 3.136: Glue matrix for the NB_STEADY_SEQUENCE constraint defined as the composition of the STEADYSEQUENCE pattern, the feature ONE, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### NB\_STRICTLY\_DECREASING\_SEQUENCE

**Description**

Based on the `STRICTLY\_DECREASING\_SEQUENCE` pattern.

**Constraint**

`NB\_STRICTLY\_DECREASING\_SEQUENCE(VALUE, VARIABLES)`

**Arguments**

- `VALUE` : `dvar`
- `VARIABLES` : `collection(var - dvar)`

**Restrictions**

\[
sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \\
VALUE \geq 0 \\
VALUE \leq \lfloor sv/2 \rfloor \\
\text{required}(VARIABLES, var)
\]

where

\[
sv = |VARIABLES| \\
rv = \text{range}(VARIABLES, var)
\]

**Purpose**

VALUE is the number of occurrences of the `STRICTLY\_DECREASING\_SEQUENCE` pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern `STRICTLY\_DECREASING\_SEQUENCE` is the maximal sub-sequence which matches the regular expression `>\+`.

**Example**

\[
(3, (4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3))
\]

Figure 3.958 provides an example where the `NB\_STRICTLY\_DECREASING\_SEQUENCE (3, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3])` constraint holds.

**Typical**

\[
|VARIABLES| > 1 \\
\text{range}(VARIABLES, var) > 1
\]

**Symmetry**

One and the same constant can be added to the `var` attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.958: Illustrating the NB_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 3.959 and 3.960 respectively depict the automaton associated with the constraint \texttt{NB\_STRICTLY\_DECREASING\_SEQUENCE} and its simplified form.

\begin{align*}
\{ & C \leftarrow \text{default} \\
& D \leftarrow 0 \\
& R \leftarrow \text{default} \}
\end{align*}

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{automaton_nb_strictly_decreasing_sequence}
\caption{Automaton for the \texttt{NB\_STRICTLY\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{STRICTLY\_DECREASING\_SEQUENCE} pattern where \texttt{default} is 0\protect\footnote{\texttt{default} is 0; \( R_i - R_{i-1} \geq 0 \) and \(-R_i + R_{i-2} + 1 \geq 0 \) are linear invariants.}}
\end{figure}

\begin{align*}
\{ & R \leftarrow \text{default} \}
\end{align*}

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{automaton_nb_strictly_decreasing_sequence_2}
\caption{Automaton for the \texttt{NB\_STRICTLY\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.37 to the seed transducer of the \texttt{STRICTLY\_DECREASING\_SEQUENCE} pattern where \texttt{default} is 0; \( R_i - R_{i-1} \geq 0 \) and \(-R_i + R_{i-2} + 1 \geq 0 \) are linear invariants.}
\end{figure}
Table 3.137: Glue matrix for the **NB_STRICTLY_DECREASING_SEQUENCE** constraint defined as the composition of the **STRICTLY_DECREASING_SEQUENCE** pattern, the feature **ONE**, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
</tr>
<tr>
<td>r</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>1</td>
</tr>
</tbody>
</table>
3.404  **NB_STRICTLY_INCREASING_SEQUENCE**

**DESCRIPTION**

**Origin**
Based on the **STRICTLY_INCREASING_SEQUENCE** pattern.

**Constraint**

\[
\text{NB\_STRICTLY\_INCREASING\_SEQUENCE}(\text{VALUE, VARIABLES})
\]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} \rightarrow \text{dvar})
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{sv} \leq 1 \lor \text{rv} \leq 1 & \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & \geq 0 \\
\text{VALUE} & \leq \lfloor \text{sv}/2 \rfloor
\end{align*}
\]

where

\[
\text{sv} = |\text{VARIABLES}| \\
\text{rv} = \text{range}(\text{VARIABLES}.\text{var})
\]

**Purpose**

VALUE is the number of occurrences of the **STRICTLY_INCREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern **STRICTLY_INCREASING_SEQUENCE** is the maximal subsequence which matches the regular expression \(<^+\).

**Example**

\[
(3, (4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3))
\]

Figure 3.961 provides an example where the **NB\_STRICTLY\_INCREASING\_SEQUENCE** (3, [4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3]) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Symmetry**

One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.961: Illustrating the NB\_STRICTLY\_INCREASING\_SEQUENCE constraint of the Example slot
Automaton

Figures 3.962 and 3.963 respectively depict the automaton associated with the constraint `NB_STRICTLY_INCREASING_SEQUENCE` and its simplified form.

Figure 3.962: Automaton for the `NB_STRICTLY_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `STRICTLY_INCREASING_SEQUENCE` pattern where `default` is 0.

Figure 3.963: Automaton for the `NB_STRICTLY_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 2.37 to the seed transducer of the `STRICTLY_INCREASING_SEQUENCE` pattern where `default` is 0; \( R_i - R_{i-1} \geq 0 \) and \(-R_i + R_{i-2} + 1 \geq 0\) are linear invariants.
Table 3.138: Glue matrix for the `NB_STRICTLY_INCREASING_SEQUENCE` constraint defined as the composition of the `STRICTLY_INCREASING_SEQUENCE` pattern, the feature `ONE`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.405  nb_summit

**Description**

Based on the `summit` pattern.

**Constraint**

`nb_summit(VALUE, VARIABLES)`

**Arguments**

- `VALUE : dvar`
- `VARIABLES : collection(var - dvar)`

**Restrictions**

\[
sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = 0
\]

\[
VALUE \geq 0\text{,} \\
VALUE \leq \max(0, [(sv - 1)/2])\text{.}
\]

\[
\text{required}(\text{VARIABLES, var})
\]

where

\[
sv = |\text{VARIABLES}|
\]

\[
rv = \text{range}(\text{VARIABLES, var})
\]

**Purpose**

VALUE is the number of occurrences of the `summit` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern `summit` is the maximal subsequence which matches the regular expression `(\langle | \langle (= | \langle)\ast \rangle \rangle | \langle (= | \langle)\ast \rangle)`.  

**Example**

\[
(3,(7,1,5,4,4,3,3,4,6,6,2,3,4,2,3,1))
\]

Figure 3.964 provides an example where the `nb_summit (3,[7,1,5,4,4,3,3,4,6,6,2,3,4,2,3,1])` constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 2
\]

\[
\text{range}(\text{VARIABLES, var}) > 1
\]

**Symmetries**

- Items of `VARIABLES` can be reversed.
- One and the same constant can be added to the `var` attribute of all items of `VARIABLES`.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.964: Illustrating the NB_SUMMIT constraint of the Example slot
Figures 3.965 and 3.966 respectively depict the automaton associated with the constraint NB_SUMMIT and its simplified form.

Figure 3.965: Automaton for the NB_SUMMIT constraint obtained by applying decoration Table 2.35 to the seed transducer of the SUMMIT pattern where default is 0 (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$)
Figure 3.966: Automaton for the NB_SUMMIT constraint obtained by applying decoration Table 2.37 to the seed transducer of the SUMMIT pattern where default is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-2} + 1 \geq 0$ are linear invariants.

Table 3.139: Glue matrix for the NB_SUMMIT constraint defined as the composition of the SUMMIT pattern, the feature ONE, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### 3.406 NB_VALLEY

#### DESCRIPTION

**Origin**
- Based on the VALLEY pattern.

**Constraint**
- \( \text{NB\_VALLEY(VALUE, VARIABLES)} \)

**Arguments**
- \( \text{VALUE} : \text{dvar} \)
- \( \text{VARIABLES} : \text{collection(var\_dvar)} \)

**Restrictions**
- \( sv \leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = 0 \)
- \( \text{VALUE} \geq 0 \)
- \( \text{VALUE} \leq \max(0, \lfloor (sv - 1)/2 \rfloor) \)
- \( \text{required(VARIABLES, var)} \)
  - where
  - \( sv = |\text{VARIABLES}| \)
  - \( rv = \text{range(\text{VARIABLES}.var)} \)

**Purpose**
- \( \text{VALUE} \) is the number of occurrences of the VALLEY pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, \( \text{VALUE} \) takes the default value 0.
- An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression \( > (= \ | >) ^* (< \ | =) ^* < \).

**Example**
- \((3, [1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 7])\)

Figure 3.967 provides an example where the NB_VALLEY \((3, [1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 7])\) constraint holds.

**Typical**
- \( |\text{VARIABLES}| > 2 \)
- \( \text{range(\text{VARIABLES}.var)} > 1 \)

**Symmetries**
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
- Functional dependency: \( \text{VALUE} \) determined by VARIABLES.
Figure 3.967: Illustrating the **NB_VALLEY** constraint of the **Example** slot
Automaton

Figures 3.968 and 3.969 respectively depict the automaton associated with the constraint NB\_VALLEY and its simplified form.

![Automaton diagram](image)

Figure 3.968: Automaton for the NB\_VALLEY constraint obtained by applying decoration Table 2.35 to the seed transducer of the VALLEY pattern where default is 0

![Automaton diagram](image)

Figure 3.969: Automaton for the NB\_VALLEY constraint obtained by applying decoration Table 2.37 to the seed transducer of the VALLEY pattern where default is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-2} + 1 \geq 0$ are linear invariants.
Table 3.140: Glue matrix for the NB_VALLEY constraint defined as the composition of the VALLEY pattern, the feature ONE, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.


3.407  NB_ZIGZAG

Origin  Based on the ZIGZAG pattern.

Constraint  \text{NB}_\text{ZIGZAG}(\text{VALUE, VARIABLES})

Arguments  
\text{VALUE} : \text{dvar} \\
\text{VARIABLES} : \text{collection} (\text{var} - \text{dvar})

Restrictions  
\text{sv} \leq 3 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} \geq 0 \\
\text{rv} = 2 \Rightarrow \text{VALUE} \leq \lfloor \text{sv}/4 \rfloor \\
\text{rv} \geq 3 \Rightarrow \text{VALUE} \leq \max(0, \lfloor (\text{sv} - 1)/3 \rfloor) \\
\text{required}(\text{VARIABLES}, \text{var})

where

\text{sv} = |\text{VARIABLES}| \\
\text{rv} = \text{range}(\text{VARIABLES}.\text{var})

Purpose \text{VALUE} is the number of occurrences of the ZIGZAG pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression \((<>)^+(< | <>) | (<>)^+(> | ><))\).

Example  
(3, (4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1))

Figure 3.970 provides an example where the \text{NB}_\text{ZIGZAG} (3, [4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1]) constraint holds.

Typical  
\(|\text{VARIABLES}| > 3 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1

Symmetries  
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties  Functional dependency: VALUE determined by VARIABLES.
Figure 3.970: Illustrating the NB_ZIGZAG constraint of the Example slot
Figures 3.971 and 3.972 respectively depict the automaton associated with the constraint \text{NB\_ZIGZAG} and its simplified form.

\begin{center}
\begin{tabular}{|l|l|l|l|l|l|l|}
\hline
 & \(s\) & \(a\) & \(b\) & \(c\) & \(d\) & \(e\) & \(f\) \\
\hline
\(s\) & \(\overrightarrow{C} + \overrightarrow{C}\) & \(\overrightarrow{C} + \overrightarrow{C}\) & \(\overrightarrow{C} + \overrightarrow{C}\) & \(\overrightarrow{C} + \overrightarrow{C}\) & \(\overrightarrow{C} + \overrightarrow{C}\) & \(\overrightarrow{C} + \overrightarrow{C}\) & \(\overrightarrow{C} + \overrightarrow{C}\) \\
\(a\) & \(\overrightarrow{C} + \overrightarrow{C}\) & \(\overrightarrow{C} + \overrightarrow{C}\) & \(\overrightarrow{C} + \overrightarrow{C}\) & \(\overrightarrow{C} + \overrightarrow{C}\) & \(\overrightarrow{C} + \overrightarrow{C}\) & \(\overrightarrow{C} + \overrightarrow{C}\) & \(\overrightarrow{C} + \overrightarrow{C}\) \\
\(b\) & \(\overrightarrow{C} + \overrightarrow{C}\) & \(\overrightarrow{C} + \overrightarrow{C}\) & 1 & \(\overrightarrow{C} + \overrightarrow{C}\) & 1 & \(\overrightarrow{C} + \overrightarrow{C}\) & 1 \\
\(c\) & \(\overrightarrow{C} + \overrightarrow{C}\) & 1 & \(\overrightarrow{C} + \overrightarrow{C}\) & 1 & \(\overrightarrow{C} + \overrightarrow{C}\) & 1 & \(\overrightarrow{C} + \overrightarrow{C}\) \\
\(d\) & \(\overrightarrow{C} + \overrightarrow{C}\) & \(\overrightarrow{C} + \overrightarrow{C}\) & 1 & \(\overrightarrow{C} + \overrightarrow{C}\) & \(\overrightarrow{C} + \overrightarrow{C}\) & \(\overrightarrow{C} + \overrightarrow{C}\) & 1 \\
\(e\) & \(\overrightarrow{C} + \overrightarrow{C}\) & 1 & \(\overrightarrow{C} + \overrightarrow{C}\) & 1 & \(\overrightarrow{C} + \overrightarrow{C}\) & 1 & \(\overrightarrow{C} + \overrightarrow{C}\) \\
\(f\) & \(\overrightarrow{C} + \overrightarrow{C}\) & \(\overrightarrow{C} + \overrightarrow{C}\) & 1 & \(\overrightarrow{C} + \overrightarrow{C}\) & 1 & \(\overrightarrow{C} + \overrightarrow{C}\) & 1 \\
\hline
\end{tabular}
\end{center}

Table 3.141: Glue matrix for the \text{NB\_ZIGZAG} constraint defined as the composition of the \text{ZIGZAG} pattern, the feature \text{ONE}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 3.971: Automaton for the NB_ZIGZAG constraint obtained by applying decoration Table 2.35 to the seed transducer of the ZIGZAG pattern where default is 0; (1) missing transitions from a, b, c, d, e, f to s are labelled by =; (2) on transitions from b, c, e, f to s the accumulator D is reset to its initial value; (3) on transitions from c, f to s the accumulator R is updated wrt C and the accumulator C is reset to its initial value.
Figure 3.972: Automaton for the NB_ZIGZAG constraint obtained by applying decoration Table 2.37 to the seed transducer of the ZIGZAG pattern where default is 0; missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=; R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-3} + 1 \geq 0$ are linear invariants.
3.408  POS_MAX_HEIGHT_DECREASING_TERRACE

**DESCRIPTION**

 Origin: Based on constraint `MAX_HEIGHT DECREASING TERRACE`.

**Constraint**

`POS_MAX_HEIGHT_DECREASING_TERRACE(VALUE, VARIABLES, FOUND)`

**Arguments**

- `VALUE` : `dvar`
- `VARIABLES` : `collection(var−dvar)`
- `FOUND` : `collection(var−dvar)`

**Restrictions**

- `sv ≤ 3 ∨ rv ≤ 2 ⇒ VALUE = −∞`
- `VALUE = −∞ ∨ VALUE ≥ minv + 1`
- `VALUE ≤ maxv − 1`
- `required(VARIABLES, var)`
- `required(FOUND, var)`

where

- `maxv = maxval(VARIABLES.var)`
- `minv = minval(VARIABLES.var)`
- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

**Purpose**

The constraint `MAX_HEIGHT DECREASING_TERRACE(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `DECREASING_TERRACE` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `DECREASING_TERRACE` is the maximal subsequence which matches the regular expression `> =+ >`.

Assume that the occurrence of the pattern `DECREASING_TERRACE` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i + 1` to index `j`.

**Example**

```
( 4, [6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3],
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0 )
```

Figure 3.973 provides an example where the `POS_MAX_HEIGHT_DECREASING_TERRACE (4, [6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0])` constraint holds.

**Typical**

- `|VARIABLES| > 3`
- `range(VARIABLES.var) > 2`
Figure 3.973: Illustrating the POS_MAX_HEIGHT_DECREASING_TERRACE constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_HEIGHT_DECREASING_TERRACE constraint but use the decoration table 2.33.
3.409  POS_MAX_HEIGHT_INCREASING_TERRACE

**Origin**  
Based on constraint `MAX_HEIGHT_INCREASING_TERRACE`.

**Constraint**  
`POS_MAX_HEIGHT_INCREASING_TERRACE(VALUE, VARIABLES, FOUND)`

**Arguments**  
- `VALUE : dvar`
- `VARIABLES : collection(var−dvar)`
- `FOUND : collection(var−dvar)`

**Restrictions**  
- `sv ≤ 3 ∨ rv ≤ 2 ⇒ VALUE = −∞`
- `VALUE = −∞ ∨ VALUE ≥ minv + 1`
- `VALUE ≤ maxv − 1`
- `required(VARIABLES, var)`
- `required(FOUND, var)`

where
- `maxv = maxval(VARIABLES, var)`
- `minv = minval(VARIABLES, var)`
- `sv = |VARIABLES|`
- `rv = range(VARIABLES, var)`

The constraint `MAX_HEIGHT_INCREASING_TERRACE(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern `INCREASING_TERRACE` for which the feature value is `VALUE`.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern `INCREASING_TERRACE` is the maximal subsequence which matches the regular expression `<=+<`.

Assume that the occurrence of the pattern `INCREASING_TERRACE` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i + 1` to index `j`.

**Example**  

Figure 3.974 provides an example where the `POS_MAX_HEIGHT_INCREASING_TERRACE ([5, 1, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4], [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])` constraint holds.

**Purpose**
An occurrence of the pattern `INCREASING_TERRACE` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i + 1` to index `j`.

**Typical**  
- `|VARIABLES| > 3`
- `range(VARIABLES, var) > 2`
Figure 3.974: Illustrating the POS_MAX_HEIGHT_INCREASING_TERRACE constraint of the Example slot

**Arg. properties**
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_HEIGHT_INCREASING_TERRACE constraint but use the decoration table 2.33.
3.410  POS_MAX_HEIGHT.PLAIN

Origin
Based on constraint MAX_HEIGHT.PLAIN.

Constraint
POS_MAX_HEIGHT.PLAIN(VALUE, VARIABLES, FOUND)

Arguments
\[
\text{VALUE} : \text{dvar} \\
\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar}) \\
\text{FOUND} : \text{collection}(\text{var} - \text{dvar})
\]

Restrictions
\[
\text{sv} \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} = -\infty \lor \text{VALUE} \geq \text{minv} \\
\text{VALUE} \leq \text{maxv} - 1 \\
\text{required}(	ext{VARIABLES, var}) \\
\text{required}(	ext{FOUND, var})
\]

where
\[
\text{maxv} = \text{maxval}(	ext{VARIABLES, var}) \\
\text{minv} = \text{minval}(	ext{VARIABLES, var}) \\
\text{sv} = |\text{VARIABLES}| \\
\text{rv} = \text{range}(	ext{VARIABLES, var})
\]

Purpose
The constraint MAX_HEIGHT.PLAIN(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern PLAIN for which the feature value is VALUE. The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression \( \geq^* < \). Assume that the occurrence of the pattern PLAIN starts at position \( i \) and ends at position \( j \). The feature MIN computes the minimum of the values from index \( i + 1 \) to index \( j \).

Example
\[
(5, (2, 3, 6, 5, 7, 6, 4, 5, 4, 3, 3, 6, 6, 3), (0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0))
\]

Figure 3.975 provides an example where the POS_MAX_HEIGHT.PLAIN
\[
(5, [2, 3, 6, 5, 7, 6, 4, 5, 4, 3, 3, 6, 6, 3], [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
\]

constraint holds.

Typical
\[
|\text{VARIABLES}| > 2 \\
\text{range}(	ext{VARIABLES, var}) > 1
\]

Arg. properties
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Figure 3.975: Illustrating the POS_MAX_HEIGHT_PLAIN constraint of the Example slot
Automaton

Similar to the automaton of the MAX\_HEIGHT\_PLAIN constraint but use the decoration table 2.33.
### 3.4.11 POS_MAX_HEIGHT_PLATEAU

**Origin**
Based on constraint `MAX_HEIGHT_PLATEAU`.

**Constraint**

\[
\text{POS_MAX_HEIGHT_PLATEAU}([\text{VALUE}, \text{VARIABLES}, \text{FOUND}])
\]

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td><code>dvar</code></td>
</tr>
<tr>
<td>VARIABLES</td>
<td><code>collection(var-dvar)</code></td>
</tr>
<tr>
<td>FOUND</td>
<td><code>collection(var-dvar)</code></td>
</tr>
</tbody>
</table>

**Restrictions**

\[
\begin{align*}
sv \leq 2 & \lor rv \leq 1 \implies \text{VALUE} = -\infty \\
\text{VALUE} = -\infty & \lor \text{VALUE} \geq \text{minv} + 1 \\
\text{VALUE} \leq \maxv & \\
\text{required} & (\text{VARIABLES}, \text{var}) \\
\text{required} & (\text{FOUND}, \text{var})
\end{align*}
\]

where

- \( \maxv = \maxval(\text{VARIABLES}.\text{var}) \)
- \( \minv = \minval(\text{VARIABLES}.\text{var}) \)
- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

The constraint `MAX_HEIGHT_PLATEAU(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `PLATEAU` for which the feature value is `VALUE`. The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `PLATEAU` is the *maximal* subsequence which matches the regular expression `<=*`.

Assume that the occurrence of the pattern `PLATEAU` starts at position \( i \) and ends at position \( j \). The feature `MIN` computes the minimum of the values from index \( i + 1 \) to index \( j \).

**Purpose**

Example

\[
\begin{pmatrix}
5, (7, 5, 2, 3, 1, 2, 4, 3, 4, 5, 5, 2, 2, 5), \\
\{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0\}
\end{pmatrix}
\]

Figure 3.976 provides an example where the `POS_MAX_HEIGHT_PLATEAU` constraint holds.

**Typical**

<table>
<thead>
<tr>
<th>Condition</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>`</td>
<td>VARIABLES</td>
</tr>
<tr>
<td><code>range(VARIABLES.var) &gt; 1</code></td>
<td></td>
</tr>
</tbody>
</table>
Figure 3.976: Illustrating the POS_MAX_HEIGHT_PLATEAU constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MAX.HEIGHT_PLATEAU} constraint but use the decoration table \texttt{2.33}. 
POS_MAX_HEIGHT_PLATEAU 1775
3.412  POS_MAX_HEIGHT_PROPER_PLAIN

**Origin**
Based on constraint MAX_HEIGHT_PROPER_PLAIN.

**Constraint**

```plaintext
POS_MAX_HEIGHT_PROPER_PLAIN(VALUE, VARIABLES, FOUND)
```

**Arguments**

- `VALUE`: dvar
- `VARIABLES`: collection(var−dvar)
- `FOUND`: collection(var−dvar)

**Restrictions**

- \( sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = -\infty \)
- \( VALUE = -\infty \lor VALUE \geq \text{minv} \)
- \( VALUE \leq \text{maxv} - 1 \)
- \( \text{required}(\text{VARIABLES}, \text{var}) \)
- \( \text{required}(\text{FOUND}, \text{var}) \)
  
  where
  - \( \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \)
  - \( \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \)
  - \( sv = |\text{VARIABLES}| \)
  - \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

The constraint MAX_HEIGHT_PROPER_PLAIN(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern PROPER_PLAIN for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern PROPER_PLAIN is the maximal subsequence which matches the regular expression \( > =^+ < \).

Assume that the occurrence of the pattern PROPER_PLAIN starts at position \( i \) and ends at position \( j \). The feature \( \text{MIN} \) computes the minimum of the values from index \( i + 1 \) to index \( j \).

**Purpose**

**Example**

\[
(5, [2, 7, 5, 6, 3, 7, 4, 5, 6, 5, 3, 3, 3, 5], [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
\]

Figure 3.977 provides an example where the POS_MAX_HEIGHT_PROPER_PLAIN \((5, [2, 7, 5, 6, 3, 7, 4, 5, 6, 5, 3, 3, 3, 5], [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])\) constraint holds.

**Typical**

\(|\text{VARIABLES}| > 3\)

\(\text{range} (\text{VARIABLES} . \text{var}) > 1\)
Figure 3.977: Illustrating the POS_MAX_HEIGHT_PROPER_PLAIN constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_HEIGHT_PROPERPLAIN constraint but use the decoration table 2.33.
### 3.413 POS_MAX_HEIGHT_PROPER_PLATEAU

**Description**

Based on constraint `MAX_HEIGHT_PROPER_PLATEAU`.

**Constraint**

`POS_MAX_HEIGHT_PROPER_PLATEAU(VALUE, VARIABLES, FOUND)`

**Arguments**

- `VALUE`: `dvar`
- `VARIABLES`: `collection(var−dvar)`
- `FOUND`: `collection(var−dvar)`

**Restrictions**

\[sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = -\infty\]

\[\text{VALUE} = -\infty \lor \text{VALUE} \geq \text{minv} + 1\]

\[\text{VALUE} \leq \text{maxv}\]

\[\text{required}(\text{VARIABLES}, \text{var})\]

\[\text{required}(\text{FOUND}, \text{var})\]

where

\[\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})\]

\[\text{minv} = \text{minval}(\text{VARIABLES}.\text{var})\]

\[sv = |\text{VARIABLES}|\]

\[rv = |\text{range}(\text{VARIABLES}.\text{var})|\]

**Purpose**

The constraint `MAX_HEIGHT_PROPER_PLATEAU(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `PROPER_PLATEAU` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `PROPER_PLATEAU` is the `maximal` subsequence which matches the regular expression `\(<=^+\>`.

Assume that the occurrence of the pattern `PROPER_PLATEAU` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i + 1` to index `j`.

**Example**

\[
\begin{pmatrix}
5, (7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3), \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0)
\end{pmatrix}
\]

Figure 3.978 provides an example where the `POS_MAX_HEIGHT_PROPER_PLATEAU` constraint holds.

**Typical**

\[|\text{VARIABLES}| > 3\]

\[|\text{range}(\text{VARIABLES}.\text{var})| > 1\]
Figure 3.978: Illustrating the POS_MAX_HEIGHT_PROPER_PLATEAU constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_HEIGHT_PROPER_PLATEAU constraint but use the decoration table 2.33.
POS_MAX_HEIGHT_PROPER_PLATEAU 1783
3.414 POS_MAX_HEIGHT_STEADY

**DESCRIPTION**

**Origin**

Based on constraint MAX_HEIGHT_STEADY.

**Constraint**

POS_MAX_HEIGHT_STEADY(VALUE, VARIABLES, FOUND)

**Arguments**

VALUE : dvar
VARIABLES : collection(var-dvar)
FOUND : collection(var-dvar)

**Restrictions**

sv ≤ 1 ⇒ VALUE = −∞
VALUE = −∞ ∨ VALUE ≥ minv
VALUE ≤ maxv
required(VARIABLES, var)
required(FOUND, var)
where
maxv = maxval(VARIABLES.var)
minv = minval(VARIABLES.var)
sv = |VARIABLES|

The constraint MAX_HEIGHT_STEADY(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern STEADY for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern STEADY is the subsequence which matches the regular expression =.

Assume that the occurrence of the pattern STEADY starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i to index j + 1.

**Example**

\[
(6, (1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 5, 7, 2, 6, 6), (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0))
\]

Figure 3.979 provides an example where the POS_MAX_HEIGHT_STEADY constraint holds.

**Typical**

|VARIABLES| > 1

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Figure 3.979: Illustrating the POS_MAX_HEIGHT_STEADY constraint of the Example slot
Automaton

Similar to the automaton of the MAX_HEIGHT_STEADY constraint but use the decoration table 2.33.
POS_MAX_HEIGHT_STEADY

1787
3.415 POS_MAX_HEIGHT_STEADY_SEQUENCE

**DESCRIPTION**

**Origin**
Based on constraint `MAX_HEIGHT_STEADY_SEQUENCE`.

**Constraint**
`POS_MAX_HEIGHT_STEADY_SEQUENCE(VALUE, VARIABLES, FOUND)`

**Arguments**
- `VALUE`: `dvar`
- `VARIABLES`: `collection(var−dvar)`
- `FOUND`: `collection(var−dvar)`

**Restrictions**
- `sv ≤ 1 ⇒ VALUE = −∞`
- `VALUE = −∞ ∨ VALUE ≥ minv`
- `VALUE ≤ maxv`

The feature `MIN` computes the minimum of the values from index `i` to index `j + 1`.

**Purpose**

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `STEADY_SEQUENCE` is the maximal subsequence which matches the regular expression `+`.

Assume that the occurrence of the pattern `STEADY_SEQUENCE` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i` to index `j + 1`.

**Example**

\[
\begin{pmatrix}
5, 3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1 \\
0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 3.980 provides an example where the `POS_MAX_HEIGHT_STEADY_SEQUENCE` constraint holds.

**Typical**

`|VARIABLES| > 1`

**Arg. properties**

- Functional dependency: `VALUE` determined by `VARIABLES`.
- Functional dependency: `FOUND` determined by `VARIABLES`. 
Figure 3.980: Illustrating the `POS_MAX_HEIGHT_STEADY_SEQUENCE` constraint of the **Example** slot
Automaton  Similar to the automaton of the \texttt{MAX\_HEIGHT\_STEADY\_SEQUENCE} constraint but use the decoration table \ref{2.33}. 
POS_MAX_HEIGHT_ SteadySEQUENCE

1791
3.416  **POS_MAX_MAX_BUMP_ON_DECREASING_SEQUENCE**

**Origin**
Based on constraint `MAX_MAX_BUMP_ON_DECREASING_SEQUENCE`.

**Constraint**
`POS_MAX_MAX_BUMP_ON_DECREASING_SEQUENCE(VALUE, VARIABLES, FOUND)`

**Arguments**
- `VALUE`: `dvar`
- `VARIABLES`: `collection(var−dvar)`
- `FOUND`: `collection(var−dvar)`

**Restrictions**
- `sv ≤ 5 ∨ rv ≤ 2 ⇒ VALUE = −∞`
- `VALUE = −∞ ∨ VALUE ≥ minv + 2`
- `VALUE ≤ maxv`
- `required(VARIABLES, var)`
- `required(FOUND, var)`
- Where
  - `maxv = maxval(VARIABLES.var)`
  - `minv = minval(VARIABLES.var)`
  - `sv = |VARIABLES|`
  - `rv = range(VARIABLES.var)`

The constraint `MAX_MAX_BUMP_ON_DECREASING_SEQUENCE(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `BUMP_ON_DECREASING_SEQUENCE` for which the feature value is `VALUE`

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `BUMP_ON_DECREASING_SEQUENCE` is the subsequence which matches the regular expression `>>>>`.

Assume that the occurrence of the pattern `BUMP_ON_DECREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `MAX` computes the maximum of the values from index `i + 2` to index `j`.

**Example**

\[
\begin{bmatrix}
6, (7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3), \\
0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
\end{bmatrix}
\]

Figure 3.981 provides an example where the `POS_MAX_MAX_BUMP_ON_DECREASING_SEQUENCE` constraint holds.

**Purpose**

An occurrence of the pattern `BUMP_ON_DECREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `MAX` computes the maximum of the values from index `i + 2` to index `j`.

**Typical**

- `|VARIABLES| > 5`
- `range(VARIABLES.var) > 2`
Figure 3.981: Illustrating the POS_MAX_MAX_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \textit{MAX\_MAX\_BUMP\_ON\_DECREASING\_SEQUENCE} constraint but use the decoration table \textit{2.33}.
### 3.417 POS\_MAX\_MAX\_DECREASING

**Description**

Based on constraint MAX\_MAX\_DECREASING.

**Constraint**

\[
\text{POS\_MAX\_MAX\_DECREASING(VALUE, VARIABLES, FOUND)}
\]

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var→dvar)
- **FOUND**: collection(var→dvar)

**Restrictions**

\[
\begin{align*}
\text{sv} \leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} = -\infty \lor \text{VALUE} \geq \text{minv} + 1 \\
\text{VALUE} \leq \text{maxv} \\
\text{required(VARIABLES, var)} \\
\text{required(FOUND, var)} \\
\text{where}
\end{align*}
\]

\[
\begin{align*}
\text{maxv} &= \text{maxval(VARIABLES.var)} \\
\text{minv} &= \text{minval(VARIABLES.var)} \\
\text{sv} &= |\text{VARIABLES}| \\
\text{rv} &= \text{range(VARIABLES.var)}
\end{align*}
\]

The constraint MAX\_MAX\_DECREASING(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern DECREASING for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern DECREASING is the subsequence which matches the regular expression `>`. Assume that the occurrence of the pattern DECREASING starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
\begin{pmatrix}
6, (3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4), \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 3.982 provides an example where the POS\_MAX\_MAX\_DECREASING (6, [3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

**Typical**

- |VARIABLES| $> 1$
- \(\text{range(VARIABLES.var)} > 1\)
Figure 3.982: Illustrating the POS\_MAX\_MAX\_DECREASING constraint of the Example slot

**Arg. properties**

- **Functional dependency**: VALUE determined by VARIABLES.
- **Functional dependency**: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the POS_MAX_MAX_DECREASING constraint but use the decoration table 2.33.
POs_MAX_MAX_DECREASING 1799
3.418   POS_MAX_MAX_DECREASING_SEQUENCE

Description

Based on constraint MAX_MAX_DECREASING_SEQUENCE.

Constraint

POS_MAX_MAX_DECREASING_SEQUENCE(VALUE, VARIABLES, FOUND)

Arguments

VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions

sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = −∞
VALUE = −∞ ∨ VALUE ≥ minv + 1
VALUE ≤ maxv
required(VARIABLES, var)
required(FOUND, var)

where

maxv =maxval(VARIABLES.var)
minv =minval(VARIABLES.var)
sv = |VARIABLES|
rv =range(VARIABLES.var)

Purpose

The constraint MAX_MAX_DECREASING_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern DECREASING_SEQUENCE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression > (>|=)* > | >.

Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i to index j + 1.

Example

Figure 3.983 provides an example where the POS_MAX_MAX_DECREASING_SEQUENCE (6,[3,4,2,2,5,6,6,4,4,3,1,1,4,6,4,4],[0,0,0,0,0,0,1,0,0,0,0,0,1,0,0]) constraint holds.

Typical

|VARIABLES| > 1
range(VARIABLES.var) > 1
Figure 3.983: Illustrating the POS_MAX_MAX_DECREASING_SEQUENCE constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX MAX DECREASING SEQUENCE constraint but use the decoration table 2.33.
POS_MAX_MAX_DECENDING_SEQUENCE

1803
### Description

**Origin**
Based on constraint \( \text{MAX\_MAX\_DIP\_ON\_INCREASING\_SEQUENCE} \).

**Constraint**

\[
\text{POS\_MAX\_MAX\_DIP\_ON\_INCREASING\_SEQUENCE}(\text{VALUE, VARIABLES, FOUND})
\]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} - \text{dvar}) \\
\text{FOUND} & : \text{collection}(\text{var} - \text{dvar})
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{sv} \leq 5 \land \text{rv} \leq 2 & \implies \text{VALUE} = -\infty \\
\text{VALUE} = -\infty \lor \text{VALUE} \geq \text{minv} + 2 \\
\text{VALUE} \leq \text{maxv} & \\
\text{required}(\text{VARIABLES, var}) \\
\text{required}(\text{FOUND, var})
\end{align*}
\]

where

\[
\begin{align*}
\text{sv} &= \text{maxv}(	ext{VARIABLES vara}) \\
\text{minv} &= \text{minval}(\text{VARIABLES vara}) \\
\text{sv} &= |\text{VARIABLES}| \\
\text{rv} &= \text{range}(\text{VARIABLES vara})
\end{align*}
\]

The constraint \( \text{MAX\_MAX\_DIP\_ON\_INCREASING\_SEQUENCE}(\text{VALUE, VARIABLES}) \) holds. In addition, \( \text{FOUND} \) is a collection of 0/1 variables where the value 1 indicates the position of the \text{found} letter in those occurrences of the pattern \( \text{DIP\_ON\_INCREASING\_SEQUENCE} \) for which the feature value is \( \text{VALUE} \).

The position of the \text{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

**Purpose**

An occurrence of the pattern \( \text{DIP\_ON\_INCREASING\_SEQUENCE} \) is the subsequence which matches the regular expression \( <><<< \).

Assume that the occurrence of the pattern \( \text{DIP\_ON\_INCREASING\_SEQUENCE} \) starts at position \( i \) and ends at position \( j \). The feature \( \text{MAX} \) computes the maximum of the values from index \( i + 2 \) to index \( j \).

**Example**

\[
\left( 6, (1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4), [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0] \right)
\]

Figure 3.984 provides an example where the \( \text{POS\_MAX\_MAX\_DIP\_ON\_INCREASING\_SEQUENCE} \) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 5 \\
\text{range}(\text{VARIABLES vara}) > 2
\]
Figure 3.984: Illustrating the POS_MAX_MAX_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MAX\_MAX\_DIP\_ON\_INCREASING\_SEQUENCE} constraint but use the decoration table 2.33.
3.420 POS_MAX_MAX_INCREASING

**DESCRIPTION**

**Origin**
Based on constraint **MAX_MAX_INCREASING**.

**Constraint**

\[
\text{POS_MAX_MAX_INCREASING}(\text{VALUE, VARIABLES, FOUND})
\]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} \rightarrow \text{dvar}) \\
\text{FOUND} & : \text{collection}(\text{var} \rightarrow \text{dvar})
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{sv} \leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} = -\infty \lor \text{VALUE} \geq \text{minv} + 1 \\
\text{VALUE} \leq \text{maxv} & \\
\text{required} & (\text{VARIABLES}, \text{var}) \\
\text{required} & (\text{FOUND}, \text{var}) \\
\text{where} & \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{minv} & = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

The constraint **MAX_MAX_INCREASING(VALUE, VARIABLES)** holds. In addition, **FOUND** is a collection of 0/1 variables where the value 1 indicates the position of the **found** letter in those occurrences of the pattern **INCREASING** for which the feature value is **VALUE**.

The position of the **found** letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern **INCREASING** is the subsequence which matches the regular expression `<`. Assume that the occurrence of the pattern **INCREASING** starts at position \(i\) and ends at position \(j\). The feature **MAX** computes the maximum of the values from index \(i\) to index \(j+1\).

**Purpose**

**Example**

\[
\begin{pmatrix}
6, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3), \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 3.985 provides an example where the **POS_MAX_MAX_INCREASING**
\(\{6, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0]\}\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
Figure 3.985: Illustrating the POS_MAX_MAX_INCREASING constraint of the **Example** slot

**Arg. properties**
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the `MAX_MAX_INCREASING` constraint but use the decoration table 2.33.
3.421 POS_MAX_MAX_INCREASING_SEQUENCE

**Description**

Based on constraint MAX_MAX_INCREASING_SEQUENCE.

**Constraint**

POS_MAX_MAX_INCREASING_SEQUENCE(VALUE, VARIABLES, FOUND)

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>dvar</td>
</tr>
<tr>
<td>VARIABLES</td>
<td>collection(var−dvar)</td>
</tr>
<tr>
<td>FOUND</td>
<td>collection(var−dvar)</td>
</tr>
</tbody>
</table>

**Restrictions**

\[
\begin{align*}
sv \leq 1 & \lor rv \leq 1 \Rightarrow VALUE = -\infty \\
VALUE = -\infty & \lor VALUE \geq minv + 1 \\
VALUE \leq maxv & \\
required(VARIABLES, var) & \\
required(FOUND, var) & \\
where & \\
maxv &= \text{maxval}(VARIABLES.var) \\
minv &= \text{minval}(VARIABLES.var) \\
sv &= |VARIABLES| \\
rv &= \text{range}(VARIABLES.var)
\end{align*}
\]

The constraint MAX_MAX_INCREASING_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern INCREASING_SEQUENCE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression \(<(<|=)^*<|<\).

Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
\begin{pmatrix}
6, 4, 3, 5, 2, 1, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3, \, \\
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 3.986 provides an example where the POS_MAX_MAX_INCREASING_SEQUENCE (6,[4,3,5,2,1,1,1,3,3,4,6,6,3,1,3,3],[0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0]) constraint holds.

**Purpose**

Typical

\[|VARIABLES| > 1\]

\[\text{range}(VARIABLES.var) > 1\]
**Figure 3.986:** Illustrating the POS_MAX_MAX_INCREASING_SEQUENCE constraint of the Example slot

**Arg. properties**
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \textit{MAX\_MAX\_INCREASING\_SEQUENCE} constraint but use the decoration table 2.33.
### 3.422 POS\_MAX\_MAX\_INFLEXION

**Origin**
Based on constraint `MAX\_MAX\_INFLEXION`.

**Constraint**
`POS\_MAX\_MAX\_INFLEXION(VALUE, VARIABLES, FOUND)`

**Arguments**
- `VALUE`: dvar
- `VARIABLES`: collection(var - dvar)
- `FOUND`: collection(var - dvar)

**Restrictions**
1. `sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = −∞`
2. `VALUE = −∞ ∨ VALUE ≥ minv`
3. `VALUE ≤ maxv` (where `maxv = maxval(VARIABLES.var)`)
4. `required(VARIABLES, var)`
5. `required(FOUND, var)`

where
- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

The constraint `MAX\_MAX\_INFLEXION(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `INFLEXION` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `INFLEXION` is the *maximal* subsequence which matches the regular expression `< (< | =)∗ | > (| =)∗ | >)∗ <`.

Assume that the occurrence of the pattern `INFLEXION` starts at position `i` and ends at position `j`. The feature `MAX` computes the maximum of the values from index `i + 1` to index `j`.

**Purpose**

**Example**

Figure 3.987 provides an example where the `POS\_MAX\_MAX\_INFLEXION` constraint holds.

**Typical**
- `|VARIABLES| > 2`
- `range(VARIABLES.var) > 1`
Figure 3.987: Illustrating the POS_MAX_MAX_INFLEXION constraint of the Example slot

**Arg. properties**

- **Functional dependency**: VALUE determined by VARIABLES.
- **Functional dependency**: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_INFLEXION constraint but use the decoration table 2.33.
**3.423  POS_MAX_MAX_PEAK**

**DESCRIPTION**

**Origin**

Based on constraint MAX_MAX_PEAK.

**Constraint**

POS_MAX_MAX_PEAK(VALUE, VARIABLES, FOUND)

**Arguments**

VALUE : dvar  
VARIABLES : collection(var−dvar)  
FOUND : collection(var−dvar)

**Restrictions**

sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = -\infty  
VALUE = -\infty \lor VALUE \geq minv + 1  
VALUE \leq maxv  
required(VARIABLES, var)  
required(FOUND, var)

where

maxv = maxval(VARIABLES.var)  
minv = minval(VARIABLES.var)  
sv = |VARIABLES|  
rv = range(VARIABLES.var)

**Purpose**

The constraint MAX_MAX_PEAK(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern PEAK for which the feature value is VALUE. The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression \(< (= | <)^* (> | \geq)\) >. Assume that the occurrence of the pattern PEAK starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i + 1\) to index \(j\).

**Example**

\[
\begin{pmatrix} 6, (7, 5, 1, 4, 5, 2, 3, 5, 6, 2, 3, 3, 1), \{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0\} \end{pmatrix}
\]

Figure 3.988 provides an example where the POS_MAX_MAX_PEAK (6, [7, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 1], [0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0]) constraint holds.

**Typical**

|VARIABLES| > 2  
range(VARIABLES.var) > 1

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Figure 3.988: Illustrating the POS_MAX_MAX_PEAK constraint of the Example slot
Automaton  

Similar to the automaton of the MAX_MAX_PEAK constraint but use the decoration table 2.33.
POS_MAX_MAX_PEAK

1823
3.424 POS_MAX_MAX_STRICTLY_DECREASING_SEQUENCE

DESCRIPTION

Based on constraint MAX_MAX_STRICTLY_DECREASING_SEQUENCE.

Constraint

POS_MAX_MAX_STRICTLY_DECREASING_SEQUENCE(VALUE, VARIABLES, FOUND)

Arguments

VALUE : dvar
VARIABLES : collection(var=dvar)
FOUND : collection(var=dvar)

Restrictions

\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = -\infty \]
\[ VALUE = -\infty \lor VALUE \geq minv + 1 \]
\[ VALUE \leq maxv \]
required(VARIABLES, var)
required(FOUND, var)
where
maxv = maxval(VARIABLES.var)
minv = minval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)

Purpose

The constraint MAX_MAX_STRICTLY_DECREASING_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern STRICTLY_DECREASING_SEQUENCE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern STRICTLY_DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression \( \geq^+ \).

Assume that the occurrence of the pattern STRICTLY_DECREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature MAX computes the maximum of the values from index \( i \) to index \( j + 1 \).

Example

\[
\begin{pmatrix}
6, (4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3), \\
(0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0)
\end{pmatrix}
\]

Figure 3.989 provides an example where the POS_MAX_MAX_STRICTLY_DECREASING_SEQUENCE (6, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3], [0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0]) constraint holds.

Typical

\[ |VARIABLES| > 1 \]
\[ range(VARIABLES.var) > 1 \]
Figure 3.989: Illustrating the POS_MAX_MAX_STATICLY_DECREASING_SEQUENCE constraint of the Example slot

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the `MAX_MAX_STRICTLY_DECREASING_SEQUENCE` constraint but use the decoration table 2.33.
### 3.425 POS_MAX_MAX STRICTLY_INCREASING_SEQUENCE

**Description**

Based on constraint `MAX_MAX STRICTLY_INCREASING_SEQUENCE`.

**Constraint**

\[ \text{POS_MAX_MAX\_STRICTLY\_INCREASING\_SEQUENCE}(\text{VALUE}, \text{VARIABLES}, \text{FOUND}) \]

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var−dvar)`
- **FOUND**: `collection(var−dvar)`

**Restrictions**

- \( sv \leq 1 \lor rv \leq 1 \Rightarrow \text{VALUE} = -\infty \)
- \( \text{VALUE} = -\infty \lor \text{VALUE} \geq \minv + 1 \)
- \( \text{VALUE} \leq \maxv \)
- \( \text{required}(\text{VARIABLES}, \text{var}) \)
- \( \text{required}(\text{FOUND}, \text{var}) \)

where
- \( \maxv = \maxval(\text{VARIABLES}.\text{var}) \)
- \( \minv = \minval(\text{VARIABLES}.\text{var}) \)
- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

**Purpose**

The constraint `MAX_MAX\_STRICTLY\_INCREASING\_SEQUENCE(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern `STRICTLY\_INCREASING\_SEQUENCE` for which the feature value is `VALUE`.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `STRICTLY\_INCREASING\_SEQUENCE` is the maximal subsequence which matches the regular expression `<+`.

Assume that the occurrence of the pattern `STRICTLY\_INCREASING\_SEQUENCE` starts at position \( i \) and ends at position \( j \). The feature `MAX` computes the maximum of the values from index \( i \) to index \( j + 1 \).

**Example**

\[
\begin{bmatrix}
6, (4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3),
(0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
\end{bmatrix}
\]

Figure 3.990 provides an example where the `POS_MAX_MAX\_STRICTLY\_INCREASING\_SEQUENCE (6, [4, 3, 5, 5, 2, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3], [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0])` constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 1\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)
Figure 3.990: Illustrating the POS_MAX_MAX STRICTLY_INCREASING_SEQUENCE constraint of the Example slot

**Arg. properties**

- **Functional dependency**: VALUE determined by VARIABLES.
- **Functional dependency**: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the `MAX_MAX_STRICTLY_INCREASING_SEQUENCE` constraint but use the decoration table 2.33.
3.426  POS_MAX_MAX_SUMMIT

DESCRIPTION  AUTOMATON

Origin
Based on constraint MAX_MAX_SUMMIT.

Constraint
POS_MAX_MAX_SUMMIT(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions
sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = −∞
VALUE = −∞ ∨ VALUE ≥ minv + 1
VALUE ≤ maxv
required(VARIABLES, var)
required(FOUND, var)
where
maxv = maxval(VARIABLES.var)
minv = minval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)

Purpose
The constraint MAX_MAX_SUMMIT(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern SUMMIT for which the feature value is VALUE. The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression (<= | <= | * <= | > | <= | > | * > ). Assume that the occurrence of the pattern SUMMIT starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i + 1 to index j.

Example
(5, 7, 1, 5, 4, 3, 4, 6, 2, 3, 4, 2, 3, 1).

Figure 3.991 provides an example where the POS_MAX_MAX_SUMMIT
(5, 7, 1, 5, 4, 3, 4, 6, 2, 3, 4, 2, 3, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) constraint holds.

Typical
|VARIABLES| > 2
range(VARIABLES.var) > 1
Figure 3.991: Illustrating the POS_MAX_MAX_SUMMIT constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MAX\_MAX\_SUMMIT} constraint but use the decoration table \texttt{2.33}. 
3.427  POS_MAX_MAX_ZIGZAG

**Origin**
Based on constraint MAX_MAX_ZIGZAG.

**Constraint**

\[
\text{POS_MAX_MAX_ZIGZAG}(\text{VALUE}, \text{VARIABLES}, \text{FOUND})
\]

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)
- **FOUND**: collection(var−dvar)

**Restrictions**

\[
\text{sv} \leq 3 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} = -\infty \lor \text{VALUE} \geq \text{minv} + 1 \\
\text{VALUE} \leq \text{maxv} \\
\text{required}(\text{VARIABLES}, \text{var}) \\
\text{required}(\text{FOUND}, \text{var})
\]

where

\[
\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{sv} = |\text{VARIABLES}| \\
\text{rv} = \text{range}(\text{VARIABLES}.\text{var})
\]

The constraint MAX_MAX_ZIGZAG(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern ZIGZAG for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression \((<>)^+ (< | <> | (<>)^+ (> | >>))\).

Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i + 1\) to index \(j\).

**Example**

\[
(7, [4, 1, 3, 1, 4, 6, 1, 5, 2, 7, 2, 3, 1, 6, 1], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
\]

Figure 3.992 provides an example where the POS_MAX_MAX_ZIGZAG constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 3 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Figure 3.992: Illustrating the POS_MAX_MAX_ZIGZAG constraint of the Example slot
Automaton

Similar to the automaton of the \texttt{MAX\_MAX\_ZIGZAG} constraint but use the decoration table \texttt{2.33}.
3.428  **POS_MAX_MIN_BUMP_ON_DECREASING_SEQUENCE**

**Origin**
Based on constraint `MAX_MIN_BUMP_ON_DECREASING_SEQUENCE`.

**Constraint**
`POS_MAX_MIN_BUMP_ON_DECREASING_SEQUENCE(VALUE, VARIABLES, FOUND)`

**Arguments**
- `VALUE`: `dvar`
- `VARIABLES`: `collection(var−dvar)`
- `FOUND`: `collection(var−dvar)`

**Restrictions**
- `sv ≤ 5 ∨ rv ≤ 2 ⇒ VALUE = −∞`
- `VALUE = −∞ ∨ VALUE ≥ minv`
- `VALUE ≤ maxv − 2`  
  required(VARIABLES, var)  
  required(FOUND, var)
  where
  - `maxv = maxval(VARIABLES.var)`
  - `minv = minval(VARIABLES.var)`
  - `sv = |VARIABLES|`
  - `rv = range(VARIABLES.var)`

The constraint `MAX_MIN_BUMP_ON_DECREASING_SEQUENCE(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `BUMP_ON_DECREASING_SEQUENCE` for which the feature value is `VALUE`. The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern `BUMP_ON_DECREASING_SEQUENCE` is the subsequence which matches the regular expression `>>><>>`. Assume that the occurrence of the pattern `BUMP_ON_DECREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i + 2` to index `j`.

**Example**

\[
\begin{pmatrix}
  5, (7, 6, 5, 6, 5, 4, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3), \\
  (0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
\end{pmatrix}
\]

Figure 3.993 provides an example where the `POS_MAX_MIN_BUMP_ON_DECREASING_SEQUENCE(5, [7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3], [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])` constraint holds.

**Purpose**

**Typical**
- `|VARIABLES| > 5`
- `range(VARIABLES.var) > 2`
Figure 3.993: Illustrating the POS_MAX_MIN_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MAX\_MIN\_BUMP\_ON\_DECREASING\_SEQUENCE} constraint but use the decoration table \texttt{2.33}.
POS_MAX_MIN_BUMP_ON_DECREASING_SEQUENCE 1843
3.429 POS_MAX_MIN_DECREEASING

**DESCRIPTION**

Based on constraint **MAX_MIN_DECREEASING**.

**Constraint**

POS_MAX_MIN_DECREEASING(VALUE, VARIABLES, FOUND)

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)
- **FOUND**: collection(var−dvar)

**Restrictions**

- \( sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = -\infty \)
- \( VALUE = -\infty \lor VALUE \geq minv \)
- \( VALUE \leq maxv - 1 \)
- required(VARIABLES, var)
- required(FOUND, var)

where

\[
\begin{align*}
maxv &= maxval(VARIABLES.var) \\
minv &= minval(VARIABLES.var) \\
sv &= |VARIABLES| \\
rsv &= range(VARIABLES.var)
\end{align*}
\]

The constraint **MAX_MIN_DECREEASING(VALUE, VARIABLES)** holds. In addition, **FOUND** is a collection of 0/1 variables where the value 1 indicates the position of the **found** letter in those occurrences of the pattern **DECREEASING** for which the feature value is **VALUE**.

The position of the **found** letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern **DECREEASING** is the subsequence which matches the regular expression >.

Assume that the occurrence of the pattern **DECREEASING** starts at position \( i \) and ends at position \( j \). The feature **MIN** computes the minimum of the values from index \( i \) to index \( j + 1 \).

**Example**

\[ (4, 3, 4, 2, 5, 6, 4, 4, 3, 1, 4, 6, 4, 4), (0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0) \]

Figure 3.994 provides an example where the **POS_MAX_MIN_DECREEASING** constraint holds.

**Typical**

- \(|VARIABLES| > 1\)
- \(range(VARIABLES.var) > 1\)
Figure 3.994: Illustrating the POS_MAX_MIN_DECREASING constraint of the Example slot.

**Arg. properties**
- **Functional dependency**: VALUE determined by VARIABLES.
- **Functional dependency**: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_MIN_DECREASING constraint but use the decoration table 2.33.
POS_MAX_MIN_DECREASING

1847
### 3.430 POS_MAX_MIN_DECREASING_SEQUENCE

**Origin**
Based on constraint MAX_MIN_DECREASING_SEQUENCE.

**Constraint**
POS_MAX_MIN_DECREASING_SEQUENCE(VALUE, VARIABLES, FOUND)

**Arguments**
- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)
- **FOUND**: collection(var−dvar)

**Restrictions**
- $sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = -\infty$
- $VALUE = -\infty \lor VALUE \geq minv$
- $VALUE \leq maxv - 1$
- required(VARIABLES, var)
- required(FOUND, var)

where
- $maxv = maxval(VARIABLES.var)$
- $minv = minval(VARIABLES.var)$
- $sv = |VARIABLES|$
- $rv = range(VARIABLES.var)$

The constraint MAX_MIN_DECREASING_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern DECREASING_SEQUENCE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression $>(>\mid =)^* > \mid >$.

Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i$ to index $j + 1$.

**Purpose**

**Example**

$$\begin{pmatrix} 4, 3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4 \end{pmatrix}, \begin{pmatrix} 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0 \end{pmatrix}$$

Figure 3.995 provides an example where the POS_MAX_MIN_DECREASING_SEQUENCE (4, [3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]) constraint holds.

**Typical**
- $|VARIABLES| > 1$
- $range(VARIABLES.var) > 1$
Figure 3.995: Illustrating the POS_MAX_MIN_DECREASING_SEQUENCE constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MAX\_MIN\_DECREASING\_SEQUENCE} constraint but use the decoration table 2.33.
POS_MAX_MIN_DECREASING_SEQUENCE

1851
3.431  POS_MAX_MIN_DIP_ON_INCREASING_SEQUENCE

**Origin**
Based on constraint \( \text{MAX_MIN_DIP_ON_INCREASING_SEQUENCE} \).

**Constraint**
\( \text{POS_MAX_MIN_DIP_ON_INCREASING_SEQUENCE} \)\((\text{VALUE, VARIABLES, FOUND})\)

**Arguments**
- **VALUE** : dvar
- **VARIABLES** : collection(var−dvar)
- **FOUND** : collection(var−dvar)

**Restrictions**
- \( sv \leq 5 \lor rv \leq 2 \Rightarrow \text{VALUE} = -\infty \)
- \( \text{VALUE} = -\infty \lor \text{VALUE} \geq \text{minv} \)
- \( \text{VALUE} \leq \text{maxv} - 2 \cdot 0 \)
- \( \text{required}(\text{VARIABLES},\text{var}) \)
- \( \text{required}(\text{FOUND},\text{var}) \)

where
- \( \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \)
- \( \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \)
- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

The constraint \( \text{MAX_MIN_DIP_ON_INCREASING_SEQUENCE}(\text{VALUE, VARIABLES}) \) holds. In addition, \( \text{FOUND} \) is a collection of 0/1 variables where the value 1 indicates the position of the \text{found} letter in those occurrences of the pattern \( \text{DIP_ON_INCREASING_SEQUENCE} \) for which the feature value is \( \text{VALUE} \).

The position of the \text{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \( \text{DIP_ON_INCREASING_SEQUENCE} \) is the subsequence which matches the regular expression \( <<><< > < < \).

Assume that the occurrence of the pattern \( \text{DIP_ON_INCREASING_SEQUENCE} \) starts at position \( i \) and ends at position \( j \). The feature \text{MIN} computes the minimum of the values from index \( i + 2 \) to index \( j \).

**Purpose**

**Example**

\[
(2, (1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4), \left(0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\right))
\]

Figure 3.996 provides an example where the \( \text{POS_MAX_MIN_DIP_ON_INCREASING_SEQUENCE} \)\((2, [1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4], [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])\) constraint holds.

**Typical**
- \( |\text{VARIABLES}| > 5 \)
- \( \text{range}(\text{VARIABLES}.\text{var}) > 2 \)
Figure 3.996: Illustrating the POS_MAX_MIN_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot

Arg. properties
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_MIN_DIP_ON_INCREASING_SEQUENCE constraint but use the decoration table 2.33.
3.432 POS_MAX_MINGORGE

**DESCRIPTION**

**Origin**
Based on constraint MAX_MINGORGE.

**Constraint**
POS_MAX_MINGORGE(VALUE, VARIABLES, FOUND)

**Arguments**
VALUE : dvar  
VARIABLES : collection(var−dvar)  
FOUND : collection(var−dvar)

**Restrictions**
sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = −∞  
VALUE = −∞ ∨ VALUE ≥ minv  
VALUE ≤ maxv − 1  
required(VARIABLES, var)  
required(FOUND, var)
where
maxv = maxval(VARIABLES.var)  
minv = minval(VARIABLES.var)  
sv = |VARIABLES|  
rv = range(VARIABLES.var)

**Purpose**
The constraint MAX_MINGORGE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern GORGE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression (> | > (= | >)* >)(< | < (= | <)* <).

Assume that the occurrence of the pattern GORGE starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i + 1 to index j.

**Example**

\[
(5, \{1, 7, 3, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7\}, \{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0\})
\]

Figure 3.997 provides an example where the POS_MAX_MINGORGE (5, [1, 7, 3, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]) constraint holds.

**Typical**
|VARIABLES| > 2  
range(VARIABLES.var) > 1

**Arg. properties**
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Figure 3.997: Illustrating the POS_MAX_MIN_GORGE constraint of the Example slot
Automaton

Similar to the automaton of the MAX_MIN_GORGE constraint but use the decoration table 2.33.
PO5_MAX_MIN_GORGE

1859
### 3.433 POS_MAX_MIN_INCREASING

**Description**

Based on constraint `MAX_MIN_INCREASING`.

**Constraint**

\[
\text{POS\_MAX\_MIN\_INCREASING}(\text{VALUE, VARIABLES, FOUND})
\]

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>dvar</td>
</tr>
<tr>
<td>VARIABLES</td>
<td>collection(var−dvar)</td>
</tr>
<tr>
<td>FOUND</td>
<td>collection(var−dvar)</td>
</tr>
</tbody>
</table>

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} & = -\infty \lor \text{VALUE} \geq \text{minv} \\
\text{VALUE} & \leq \text{maxv} - 1 \Rightarrow \\
\text{required}(\text{VARIABLES, var}) & \\
\text{required}(\text{FOUND, var}) & \\
\text{where} & \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}\text{.var}) \\
\text{minv} & = \text{minval}(\text{VARIABLES}\text{.var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES}\text{.var})
\end{align*}
\]

The constraint `MAX_MIN_INCREASING(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `INCREASING` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `INCREASING` is the subsequence which matches the regular expression `<`. Assume that the occurrence of the pattern `INCREASING` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i` to index `j + 1`.

**Example**

\[
\left[4, 4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3\right], \\
\left[0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0\right]
\]

Figure 3.998 provides an example where the `POS_MAX_MIN_INCREASING (4, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0])` constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}\text{.var}) > 1
\]
Figure 3.998: Illustrating the POS_MAX_MIN_INCREASING constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_MIN_INCREASING constraint but use the decoration table 2.33.
3.434 POS_MAX_MIN_INCREASING_SEQUENCE

**DESCRIPTION**

**Origin**
Based on constraint `MAX_MIN_INCREASING_SEQUENCE`.

**Constraint**
`POS_MAX_MIN_INCREASING_SEQUENCE(VALUE, VARIABLES, FOUND)`

**Arguments**
- `VALUE`: `dvar`
- `VARIABLES`: `collection(var−dvar)`
- `FOUND`: `collection(var−dvar)`

**Restrictions**
- `sv ≤ 1 ∨ rv ≤ 1 → VALUE = −∞`
- `VALUE = −∞ ∨ VALUE ≥ minv`
- `VALUE ≤ maxv − 1`
- `required(VARIABLES, var)`
- `required(FOUND, var)`

where
- `maxv = maxval(VARIABLES.var)`
- `minv = minval(VARIABLES.var)`
- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

The constraint `MAX_MIN_INCREASING_SEQUENCE(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `INCREASING_SEQUENCE` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern `INCREASING_SEQUENCE` is the maximal subsequence which matches the regular expression `< (< | =)* < |`.

Assume that the occurrence of the pattern `INCREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i` to index `j + 1`.

**Example**

```
(3, 4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3, 3),
(0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
```

Figure 3.999 provides an example where the `POS_MAX_MIN_INCREASING_SEQUENCE`

\( (3, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3], [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) \) constraint holds.

**Purpose**

**Typical**
- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`
Figure 3.999: Illustrating the POS_MAX_MIN_INCREASING_SEQUENCE constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_MIN_INCREASING_SEQUENCE constraint but use the decoration table 2.33.
POS_MAX_MIN, INCREASING_SEQUENCE

1867
3.435  POS_MAX_MIN_INFLEXION

**DESCRIPTION**

Based on constraint MAX_MIN_INFLEXION.

**Automaton**

\[< (< | =) > | > (>| =) > <\]

**Origin**

Based on constraint MAX_MIN_INFLEXION.

**Constraint**

POS_MAX_MIN_INFLEXION(VALUE, VARIABLES, FOUND)

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)
- **FOUND**: collection(var−dvar)

**Restrictions**

\[sv \leq 2 \vee rv \leq 1 \Rightarrow VALUE = -\infty\]
\[VALUE = -\infty \vee VALUE \geq minv\]
\[VALUE \leq maxv\]
\[required(VARIABLES, var)\]
\[required(FOUND, var)\]

where

\[maxv = maxval(VARIABLES, var)\]
\[minv = minval(VARIABLES, var)\]
\[sv = |VARIABLES|\]
\[rv = range(VARIABLES, var)\]

The constraint MAX_MIN_INFLEXION(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern INFLEXION for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression \(< (< | =) > | > (>| =) > <\).

Assume that the occurrence of the pattern INFLEXION starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**

\[
\begin{pmatrix}
5, 1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4 \\
0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 3.1000 provides an example where the POS_MAX_MIN_INFLEXION (5, [1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4], [0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0]) constraint holds.

**Purpose**

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

**Typical**

\[|VARIABLES| > 2\]
\[range(VARIABLES, var) > 1\]
Figure 3.1000: Illustrating the POS_MAX_MIN_INFLEXION constraint of the **Example** slot

**Arg. properties**

- **Functional dependency**: VALUE determined by VARIABLES.
- **Functional dependency**: FOUND determined by VARIABLES.
Automaton: Similar to the automaton of the MAX_MIN_INFLEXION constraint but use the decoration table 2.33.
| POS_MAX_MIN_INFLEXION | 1871 |


### 3.436 POS_MAX_MIN斯特林严格递减序列

**Origin**
Based on constraint `MAX_MIN斯特林严格递减序列`.

**Constraint**
POS_MAX_MIN斯特林严格递减序列\((\text{VALUE}, \text{VARIABLES}, \text{FOUND})\)

**Arguments**
- **VALUE**: `dvar`
- **VARIABLES**: `collection(var-dvar)`
- **FOUND**: `collection(var-dvar)`

**Restrictions**
- `sv \leq 1 \lor rv \leq 1 \Rightarrow \text{VALUE} = -\infty`
- `\text{VALUE} = -\infty \lor \text{VALUE} \geq \text{minv}`
- `\text{VALUE} \leq \text{maxv} - 1` if `\text{required(VARIABLES, var)}` and `\text{required(FOUND, var)}`

where
- `\text{maxv} = \text{maxval(VARIABLES.var)}`
- `\text{minv} = \text{minval(VARIABLES.var)}`
- `sv = |\text{VARIABLES}|`
- `rv = \text{range(VARIABLES.var)}`

The constraint `MAX_MIN斯特林严格递减序列(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `斯特林严格递减序列` for which the feature value is `VALUE`. The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern `斯特林严格递减序列` is the maximal subsequence which matches the regular expression `>^+`. Assume that the occurrence of the pattern `斯特林严格递减序列` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i` to index `j + 1`.

**Example**

\[
\begin{pmatrix}
3, (4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3), \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0
\end{pmatrix}
\]

Figure 3.1001 provides an example where the `POS_MAX_MIN斯特林严格递减序列` constraint holds.

**Purpose**

**Typical**
- `|\text{VARIABLES}| > 1`
- `\text{range(VARIABLES.var)} > 1`
Figure 3.1001: Illustrating the **POS_MAX_MIN STRICTLY DECREASING SEQUENCE** constraint of the **Example** slot

**Arg. properties**
- **Functional dependency**: VALUE determined by **VARIABLES**.
- **Functional dependency**: FOUND determined by **VARIABLES**.
Automaton

Similar to the automaton of the MAX_MIN_STRICTLY_DECREASING_SEQUENCE constraint but use the decoration table 2.33.
3.437  **POS_MAX_MIN STRICTLY_INCREASING_SEQUENCE**

**DESCRIPTION**

**Constraint**

Based on constraint **MAX_MIN STRICTLY_INCREASING_SEQUENCE.**

**Arguments**

| VALUE   | : dvar          |
| VARIABLES | : collection(var−dvar) |
| FOUND   | : collection(var−dvar) |

**Restrictions**

\[
sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = -\infty \\
VALUE = -\infty \lor VALUE \geq minv \\
VALUE \leq maxv - 1 \Rightarrow \text{required(VARIABLES, var)} \\
\text{required(FOUND, var)} \\
\text{where} \\
maxv = \text{maxval(VARIABLES.var)} \\
minv = \text{minval(VARIABLES.var)} \\
sv = |\text{VARIABLES}| \\
rv = \text{range}(\text{VARIABLES.var})
\]

The constraint **MAX_MIN STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES)** holds. In addition, **FOUND** is a collection of 0/1 variables where the value 1 indicates the position of the **found** letter in those occurrences of the pattern **STRICTLY_INCREASING_SEQUENCE** for which the feature value is **VALUE**.

The position of the **found** letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern **STRICTLY_INCREASING_SEQUENCE** is the **maximal** subsequence which matches the regular expression **<+**.

Assume that the occurrence of the pattern **STRICTLY_INCREASING_SEQUENCE** starts at position **i** and ends at position **j**. The feature **MIN** computes the minimum of the values from index **i** to index **j + 1**.

**Example**

\[
(3, 4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3), \\
(0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
\]

Figure 3.1002 provides an example where the **POS_MAX_MIN STRICTLY_INCREASING_SEQUENCE** \((3, 4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3), [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] \) constraint holds.

**Purpose**

**Typical**

\[|\text{VARIABLES}| > 1 \]

\[\text{range(VARIABLES.var)} > 1 \]
Figure 3.1002: Illustrating the POS_MAX_MIN STRICTLY_INCREASING_SEQUENCE constraint of the Example slot

**Arg. properties**
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_MIN STRICTLY INCREASING SEQUENCE constraint but use the decoration table 2.33.
POS_MAX_MIN_STRICTLY_INCREASING_SEQUENCE

1879
### 3.438 POS_MAX_MIN_VALLEY

**Origin**
Based on constraint `MAX_MIN_VALLEY`.

**Constraint**
`POS_MAX_MIN_VALLEY(VALUE, VARIABLES, FOUND)`

**Arguments**
- **VALUE**: `dvar`
- **VARIABLES**: `collection(var−dvar)`
- **FOUND**: `collection(var−dvar)`

**Restrictions**
- `sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = −∞`
- `VALUE = −∞ ∨ VALUE ≥ minv`
- `VALUE ≤ maxv − 1 ⋄`
- `required(VARIABLES, var)`
- `required(FOUND, var)`

where
- `maxv = maxval(VARIABLES.var)`
- `minv = minval(VARIABLES.var)`
- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

The constraint `POS_MAX_MIN_VALLEY(VALUE, VARIABLES, FOUND)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `VALLEY` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `VALLEY` is the maximal subsequence which matches the regular expression `>(=|>)*(<|=)∗<`. Assume that the occurrence of the pattern `VALLEY` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i+1` to index `j`.

**Purpose**

**Example**

```
(5, ⟨1, 3, 7, 4, 3, 6, 5, 3, 3, 2, 6, 5, 5, 7⟩, ⟨0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0⟩)
```

Figure 3.1003 provides an example where the `POS_MAX_MIN_VALLEY (5, [1, 3, 7, 4, 3, 6, 5, 3, 3, 2, 6, 5, 5, 7], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0])` constraint holds.

**Typical**
- `|VARIABLES| > 2`
- `range(VARIABLES.var) > 1`

**Arg. properties**
- Functional dependency: `VALUE` determined by `VARIABLES`.
- Functional dependency: `FOUND` determined by `VARIABLES`.

---

Origin Based on constraint `MAX_MIN_VALLEY`.

Constraint `POS_MAX_MIN_VALLEY(VALUE, VARIABLES, FOUND)`

Arguments
- **VALUE**: `dvar`
- **VARIABLES**: `collection(var−dvar)`
- **FOUND**: `collection(var−dvar)`

Restrictions
- `sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = −∞`
- `VALUE = −∞ ∨ VALUE ≥ minv`
- `VALUE ≤ maxv − 1 ⋄`
- `required(VARIABLES, var)`
- `required(FOUND, var)`

where
- `maxv = maxval(VARIABLES.var)`
- `minv = minval(VARIABLES.var)`
- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

The constraint `POS_MAX_MIN_VALLEY(VALUE, VARIABLES, FOUND)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `VALLEY` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `VALLEY` is the maximal subsequence which matches the regular expression `>(=|>)*(<|=)∗<`. Assume that the occurrence of the pattern `VALLEY` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i+1` to index `j`.

Purpose

Example

```
(5, ⟨1, 3, 7, 4, 3, 6, 5, 3, 3, 2, 6, 5, 5, 7⟩, ⟨0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0⟩)
```

Figure 3.1003 provides an example where the `POS_MAX_MIN_VALLEY (5, [1, 3, 7, 4, 3, 6, 5, 3, 3, 2, 6, 5, 5, 7], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0])` constraint holds.

Typical
- `|VARIABLES| > 2`
- `range(VARIABLES.var) > 1`

Arg. properties
- Functional dependency: `VALUE` determined by `VARIABLES`.
- Functional dependency: `FOUND` determined by `VARIABLES`. 
Figure 3.1003: Illustrating the POS_MAX_MIN_VALLEY constraint of the Example slot
Automaton  Similar to the automaton of the \texttt{MAX\_MIN\_VALLEY} constraint but use the decoration table \ref{table:2.33}.
3.439  

**POS_MAX_MIN_ZIGZAG**

**DESCRIPTION**

Based on constraint **MAX_MIN_ZIGZAG**.

**AUTOMATON**

\[(<=)(< | <>)(<>)(< | <>)(><)(> | <>)]

**Origin**

Based on constraint **MAX_MIN_ZIGZAG**.

**Constraint**

**POS_MAX_MIN_ZIGZAG(VALUE, VARIABLES, FOUND)**

**Arguments**

- **VALUE** : dvar
- **VARIABLES** : collection(var−dvar)
- **FOUND** : collection(var−dvar)

**Restrictions**

- \(sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = -\infty\)
- \(VALUE = -\infty \lor VALUE \geq \min v\)
- \(VALUE \leq \max v - 1\)
- \(\text{required}(\text{VARIABLES}, \text{var})\)
- \(\text{required}(\text{FOUND}, \text{var})\)

where

- \(\max v = \text{maxval}(\text{VARIABLES}.\text{var})\)
- \(\min v = \text{minval}(\text{VARIABLES}.\text{var})\)
- \(sv = |\text{VARIABLES}|\)
- \(rv = \text{range}(\text{VARIABLES}.\text{var})\)

**Purpose**

The constraint **MAX_MIN_ZIGZAG(VALUE, VARIABLES)** holds. In addition, **FOUND** is a collection of 0/1 variables where the value 1 indicates the position of the **found** letter in those occurrences of the pattern **ZIGZAG** for which the feature value is **VALUE**.

The position of the **found** letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern **ZIGZAG** is the maximal subsequence which matches the regular expression \((<>)^{+}(<>)(< | <>)(<>)(< | <>)(><)(> | <>)).\)

Assume that the occurrence of the pattern **ZIGZAG** starts at position \(i\) and ends at position \(j\). The feature **MIN** computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**

\[\left(1, \{4, 1, 3, 1, 4, 6, 1, 5, 2, 7, 2, 3, 1, 6, 1\}, \{0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0\}\right)\]

Figure 3.1004 provides an example where the **POS_MAX_MIN_ZIGZAG** constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 3\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Arg. properties**

- Functional dependency: **VALUE** determined by **VARIABLES**.
- Functional dependency: **FOUND** determined by **VARIABLES**.
Figure 3.1004: Illustrating the POS_MAX_MIN_ZIGZAG constraint of the Example slot
Automaton

Similar to the automaton of the MAX_MIN_ZIGZAG constraint but use the decoration table 2.33.
3.440 POS_MAX_SURF_BUMP_ON_DECREASING_SEQUENCE

DESCRIPTION

Origin
Based on constraint MAX_SURF_BUMP_ON_DECREASING_SEQUENCE.

Constraint
POS_MAX_SURF_BUMP_ON_DECREASING_SEQUENCE(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions
\[ sv \leq 5 \lor rv \leq 2 \Rightarrow VALUE = -\infty \]
\[ VALUE = -\infty \lor VALUE \geq 3 \times \text{minv} + 3 \]
\[ VALUE \leq 3 \times \text{maxv} - 3 \]
\[
\text{required}(\text{VARIABLES}, \text{var})
\]
\[
\text{required}(\text{FOUND}, \text{var})
\]

where
\[ \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \]
\[ \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \]
\[ sv = |\text{VARIABLES}| \]
\[ rv = \text{range}(\text{VARIABLES}.\text{var}) \]

The constraint MAX_SURF_BUMP_ON_DECREASING_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern BUMP_ON_DECREASING_SEQUENCE for which the feature value is VALUE. The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence which matches the regular expression >>><>>.

Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position i and ends at position j. The feature SURF computes the sum of the values from index \(i + 2\) to index \(j\).

Example

\[
\begin{pmatrix}
16, \langle 7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3 \rangle, \\
\{0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 \}
\end{pmatrix}
\]

Figure 3.1005 provides an example where the POS_MAX_SURF_BUMP_ON_DECREASING_SEQUENCE (16, [7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3], [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

Purpose

Typical

\[ |\text{VARIABLES}| > 5 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 2 \]
Figure 3.1005: Illustrating the `POS_MAX_SURF_BUMP_ON_DECREASING_SEQUENCE` constraint of the Example slot

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton Similar to the automaton of the \texttt{MAX_SURF_BUMP_ON_DECREASING_SEQUENCE} constraint but use the decoration table 2.33.
3.441 **POS_MAX_SURF_DECREEASING**

**Description**

Based on constraint `MAX_SURF_DECREEASING`.

**Constraint**

`POS_MAX_SURF_DECREEASING(VALUE, VARIABLES, FOUND)`

**Arguments**

- `VALUE`: `dvar`
- `VARIABLES`: `collection(var-dvar)`
- `FOUND`: `collection(var-dvar)`

**Restrictions**

\[
\begin{align*}
sv \leq 1 \lor rv \leq 1 & \Rightarrow VALUE = -\infty \\
VALUE = -\infty \lor VALUE \geq 2 \times \text{minv} + 1 & \\
VALUE \leq 2 \times \text{maxv} - 1 & \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{required}(\text{FOUND}, \text{var}) & \\
\text{where} & \\
\text{maxv} &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{minv} &= \text{minval}(\text{VARIABLES}.\text{var}) \\
sv &= |\text{VARIABLES}| \\
rv &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

The constraint `MAX_SURF_DECREEASING(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `DECREEASING` for which the feature value is `VALUE`.

**Purpose**

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `DECREEASING` is the subsequence which matches the regular expression `>`.

Assume that the occurrence of the pattern `DECREEASING` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i` to index `j + 1`.

**Example**

\[
\begin{pmatrix}
10, \langle 3, 4, 2, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4 \rangle, \\
\langle 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0 \rangle
\end{pmatrix}
\]

Figure 3.1006 provides an example where the `POS_MAX_SURF_DECREEASING` constraint holds.

**Typical**

- `|\text{VARIABLES}| > 1`
- `\text{range}(\text{VARIABLES}.\text{var}) > 1`
Figure 3.1006: Illustrating the POS_MAX_SURF_DECREASING constraint of the Example slot

Arg. properties

- **Functional dependency**: VALUE determined by VARIABLES.
- **Functional dependency**: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the `MAX_SURF_DECREASING` constraint but use the decoration table 2.33.
POS_MAX_SURF_DECREASING 1895
3.442 POS_MAX_SURF_DECREASING_SEQUENCE

**DESCRIPTION**

Based on constraint **MAX_SURF_DECREASING_SEQUENCE**.

**Arguments**

\[
\text{VALUE} : \ d\text{var} \\
\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar}) \\
\text{FOUND} : \text{collection}(\text{var} - \text{dvar})
\]

**Restrictions**

\[
\text{sv} \leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{rv} = 2 \Rightarrow \text{VALUE} = -\infty \lor \text{VALUE} \geq 2 \times \text{minv} + 1 \\
\text{rv} \geq 3 \Rightarrow \text{VALUE} = -\infty \lor \text{VALUE} \geq \min(2 \times \text{minv} + 1, \text{sv} \times (\text{minv} + 1)) \\
\text{rv} = 2 \Rightarrow \text{VALUE} \leq 2 \times \text{maxv} - 1 \\
\text{rv} \geq 3 \Rightarrow \text{VALUE} \leq \max(2 \times \text{maxv} - 1, \text{sv} \times (\text{maxv} - 1)) \\
\text{required} \text{(VARIABLES, var)} \\
\text{required} \text{(FOUND, var)}
\]

where

\[
\text{maxv} = \max\text{val}(\text{VARIABLES}.\text{var}) \\
\text{rv} = \text{range}(\text{VARIABLES}.\text{var}) \\
\text{sv} = |\text{VARIABLES}| \\
\text{minv} = \min\text{val}(\text{VARIABLES}.\text{var})
\]

The constraint **MAX_SURF_DECREASING_SEQUENCE(VALUE, VARIABLES)** holds. In addition, **FOUND** is a collection of 0/1 variables where the value 1 indicates the position of the **found** letter in those occurrences of the pattern **DECREASING_SEQUENCE** for which the feature value is **VALUE**.

The position of the **found** letter in an occurrence of a pattern is the first position where the occurrence of the pattern is identified, even if the pattern is not complete.

An occurrence of the pattern **DECREASING_SEQUENCE** is the **maximal** subsequence which matches the regular expression \( (> | =)* > | > \).

Assume that the occurrence of the pattern **DECREASING_SEQUENCE** starts at position \( i \) and ends at position \( j \). The feature **SURF** computes the sum of the values from index \( i \) to index \( j + 1 \).

**Example**

\[
\begin{pmatrix}
18, \langle 3, 4, 2, 5, 6, 4, 3, 1, 4, 6, 4, 4 \rangle, \\
\langle 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0 \rangle
\end{pmatrix}
\]

Figure 3.1007 provides an example where the **POS_MAX_SURF_DECREASING_SEQUENCE** (18, [3, 4, 2, 5, 6, 4, 3, 1, 4, 6, 4, 4], [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0]) constraint holds.
Figure 3.1007: Illustrating the POS_MAX_SURF_DECREASING_SEQUENCE constraint of the Example slot

**Typical**

| VARIABLES | > 1  
| range(VARIABLES.var) | > 1 |

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_SURF_DECREASING_SEQUENCE constraint but use the decoration table 2.33.
3.443 POS_MAX_SURF_DECREASING_TERRACE

**DESCRIPTION**

**Origin**
Based on constraint `MAX_SURF_DECREASING_TERRACE`.

**Constraint**

`POS_MAX_SURF_DECREASING_TERRACE(VALUE, VARIABLES, FOUND)`

**Arguments**

- `VALUE`: `dvar`
- `VARIABLES`: `collection(var−dvar)`
- `FOUND`: `collection(var−dvar)`

**Restrictions**

\[
\text{sv} \leq 3 \lor \text{rv} \leq 2 \Rightarrow \text{VALUE} = -\infty
\]

\[
\text{VALUE} = -\infty \lor \text{VALUE} \geq \min(2 + (\text{minv} + 1), (\text{sv} - 2) \ast (\text{minv} + 1))
\]

\[
\text{VALUE} \leq \max(2 + (\text{maxv} - 1) \ast (\text{sv} - 2) \ast (\text{maxv} - 1) \ast (\text{found} - \text{var}))
\]

where

\[
\text{maxv} = \maxval(VARIABLES.\text{var})
\]

\[
\text{sv} = |VARIABLES|
\]

\[
\text{minv} = \minval(VARIABLES.\text{var})
\]

\[
\text{rv} = \range(VARIABLES.\text{var})
\]

The constraint `MAX_SURF_DECREASING_TERRACE(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `DECREASING_TERRACE` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `DECREASING_TERRACE` is the `maximal` subsequence which matches the regular expression `>=+>`.

Assume that the occurrence of the pattern `DECREASING_TERRACE` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i+1` to index `j`.

**Example**

\[
\begin{pmatrix}
8, (6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3), \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0
\end{pmatrix}
\]

Figure 3.1008 provides an example where the `POS_MAX_SURF_DECREASING_TERRACE` constraint holds.

**Purpose**

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `DECREASING_TERRACE` is the `maximal` subsequence which matches the regular expression `>=+>`.

Assume that the occurrence of the pattern `DECREASING_TERRACE` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i+1` to index `j`.

**Typical**

\[
|VARIABLES| > 3
\]

\[
\text{range}(\text{VARIABLES.\text{var}}) > 2
\]
Figure 3.1008: Illustrating the POS_MAX_SURF_DECREASING_TERRACE constraint of the Example slot

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the $\text{MAX\_SURF\_DECREASING\_TERRACE}$ constraint but use the decoration table 2.33.
POS_MAX_SURF_DECREASING_TERRACE 1903
### 3.444 POS\_MAX\_SURF\_DIP\_ON\_INCREASING\_SEQUENCE

**Description**

Based on constraint `MAX\_SURF\_DIP\_ON\_INCREASING\_SEQUENCE`.

**Constraint**

\[
\text{POS\_MAX\_SURF\_DIP\_ON\_INCREASING\_SEQUENCE}(\text{VALUE, VARIABLES, FOUND})
\]

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var=dvar)`
- **FOUND**: `collection(var=dvar)`

**Restrictions**

\[
\begin{align*}
sv & \leq 5 \lor rv \leq 2 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} & = -\infty \lor \text{VALUE} \geq 3 \times \text{minv} + 3 \\
\text{VALUE} & \leq 3 \times \text{maxv} - 3 \\
\text{required}(\text{VARIABLES, var}) & \\
\text{required}(\text{FOUND, var}) & \\
\text{where} & \\
\text{maxv} & = \maxval(\text{VARIABLES}.\text{var}) \\
\text{minv} & = \minval(\text{VARIABLES}.\text{var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

The constraint `MAX\_SURF\_DIP\_ON\_INCREASING\_SEQUENCE(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `DIP\_ON\_INCREASING\_SEQUENCE` for which the feature value is `VALUE`. The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

**Purpose**

An occurrence of the pattern `DIP\_ON\_INCREASING\_SEQUENCE` is the subsequence which matches the regular expression `<<<`.

Assume that the occurrence of the pattern `DIP\_ON\_INCREASING\_SEQUENCE` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i + 2` to index `j`.

**Example**

\[
\begin{pmatrix}
10, (1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4), \\
0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 3.1009 provides an example where the `POS\_MAX\_SURF\_DIP\_ON\_INCREASING\_SEQUENCE` constraint holds.

**Typical**

- `|\text{VARIABLES}| > 5`
- `\text{range}(\text{VARIABLES}.\text{var}) > 2`
Figure 3.1009: Illustrating the POS_MAX_SURF_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MAX\_SURF\_DIP\_ON\_INCREASING\_SEQUENCE} constraint but use the decoration table 2.33.
3.445  POS_MAX_SURF_GORGE

**Description**

Based on constraint `MAX_SURF_GORGE`.

**Constraint**

`POS_MAX_SURF_GORGE(VALUE, VARIABLES, FOUND)`

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var=dvar)`
- **FOUND**: `collection(var=dvar)`

**Restrictions**

\[
sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = -\infty \\
rv = 2 \Rightarrow VALUE = -\infty \lor VALUE \geq minv \\
rv \geq 3 \Rightarrow VALUE = -\infty \lor VALUE \geq \min(v(sv - 2 + (minv + 1) - 1)) \\
rv = 2 \Rightarrow VALUE \leq maxv - 1 \\
rv \geq 3 \Rightarrow VALUE \leq \max(maxv - 1, sv - 2 + (maxv - 1) - 1) \\
required(VARIABLES, var) \\
required(FOUND, var)
\]

where

- `maxv = \maxval(VARIABLES.var)`
- `rv = \range(VARIABLES.var)`
- `sv = |VARIABLES|`
- `minv = \minval(VARIABLES.var)`

The constraint `MAX_SURF_GORGE(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `GORGE` for which the feature value is `VALUE`.

**Purpose**

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `GORGE` is the maximal subsequence which matches the regular expression ` (> | > (= | >)* >)(< | < (= | <)* <)`.

Assume that the occurrence of the pattern `GORGE` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i + 1` to index `j`.

**Example**

\[
(11, (1, 7, 3, 4, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7), [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
\]

Figure 3.1010 provides an example where the `POS_MAX_SURF_GORGE (11, [1, 7, 3, 4, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7], [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])` constraint holds.

**Typical**

- `|VARIABLES| > 2`
- `\range(VARIABLES.var) > 1`
Figure 3.1010: Illustrating the POS_MAX_SURF_GORGE constraint of the **Example** slot

**Arg. properties**

- **Functional dependency:** VALUE determined by VARIABLES.
- **Functional dependency:** FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_SURF_GORGE constraint but use the decoration table 2.33.
**3.46  ** POS\_MAX\_SURF\_INCREASING

**DESCRIPTION  ** AUTOMATON

**Origin**  
Based on constraint MAX\_SURF\_INCREASING.

**Constraint**  
POS\_MAX\_SURF\_INCREASING(VALUE, VARIABLES, FOUND)

**Arguments**  
VALUE : dvar  
VARIABLES : collection(var-dvar)  
FOUND : collection(var-dvar)

**Restrictions**  
\[ \text{sv} \leq 1 \vee \text{rv} \leq 1 \Rightarrow \text{VALUE} = -\infty \]  
\[ \text{VALUE} = -\infty \vee \text{VALUE} \geq 2 \times \text{minv} + 1 \]  
\[ \text{VALUE} \leq 2 \times \text{maxv} - 1 \times \text{required} (\text{VARIABLES}, \text{var}) \]  
\[ \text{required} (\text{FOUND}, \text{var}) \]  
where  
\[ \text{maxv} = \text{maxval} (\text{VARIABLES}.\text{var}) \]  
\[ \text{minv} = \text{minval} (\text{VARIABLES}.\text{var}) \]  
\[ \text{sv} = |\text{VARIABLES}| \]  
\[ \text{rv} = \text{range} (\text{VARIABLES}.\text{var}) \]

**Purpose**  
The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.  
An occurrence of the pattern INCREASING is the subsequence which matches the regular expression <.  
Assume that the occurrence of the pattern INCREASING starts at position i and ends at position j. The feature SURF computes the sum of the values from index i to index j + 1.

**Example**  
\[ (10, \{4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3\}, \{0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0\}) \]

Figure 3.1011 provides an example where the POS\_MAX\_SURF\_INCREASING (10, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3], [0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0]) constraint holds.

**Typical**  
\[ |\text{VARIABLES}| > 1 \]  
\[ \text{range} (\text{VARIABLES}.\text{var}) > 1 \]
Figure 3.1011: Illustrating the **POS_MAX_SURF_INCREASING** constraint of the **Example** slot

**Arg. properties**

- **Functional dependency**: **VALUE** determined by **VARIABLES**.
- **Functional dependency**: **FOUND** determined by **VARIABLES**.
Automaton

Similar to the automaton of the \texttt{MAX\_SURF\_INCREASING} constraint but use the decoration table \texttt{2.33}.  
POS\_MAX\_SURF\_INCREASING

1915
3.447 POS_MAX_SURF_INCREASING_SEQUENCE

**Origin**
Based on constraint \textsc{max} \textsc{surf} \textsc{increasing} \textsc{sequence}.

**Constraint**
\textsc{pos} \textsc{max} \textsc{surf} \textsc{increasing} \textsc{sequence}(\textsc{value}, \textsc{variables}, \textsc{found})

**Arguments**
\begin{align*}
\textsc{value} & : \text{dvar} \\
\textsc{variables} & : \text{collection}(\text{var} - \text{dvar}) \\
\textsc{found} & : \text{collection}(\text{var} - \text{dvar})
\end{align*}

**Restrictions**
\begin{align*}
\text{sv} & \leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{value} = -\infty \\
\text{rv} = 2 & \Rightarrow \text{value} = -\infty \lor \text{value} \geq 2 \ast \text{minv} + 1 \\
\text{rv} & \geq 3 \Rightarrow \text{value} = -\infty \lor \text{value} \geq \min(2 \ast \text{minv} + 1, \text{sv} \ast (\text{minv} + 1)) \\
\text{rv} = 2 & \Rightarrow \text{value} \leq 2 \ast \text{maxv} - 1 \\
\text{rv} \geq 3 & \Rightarrow \text{value} \leq \max(2 \ast \text{maxv} - 1 \ast \text{sv}, \text{sv} \ast (\text{maxv} - 1) \ast \text{sv}) \\
\text{required}(\text{variables}, \text{var}) & \\
\text{required}(\text{found}, \text{var}) & \\
\text{where} \\
\text{maxv} & = \text{maxval}(\text{variables}.\text{var}) \\
\text{rv} & = \text{range}(\text{variables}.\text{var}) \\
\text{sv} & = |\text{variables}| \\
\text{minv} & = \text{minval}(\text{variables}.\text{var})
\end{align*}

The constraint \textsc{max} \textsc{surf} \textsc{increasing} \textsc{sequence}(\textsc{value}, \textsc{variables}) holds. In addition, \textsc{found} is a collection of 0/1 variables where the value 1 indicates the position of the \textsc{found} letter in those occurrences of the pattern \textsc{increasing} \textsc{sequence} for which the feature \textsc{value} is \textsc{value}.

The position of the \textsc{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \textsc{increasing} \textsc{sequence} is the maximal subsequence which matches the regular expression \(< (< | =)^{\ast} < | <\).

Assume that the occurrence of the pattern \textsc{increasing} \textsc{sequence} starts at position \(i\) and ends at position \(j\). The feature \textsc{surf} computes the sum of the values from index \(i\) to index \(j+1\).

**Example**
\begin{align*}
(17, \{4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3\}, \\
\{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\})
\end{align*}

Figure 3.1012 provides an example where the \textsc{pos} \textsc{max} \textsc{surf} \textsc{increasing} \textsc{sequence} (17, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3], [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.
Figure 3.1012: Illustrating the POS_MAX_SURF_INCREASING_SEQUENCE constraint of the Example slot

Typical

\[ |\text{VARIABLES}| > 1 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_SURF_INCREASING_SEQUENCE constraint but use the decoration table 2.33.
POS_MAX_SURF_INCREASING_SEQUENCE  1919
3.448 **POS_MAX_SURF_INCREASING_TERRACE**

**DESCRIPTION**

**Origin**

Based on constraint **MAX_SURF_INCREASING_TERRACE**.

**Constraint**

**POS_MAX_SURF_INCREASING_TERRACE**(VALUE, VARIABLES, FOUND)

**Arguments**

VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

**Restrictions**

\[
sv \leq 3 \lor rv \leq 2 \Rightarrow VALUE = -\infty
\]

VALUE = -\infty \lor VALUE \geq \min(2 + (\text{minv} + 1), (sv - 2) \times (\text{minv} + 1))

VALUE \leq \max(2 + (\text{maxv} - 1)\#,(sv - 2) \times (\text{maxv} - 1)\#)

\text{required}(\text{VARIABLES}, \text{var})

\text{required}(\text{FOUND}, \text{var})

where

\[
\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})
\]

\[
sv = |\text{VARIABLES}|
\]

\[
\text{minv} = \text{minval}(\text{VARIABLES}.\text{var})
\]

\[
rv = \text{range}(\text{VARIABLES}.\text{var})
\]

The constraint **MAX_SURF_INCREASING_TERRACE**(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the **found** letter in those occurrences of the pattern **INCREASING_TERRACE** for which the feature value is **VALUE**.

The position of the **found** letter in an occurrence of a pattern is the first position where the occurrence of a pattern is identified, even if the pattern is not complete.

An occurrence of the pattern **INCREASING_TERRACE** is the maximal subsequence which matches the regular expression \(<\equiv^+<\).

Assume that the occurrence of the pattern **INCREASING_TERRACE** starts at position \(i\) and ends at position \(j\). The feature **SURF** computes the sum of the values from index \(i + 1\) to index \(j\).

**Purpose**

**Example**

\[
\begin{align*}
10, (1, 3, 3, 2, 5, 6, 4, 4, 2, 3, 3, 4, 4), \\
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
\end{align*}
\]

Figure 3.1013 provides an example where the **POS_MAX_SURF_INCREASING_TERRACE** \((10, [1, 3, 3, 3, 2, 5, 6, 4, 4, 2, 3, 3, 4, 4], [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])\) constraint holds.

**Typical**

\(|\text{VARIABLES}| > 3\)

\text{range}(\text{VARIABLES}.\text{var}) > 2
Figure 3.1013: Illustrating the POS_MAX_SURF_INCREASING_TERRACE constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the `MAX_SURF_INCREASING_TERRACE` constraint but use the decoration table 2.33.
3.449  POS_MAX_SURF_INFLEXION

**Origin**
Based on constraint MAX_SURF_INFLEXION.

**Constraint**
POS_MAX_SURF_INFLEXION(VALUE, VARIABLES, FOUND)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var−dvar)
- FOUND : collection(var−dvar)

**Restrictions**
- \( sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = -\infty \)
- \( VALUE = -\infty \lor VALUE \geq \min (\text{minv}, (sv - 2) \times \text{minv}) \)
- \( VALUE \leq \max (\text{maxv}, (sv - 2) \times \text{maxv}) \)
- \( \text{required}(\text{VARIABLES}, \text{var}) \)
- \( \text{required}(\text{FOUND}, \text{var}) \)

where
- \( \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \)
- \( sv = |\text{VARIABLES}| \)
- \( \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \)
- \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

The constraint MAX_SURF_INFLEXION(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern INFLEXION for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression \(< (< | =) > | > (>| =) ^* <\). Assume that the occurrence of the pattern INFLEXION starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i + 1 \) to index \( j \).

**Example**

\[
\begin{align*}
14, (1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4), \\
0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
\end{align*}
\]

Figure 3.1014 provides an example where the POS_MAX_SURF_INFLEXION (14, [1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4], [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

**Typical**
- \(|\text{VARIABLES}| > 2\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)
Figure 3.1014: Illustrating the POS_MAX_SURF_INFLEXION constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton Similar to the automaton of the MAX_SURF_INFLEXION constraint but use the decoration table 2.33.
3.450 POS_MAX_SURF_PEAK

**Description**

Based on constraint \(\text{MAX\_SURF\_PEAK}\).

**Arguments**

- **VALUE** : dvar
- **VARIABLES** : collection(var−dvar)
- **FOUND** : collection(var−dvar)

**Restrictions**

\[
\begin{align*}
sv & \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} & = -\infty \lor \text{VALUE} \geq \min(\text{minv} + 1, (sv - 2) + (\text{minv} + 1)) \\
\text{VALUE} & \leq \max(\text{maxv}, (sv - 2) + \text{maxv}) \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{required}(\text{FOUND}, \text{var}) & \\
\text{where} & \\
\text{minv} & = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var}) \\
\end{align*}
\]

The constraint \(\text{MAX\_SURF\_PEAK}(\text{VALUE}, \text{VARIABLES}, \text{FOUND})\) holds. In addition, **FOUND** is a collection of 0/1 variables where the value 1 indicates the position of the **found** letter in those occurrences of the pattern **PEAK** for which the feature value is **VALUE**.

**Purpose**

- An occurrence of the pattern **PEAK** is the **maximal** subsequence which matches the regular expression \(<(=|<)>(>|=)^*\).
- Assume that the occurrence of the pattern **PEAK** starts at position \(i\) and ends at position \(j\). The feature **SURF** computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

\[
(14, \langle 7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 3, 1 \rangle, \\
\langle 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0 \rangle)
\]

Figure 3.1015 provides an example where the **POS\_MAX\_SURF\_PEAK** constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 2\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Arg. properties**

- Functional dependency: **VALUE** determined by **VARIABLES**.
- Functional dependency: **FOUND** determined by **VARIABLES**.
Figure 3.1015: Illustrating the POS_MAX_SURF_PEAK constraint of the Example slot
Automaton

Similar to the automaton of the \texttt{MAX\_SURF\_PEAK} constraint but use the decoration table \texttt{2.33}.
### 3.451 POS_MAX_SURF_PLAIN

**DESCRIPTION**

**Origin**
Based on constraint `MAX_SURF_PLAIN`.

**Constraint**

```plaintext
POS_MAX_SURF_PLAIN(VALUE, VARIABLES, FOUND)
```

**Arguments**

- `VALUE`: `dvar`
- `VARIABLES`: `collection(var−dvar)`
- `FOUND`: `collection(var−dvar)`

**Restrictions**

- `sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = −∞`
- `VALUE = −∞ ∨ VALUE ≥ min(minv, (sv − 2) * minv)`
- `VALUE ≤ max(maxv − 10, (sv − 2) * (maxv − 1))`

Required:

- `VARIABLES`, `var`
- `FOUND`, `var`

where:

- `maxv = maxval(VARIABLES.var)`
- `sv = |VARIABLES|`
- `minv = minval(VARIABLES.var)`
- `rv = range(VARIABLES.var)`

The constraint `POS_MAX_SURF_PLAIN(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `PLAIN` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `PLAIN` is the **maximal** subsequence which matches the regular expression `>` = "<".

Assume that the occurrence of the pattern `PLAIN` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i + 1` to index `j`.

**Purpose**

**Example**

```
(6, (2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3),
  ⟨0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0⟩)
```

Figure 3.1016 provides an example where the `POS_MAX_SURF_PLAIN (6, [2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0])` constraint holds.

**Typical**

- `|VARIABLES| > 2`
- `range(VARIABLES.var) > 1`

**Arg. properties**

- Functional dependency: `VALUE` determined by `VARIABLES`.
- Functional dependency: `FOUND` determined by `VARIABLES`. 
Figure 3.1016: Illustrating the POS_MAX_SURF_PLAIN constraint of the Example slot
Automaton
Similar to the automaton of the MAX_SURF/plain constraint but use the decoration table 2.33.
### Origin
Based on constraint **MAX_SURF_PLATEAU**.

### Constraint
**POS_MAX_SURF_PLATEAU(VALUE, VARIABLES, FOUND)**

### Arguments
- **VALUE**: `dvar`
- **VARIABLES**: `collection(var−dvar)`
- **FOUND**: `collection(var−dvar)`

### Restrictions
- `sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = −∞`
- `VALUE = −∞ ∨ VALUE ≥ min(minv + 1, (sv − 2) ∗ (minv + 1))`
- `VALUE ≤ max(maxv, (sv − 2) ∗ maxv)`
- `required(VARIABLES, var)`
- `required(FOUNDCollection, var)`

where
- `maxv = maxval(VARIABLES.var)`
- `sv = |VARIABLES|`
- `minv = minval(VARIABLES.var)`
- `rv = range(VARIABLES.var)`

The constraint **MAX_SURF_PLATEAU(VALUE, VARIABLES)** holds. In addition, **FOUND** is a collection of 0/1 variables where the value 1 indicates the position of the **found** letter in those occurrences of the pattern **PLATEAU** for which the feature value is **VALUE**.

### Purpose
The position of the **found** letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern **PLATEAU** is the maximal subsequence which matches the regular expression `<=* >`.

Assume that the occurrence of the pattern **PLATEAU** starts at position `i` and ends at position `j`. The feature **SURF** computes the sum of the values from index `i + 1` to index `j`.

### Example

Figure 3.1017 provides an example where the **POS_MAX_SURF_PLATEAU** constraint holds.

### Typical
- `|VARIABLES| > 2`
- `range(VARIABLES.var) > 1`
Figure 3.1017: Illustrating the POS_MAX_SURF_PLATEAU constraint of the Example slot

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_SURF_PLATEAU constraint but use the decoration table 2.33.
3.453 POS_MAX_SURF_PROPERPLAIN

DESCRIPTION

Origin
Based on constraint MAX_SURF_PROPERPLAIN.

Constraint
POS_MAX_SURF_PROPERPLAIN(VALUE, VARIABLES, FOUND)

Arguments
- VALUE : dvar
- VARIABLES : collection(var--dvar)
- FOUND : collection(var--dvar)

Restrictions
- sv ≤ 3 ∨ rv ≤ 1 ⇒ VALUE = −∞
- VALUE = −∞ ∨ VALUE ≥ min(2 + minv, (sv − 2) * minv)
- VALUE ≤ max(2 * (maxv − 1) − (sv − 2) * (maxv − 1) − 1) required(VARIABLES, var)
- required(FOUND, var)

where
- maxv = maxval(VARIABLES.var)
- sv = |VARIABLES|
- minv = minval(VARIABLES.var)
- rv = range(VARIABLES.var)

The constraint MAX_SURF_PROPERPLAIN(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern PROPERPLAIN for which the feature value is VALUE.

Purpose
The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern PROPERPLAIN is the maximal subsequence which matches the regular expression > = + <.

Assume that the occurrence of the pattern PROPERPLAIN starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 1 to index j.

Example
\[
\begin{align*}
10, & (2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5) \\
& (0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
\end{align*}
\]

Figure 3.1018 provides an example where the POS_MAX_SURF_PROPERPLAIN (10, [2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5], [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

Typical
- |VARIABLES| > 3
- range(VARIABLES.var) > 1
Figure 3.1018: Illustrating the POS_MAX_SURF_PROPER_PLAIN constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \textit{MAX SURF PROPER PLAIN} constraint but use the decoration table 2.33.
3.454 POS_MAX_SURF_PROPER_PLATEAU

**Origin**
Based on constraint \textsc{max_surf_proper_plateau}.

**Constraint**
\[ \text{POS\_MAX\_SURF\_PROPER\_PLATEAU}(\text{VALUE, VARIABLES, FOUND}) \]

**Arguments**
- \text{VALUE} : dvar
- \text{VARIABLES} : collection(var−dvar)
- \text{FOUND} : collection(var−dvar)

**Restrictions**
- \( \text{sv} \leq 3 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = -\infty \)
- \( \text{VALUE} = -\infty \lor \text{VALUE} \geq \min(2 + (\text{minv} + 1), (\text{sv} - 2) \ast (\text{minv} + 1)) \)
- \( \text{VALUE} \leq \max(2 + \text{maxv}, (\text{sv} - 2) \ast \text{maxv}) \)
- \text{required}(\text{VARIABLES, var})
- \text{required}(\text{FOUND, var})
- \text{where}
  - \( \text{maxv} = \maxval(\text{VARIABLES}\_\text{var}) \)
  - \( \text{sv} = |\text{VARIABLES}| \)
  - \( \text{minv} = \minval(\text{VARIABLES}\_\text{var}) \)
  - \( \text{rv} = \text{range}(\text{VARIABLES}\_\text{var}) \)

The constraint \textsc{max_surf_proper_plateau}(\text{VALUE, VARIABLES}) holds. In addition, \text{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \text{found} letter in those occurrences of the pattern \text{proper_plateau} for which the feature value is \text{VALUE}.

The position of the \text{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \text{proper_plateau} is the \text{maximal} subsequence which matches the regular expression \( < \equiv^{+} > \).

Assume that the occurrence of the pattern \text{proper_plateau} starts at position \( i \) and ends at position \( j \). The feature \text{surf} computes the sum of the values from index \( i + 1 \) to index \( j \).

**Example**
\[
(15, [7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0])
\]

Figure 3.1019 provides an example where the \text{pos\_max\_surf\_proper\_plateau} \( (15, [7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]) \) constraint holds.

**Purpose**
The feature \text{surf} computes the sum of the values from index \( i + 1 \) to index \( j \).

**Typical**
- \(|\text{VARIABLES}| > 3\)
- \( \text{range}(\text{VARIABLES}\_\text{var}) > 1 \)
Figure 3.1019: Illustrating the POS_MAX_SURF_PROPER_PLATEAU constraint of the Example slot

**Arg. properties**

- **Functional dependency**: VALUE determined by VARIABLES.
- **Functional dependency**: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MAX\_SURF\_PROPER\_PLATEAU} constraint but use the decoration table 2.33.
3.455  POS_MAX_SURF_STEADY

**DESCRIPTION**

**Origin**
Based on constraint MAX_SURF_STEADY.

**Constraint**
POS_MAX_SURF_STEADY(VALUE, VARIABLES, FOUND)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var−dvar)
- FOUND : collection(var−dvar)

**Restrictions**
- \( sv \leq 1 \Rightarrow VALUE = -\infty \)
- \( VALUE = -\infty \lor VALUE \geq 2 \times \text{minv} \)
- \( VALUE \leq 2 \times \text{maxv} \)
- \( \text{required}(\text{VARIABLES}, \text{var}) \)
- \( \text{required}(\text{FOUND}, \text{var}) \)

where
- \( \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \)
- \( \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \)
- \( sv = |\text{VARIABLES}| \)

The constraint MAX_SURF_STEADY(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern STEADY for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

**Purpose**

An occurrence of the pattern STEADY is the subsequence which matches the regular expression =.

Assume that the occurrence of the pattern STEADY starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i \) to index \( j + 1 \).

**Example**

\[
\begin{pmatrix}
12, (1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6), \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0
\end{pmatrix}
\]

Figure 3.1020 provides an example where the POS_MAX_SURF_STEADY (12, [1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]) constraint holds.

**Typical**

\(|\text{VARIABLES}| > 1\)

**Arg. properties**
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Figure 3.1020: Illustrating the POS_MAX_SURF_STEADY constraint of the **Example** slot
Automaton

Similar to the automaton of the `MAX_SURF_STEADY` constraint but use the decoration table 2.33.
POS_MAX_SURF_STEADY

1951
3.456  POS_MAX_SURF_STEADY_SEQUENCE

**Origin**
Based on constraint MAX_SURF_STEADY_SEQUENCE.

**Constraint**
POS_MAX_SURF_STEADY_SEQUENCE(VALUE, VARIABLES, FOUND)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var − dvar)
- FOUND : collection(var − dvar)

**Restrictions**
- \( sv \leq 1 \Rightarrow VALUE = -\infty \)
- \( rv = 1 \Rightarrow VALUE = -\infty \lor VALUE \geq sv \times minv \)
- \( rv \geq 2 \Rightarrow VALUE = -\infty \lor VALUE \geq \min(2 \times minv, sv \times minv) \)
- \( rv = 1 \Rightarrow VALUE \leq sv \times maxv \)
- \( rv \geq 2 \Rightarrow VALUE \leq \max(2 \times maxv, sv \times maxv) \)

**Purpose**
The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `STEADY_SEQUENCE` is the `maximal` subsequence which matches the regular expression `\^*`.

Assume that the occurrence of the pattern `STEADY_SEQUENCE` starts at position \(i\) and ends at position \(j\). The feature `SURF` computes the sum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
(15, [3, 1, 1, 4, 5, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1], 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
\]

Figure 3.1021 provides an example where the `POS_MAX_SURF_STEADY_SEQUENCE` constraint holds.
Figure 3.1021: Illustrating the POS_MAX_SURF_STEADY_SEQUENCE constraint of the Example slot

Typical $|\text{VARIABLES}| > 1$

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MAX_SURF_STEADY_SEQUENCE constraint but use the decoration table 2.33.
3.457 POS_MAX_SURF_STRICTLY_DECREASING_SEQUENCE

**Origin**
Based on constraint MAX_SURF_STRICTLY_DECREASING_SEQUENCE.

**Constraint**
POS_MAX_SURF_STRICTLY_DECREASING_SEQUENCE(VALUE, VARIABLES, FOUND)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var−dvar)
- FOUND : collection(var−dvar)

**Restrictions**
- \(sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = -\infty\)
- \(\minv < 0 \Rightarrow VALUE = -\infty \lor VALUE \geq \ell_1 \ast \minv + |\ell_1 \ast (\ell_1 - 1)/2|\)
- \(\minv \geq 0 \Rightarrow VALUE = -\infty \lor VALUE \geq 2 \ast \minv + 1\)
- \(\maxv > 0 \Rightarrow VALUE \leq \ell_2 \ast \maxv - |\ell_2 \ast (\ell_2 - 1)/2|\)
- \(\maxv \leq 0 \Rightarrow VALUE \leq 2 \ast \maxv - 1\)
- \(\text{required}(\text{VARIABLES}, \text{var})\)
- \(\text{required}(\text{FOUND, var})\)

where
- \(\maxv = \text{maxval}(\text{VARIABLES}, \text{var})\)
- \(\text{rv} = \text{range}(\text{VARIABLES}, \text{var})\)
- \(sv = |\text{VARIABLES}|\)
- \(\ell_1 = \min(\text{min}(sv, rv), \text{minv})\)
- \(\ell_2 = \min(\text{min}(sv, rv), \maxv)\)
- \(\minv = \text{minval}(\text{VARIABLES}, \text{var})\)

The constraint \(\text{MAX\_SURF\_STRICTLY\_DECREASING\_SEQUENCE}(\text{VALUE}, \text{VARIABLES})\) holds. In addition, \(\text{FOUND}\) is a collection of 0/1 variables where the value 1 indicates the position of the \text{found} letter in those occurrences of the pattern \text{STRICTLY\_DECREASING\_SEQUENCE} for which the feature value is \text{VALUE}.

The position of the \text{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \text{STRICTLY\_DECREASING\_SEQUENCE} is the maximal subsequence which matches the regular expression \(>^{+}\).

Assume that the occurrence of the pattern \text{STRICTLY\_DECREASING\_SEQUENCE} starts at position \(i\) and ends at position \(j\). The feature \text{SURF} computes the sum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
\begin{pmatrix}
13, 4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3, \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 3.1022 provides an example where the POS_MAX_SURF_STRICTLY_DECREASING_SEQUENCE (13, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0]) constraint holds.
Figure 3.1022: Illustrating the POS_MAX_SURF_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot

**Typical**

\[ |\text{VARIABLES}| > 1 \]

\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

**Arg. properties**

- **Functional dependency**: VALUE determined by VARIABLES.
- **Functional dependency**: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the `MAX_SURF_STRICTLY_DECREASING_SEQUENCE` constraint but use the decoration table 2.33.
POS_MAX_SURF_STRICTLY_DECREASING_SEQUENCE 1959
3.458 **POS_MAX_SURF_STRICTLY_INCREASING_SEQUENCE**

**DESCRIPTION**

Based on constraint **MAX_SURF_STRICTLY_INCREASING_SEQUENCE**.

**AUTOMATON**

**Constraint**

POS_MAX_SURF_STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES, FOUND)

**Arguments**

VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

**Restrictions**

\[
\begin{align*}
 sv & \leq 1 \lor rv \leq 1 \Rightarrow VALUE = -\infty \\
 minv < 0 & \Rightarrow VALUE = -\infty \lor VALUE \geq \ell_1 + minv + [\ell_1 \ast (\ell_1 - 1) / 2] \\
 minv \geq 0 & \Rightarrow VALUE = -\infty \lor VALUE \geq 2 \ast minv + 1 \\
 maxv > 0 & \Rightarrow VALUE \leq \ell_2 \ast maxv - [\ell_2 \ast (\ell_2 - 1) / 2] \\
 maxv \leq 0 & \Rightarrow VALUE \leq 2 \ast maxv - 1 \\
 \text{required}(VARIABLES, var) \\
 \text{required}(FOUND, var)
\end{align*}
\]

where

\[
\begin{align*}
 maxv &= \text{maxval}(VARIABLES, var) \\
 rv &= \text{range}(VARIABLES, var) \\
 sv &= |VARIABLES| \\
 \ell_1 &= \min(\min(sv, rv), \minv) \\
 \ell_2 &= \min(\min(sv, rv), \maxv) \\
 minv &= \minval(VARIABLES, var)
\end{align*}
\]

The constraint **MAX_SURF_STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES)** holds. In addition, **FOUND** is a collection of 0/1 variables where the value 1 indicates the position of the **found** letter in those occurrences of the pattern **STRICTLY_INCREASING_SEQUENCE** for which the feature value is **VALUE**.

The position of the **found** letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern **STRICTLY_INCREASING_SEQUENCE** is the *maximal* subsequence which matches the regular expression \(<^+\).

Assume that the occurrence of the pattern **STRICTLY_INCREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **SURF** computes the sum of the values from index \(i\) to index \(j + 1\).

**Purpose**

**Example**

\[
(\begin{array}{c}
16, \{4, 3, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3\}, \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
\end{array})
\]

Figure 3.1023 provides an example where the **POS_MAX_SURF_STRICTLY_INCREASING_SEQUENCE** (16, [4, 3, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.
Figure 3.1023: Illustrating the **POS_MAX_SURF STRICTLY INCREASING_SEQUENCE** constraint of the **Example** slot

**Typical**

\[ |\text{VARIABLES}| > 1 \]

\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

**Arg. properties**

- Functional dependency: **VALUE** determined by **VARIABLES**.
- Functional dependency: **FOUND** determined by **VARIABLES**.
Automaton

Similar to the automaton of the MAX_SURF.Strictly_Increasing_Sequence constraint but use the decoration table 2.33.
POS_MAX_SURF STRICTLY_INCREASING_SEQUENCE
3.459  POS_MAX_SURF_SUMMIT

DESCRIPTION

Based on constraint MAX_SURF_SUMMIT.

Arguments

\[
\begin{align*}
\text{VALUE} & : \ dvar \\
\text{VARIABLES} & : \ \text{collection}(\text{var} - \ dvar) \\
\text{FOUND} & : \ \text{collection}(\text{var} - \ dvar)
\end{align*}
\]

Restrictions

\[
\begin{align*}
sv \leq 2 & \lor rv \leq 1 \Rightarrow VALUE = -\infty \\
rv = 2 & \Rightarrow VALUE = -\infty \lor VALUE \geq \minv + 1 \\
rv \geq 3 & \Rightarrow \\
VALUE & = -\infty \lor VALUE \geq \min(\minv + 1, (sv - 2) \ast (\minv + 1) + 1) \\
rv = 2 & \Rightarrow VALUE \leq \maxv \\
rv \geq 3 & \Rightarrow VALUE \leq \max(\maxv, (sv - 2) \ast (\maxv - 1) + 1) \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{required}(\text{FOUND}, \text{var}) & \\
\text{where} & \\
\maxv & = \text{maxval}(\text{VARIABLES}, \text{var}) \\
\rv & = \text{range}(\text{VARIABLES}, \text{var}) \\
sv & = |\text{VARIABLES}| \\
\minv & = \text{minval}(\text{VARIABLES}, \text{var})
\end{align*}
\]

Purpose

The constraint $\text{MAX\_SURF\_SUMMIT}(\text{VALUE}, \text{VARIABLES})$ holds. In addition, $\text{FOUND}$ is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern SUMMIT for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression $< < (|=|<)^* <) | > (|=|>)^* >)$.

Assume that the occurrence of the pattern SUMMIT starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 1$ to index $j$.

Example

\[
\begin{pmatrix}
13, \langle 7, 1, 5, 4, 3, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1 \rangle, \\
\langle 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 \rangle
\end{pmatrix}
\]

Figure 3.1024 provides an example where the $\text{POS\_MAX\_SURF\_SUMMIT}$ constraint holds.
Figure 3.1024: Illustrating the POS_MAX_SURF_SUMMIT constraint of the Example slot

**Typical**

\[|\text{VARIABLES}| > 2\]

\[\text{range(} \text{VARIABLES.var}) > 1\]

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MAX\_SURF\_SUMMIT} constraint but use the decoration table \ref{table2}.3.
## 3.460 POS_MAX_SURF_VALLEY

### Origin
Based on constraint MAX_SURF_VALLEY.

### Constraint
\[
\text{POS_MAX_SURF_VALLEY}(\text{VALUE, VARIABLES, FOUND})
\]

### Arguments
- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)
- **FOUND**: collection(var−dvar)

### Restrictions
\[
\begin{align*}
\text{sv} & \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = -\infty \\
\text{VALUE} & = -\infty \lor \text{VALUE} \geq \min(\text{minv}, (\text{sv} - 2) \ast \text{minv}) \\
\text{VALUE} & \leq \max(\text{maxv} - 10, (\text{sv} - 2) \ast (\text{maxv} - 1)\#) \\
\text{required}(\text{VARIABLES, var}) \\
\text{required}(\text{FOUND, var})
\end{align*}
\]

where
- \(\text{maxv} = \text{maxval}(\text{VARIABLES, var})\)
- \(\text{sv} = |\text{VARIABLES}|\)
- \(\text{minv} = \text{minval}(\text{VARIABLES, var})\)
- \(\text{rv} = \text{range}(\text{VARIABLES, var})\)

The constraint \(\text{POS_MAX_SURF_VALLEY}(\text{VALUE, VARIABLES})\) holds. In addition, \(\text{FOUND}\) is a collection of \(0/1\) variables where the value 1 indicates the position of the found letter in those occurrences of the pattern VALLEY for which the feature value is VALUE.

### Purpose
The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression \(>(= | >)^* (< | =)^* <\).

Assume that the occurrence of the pattern VALLEY starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

### Example
\[
\left( 15, \langle 1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 7 \rangle, \langle 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0 \rangle \right)
\]

Figure 3.1025 provides an example where the \(\text{POS_MAX_SURF_VALLEY}\) constraint holds.

### Typical
- |\(\text{VARIABLES}\)| > 2
- \(\text{range}(\text{VARIABLES, var}) > 1\)

### Arg. properties
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Figure 3.1025: Illustrating the POS_MAX_SURF_VALLEY constraint of the Example slot
Automaton

Similar to the automaton of the MAX_SURF_VALLEY constraint but use the decoration table 2.33.
POS_MAX_SURF_VALLEY

1971
3.461  POS_MAX_SURF_ZIGZAG

**Description**

Based on constraint MAX_SURF_ZIGZAG.

**Constraint**

POS_MAX_SURF_ZIGZAG(VALUE, VARIABLES, FOUND)

**Arguments**

VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

**Restrictions**

\[
sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = -\infty
\]

\[
VALUE \geq \min \left( 2 \ast \minv + 1, \left\lfloor (sv - 1)/2 \right\rfloor \ast \minv + \left\lfloor (sv - 2)/2 \right\rfloor \ast (\minv + 1) \right)
\]

\[
VALUE \leq \max \left( 2 \ast \maxv - 10, \left\lfloor (sv - 1)/2 \right\rfloor \ast \maxv + \left\lfloor (sv - 2)/2 \right\rfloor \ast (\maxv - 1) \right)
\]

required(VARIABLES, var)
required(FOUND, var)

where

minv = minval(VARIABLES.var)
maxv = maxval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)

**Purpose**

The constraint MAX_SURF_ZIGZAG(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern ZIGZAG for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression \((<>)^+(< | <>)(<>)(| > <))\).

Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

\[
\left( 21, \langle 4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1 \rangle, \langle 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 \rangle \right)
\]

Figure 3.1026 provides an example where the POS_MAX_SURF_ZIGZAG (21, [4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

**Typical**

\(|VARIABLES| > 3\)
\(\text{range}(\text{VARIABLES}.\text{var}) > 1\)
Figure 3.1026: Illustrating the POS_MAX_SURF_ZIGZAG constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MAX\_SURF\_ZIGZAG} constraint but use the decoration table \texttt{2.33}.
3.462  POS\_MAX\_WIDTH\_DECREASING\_SEQUENCE

**Description**

Based on constraint \(\text{MAX\_WIDTH\_DECREASING\_SEQUENCE}\).

**Constraint**

\(\text{POS\_MAX\_WIDTH\_DECREASING\_SEQUENCE}(\text{VALUE, VARIABLES, FOUND})\)

**Arguments**

- \(\text{VALUE} : \text{dvar}\)
- \(\text{VARIABLES} : \text{collection}(\text{var}\_\text{dvar})\)
- \(\text{FOUND} : \text{collection}(\text{var}\_\text{dvar})\)

**Restrictions**

\(sv \leq 1 \lor rv \leq 1 \Rightarrow \text{VALUE} = 0\)
\(\text{VALUE} = 0 \lor \text{VALUE} \geq 2\)
\(rv = 2 \Rightarrow \text{VALUE} \leq 2\)
\(rv \geq 3 \Rightarrow \text{VALUE} \leq sv\)
\(\text{required} (\text{VARIABLES}, \text{var})\)
\(\text{required} (\text{FOUND}, \text{var})\)

where

\(sv = |\text{VARIABLES}|\)
\(rv = \text{range} (\text{VARIABLES}.\text{var})\)

**Purpose**

The position of the \text{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \text{DECREASING\_SEQUENCE} is the \text{maximal} subsequence which matches the regular expression \((>|=)^*>|>.\)

Assume that the occurrence of the pattern \text{DECREASING\_SEQUENCE} starts at position \(i\) and ends at position \(j\). The feature \text{WIDTH} computes the value \(j - i + 2\).

**Example**

\[
(5, [3, 4, 2, 5, 6, 6, 4, 3, 1, 1, 4, 6, 4, 4], [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0])
\]

Figure 3.1027 provides an example where the \text{POS\_MAX\_WIDTH\_DECREASING\_SEQUENCE} constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 1\)
- \(\text{range} (\text{VARIABLES}.\text{var}) > 1\)

**Arg. properties**

- Functional dependency: \(\text{VALUE}\) determined by \text{VARIABLES}.
- Functional dependency: \(\text{FOUND}\) determined by \text{VARIABLES}.
Figure 3.1027: Illustrating the POS_MAX_WIDTH_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Similar to the automaton of the `MAX_WIDTH_DECREASING_SEQUENCE` constraint but use the decoration table 2.33.
3.463  POS_MAX_WIDTH_DECREASING_TERRACE

**DESCRIPTION**

Origin

Based on constraint MAX_WIDTH_DECREASING_TERRACE.

Constraint

POS_MAX_WIDTH_DECREASING_TERRACE(VALUE, VARIABLES, FOUND)

Arguments

VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions

\[
\begin{align*}
sv & \leq 3 \lor rv \leq 2 \Rightarrow VALUE = 0 \\
VALUE & = 0 \lor VALUE \geq 2 \\
VALUE & \leq \max(0, sv - 2x) \\
\text{required}(VARIABLES, var) \\
\text{required}(FOUND, var)
\end{align*}
\]

where

\[
\begin{align*}
sv & = |VARIABLES| \\
rv & = \text{range}(VARIABLES.var)
\end{align*}
\]

The constraint MAX_WIDTH_DECREASING_TERRACE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern DECREASING_TERRACE for which the feature value is VALUE.

Purpose

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern DECREASING_TERRACE is the maximal subsequence which matches the regular expression \( > = + > \).

Assume that the occurrence of the pattern DECREASING_TERRACE starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

Example

\[
\begin{align*}
(2, 6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3), \\
(0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0)
\end{align*}
\]

Figure 3.1028 provides an example where the POS_MAX_WIDTH_DECREASING_TERRACE (2, [6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3], [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

Typical

\(|VARIABLES| > 3 \\
\text{range}(VARIABLES.var) > 2 \)

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Figure 3.1028: Illustrating the POS_MAX_WIDTH_DECREASING_TERRACE constraint of the Example slot
Automaton

Similar to the automaton of the \texttt{MAX\_WIDTH\_DECREASING\_TERRACE} constraint but use the decoration table \texttt{2.33}.
3.464  POS_MAX_WIDTH_GORGE

**DESCRIPTION**

Based on constraint **MAX_WIDTH_GORGE**.

**AUTOMATON**

\[(> | > \ (|= >)^* >) < | < \ (|= <)^* <)\]

**Origin**

Based on constraint **MAX_WIDTH_GORGE**.

**Constraint**

\[\text{POS_MAX_WIDTH_GORGE(VALUE, VARIABLES, FOUND)}\]

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>dvar</td>
</tr>
<tr>
<td>VARIABLES</td>
<td>collection(var−dvar)</td>
</tr>
<tr>
<td>FOUND</td>
<td>collection(var−dvar)</td>
</tr>
</tbody>
</table>

**Restrictions**

- \(sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = 0\)
- \(VALUE = 0 \lor VALUE \geq 1\)
- \(rv = 2 \Rightarrow VALUE \leq 1\)
- \(rv \geq 3 \Rightarrow VALUE \leq \max(0, sv - 2)\)
- \(\text{required}(\text{VARIABLES, var})\)
- \(\text{required}(\text{FOUND, var})\)

where

\(sv = |\text{VARIABLES}|\)
\(rv = \text{range}(\text{VARIABLES.var})\)

The constraint **MAX_WIDTH_GORGE(VALUE, VARIABLES)** holds. In addition, **FOUND** is a collection of 0/1 variables where the value 1 indicates the position of the **found** letter in those occurrences of the pattern **GORGE** for which the feature value is **VALUE**.

The position of the **found** letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern **GORGE** is the **maximal** subsequence which matches the regular expression \((> | > \ (|= >)^* >) < | < \ (|= <)^* <)\).

Assume that the occurrence of the pattern **GORGE** starts at position \(i\) and ends at position \(j\). The feature **WIDTH** computes the value \(j - i\).

**Purpose**

**Example**

\[
(3, (1, 7, 3, 4, 4, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7), (0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0))
\]

Figure 3.1029 provides an example where the **POS_MAX_WIDTH_GORGE** constraint holds.

**Typical**

\(|\text{VARIABLES}| > 2\)
\(\text{range}(\text{VARIABLES.var}) > 1\)

**Arg. properties**

- Functional dependency: **VALUE** determined by **VARIABLES**.
- Functional dependency: **FOUND** determined by **VARIABLES**.
Figure 3.1029: Illustrating the POS_MAX_WIDTH_GORGE constraint of the Example slot
Automaton

Similar to the automaton of the `MAX_WIDTH_GORGE` constraint but use the decoration table 2.33.
POS_MAX_WIDTH_GORGE 1987
### Description

- **Origin:** Based on constraint `MAX_WIDTH_INCREASING_SEQUENCE`.
- **Constraint:**
  
  \[
  \text{POS\_MAX\_WIDTH\_INCREASING\_SEQUENCE}(VALUE, VARIABLES, FOUND)
  \]

- **Arguments**:
  
  - `VALUE`: `dvar`
  - `VARIABLES`: `collection(var – dvar)`
  - `FOUND`: `collection(var – dvar)`

- **Restrictions**:
  
  - \( sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \)
  - \( VALUE = 0 \lor VALUE \geq 2 \)
  - \( rv = 2 \Rightarrow VALUE \leq 2 \)
  - \( rv \geq 3 \Rightarrow VALUE \leq sv^2 \)
  - `required(VARIABLES, var)`
  - `required(FOUND, var)`
  
  where
  
  - \( sv = |VARIABLES| \)
  - \( rv = \text{range}(VARIABLES.var) \)

- **Purpose**

  The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

  An occurrence of the pattern `INCREASING_SEQUENCE` is the maximal subsequence which matches the regular expression \(< (< | =)^* < | < \).

  Assume that the occurrence of the pattern `INCREASING_SEQUENCE` starts at position \( i \) and ends at position \( j \). The feature `WIDTH` computes the value \( j - i + 2 \).

- **Example**

  \[
  (5, [4, 3, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3], [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
  \]

  Figure 3.1030 provides an example where the `POS\_MAX\_WIDTH\_INCREASING\_SEQUENCE` constraint holds.

- **Typical**

  \[ |VARIABLES| > 1 \]

  \[ \text{range}(VARIABLES.var) > 1 \]

- **Arg. properties**

  - Functional dependency: `VALUE` determined by `VARIABLES`.
  - Functional dependency: `FOUND` determined by `VARIABLES`.
Fig. 3.1030: Illustrating the POS_MAX_WIDTH_INCREASING_SEQUENCE constraint of the Example slot
Automaton: Similar to the automaton of the MAX_WIDTH_INCREASING_SEQUENCE constraint but use the decoration table 2.33.
### 3.466 POS_MAX_WIDTH_INCREASING_TERRACE

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
</table>

#### Origin
Based on constraint `MAX_WIDTH_INCREASING_TERRACE`.

#### Constraint
`POS_MAX_WIDTH_INCREASING_TERRACE(VALUE, VARIABLES, FOUND)`

#### Arguments
- **VALUE**: `dvar`
- **VARIABLES**: `collection(var-dvar)`
- **FOUND**: `collection(var-dvar)`

#### Restrictions
\[
\begin{align*}
sv \leq 3 & \lor rv \leq 2 \Rightarrow VALUE = 0 \\
\text{VALUE} & = 0 \lor \text{VALUE} \geq 2 \\
\text{VALUE} & \leq \max(0, sv - 2k) \\
\text{required}(\text{VARIABLES}, \text{var}) & \Rightarrow \text{required}(\text{FOUND}, \text{var}) \\
\end{align*}
\]

where
\[
sv = |\text{VARIABLES}| \\
rv = \text{range}(\text{VARIABLES}.\text{var})
\]

The constraint `MAX_WIDTH_INCREASING_TERRACE(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `INCREASING_TERRACE` for which the feature value is `VALUE`.

#### Purpose
The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `INCREASING_TERRACE` is the maximal subsequence which matches the regular expression `< = + <`.

Assume that the occurrence of the pattern `INCREASING_TERRACE` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j - i`.

#### Example
\[
\begin{pmatrix}
3, (1, 3, 3, 3, 2, 5, 6, 4, 2, 3, 3, 3, 4, 4) , \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0)
\end{pmatrix}
\]

Figure 3.103 provides an example where the `POS_MAX_WIDTH_INCREASING_TERRACE (3, [1, 3, 3, 3, 2, 5, 6, 4, 2, 3, 3, 3, 4, 4], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])` constraint holds.

#### Typical
- `|VARIABLES| > 3`
- `\text{range}(\text{VARIABLES}.\text{var}) > 2`

#### Arg. properties
- **Functional dependency**: `VALUE` determined by `VARIABLES`.
- **Functional dependency**: `FOUND` determined by `VARIABLES`.
Figure 3.1031: Illustrating the **POS_MAX_WIDTH_INCREASING_TERRACE** constraint of the **Example** slot
Automaton

Similar to the automaton of the \textit{MAX WIDTH INCREASING TERRACE} constraint but use the decoration table 2.33.
3.467  POS_MAX_WIDTH_INFLEXION

Description  Automaton

Origin  Based on constraint MAX_WIDTH_INFLEXION.

Constraint  POS_MAX_WIDTH_INFLEXION(VALUE, VARIABLES, FOUND)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>dvar</td>
</tr>
<tr>
<td>VARIABLES</td>
<td>collection(var−dvar)</td>
</tr>
<tr>
<td>FOUND</td>
<td>collection(var−dvar)</td>
</tr>
</tbody>
</table>

Restrictions

\[ sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = 0 \]
VALUE \geq 0
VALUE \leq \max(0, sv − 2v)
required(VARIABLES, var)
required(FOUND, var)
where
\[ sv = |VARIABLES| \]
\[ rv = \text{range}(VARIABLES\.var) \]

Purpose

The constraint MAX_WIDTH_INFLEXION(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern INFLEXION for which the feature value is VALUE. The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression \(< (< | = > | > | = >) > | > (> | = >) > <\). Assume that the occurrence of the pattern INFLEXION starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

Example

\[
\begin{pmatrix}
3, (1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 4, 4), (0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0)
\end{pmatrix}
\]

Figure 3.1032 provides an example where the POS_MAX_WIDTH_INFLEXION (3, [1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 4, 4], [0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

Typical

\[ |VARIABLES| > 2 \]
\[ \text{range}(VARIABLES\.var) > 1 \]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Figure 3.1032: Illustrating the POS_MAX_WIDTH_INFLEXION constraint of the Example slot
Automaton

Similar to the automaton of the MAX_WIDTH_INFLEXION constraint but use the decoration table 2.33.
### 3.468 POS_MAX_WIDTH_PEAK

**DESCRIPTION AUTOMATON**

**Origin**
Based on constraint \( \text{MAX_WIDTH_PEAK} \).

**Constraint**
POS_MAX_WIDTH_PEAK(VALUE, VARIABLES, FOUND)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var−dvar)
- FOUND : collection(var−dvar)

**Restrictions**
- \( \text{sv} \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \)
- \( \text{VALUE} \geq 0 \)
- \( \text{VALUE} < \max(0, \text{sv} - 2) \)
- required(VARIABLES, var)
- required(FOUND, var)
  
  where
  - \( \text{sv} = |\text{VARIABLES}| \)
  - \( \text{rv} = \text{range}(\text{VARIABLES}.\text{var}) \)

The constraint \( \text{MAX_WIDTH_PEAK(VALUE, VARIABLES)} \) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern PEAK for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

**Purpose**
An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression \( < (= | <)\*(> | =)\*> \).

Assume that the occurrence of the pattern PEAK starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

**Example**
\[
\begin{pmatrix}
3, 7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 3, 1, \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0
\end{pmatrix}
\]

Figure 3.1033 provides an example where the \( \text{POS_MAX_WIDTH_PEAK} \)
\((3, [7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 3, 1], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0])\)
constraint holds.

**Typical**
- \(|\text{VARIABLES}| > 2\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Arg. properties**
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Figure 3.1033: Illustrating the POS_MAX_WIDTH_PEAK constraint of the Example slot
Automaton

Similar to the automaton of the MAX_WIDTH_PEAK constraint but use the decoration table 2.33.
3.469 POS_MAX_WIDTH_PLAIN

Origin: Based on constraint MAX_WIDTH_PLAIN.

Constraint: \( \text{POS\_MAX\_WIDTH\_PLAIN}(\text{VALUE}, \text{VARIABLES}, \text{FOUND}) \)

Arguments:
- \( \text{VALUE} : \text{dvar} \)
- \( \text{VARIABLES} : \text{collection}(\text{var} \rightarrow \text{dvar}) \)
- \( \text{FOUND} : \text{collection}(\text{var} \rightarrow \text{dvar}) \)

Restrictions:
- \( sv \leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = 0 \)
- \( \text{VALUE} \geq 0 \)
- \( \text{VALUE} \leq \max(0, sv - 2v) \)
- \( \text{required}(\text{VARIABLES}, \text{var}) \)
- \( \text{required}(\text{FOUND}, \text{var}) \)

where
- \( sv = \lvert \text{VARIABLES} \rvert \)
- \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

Purpose:
The constraint \( \text{MAX\_WIDTH\_PLAIN}(\text{VALUE}, \text{VARIABLES}) \) holds. In addition, \( \text{FOUND} \) is a collection of 0/1 variables where the value 1 indicates the position of the \text{found} letter in those occurrences of the pattern \text{PLAIN} for which the feature value is \text{VALUE}.

The position of the \text{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \text{PLAIN} is the \text{maximal} subsequence which matches the regular expression \( > = < \).

Assume that the occurrence of the pattern \text{PLAIN} starts at position \( i \) and ends at position \( j \). The feature \text{WIDTH} computes the value \( j - i \).

Example:

\[
(2, (2, 3, 6, 5, 7, 6, 6, 4, 5, 4, 3, 3, 6, 6, 3), (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0))
\]

Figure 3.1034 provides an example where the \( \text{POS\_MAX\_WIDTH\_PLAIN} \) constraint holds.

Typical:
- \( \lvert \text{VARIABLES} \rvert > 2 \)
- \( \text{range}(\text{VARIABLES}.\text{var}) > 1 \)

Arg. properties:
- Functional dependency: \text{VALUE} determined by \text{VARIABLES}.
- Functional dependency: \text{FOUND} determined by \text{VARIABLES}. 
Figure 3.1034: Illustrating the POS_MAX_WIDTHPlain constraint of the Example slot
Automaton

Similar to the automaton of the MAX WIDTH constraint but use the decoration table 2.33.
### 3.470 POS_MAX_WIDTH_PLATEAU

**DESCRIPTION**

**AUTOMATON**

**Origin**

Based on constraint `MAX_WIDTH_PLATEAU`.

**Constraint**

`POS_MAX_WIDTH_PLATEAU(VALUE, VARIABLES, FOUND)`

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var−dvar)`
- **FOUND**: `collection(var−dvar)`

**Restrictions**

- `sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = 0`
- `VALUE ≥ 0`
- `VALUE ≤ max(0, sv − 2v)`
- `required(VARIABLES, var)`
- `required(FOUND, var)`

where

- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

The constraint `MAX_WIDTH_PLATEAU(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern `PLATEAU` for which the feature value is `VALUE`. The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

**Purpose**

An occurrence of the pattern `PLATEAU` is the maximal subsequence which matches the regular expression `<=∗>`. Assume that the occurrence of the pattern `PLATEAU` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j − i`.

**Example**

```
( 4, 1, 3, 3, 5, 5, 5, 2, 4, 4, 4, 3, 3, 1, 5, 5),
( 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
```

Figure 3.1035 provides an example where the `POS_MAX_WIDTH_PLATEAU` constraint holds.

**Typical**

- `|VARIABLES| > 2`
- `range(VARIABLES.var) > 1`

**Arg. properties**

- **Functional dependency**: `VALUE` determined by `VARIABLES`.
- **Functional dependency**: `FOUND` determined by `VARIABLES`.
Figure 3.1035: Illustrating the POS_MAX_WIDTH_PLATEAU constraint of the Example slot
Automaton  Similar to the automaton of the MAX_WIDTH_PLATEAU constraint but use the decoration table 2.33.
### 3.471  **POS_MAX_WIDTH_PROPER_PLAIN**

#### Description

Based on constraint MAX_WIDTH_PROPER_PLAIN.

**Constraint**

\[
\text{POS_MAX_WIDTH_PROPER_PLAIN}(\text{VALUE}, \text{VARIABLES}, \text{FOUND})
\]

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)
- **FOUND**: collection(var−dvar)

**Restrictions**

\[
\begin{align*}
\text{sv} \leq 3 \lor \text{rv} \leq 1 & \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} = 0 \lor \text{VALUE} \geq 2 & \\
\text{VALUE} \leq \max(0, \text{sv} - 2x) & \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{required}(\text{FOUND}, \text{var}) & \\
\text{where} & \\
\text{sv} = |\text{VARIABLES}| & \\
\text{rv} = \text{range}(\text{VARIABLES}.\text{var}) &
\end{align*}
\]

The constraint MAX_WIDTH_PROPER_PLAIN(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern PROPER_PLAIN for which the feature value is VALUE.

**Purpose**

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern PROPER_PLAIN is the maximal subsequence which matches the regular expression \(> =^+ <\).

Assume that the occurrence of the pattern PROPER_PLAIN starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

**Example**

\[
(3, [2, 7, 5, 6, 3, 7, 4, 5, 6, 5, 3, 3, 3, 3], \langle 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0 \rangle)
\]

Figure 3.1036 provides an example where the POS_MAX_WIDTH_PROPER_PLAIN (3, [2, 7, 5, 6, 3, 7, 4, 5, 6, 5, 3, 3, 3, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]) constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 3\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Figure 3.1036: Illustrating the POS_MAX_WIDTH_PROPER_PLAIN constraint of the Example slot
Automaton

Similar to the automaton of the MAX_WIDTH_PROPERPlain constraint but use the decoration table 2.33.
### 3.472 POS_MAX_WIDTH_PROPER_PLATEAU

#### Description

Based on constraint `MAX_WIDTH_PROPER_PLATEAU`.

#### Constraint

`POS_MAX_WIDTH_PROPER_PLATEAU(VALUE, VARIABLES, FOUND)`

#### Arguments

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var−dvar)`
- **FOUND**: `collection(var−dvar)`

#### Restrictions

- \( sv \leq \lfloor \frac{3}{2} \rfloor \lor rv \leq 1 \Rightarrow VALUE = 0 \)
- \( VALUE = 0 \lor VALUE \geq 2 \)
- \( VALUE \leq \max(0, sv - 2n) \)
- \( \text{required}(\text{VARIABLES}, \text{var}) \)
- \( \text{required}(\text{FOUND}, \text{var}) \)

where

- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range}(\text{VARIABLES} var) \)

The constraint `MAX_WIDTH_PROPER_PLATEAU(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern `PROPER_PLATEAU` for which the feature value is `VALUE`.

#### Purpose

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `PROPER_PLATEAU` is the maximal subsequence which matches the regular expression `< =+ >`.

Assume that the occurrence of the pattern `PROPER_PLATEAU` starts at position \( i \) and ends at position \( j \). The feature `WIDTH` computes the value \( j - i \).

#### Example

\[
\left( 3, (7, 1, 3, 2, 5, 1, 4, 3, 2, 3, 5, 5, 3), (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0) \right)
\]

Figure 3.1037 provides an example where the `POS_MAX_WIDTH_PROPER_PLATEAU (3, [7, 1, 3, 2, 5, 1, 4, 3, 2, 3, 5, 5, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0])` constraint holds.

#### Typical

- \(|\text{VARIABLES}| > 3\)
- \(\text{range}(\text{VARIABLES}. \text{var}) > 1\)

#### Arg. properties

- Functional dependency: `VALUE` determined by `VARIABLES`.
- Functional dependency: `FOUND` determined by `VARIABLES`.

---

**PL**

2016
Figure 3.1037: Illustrating the POS_MAX_WIDTH_PROPER_PLATEAU constraint of the Example slot
Automaton

Similar to the automaton of the MAX_WIDTH_PROPER_PLATEAU constraint but use the decoration table 2.33.
### 3.473 POS_MAX_WIDTH_STEADY_SEQUENCE

#### DESCRIPTION

**Origin**
Based on constraint `MAX_WIDTH_STEADY_SEQUENCE`.

**Constraint**
`POS_MAX_WIDTH_STEADY_SEQUENCE(VALUE, VARIABLES, FOUND)`

**Arguments**
- `VALUE`: `dvar`
- `VARIABLES`: `collection(var−dvar)`
- `FOUND`: `collection(var−dvar)`

**Restrictions**
- `sv ≤ 1 ⇒ VALUE = 0`
- `VALUE = 0 ∨ VALUE ≥ 2`
- `rv = 1 ∧ sv ≥ 2 ⇒ VALUE ≥ sv`
- `rv ≥ 2 ∧ sv ≥ 2 ⇒ VALUE ≥ 0`
- `VALUE ≤ av`*
- `required(VARIABLES, var)`
- `required(FOUND, var)`

where
- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

*The constraint `MAX_WIDTH_STEADY_SEQUENCE(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `STEADY_SEQUENCE` for which the feature value is `VALUE`.

**Purpose**
- The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.
- An occurrence of the pattern `STEADY_SEQUENCE` is the *maximal* subsequence which matches the regular expression `=+`.
- Assume that the occurrence of the pattern `STEADY_SEQUENCE` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j − i + 2`.

**Example**

Figure 3.1038 provides an example where the `POS_MAX_WIDTH_STEADY_SEQUENCE (3,[3,1,1,4,5,5,6,2,2,4,4,3,2,1,1], [0,0,0,0,1,0,0,0,0,0,0,0,0,0])` constraint holds.

**Typical**
- `|VARIABLES| > 1`

**Arg. properties**
- Functional dependency: `VALUE` determined by `VARIABLES`.
- Functional dependency: `FOUND` determined by `VARIABLES`.
Figure 3.1038: Illustrating the POS_MAX_WIDTH_STEADY_SEQUENCE constraint of the Example slot
Automaton

Similar to the automaton of the MAX_WIDTH_STEADY_SEQUENCE constraint but use the decoration table 2.33.
3.474 **POS_MAX_WIDTH.Strictly Decreasing Sequence**

**Description**

Based on constraint `MAX_WIDTH.Strictly Decreasing Sequence`.

**Constraint**

`POS_MAX_WIDTH.Strictly Decreasing Sequence(VALUE, VARIABLES, FOUND)`

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var-dvar)`
- **FOUND**: `collection(var-dvar)`

**Restrictions**

\[
\begin{align*}
\text{sv} \leq 1 & \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} = 0 & \lor \text{VALUE} \geq 2 \\
\text{VALUE} \leq \min(\text{sv}, \text{rv}) & \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{required}(\text{FOUND}, \text{var}) & \\
\text{where} & \\
\text{sv} = |\text{VARIABLES}| & \\
\text{rv} = \text{range}(\text{VARIABLES}.\text{var}) &
\end{align*}
\]

The constraint `MAX_WIDTH.Strictly Decreasing Sequence(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `Strictly Decreasing Sequence` for which the feature value is `VALUE`.

**Purpose**

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `Strictly Decreasing Sequence` is the *maximal* subsequence which matches the regular expression `>^+`.

Assume that the occurrence of the pattern `Strictly Decreasing Sequence` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j - i + 2`.

**Example**

\[
\left(\begin{array}{cccccccccccc}
3, & 4, & 4, & 6, & 4, & 1, & 1, & 3, & 4, & 6, & 6, & 5, & 2, & 2, & 4, & 3,\\
0, & 0, & 1, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0
\end{array}\right)
\]

Figure 3.1039 provides an example where the `POS_MAX_WIDTH.Strictly Decreasing Sequence` constraint holds.

**Typical**

\[|\text{VARIABLES}| > 1\]
\[\text{range}(\text{VARIABLES}.\text{var}) > 1\]

**Arg. properties**

- Functional dependency: `VALUE` determined by `VARIABLES`.
- Functional dependency: `FOUND` determined by `VARIABLES`. 
Figure 3.1039: Illustrating the POS_MAX_WIDTH_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Similar to the automaton of the MAX_WIDTH STRICTLY_DECREASING_SEQUENCE constraint but use the decoration table 2.33.
3.475  **POS_MAX_WIDTH STRICTLY_INCREASING_SEQUENCE**

**DESCRIPTION**

Based on constraint **MAX_WIDTH STRICTLY_INCREASING_SEQUENCE**.

**Constraint**

**POS_MAX_WIDTH STRICTLY_INCREASING_SEQUENCE**(VALUE, VARIABLES, FOUND)

**Arguments**

- VALUE: dvar
- VARIABLES: collection(var−dvar)
- FOUND: collection(var−dvar)

**Restrictions**

\[
\begin{align*}
sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \\
VALUE = 0 \lor VALUE \geq 2 \\
VALUE \leq \min(sv, rv) \\
required(VARIABLES, var) \\
required(FOUND, var)
\end{align*}
\]

where

\[
\begin{align*}
sv &= |VARIABLES| \\
rv &= \text{range}(VARIABLES.var)
\end{align*}
\]

**Purpose**

The constraint **MAX_WIDTH STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES)** holds. In addition, **FOUND** is a collection of 0/1 variables where the value 1 indicates the position of the **found** letter in those occurrences of the pattern **STRICTLY_INCREASING_SEQUENCE** for which the feature value is **VALUE**.

The position of the **found** letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern **STRICTLY_INCREASING_SEQUENCE** is the **maximal** subsequence which matches the regular expression \(<^+\).

Assume that the occurrence of the pattern **STRICTLY_INCREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **WIDTH** computes the value \(j - i + 2\).

**Example**

\[
\begin{pmatrix}
5, (4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3), \\
(0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0)
\end{pmatrix}
\]

Figure 3.1040 provides an example where the **POS_MAX_WIDTH STRICTLY_INCREASING_SEQUENCE** (5, [4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3], [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

**Typical**

\[
|VARIABLES| > 1 \\
range(VARIABLES.var) > 1
\]

**Arg. properties**

- **Functional dependency**: **VALUE** determined by **VARIABLES**.
- **Functional dependency**: **FOUND** determined by **VARIABLES**.
Figure 3.1040: Illustrating the \textit{POSMAX_WIDTH_STRICTLY_INCREASING_SEQUENCE} constraint of the \textbf{Example} slot
| Automaton | Similar to the automaton of the MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE constraint but use the decoration table 2.33. |
POS_MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE

2031
### 3.476 POS_MAX_WIDTH_SUMMIT

**Description**

Based on constraint `MAX_WIDTH_SUMMIT`.

**Constraint**

`POS_MAX_WIDTH_SUMMIT(VALUE, VARIABLES, FOUND)`

**Arguments**

- `VALUE` : `dvar`
- `VARIABLES` : `collection(var−dvar)`
- `FOUND` : `collection(var−dvar)`

**Restrictions**

- `sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = 0`
- `VALUE ≥ 0`
- `rv = 2 ⇒ VALUE ≤ 1` *required*(`VARIABLES.var`)
- `rv ≥ 3 ⇒ VALUE ≤ max(0, sv − 2)` *required*(`VARIABLES.var`, `FOUND.var`)

where

- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

The constraint `MAX_WIDTH_SUMMIT(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `SUMMIT` for which the feature value is `VALUE`. The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `SUMMIT` is the `maximal` subsequence which matches the regular expression `(< | < (= | <) | > (|= | >))`

Assume that the occurrence of the pattern `SUMMIT` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j − i`.

**Example**

```
( [ 3, 7, 1, 5, 4, 4, 3, 4, 6, 2, 3, 4, 2, 3, 1 ], [ 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 ] )
```

Figure 3.1041 provides an example where the `POS_MAX_WIDTH_SUMMIT` constraint holds.

**Typical**

- `|VARIABLES| > 2`
- `range(VARIABLES.var) > 1`

**Arg. properties**

- Functional dependency: `VALUE` determined by `VARIABLES`.
- Functional dependency: `FOUND` determined by `VARIABLES`.

---

**Notes**

[1] Typical properties are required to be satisfied in order for the constraint to be useful.

---

**Footnote**

[1] typical properties are required to be satisfied in order for the constraint to be useful.
Figure 3.1041: Illustrating the POS_MAX_WIDTH_SUMMIT constraint of the Example slot
Automaton

Similar to the automaton of the MAX_WIDTH_SUMMIT constraint but use the decoration table 2.33.
3.477 POS_MAX_WIDTH_VALLEY

**DESCRIPTION**

**Origin**
Based on constraint `MAX_WIDTH_VALLEY`.

**Constraint**

```plaintext
POS_MAX_WIDTH_VALLEY(VALUE, VARIABLES, FOUND)
```

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var−dvar)`
- **FOUND**: `collection(var−dvar)`

**Restrictions**

- `sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = 0`
- `VALUE ≥ 0`
- `VALUE ≤ \max(0, sv − 2v)`
- `required(VARIABLES, var)`
- `required(FOUND, var)`

where

- `sv = \|VARIABLES\|`
- `rv = \text{range}(VARIABLES.var)`

**Purpose**

The constraint `MAX_WIDTH_VALLEY(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `VALLEY` for which the feature value is `VALUE`. The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `VALLEY` is the `maximal` subsequence which matches the regular expression `> (= | >^*(< | =)^* <`.

Assume that the occurrence of the pattern `VALLEY` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j − i`.

**Example**

```
(4, (1, 3, 7, 4, 3, 6, 5, 3, 3, 2, 6, 5, 5, 7), (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0))
```

Figure 3.1042 provides an example where the `POS_MAX_WIDTH_VALLEY (4, [1, 3, 7, 4, 3, 6, 5, 3, 3, 2, 6, 5, 5, 7], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0])` constraint holds.

**Typical**

- `|VARIABLES| > 2`
- `\text{range}(VARIABLES.var) > 1`

**Arg. properties**

- Functional dependency: `VALUE` determined by `VARIABLES`.
- Functional dependency: `FOUND` determined by `VARIABLES`.
Figure 3.1042: Illustrating the POS_MAX_WIDTH_VALLEY constraint of the Example slot
Automaton

Similar to the automaton of the MAX_WIDTH_VALLEY constraint but use the decoration table 2.33.
POS_MAX_WIDTH_VALLEY

2039
### 3.478 POS_MAX_WIDTH_ZIGZAG

**Origin**

Based on constraint MAX_WIDTH_ZIGZAG.

**Constraint**

POS_MAX_WIDTH_ZIGZAG(VALUE, VARIABLES, FOUND)

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var–dvar)
- **FOUND**: collection(var–dvar)

**Restrictions**

\[ sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = 0 \]

\[ VALUE = 0 \lor VALUE \geq 2 \]

\[ VALUE \leq \max(0, sv - 2) \]

\[ \text{required}(\text{VARIABLES}, \text{var}) \]

\[ \text{required}(\text{FOUND}, \text{var}) \]

where

\[ sv = |\text{VARIABLES}| \]

\[ rv = \text{range}(\text{VARIABLES}.\text{var}) \]

The constraint MAX_WIDTH_ZIGZAG(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern ZIGZAG for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

**Purpose**

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression \((<>)^+ (< | <> ) | (<>)^+ (> | >> ))\).

Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature \(\text{WIDTH}\) computes the value \(j - i\).

**Example**

\[
\begin{pmatrix}
6, (4, 1, 3, 1, 4, 6, 1, 5, 2, 2, 3, 1, 6, 1), \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
\end{pmatrix}
\]

Figure 3.1043 provides an example where the POS_MAX_WIDTH_ZIGZAG (6, [4, 1, 3, 1, 4, 6, 1, 5, 2, 7, 2, 3, 1, 6, 1], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]) constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 3\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Figure 3.1043: Illustrating the POS_MAX_WIDTH_ZIGZAG constraint of the Example slot.
Automaton

Similar to the automaton of the \texttt{MAX\_WIDTH\_ZIGZAG} constraint but use the decoration table \texttt{2.33}.
3.479  POS_MIN_HEIGHT_DECREASING_TERRACE

**DESCRIPTION**

**Constraint**

Based on constraint MIN_HEIGHT_DECREASING_TERRACE.

**Arguments**

```
VALUE : dvar
VARIABLES : collection(var − dvar)
FOUND : collection(var − dvar)
```

**Restrictions**

```
sv ≤ 3 ∨ rv ≤ 2 ⇒ VALUE = +∞
VALUE ≥ minv + 1 ≤
VALUE = +∞ ∨ VALUE ≤ maxv − 1
required(VARIABLES, var)
required(FOUND, var)
```

where

```
minv = minval(VARIABLES.var)
maxv = maxval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)
```

The constraint MIN_HEIGHT_DECREASING_TERRACE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the **found** letter in those occurrences of the pattern **DECREASING_TERRACE** for which the feature value is VALUE.

The position of the **found** letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern **DECREASING_TERRACE** is the **maximal** subsequence which matches the regular expression `>^=^+>`. Assume that the occurrence of the pattern **DECREASING_TERRACE** starts at position `i` and ends at position `j`. The feature **MIN** computes the minimum of the values from index `i + 1` to index `j`.

**Example**

```
(2, [6, 4, 4, 5, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
```

Figure 3.1044 provides an example where the POS_MIN_HEIGHT_DECREASING_TERRACE (2, [6, 4, 4, 5, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

**Typical**

```
|VARIABLES| > 3
range(VARIABLES.var) > 2
```
Figure 3.1044: Illustrating the POS_MIN_HEIGHT_DECREASING_TERRACE constraint of the Example slot

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton  Similar to the automaton of the `MIN_HEIGHT_DECREASING_TERRACE` constraint but use the decoration table 2.33.
3.480  POS_MIN_HEIGHT_INCREASING_TERRACE

Description

Origin
Based on constraint MIN_HEIGHT_INCREASING_TERRACE.

Constraint
POS_MIN_HEIGHT_INCREASING_TERRACE(VALUE, VARIABLES, FOUND)

Arguments
VALUE: dvar
VARIABLES: collection(var−dvar)
FOUND: collection(var−dvar)

Restrictions
sv ≤ 3 ∨ rv ≤ 2 ⇒ VALUE = +∞
VALUE ≥ minv + 1
VALUE = +∞ ∨ VALUE ≤ maxv − 1
required(VARIABLES, var)
required(FOUND, var)
where
minv = minval(VARIABLES.var)
maxv = maxval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)

Purpose
The constraint MIN_HEIGHT_INCREASING_TERRACE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern INCREASING_TERRACE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.
An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression < = + <.
Assume that the occurrence of the pattern INCREASING_TERRACE starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i + 1 to index j.

Example

Figure 3.1045 provides an example where the POS_MIN_HEIGHT_INCREASING_TERRACE (3, [1, 3, 3, 2, 5, 6, 4, 2, 3, 3, 4, 4], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

Typical

|VARIABLES| > 3
range(VARIABLES.var) > 2


Figure 3.1045: Illustrating the POS_MIN_HEIGHT_INCREASING_TERRACE constraint of the Example slot

**Arg. properties**

- **Functional dependency**: VALUE determined by VARIABLES.
- **Functional dependency**: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MIN_HEIGHT_INCREASING_TERRACE} constraint but use the decoration table \texttt{2.33}.
POS_MIN_HEIGHT_INCREASING_TERRACE

2051
### 3.481  POS_MIN_HEIGHT_PLAIN

#### Origin
Based on constraint MIN_HEIGHT_PLAIN.

#### Constraint
POS_MIN_HEIGHT_PLAIN(VALUE, VARIABLES, FOUND)

#### Arguments
- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)
- **FOUND**: collection(var−dvar)

#### Restrictions
\[
sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = +\infty \\
VALUE \geq minv \lor \\
VALUE = +\infty \lor VALUE \leq maxv - 1 \\
required(VARIABLES, var) \\
required(FOUND, var)
\]

where
\[
minv = minval(VARIABLES.var) \\
maxv = maxval(VARIABLES.var) \\
sv = |VARIABLES| \\
rv = range(VARIABLES.var)
\]

The constraint MIN_HEIGHT_PLAIN(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern PLAIN for which the feature value is VALUE. The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression > =∗ <. Assume that the occurrence of the pattern PLAIN starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\).

#### Purpose

#### Example
\[
\left( \begin{array}{c}
3, (2, 3, 6, 5, 7, 6, 4, 5, 4, 3, 3, 6, 6, 3), \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0) 
\end{array} \right)
\]

Figure 3.1046 provides an example where the POS_MIN_HEIGHT_PLAIN (3, [2, 3, 6, 5, 7, 6, 4, 5, 4, 3, 3, 6, 6, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0]) constraint holds.

#### Typical
- \(|VARIABLES| > 2\)
- \(range(VARIABLES.var) > 1\)

#### Arg. properties
- **Functional dependency**: VALUE determined by VARIABLES.
- **Functional dependency**: FOUND determined by VARIABLES.
Figure 3.1046: Illustrating the POS_MIN_HEIGHT_PLAIN constraint of the Example slot
Automaton

Similar to the automaton of the MIN HEIGHT constraint but use the decoration table 2.33.
### 3.482 POS_MIN_HEIGHT_PLATEAU

**Description**

**Origin**

Based on constraint MIN_HEIGHT_PLATEAU.

**Constraint**

POS_MIN_HEIGHT_PLATEAU(VALUE, VARIABLES, FOUND)

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)
- **FOUND**: collection(var−dvar)

**Restrictions**

\[
\begin{align*}
sv &\leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} &\geq \text{minv} + 1 \\
\text{VALUE} &\geq +\infty \lor \text{VALUE} \leq \text{maxv} \\
\text{required}(\text{VARIABLES}.\text{var}) &\Rightarrow \text{required}(\text{FOUND}.\text{var})
\end{align*}
\]

where

- \(\text{minv} = \text{minval}(\text{VARIABLES}.\text{var})\)
- \(\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})\)
- \(\text{sv} = |\text{VARIABLES}|\)
- \(\text{rv} = \text{range}(\text{VARIABLES}.\text{var})\)

The constraint MIN_HEIGHT_PLATEAU(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern PLATEAU for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression \(<=*\>.

Assume that the occurrence of the pattern PLATEAU starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**

\[
\left(\begin{array}{c}
3, 7, 5, 2, 3, 1, 2, 2, 4, 3, 4, 5, 5, 2, 2, 5 \\
0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
\end{array}\right)
\]

Figure 3.1047 provides an example where the POS_MIN_HEIGHT_PLATEAU \((3, [7, 5, 2, 3, 1, 2, 2, 4, 3, 4, 5, 5, 2, 2, 5], [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])\) constraint holds.

**Purpose**

**Typical**

- \(|\text{VARIABLES}| > 2\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)
Figure 3.1047: Illustrating the POS_MIN_HEIGHT_PLATEAU constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_HEIGHT_PLATEAU constraint but use the decoration table 2.33.
3.483 POS_MIN_HEIGHT_PROPER Plain

**Description**

Based on constraint MIN_HEIGHT_PROPERPLAIN.

**Constraint**

\[
\text{POS_MIN_HEIGHT_PROPERPLAIN}({\text{VALUE, VARIABLES, FOUND}})
\]

**Arguments**

- \(\text{VALUE} : \text{dvar}\)
- \(\text{VARIABLES} : \text{collection(var-dvar)}\)
- \(\text{FOUND} : \text{collection(var-dvar)}\)

**Restrictions**

\[
\begin{align*}
\text{sv} \leq 3 & \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} \geq \minv & \\
\text{VALUE} = +\infty & \lor \text{VALUE} \leq \maxv - 1 \\
\text{required} & (\text{VARIABLES}, \text{var}) \\
\text{required} & (\text{FOUND}, \text{var})
\end{align*}
\]

where

- \(\minv = \text{minval} (\text{VARIABLES}.\text{var})\)
- \(\maxv = \text{maxval} (\text{VARIABLES}.\text{var})\)
- \(\text{sv} = |\text{VARIABLES}|\)
- \(\text{rv} = \text{range} (\text{VARIABLES}.\text{var})\)

The constraint \(\text{MIN_HEIGHT_PROPERPLAIN}(\text{VALUE, VARIABLES})\) holds. In addition, \(\text{FOUND}\) is a collection of 0/1 variables where the value 1 indicates the position of the \(\text{found}\) letter in those occurrences of the pattern \(\text{PROPERPLAIN}\) for which the feature value is \(\text{VALUE}\).

The position of the \(\text{found}\) letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \(\text{PROPERPLAIN}\) is the maximal subsequence which matches the regular expression \(> = + <\).

Assume that the occurrence of the pattern \(\text{PROPERPLAIN}\) starts at position \(i\) and ends at position \(j\). The feature \(\text{MIN}\) computes the minimum of the values from index \(i + 1\) to index \(j\).

**Purpose**

**Example**

Figure 3.1048 provides an example where the \(\text{POS_MIN_HEIGHT_PROPERPLAIN}\)

\[
(3, 2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3, 5), \\
\langle 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0 \rangle
\]

\(\text{constraint holds.}\)

**Typical**

\(|\text{VARIABLES}| > 3 \]

\(\text{range}(\text{VARIABLES}.\text{var}) > 1\)
Figure 3.1048: Illustrating the POS_MIN_HEIGHT_PROPER.PLAIN constraint of the Example slot

Arg. properties
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_HEIGHT_PROPER.PLAIN constraint but use the decoration table 2.33.
3.484 POS_MIN_HEIGHT_PROPER_PLATEAU

 Origin
 Based on constraint MIN_HEIGHT_PROPER_PLATEAU.

 Constraint
 POS_MIN_HEIGHT_PROPER_PLATEAU(VALUE, VARIABLES, FOUND)

 Arguments
 VALUE : dvar
 VARIABLES : collection(var-dvar)
 FOUND : collection(var-dvar)

 Restrictions
 sv ≤ 3 ∨ rv ≤ 1 ⇒ VALUE = +∞
 VALUE ≥ minv + 10
 VALUE = +∞ ∨ VALUE ≤ maxv
 required(VARIABLES, var)
 required(FOUND, var)
 where
 minv = minval(VARIABLES.var)
 maxv = maxval(VARIABLES.var)
 sv = |VARIABLES|
 rv = range(VARIABLES.var)

 Purpose
 The constraint MIN_HEIGHT_PROPER_PLATEAU(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern PROPER_PLATEAU for which the feature value is VALUE.

 The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

 An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression < =+ >.

 Assume that the occurrence of the pattern PROPER_PLATEAU starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i + 1 to index j.

 Example
 (3, [7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3], [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])

 Figure 3.1049 provides an example where the POS_MIN_HEIGHT_PROPER_PLATEAU (3, [7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3], [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

 Typical
 |VARIABLES| > 3
 range(VARIABLES.var) > 1
Figure 3.1049: Illustrating the POS_MIN_HEIGHT_PROPER_PLATEAU constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the `MIN_HEIGHT_PROPER_PLATEAU` constraint but use the decoration table 2.33.
3.485 POS_MIN_HEIGHT_STEADY

**Origin**
Based on constraint MIN_HEIGHT_STEADY.

**Constraint**

```
POS_MIN_HEIGHT_STEADY(VALUE, VARIABLES, FOUND)
```

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)
- **FOUND**: collection(var−dvar)

**Restrictions**

```
sv ≤ 1 ⇒ VALUE = +∞
VALUE ≥ minv
VALUE = +∞ V VALUE ≤ maxv
required(VARIABLES, var)
required(FOUND, var)
```

where

- `minv = minval(VARIABLES.var)`
- `maxv = maxval(VARIABLES.var)`
- `sv = |VARIABLES|`

The constraint `MIN_HEIGHT_STEADY(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `STEADY` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `STEADY` is the subsequence which matches the regular expression `= `. Assume that the occurrence of the pattern `STEADY` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i` to index `j + 1`.

**Example**

```
( 1, (1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6) ,
 (1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
)
```

Figure 3.1050 provides an example where the `POS_MIN_HEIGHT_STEADY` constraint holds.

**Typical**
```
|VARIABLES| > 1
```

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Figure 3.1050: Illustrating the POS_MIN_HEIGHT_STEADY constraint of the Example slot
Automaton

Similar to the automaton of the \texttt{MIN\_HEIGHT\_STEADY} constraint but use the decoration table \texttt{2.33}.
### 3.466 **POS_MIN_HEIGHT_STEADY_SEQUENCE**

#### Description

Based on constraint `MIN_HEIGHT_STEADY_SEQUENCE`.

#### Automaton

<table>
<thead>
<tr>
<th>Constraint</th>
<th>POS_MIN_HEIGHT_STEADY_SEQUENCE(VALUE, VARIABLES, FOUND)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arguments</td>
<td>VALUE : dvar, VARIABLES : collection(var−dvar), FOUND : collection(var−dvar)</td>
</tr>
</tbody>
</table>
| Restrictions | sv ≤ 1 ⇒ VALUE = +∞  
VALUE ≥ minv  
VALUE = +∞ ∨ VALUE ≤ maxv  
required(VARIABLES, var)  
required(FOUND, var)  
where  
minv = minval(VARIABLES, var)  
maxv = maxval(VARIABLES, var)  
sv = |VARIABLES| |

The constraint `MIN_HEIGHT_STEADY_SEQUENCE(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `STEADY_SEQUENCE` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `STEADY_SEQUENCE` is the *maximal* subsequence which matches the regular expression `^+`.

Assume that the occurrence of the pattern `STEADY_SEQUENCE` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i` to index `j + 1`.

#### Example

\[
\begin{pmatrix}
1, (3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1),
\{0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0}\,
\end{pmatrix}
\]

Figure 3.1051 provides an example where the `POS_MIN_HEIGHT_STEADY_SEQUENCE` (1, [3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1], [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]) constraint holds.

#### Typical

| VARIABLES | > 1 |

#### Arg. properties

- **Functional dependency**: `VALUE` determined by `VARIABLES`.
- **Functional dependency**: `FOUND` determined by `VARIABLES`. 
Figure 3.1051: Illustrating the POS\_MIN\_HEIGHT\_STEADY\_SEQUENCE constraint of the Example slot
Automaton

Similar to the automaton of the MIN_HEIGHT_STEADY_SEQUENCE constraint but use the decoration table 2.33.
3.487  POS_MIN_MAX_BUMP_ON_DECREASING_SEQUENCE

**Description**

Based on constraint MIN_MAX_BUMP_ON_DECREASING_SEQUENCE.

**Constraint**

POS_MIN_MAX_BUMP_ON_DECREASING_SEQUENCE(VALUE, VARIABLES, FOUND)

**Arguments**

- VALUE : dvar
- VARIABLES : collection(var-dvar)
- FOUND : collection(var-dvar)

**Restrictions**

\[ sv \leq 5 \lor rv \leq 2 \Rightarrow VALUE = +\infty \]
\[ VALUE \geq minv + 2 \]
\[ VALUE = +\infty \lor VALUE \leq maxv \]
\[ required(VARIABLES, var) \]
\[ required(FOUND, var) \]

where
- \( minv = minval(VARIABLES.var) \)
- \( maxv = maxval(VARIABLES.var) \)
- \( sv = |VARIABLES| \)
- \( rv = range(VARIABLES.var) \)

The constraint MIN_MAX_BUMP_ON_DECREASING_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern BUMP_ON_DECREASING_SEQUENCE for which the feature value is VALUE. The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

**Purpose**

An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence which matches the regular expression >>><>. Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature MAX computes the maximum of the values from index \( i + 2 \) to index \( j \).

**Example**

\[
\begin{pmatrix}
5, (7, 6, 5, 6, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3), \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0
\end{pmatrix}
\]

Figure 3.1052 provides an example where the POS_MIN_MAX_BUMP_ON_DECREASING_SEQUENCE (5, [7, 6, 5, 6, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]) constraint holds.

**Typical**

\[ |VARIABLES| > 5 \]
\[ range(VARIABLES.var) > 2 \]
Figure 3.1052: Illustrating the **POS_MIN_MAX_BUMP_ON_DECREASING_SEQUENCE** constraint of the **Example** slot

Arg. properties

- **Functional dependency**: VALUE determined by VARIABLES.
- **Functional dependency**: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_MAX_BUMP_ON_DECREASING_SEQUENCE constraint but use the decoration table 2.33.
3.488 POS_MIN_MAX_DECREASING

**Description**

Based on constraint MIN_MAX_DECREASING.

**Constraint**

POS_MIN_MAX_DECREASING(VALUE, VARIABLES, FOUND)

**Arguments**

\[
\begin{align*}
\text{VALUE} &: \ dvar \\
\text{VARIABLES} &: \ \text{collection}(\text{var} - \ dvar) \\
\text{FOUND} &: \ \text{collection}(\text{var} - \ dvar)
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
sv \leq 1 \lor rv \leq 1 & \Rightarrow VALUE = +\infty \\
VALUE \geq \minv + 1 & \\
VALUE = +\infty \lor VALUE \leq \maxv & \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{required}(\text{FOUND}, \text{var}) & \\
\text{where} & \\
\minv &= \text{minval}(\text{VARIABLES}.\text{var}) \\
\maxv &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
sv &= |\text{VARIABLES}| \\
rv &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

The constraint MIN_MAX_DECREASING(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern DECREASING for which the feature value is VALUE. The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern DECREASING is the subsequence which matches the regular expression >. Assume that the occurrence of the pattern DECREASING starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i\) to index \(j + 1\).

**Purpose**

An occurrence of the pattern DECREASING holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern DECREASING for which the feature value is VALUE.

**Example**

\[
\begin{pmatrix}
3, (3, 4, 2, 2, 5, 6, 4, 4, 3, 1, 4, 6, 4, 4), \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
\end{pmatrix}
\]

Figure 3.1053 provides an example where the POS_MIN_MAX_DECREASING constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
POS_MIN_MAX_DECREASING

Figure 3.1053: Illustrating the POS_MIN_MAX_DECREASING constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_MAX_DECREASING constraint but use the decoration table 2.33.
POS_MIN_MAX_DECREASING

2083
3.489 POS_MIN_MAX_DECREASING_SEQUENCE

**DESCRIPTION**

**Origin**
Based on constraint MIN_MAX_DECREASING_SEQUENCE.

**Constraint**
POS_MIN_MAX_DECREASING_SEQUENCE(VALUE, VARIABLES, FOUND)

**Arguments**

```
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)
```

**Restrictions**

\[
sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty
\]
\[
VALUE \geq minv + 1 \lor
\]
\[
VALUE = +\infty \lor VALUE \leq maxv
\]
\[
required(VARIABLES, var)
\]
\[
required(FOUND, var)
\]

where

\[
minv = minval(VARIABLES.var)
\]
\[
maxv = maxval(VARIABLES.var)
\]
\[
sv = |VARIABLES|
\]
\[
rv = range(VARIABLES.var)
\]

The constraint MIN_MAX_DECREASING_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern DECREASING_SEQUENCE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression \( (> | =)^* > | > \).

Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature MAX computes the maximum of the values from index \( i \) to index \( j + 1 \).

**Example**

\[
(4, (3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 4, 6, 4, 4), (0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0))
\]

Figure 3.1054 provides an example where the POS_MIN_MAX_DECREASING_SEQUENCE \((4, [3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 4, 6, 4, 4], [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])\) constraint holds.

**Purpose**

**Typical**

\[
|VARIABLES| > 1
\]
\[
range(VARIABLES.var) > 1
\]
Figure 3.1054: Illustrating the POS_MIN_MAX_DECREASING_SEQUENCE constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MIN_MAX_DECREASING_SEQUENCE} constraint but use the decoration table \ref{2.33}. 
### 3.490 POS_MIN_MAX_DIP_ON_INCREASING_SEQUENCE

**Description**

**Origin**
Based on constraint MIN_MAX_DIP_ON_INCREASING_SEQUENCE.

**Constraint**
POS_MIN_MAX_DIP_ON_INCREASING_SEQUENCE(VALUE, VARIABLES, FOUND)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var-dvar)
- FOUND : collection(var-dvar)

**Restrictions**
- \( sv \leq 5 \lor rv \leq 2 \Rightarrow VALUE = +\infty \)
- \( VALUE \geq minv + 2 \)
- \( VALUE = +\infty \lor VALUE \leq maxv \)
- required(VARIABLES.var)
- required(FOUND, var)
  
  where
  - \( minv = minval(VARIABLES.var) \)
  - \( maxv = maxval(VARIABLES.var) \)
  - \( sv = |VARIABLES| \)
  - \( rv = range(VARIABLES.var) \)

**Purpose**

The constraint MIN_MAX_DIP_ON_INCREASING_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern DIP_ON_INCREASING_SEQUENCE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence which matches the regular expression \(<<<><\). Assume that the occurrence of the pattern DIP_ON_INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature \(max\) computes the maximum of the values from index \(i + 2\) to index \(j\).

**Example**

\[
\begin{pmatrix}
5, \{1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4\}, \\
\{0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\}
\end{pmatrix}
\]

Figure 3.1055 provides an example where the POS_MIN_MAX_DIP_ON_INCREASING_SEQUENCE (5, \([1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4]\), \([0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]\)) constraint holds.

**Typical**

\(|VARIABLES| \geq 5\)

range(VARIABLES.var) \geq 2
Figure 3.1055: Illustrating the POS_MIN_MAX_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton  Similar to the automaton of the \texttt{MIN\_MAX\_DIP\_ON\_INCREASING\_SEQUENCE} constraint but use the decoration table \texttt{2.33}.
3.491 POS_MIN_MAX_INCREASING

**DESCRIPTION**

**Origin**
Based on constraint MIN_MAX_INCREASING.

**Constraint**
POS_MIN_MAX_INCREASING(VALUE, VARIABLES, FOUND)

**Arguments**
- `VALUE` : dvar
- `VARIABLES` : collection(var−dvar)
- `FOUND` : collection(var−dvar)

**Restrictions**
- `sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = +∞`
- `VALUE ≥ minv + 1`
- `VALUE = +∞ ∨ VALUE ≤ maxv`
- `required(VARIABLES, var)`
- `required(FOUND, var)`

where
- `minv = minval(VARIABLES.var)`
- `maxv = maxval(VARIABLES.var)`
- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

The constraint MIN_MAX_INCREASING(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the **found** letter in those occurrences of the pattern INCREASING for which the feature value is **VALUE**.

The position of the **found** letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern INCREASING is the subsequence which matches the regular expression `<`.

Assume that the occurrence of the pattern INCREASING starts at position `i` and ends at position `j`. The feature MAX computes the maximum of the values from index `i` to index `j + 1`.

**Example**

```
( 3, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3],
  [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0])
```

Figure 3.1056 provides an example where the POS_MIN_MAX_INCREASING (3, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3], [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0]) constraint holds.

**Purpose**

**Typical**
- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`
Figure 3.1056: Illustrating the POS_MIN_MAX_INCREASING constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MIN\_MAX\_INCREASING} constraint but use the decoration table \texttt{2.33}. 
3.492 **POS_MIN_MAX_INCREASING_SEQUENCE**

**Description**

Based on constraint **MIN_MAX_INCREASING_SEQUENCE**.

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td><code>dvar</code></td>
</tr>
<tr>
<td>VARIABLES</td>
<td><code>collection(var−dvar)</code></td>
</tr>
<tr>
<td>FOUND</td>
<td><code>collection(var−dvar)</code></td>
</tr>
</tbody>
</table>

**Restrictions**

\[
sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty \\
VALUE \geq minv + 1 \lor \\
VALUE = +\infty \lor VALUE \leq maxv \\
required(VARIABLES, var) \\
required(FOUND, var)
\]

where

\[
minv = minval(VARIABLES.var) \\
maxv = maxval(VARIABLES.var) \\
sv = |VARIABLES| \\
rv = range(VARIABLES.var)
\]

The constraint **MIN_MAX_INCREASING_SEQUENCE(VALUE, VARIABLES)** holds. In addition, **FOUND** is a collection of 0/1 variables where the value 1 indicates the position of the **found** letter in those occurrences of the pattern **INCREASING_SEQUENCE** for which the feature value is **VALUE**.

The position of the **found** letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern **INCREASING_SEQUENCE** is the maximal subsequence which matches the regular expression \(< (< | =)^* | <\). Assume that the occurrence of the pattern **INCREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **MAX** computes the maximum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
(3, 4, 3, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3), (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0)
\]

Figure 3.1057 provides an example where the **POS_MIN_MAX_INCREASING_SEQUENCE (3, [4, 3, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0])** constraint holds.

**Typical**

| `|VARIABLES|` > 1 \\
| `range(VARIABLES.var)` > 1 |
Figure 3.1057: Illustrating the POS_MIN_MAX_INCREASING_SEQUENCE constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MIN\_MAX\_INCREASING\_SEQUENCE} constraint but use the decoration table 2.33.
3.493  POS_MIN_MAX_INFLEXION

Origin  Based on constraint MIN_MAX_INFLEXION.

Constraint  POS_MIN_MAX_INFLEXION(VALUE, VARIABLES, FOUND)

Arguments

\[
\begin{align*}
\text{VALUE} &: \text{dvar} \\
\text{VARIABLES} &: \text{collection} (\text{var} - \text{dvar}) \\
\text{FOUND} &: \text{collection} (\text{var} - \text{dvar})
\end{align*}
\]

Restrictions

\[
\begin{align*}
sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = +\infty \\
VALUE \geq \text{minv} \\
VALUE = +\infty \lor VALUE \leq \text{maxv} \\
\text{required}(\text{VARIABLES}, \text{var}) \\
\text{required}(\text{FOUND}, \text{var})
\end{align*}
\]

where

\[
\begin{align*}
\text{minv} &= \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{maxv} &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
sv &= |\text{VARIABLES}| \\
rv &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

Purpose

The constraint MIN_MAX_INFLEXION(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the \textit{found} letter in those occurrences of the pattern \textit{INFLEXION} for which the feature value is \textit{VALUE}.

The position of the \textit{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \textit{INFLEXION} is the \textit{maximal} subsequence which matches the regular expression \(< (\langle \mid \rangle)^* \rangle > (\rangle \mid \rangle)^* < \).

Assume that the occurrence of the pattern \textit{INFLEXION} starts at position \(i\) and ends at position \(j\). The feature \textit{MAX} computes the maximum of the values from index \(i + 1\) to index \(j\).

Example

\[
\begin{align*}
(1, \{1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4\}, \{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0\})
\end{align*}
\]

Figure 3.1058 provides an example where the POS_MIN_MAX_INFLEXION (1, \([1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4]\), \([0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0]\)) constraint holds.

Typical

\[
|\text{VARIABLES}| \geq 2 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
Figure 3.1058: Illustrating the POS_MIN_MAX_INFLEXION constraint of the Example slot

**Arg. properties**

- **Functional dependency**: VALUE determined by VARIABLES.
- **Functional dependency**: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_MAX_INFLEXION constraint but use the decoration table 2.33.
### 3.494 POS_MIN_MAX_PEAK

#### DESCRIPTION

**Origin**
Based on constraint MIN_MAX_PEAK.

**Constraint**

\[
\text{POS_MIN_MAX_PEAK}(\text{VALUE}, \text{VARIABLES}, \text{FOUND})
\]

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)
- **FOUND**: collection(var−dvar)

**Restrictions**

- \(sv \leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = +\infty\)
- \(\text{VALUE} \geq \minv + 1\)
- \(\text{VALUE} = +\infty \lor \text{VALUE} \leq \maxv\)
- \(\text{required}(\text{VARIABLES}, \text{var})\)
- \(\text{required}(\text{FOUND}, \text{var})\)

where
- \(\minv = \text{minval}(\text{VARIABLES}.\text{var})\)
- \(\maxv = \text{maxval}(\text{VARIABLES}.\text{var})\)
- \(sv = |\text{VARIABLES}|\)
- \(rv = \text{range}(\text{VARIABLES}.\text{var})\)

#### Purpose

The constraint MIN_MAX_PEAK(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern PEAK for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression \(< (= | <)^* (> | =)^* >\).

Assume that the occurrence of the pattern PEAK starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i + 1\) to index \(j\).

#### Example

\[
\left(3, \langle 7, 5, 5, 1, 4, 5, 2, 3, 5, 6, 2, 3, 3, 1 \rangle, \langle 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0 \rangle \right)
\]

Figure 3.1059 provides an example where the POS_MIN_MAX_PEAK (3, [7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 1], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]) constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 2\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Figure 3.1059: Illustrating the POS_MIN_MAX_PEAK constraint of the Example slot
Automaton

Similar to the automaton of the MIN_MAX.Peak constraint but use the decoration table \texttt{2.33}.
POS_MIN_MAX_PEAK

2107
3.495 POS_MIN_MAX_STRICTLY_DECREASING_SEQUENCE

**DESCRIPTION**

Based on constraint **MIN_MAX_STRICTLY_DECREASING_SEQUENCE**.

**Constraint**

\[
\text{POS_MIN_MAX_STRICTLY_DECREASING_SEQUENCE}(\text{VALUE}, \text{VARIABLES}, \text{FOUND})
\]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} - \text{dvar}) \\
\text{FOUND} & : \text{collection}(\text{var} - \text{dvar})
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
sv & \leq 1 \lor rv \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} & \geq minv + 1 \\
\text{VALUE} & = +\infty \lor \text{VALUE} \leq maxv \\
\text{required}(\text{VARIABLES}, \text{var}) \\
\text{required}(\text{FOUND}, \text{var}) \\
\text{where} & \\
minv & = \text{minval}(\text{VARIABLES}.\text{var}) \\
maxv & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
sv & = |\text{VARIABLES}| \\
rv & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

The constraint **MIN_MAX_STRICTLY_DECREASING_SEQUENCE(VALUE, VARIABLES)** holds. In addition, **FOUND** is a collection of 0/1 variables where the value 1 indicates the position of the **found** letter in those occurrences of the pattern **STRICTLY_DECREASING_SEQUENCE** for which the feature value is **VALUE**.

The position of the **found** letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

**Purpose**

An occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** is the **maximal** subsequence which matches the regular expression \(>^+\).

Assume that the occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **MAX** computes the maximum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
\left(\begin{array}{c}
4, 4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3 \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
\end{array}\right)
\]

Figure 3.1060 provides an example where the **POS_MIN_MAX_STRICTLY_DECREASING_SEQUENCE** constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
Figure 3.1060: Illustrating the POS\_MIN\_MAX\_STRICKLY\_DECREASING\_SEQUENCE constraint of the Example slot

Arg. properties
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_MAX STRICTLY DECREASING SEQUENCE constraint but use the decoration table 2.33.
3.496 POS_MIN_MAX_STRICTLY_INCREASING_SEQUENCE

**Description**

Based on constraint MIN_MAX_STRICTLY_INCREASING_SEQUENCE.

**Constraint**

POS_MIN_MAX_STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES, FOUND)

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>dvar</td>
</tr>
<tr>
<td>VARIABLES</td>
<td>collection(var-dvar)</td>
</tr>
<tr>
<td>FOUND</td>
<td>collection(var-dvar)</td>
</tr>
</tbody>
</table>

**Restrictions**

\[
sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty \\
VALUE \geq minv + 1 \\
VALUE = +\infty \lor VALUE \leq maxv \\
required(VARIABLES, var) \\
required(FOUND, var) \\
where \\
minv = minval(VARIABLES.var) \\
maxv = maxval(VARIABLES.var) \\
sv = |VARIABLES| \\
rv = range(VARIABLES.var)
\]

The constraint MIN_MAX_STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern STRICTLY_INCREASING_SEQUENCE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression \(<+\).

Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i\) to index \(j+1\).

**Example**

\[
(3, 4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0)
\]

Figure 3.1061 provides an example where the POS_MIN_MAX_STRICTLY_INCREASING_SEQUENCE (3, [4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]) constraint holds.

**Typical**

\|
\]

<table>
<thead>
<tr>
<th>VARIABLES</th>
<th>(&gt; 1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>range(VARIABLES.var)</td>
<td>(&gt; 1)</td>
</tr>
</tbody>
</table>
Figure 3.1061: Illustrating the POS_MIN_MAX.Strictly_Increasing_Sequence constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_MAX_STRICTLY_INCREASING_SEQUENCE constraint but use the decoration table 2.33.
3.497 POS_MIN_MAX_SUMMIT

Description

Based on constraint MIN_MAX_SUMMIT.

Constraint

POS_MIN_MAX_SUMMIT(VALUE, VARIABLES, FOUND)

Arguments

VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions

sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = +∞
VALUE ≥ minv + 1
VALUE = +∞ ∨ VALUE ≤ maxv
required(VARIABLES, var)
required(FOUND, var)
where
minv = minval(VARIABLES.var)
maxv = maxval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)

Purpose

The constraint MIN_MAX_SUMMIT(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern SUMMIT for which the feature value is VALUE. The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression (< | < (= | <)* <)|(>| (> (= | >)* )>.
Assume that the occurrence of the pattern SUMMIT starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i + 1 to index j.

Example

\[
\begin{pmatrix}
3, \{7, 1, 5, 4, 3, 3, 4, 6, 2, 3, 4, 2, 3, 1\},
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0
\end{pmatrix}
\]

Figure 3.1062 provides an example where the POS_MIN_MAX_SUMMIT (3, [7, 1, 5, 4, 3, 3, 4, 6, 2, 3, 4, 2, 3, 1], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]) constraint holds.

Typical

|VARIABLES| > 2
range(VARIABLES.var) > 1
Figure 3.1062: Illustrating the POS_MIN_MAX_SUMMIT constraint of the Example slot

Arg. properties
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_MAX_SUMMIT constraint but use the decoration table 2.33.
### 3.498  **POS_MIN_MAX_ZIGZAG**

**DESCRIPTION**

Based on constraint \( \text{MIN_MAX_ZIGZAG} \).

**AUTOMATON**

\[
(<>)^* (<> | <> ) | (<>)^* (> | <>)
\]

**Origin**

Based on constraint \( \text{MIN_MAX_ZIGZAG} \).

**Constraint**

\[
\text{POS_MIN_MAX_ZIGZAG}(\text{VALUE}, \text{VARIABLES}, \text{FOUND})
\]

**Arguments**

- **VALUE** : \( \text{dvar} \)
- **VARIABLES** : collection(var-dvar)
- **FOUND** : collection(var-dvar)

**Restrictions**

\[
\text{sv} \leq 3 \lor \text{rv} \leq 1 \implies \text{VALUE} = +\infty \\
\text{VALUE} \geq \text{minv} + 1 \\
\text{VALUE} = +\infty \lor \text{VALUE} \leq \text{maxv} \\
\text{required}(\text{VARIABLES}, \text{var}) \\
\text{required}(\text{FOUND}, \text{var})
\]

where

\[
\text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \\
\text{sv} = |\text{VARIABLES}| \\
\text{rv} = \text{range}(\text{VARIABLES}.\text{var})
\]

The constraint \( \text{MIN_MAX_ZIGZAG}(\text{VALUE}, \text{VARIABLES}) \) holds. In addition, \( \text{FOUND} \) is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern ZIGZAG for which the feature value is VALUE. The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression \((<>)^* (< | <>) | (<>)^* (> | <>)\).

Assume that the occurrence of the pattern ZIGZAG starts at position \( i \) and ends at position \( j \). The feature MAX computes the maximum of the values from index \( i + 1 \) to index \( j \).

**Purpose**

**Example**

\[
\left( 3, (4, 1, 3, 1, 4, 6, 1, 5, 2, 7, 2, 3, 1, 6, 1), (0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \right)
\]

Figure 3.1063 provides an example where the \( \text{POS_MIN_MAX_ZIGZAG} \)

(3, [4, 1, 3, 1, 4, 6, 1, 5, 2, 7, 2, 3, 1, 6, 1], [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])

constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 3\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Arg. properties**

- **Functional dependency**: \( \text{VALUE} \) determined by \( \text{VARIABLES} \).
- **Functional dependency**: \( \text{FOUND} \) determined by \( \text{VARIABLES} \).
Figure 3.1063: Illustrating the POS_MIN_MAX_ZIGZAG constraint of the Example slot
Automaton

Similar to the automaton of the MIN_MAX_ZIGZAG constraint but use the decoration table 2.33.
3.499 POS_MIN_MIN_BUMP_ON_DECREASING_SEQUENCE

**Description**

Based on constraint MIN_MIN_BUMP_ON_DECREASING_SEQUENCE.

**Constraint**

POS_MIN_MIN_BUMP_ON_DECREASING_SEQUENCE(VALUE, VARIABLES, FOUND)

**Arguments**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>: dvar</td>
</tr>
<tr>
<td>VARIABLES</td>
<td>: collection(var−dvar)</td>
</tr>
<tr>
<td>FOUND</td>
<td>: collection(var−dvar)</td>
</tr>
</tbody>
</table>

**Restrictions**

\[
\begin{align*}
sv & \leq 5 \lor rv \leq 2 \Rightarrow VALUE = +\infty \\
VALUE & \geq minv \\
VALUE & = +\infty \lor VALUE \leq maxv - 2 \\
required & (VARIABLES, var) \\
required & (FOUND, var)
\end{align*}
\]

where

\[
\begin{align*}
minv & = \text{minval}(\text{VARIABLES}.\text{var}) \\
maxv & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
sv & = |\text{VARIABLES}| \\
rv & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

The constraint MIN_MIN_BUMP_ON_DECREASING_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern BUMP_ON_DECREASING_SEQUENCE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence which matches the regular expression >>><>.

Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 2\) to index \(j\).

**Example**

\[
\begin{pmatrix}
2, (7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3), \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0)
\end{pmatrix}
\]

Figure 3.1064 provides an example where the POS_MIN_MIN_BUMP_ON_DECREASING_SEQUENCE (2, [7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 5 \\
\text{range}(\text{VARIABLES}.\text{var}) > 2
\]
Figure 3.1064: Illustrating the POS_MIN_MIN_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the `MIN_MIN_BUMP_ON_DECREASING_SEQUENCE` constraint but use the decoration table 2.33.
3.500  POS_MIN_MIN_DECREASING

Description

Based on constraint MIN_MIN_DECREASING.

Constraint

POS_MIN_MIN_DECREASING(VALUE, VARIABLES, FOUND)

Arguments

VALUE : dvar
VARIABLES : collection(var − dvar)
FOUND : collection(var − dvar)

Restrictions

sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty
VALUE \geq minv
VALUE = +\infty \lor VALUE \leq maxv - 1
required(VARIABLES, var)
required(FOUND, var)

where

minv = minval(VARIABLES.var)
maxv = maxval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)

Purpose

The constraint MIN_MIN_DECREASING(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern DECREASING for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern DECREASING is the subsequence which matches the regular expression >.

Assume that the occurrence of the pattern DECREASING starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i to index j + 1.

Example

Figure 3.1065 provides an example where the POS_MIN_MIN_DECREASING (1, [3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4], [0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0]) constraint holds.

Typical

|VARIABLES| > 1
range(VARIABLES.var) > 1
Figure 3.1065: Illustrating the POS_MIN_MIN_DECREASING constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_MIN_DECREASING constraint but use the decoration table 2.33.
### 3.501 POS_MIN_MIN_DECREASING_SEQUENCE

**DESCRIPTION**

**Origin**
Based on constraint MIN_MIN_DECREASING_SEQUENCE.

**Constraint**
POS_MIN_MIN_DECREASING_SEQUENCE(VALUE, VARIABLES, FOUND)

**Arguments**
- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)
- **FOUND**: collection(var−dvar)

**Restrictions**
\[
sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty \\
VALUE \geq \text{minv} \\
VALUE = +\infty \lor VALUE \leq \text{maxv} - 1 \\
\text{required}(\text{VARIABLES}, \text{var}) \\
\text{required}(\text{FOUND}, \text{var})
\]

where
- \( \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \)
- \( \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \)
- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

The constraint MIN_MIN_DECREASING_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern DECREASING_SEQUENCE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression \(>(>|=)^*>|\).

Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature MIN computes the minimum of the values from index \( i \) to index \( j + 1 \).

**Purpose**

**Example**

\[
\begin{pmatrix}
1, 3, 4, 2, 5, 6, 4, 4, 3, 1, 4, 6, 4, 4 \\
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 3.1066 provides an example where the POS_MIN_MIN_DECREASING_SEQUENCE constraint holds.

**Typical**

\( |\text{VARIABLES}| > 1 \)
\( \text{range}(\text{VARIABLES}.\text{var}) > 1 \)
Figure 3.1066: Illustrating the POS_MIN_MIN_DECREASING_SEQUENCE constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_MIN_DECREASING_SEQUENCE constraint but use the decoration table 2.33.
3.502 POS_MIN_MIN_DIP_ON_INCREASING_SEQUENCE

**Description**

Based on constraint MIN_MIN_DIP_ON_INCREASING_SEQUENCE.

**Constraint**

POS_MIN_MIN_DIP_ON_INCREASING_SEQUENCE(VALUE, VARIABLES, FOUND)

**Arguments**

- VALUE: dvar
- VARIABLES: collection(var–dvar)
- FOUND: collection(var–dvar)

**Restrictions**

\[ sv \leq 5 \lor rv \leq 2 \Rightarrow VALUE = +\infty \]

\[ VALUE \geq \text{minval} \]

\[ VALUE = +\infty \lor VALUE \leq \text{maxval} - 2 \]

\[ \text{required}(\text{VARIABLES}, \text{var}) \]

\[ \text{required}(\text{FOUND}, \text{var}) \]

where

\[ \text{minval} = \text{minval}(\text{VARIABLES}.\text{var}) \]

\[ \text{maxval} = \text{maxval}(\text{VARIABLES}.\text{var}) \]

\[ sv = |\text{VARIABLES}| \]

\[ rv = \text{range}(\text{VARIABLES}.\text{var}) \]

The constraint MIN_MIN_DIP_ON_INCREASING_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern DIP_ON_INCREASING_SEQUENCE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence which matches the regular expression <<><<<.

Assume that the occurrence of the pattern DIP_ON_INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 2\) to index \(j\).

**Example**

\[
\begin{pmatrix}
1, (1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4), \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0
\end{pmatrix}
\]

Figure 3.1067 provides an example where the POS_MIN_MIN_DIP_ON_INCREASING_SEQUENCE (1, [1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]) constraint holds.

**Purpose**

Typical

\[ |\text{VARIABLES}| > 5 \]

\[ \text{range}(\text{VARIABLES}.\text{var}) > 2 \]
Figure 3.1067: Illustrating the **POS_MIN_MIN_DIP_ON_INCREASING_SEQUENCE** constraint of the **Example** slot

**Arg. properties**

- **Functional dependency**: VALUE determined by VARIABLES.
- **Functional dependency**: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_MIN_DIP_ON_INCREASING_SEQUENCE constraint but use the decoration table 2.33.
3.503 **POS_MIN_MIN_GORGE**

**DESCRIPTION**

- **Origin**
  - Based on constraint MIN_MIN_GORGE.

- **Constraint**
  - \( \text{POS_MIN_MIN_GORGE}(\text{VALUE}, \text{VARIABLES}, \text{FOUND}) \)

- **Arguments**
  - \( \text{VALUE} : \text{dvar} \)
  - \( \text{VARIABLES} : \text{collection(var−dvar)} \)
  - \( \text{FOUND} : \text{collection(var−dvar)} \)

- **Restrictions**
  - \( \text{sv} \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = +\infty \)
  - \( \text{VALUE} \geq \text{minv} \)
  - \( \text{VALUE} = +\infty \lor \text{VALUE} \leq \text{maxv} - 1 \)
  - \( \text{required} (\text{VARIABLES}, \text{var}) \)
  - \( \text{required} (\text{FOUND}, \text{var}) \)

  where
  - \( \text{minv} = \text{minval} (\text{VARIABLES}, \text{var}) \)
  - \( \text{maxv} = \text{maxval} (\text{VARIABLES}, \text{var}) \)
  - \( \text{sv} = |\text{VARIABLES}| \)
  - \( \text{rv} = \text{range} (\text{VARIABLES}, \text{var}) \)

**Purpose**

- The constraint \( \text{MIN_MIN_GORGE} (\text{VALUE}, \text{VARIABLES}) \) holds. In addition, \( \text{FOUND} \) is a collection of 0/1 variables where the value 1 indicates the position of the **found** letter in those occurrences of the pattern **GORGE** for which the feature value is **VALUE**.
- The position of the **found** letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.
- An occurrence of the pattern **GORGE** is the _maximal_ subsequence which matches the regular expression \( (> | > (= | >)^* >) (< | < (= | <)^* <) \).
- Assume that the occurrence of the pattern **GORGE** starts at position \( i \) and ends at position \( j \). The feature **MIN** computes the minimum of the values from index \( i + 1 \) to index \( j \).

- **Example**
  - \( (3, \{1, 7, 3, 4, 4, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7\}, \{0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\}) \)

- **Typical**
  - \( |\text{VARIABLES}| > 2 \)
  - \( \text{range} (\text{VARIABLES}, \text{var}) > 1 \)

- **Arg. properties**
  - Functional dependency: **VALUE** determined by **VARIABLES**.
  - Functional dependency: **FOUND** determined by **VARIABLES**.

---

Figure 3.1068 provides an example where the **POS_MIN_MIN_GORGE** constraint holds.
Figure 3.1068: Illustrating the POS_MIN_MIN_GORGE constraint of the Example slot
Automaton

Similar to the automaton of the MIN_MIN_GORGE constraint but use the decoration table 2.33.
POS_MIN_MIN_GORGE

2143
3.504  POS_MIN_MIN_INCREASING

**DESCRIPTION**

Based on constraint MIN_MIN_INCREASING.

**Constraint**

POS_MIN_MIN_INCREASING(VALUE, VARIABLES, FOUND)

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)
- **FOUND**: collection(var−dvar)

**Restrictions**

- \( sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty \)
- \( VALUE \geq \text{minv} \)
- \( VALUE = +\infty \lor VALUE \leq \text{maxv} - 1 \)
- \( \text{required}(\text{VARIABLES}.\text{var}) \)
- \( \text{required}(\text{FOUND}.\text{var}) \)

where

- \( \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \)
- \( \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \)
- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

The constraint MIN_MIN_INCREASING(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern INCREASING for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern INCREASING is the subsequence which matches the regular expression <.

Assume that the occurrence of the pattern INCREASING starts at position \( i \) and ends at position \( j \). The feature MIN computes the minimum of the values from index \( i \) to index \( j + 1 \).

**Example**

\[
\begin{pmatrix}
1, (4, 3, 5, 2, 1, 1, 3, 3, 4, 6, 3, 1, 3, 3), \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0
\end{pmatrix}
\]

Figure 3.1069 provides an example where the POS_MIN_MIN_INCREASING \( (1, [4, 3, 5, 2, 1, 1, 3, 3, 4, 6, 3, 1, 3, 3], [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]) \) constraint holds.

**Purpose**

Typical

- \( |\text{VARIABLES}| > 1 \)
- \( \text{range}(\text{VARIABLES}.\text{var}) > 1 \)
Figure 3.1069: Illustrating the `POS_MIN_MIN_INCREASING` constraint of the `Example` slot

**Arg. properties**

- Functional dependency: `VALUE` determined by `VARIABLES`.
- Functional dependency: `FOUND` determined by `VARIABLES`.
Automaton

Similar to the automaton of the MIN_MIN_INCREASING constraint but use the decoration table 2.33.
3.505 POS_MIN_MIN_INCREASING_SEQUENCE

**Description**

Based on constraint MIN_MIN_INCREASING_SEQUENCE.

**Constraint**

POS_MIN_MIN_INCREASING_SEQUENCE(VALUE, VARIABLES, FOUND)

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)
- **FOUND**: collection(var−dvar)

**Restrictions**

\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty \]

\[ VALUE \geq minv \]

\[ VALUE = +\infty \lor VALUE \leq maxv - 1 \]

\[ \text{required}(\text{VARIABLES}, \text{var}) \]

\[ \text{required}(\text{FOUND}, \text{var}) \]

where

\[ minv = \text{minval}(\text{VARIABLES}.\text{var}) \]

\[ maxv = \text{maxval}(\text{VARIABLES}.\text{var}) \]

\[ sv = |\text{VARIABLES}| \]

\[ rv = \text{range}(\text{VARIABLES}.\text{var}) \]

The constraint MIN_MIN_INCREASING_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern INCREASING_SEQUENCE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression \(< (<|=)^* | < \). Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
\begin{pmatrix}
1,\{4,3,5,5,2,1,1,3,3,4,6,6,3,1,3,3\},
0,0,0,0,0,0,1,0,0,0,0,0,1,0,0
\end{pmatrix}
\]

Figure 3.1070 provides an example where the POS_MIN_MIN_INCREASING_SEQUENCE (1, [4,3,5,5,2,1,1,3,3,4,6,6,3,1,3,3], [0,0,0,0,0,0,1,0,0,0,0,0,1,0,0]) constraint holds.

**Purpose**

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression \(< (<|=)^* | < \). Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i\) to index \(j + 1\).

**Typical**

\[ |\text{VARIABLES}| > 1 \]

\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]
Figure 3.1070: Illustrating the POS_MIN_MIN_INCREASING_SEQUENCE constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton  Similar to the automaton of the MIN\_MIN\_INCREASING\_SEQUENCE constraint but use the decoration table 2.33.
### 3.506 POS_MIN_MIN_INFLEXION

**Description**

Origin

Based on constraint MIN_MIN_INFLEXION.

Constraint

POS_MIN_MIN_INFLEXION(VALUE, VARIABLES, FOUND)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>dvar</td>
</tr>
<tr>
<td>VARIABLES</td>
<td>collection(var−dvar)</td>
</tr>
<tr>
<td>FOUND</td>
<td>collection(var−dvar)</td>
</tr>
</tbody>
</table>

Restrictions

\[
\begin{align*}
sv & \leq 2 \lor rv \leq 1 \Rightarrow VALUE = +\infty \\
VALUE & \geq \text{minv} \\
VALUE & = +\infty \lor VALUE \leq \text{maxv} \\
\text{required}(\text{VARIABLES}, \text{var}) & \Rightarrow \text{required}(\text{FOUND}, \text{var}) \\
\text{where} & \\
\text{minv} & = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
sv & = |\text{VARIABLES}| \\
rv & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

The constraint MIN_MIN_INFLEXION(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern INFLEXION for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression \(< (| =)^* > | > (| =)^* <\). Assume that the occurrence of the pattern INFLEXION starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\).

Purpose

**Example**

\[
\begin{pmatrix}
1, \{1, 2, 6, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4\} \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 3.1071 provides an example where the POS_MIN_MIN_INFLEXION (1,[1,2,6,4,3,5,2,5,1,5,3,3,4,4],[0,0,0,0,0,0,0,0,0,0,0,0,0]) constraint holds.

Typical

<table>
<thead>
<tr>
<th>VARIABLES</th>
<th>&gt; 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>range(VARIABLES.var)</td>
<td>&gt; 1</td>
</tr>
</tbody>
</table>
Figure 3.1071: Illustrating the POS_MIN_MIN_INFLEXION constraint of the Example slot

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MIN\_MIN\_INFLEXION} constraint but use the decoration table \texttt{2.33}.
3.507  POS_MIN_MIN_STRICTLY_DECREASING_SEQUENCE

### Description

**Origin**
Based on constraint `MIN_MIN_STRICTLY_DECREASING_SEQUENCE`.

**Constraint**

`POS_MIN_MIN_STRICTLY_DECREASING_SEQUENCE(VALUE, VARIABLES, FOUND)`

**Arguments**

- `VALUE : dvar`
- `VARIABLES : collection(var-dvar)`
- `FOUND : collection(var-dvar)`

**Restrictions**

\[
\begin{align*}
sv \leq 1 & \lor rv \leq 1 \Rightarrow VALUE = +\infty \\
VALUE \geq minv \land \\
VALUE = +\infty & \lor VALUE \leq maxv - 1 \\
required(VARIABLES, var) & \\
required(FOUND, var) & \\
\text{where} & \\
minv &= minval(VARIABLES.var) \\
maxv &= maxval(VARIABLES.var) \\
sv &= |VARIABLES| \\
rv &= range(VARIABLES.var)
\end{align*}
\]

The constraint `MIN_MIN_STRICTLY_DECREASING_SEQUENCE(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern `STRICTLY_DECREASING_SEQUENCE` for which the feature value is `VALUE`. The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` is the maximal subsequence which matches the regular expression `><+`. Assume that the occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `MIN` computes the minimum of the values from index `i` to index `j + 1`.

**Example**

\[
\begin{pmatrix}
1, (4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3), \\
(0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
\end{pmatrix}
\]

Figure 3.1072 provides an example where the `POS_MIN_MIN_STRICTLY_DECREASING_SEQUENCE (1, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3], [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])` constraint holds.

**Purpose**

An occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` is the maximal subsequence which matches the regular expression `><+`.

**Typical**

\[
|VARIABLES| > 1 \\
range(VARIABLES.var) > 1
\]
Figure 3.1072: Illustrating the POS_MIN_MIN_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot

**Arg. properties**
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_MIN_STRICTLY_DECREASING_SEQUENCE constraint but use the decoration table 2.33.
3.508   POS_MIN_MIN_STRICTLY_INCREASING_SEQUENCE

**DESCRIPTION**

**Origin**
Based on constraint MIN_MIN_STRICTLY_INCREASING_SEQUENCE.

**Constraint**
POS_MIN_MIN_STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES, FOUND)

**Arguments**
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

**Restrictions**
\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty \]
\[ VALUE \geq \minv \]
\[ VALUE = +\infty \lor VALUE \leq \maxv - 1 \]
\[ \text{required}(\text{VARIABLES}, \text{var}) \]
\[ \text{required}(\text{FOUND}, \text{var}) \]

where
\[ \minv = \minval(\text{VARIABLES}.\text{var}) \]
\[ \maxv = \maxval(\text{VARIABLES}.\text{var}) \]
\[ sv = |\text{VARIABLES}| \]
\[ rv = \text{range}(\text{VARIABLES}.\text{var}) \]

The constraint MIN_MIN_STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern STRICTLY_INCREASING_SEQUENCE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression \(<^+\).

Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i to index j + 1.

**Example**

\[
(1, [4, 3, 5, 5, 2, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3], [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0])
\]

Figure 3.1073 provides an example where the POS_MIN_MIN_STRICTLY_INCREASING_SEQUENCE (1, [4, 3, 5, 5, 2, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3], [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 1 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]
Figure 3.1073: Illustrating the POS_MIN_MIN_STRICTLY_INCREASING_SEQUENCE constraint of the Example slot

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MIN\_MIN\_STRICLTY\_INCREASING\_SEQUENCE} constraint but use the decoration table \ref{2.33}. 
3.509   POS_MIN_MIN_VALLEY

**DESCRIPTION**

**Origin**

Based on constraint MIN_MIN_VALLEY.

**Constraint**

POS_MIN_MIN_VALLEY(VALUE, VARIABLES, FOUND)

**Arguments**

| VALUE    | : dvar |
| VARIABLES | : collection(var−dvar) |
| FOUND    | : collection(var−dvar) |

**Restrictions**

\[
\begin{align*}
sv & \leq 2 \lor rv \leq 1 \Rightarrow VALUE = +\infty \\
VALUE & \geq \text{minv}
\end{align*}
\]

\[
\begin{align*}
VALUE & = +\infty \lor VALUE \leq \text{maxv} - 1 \\
\text{required}(\text{VARIABLES}.\text{var}) \\
\text{required}(\text{FOUND}.\text{var})
\end{align*}
\]

where

\[
\begin{align*}
\text{minv} & = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
sv & = |\text{VARIABLES}| \\
rv & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

The constraint MIN_MIN_VALLEY(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern VALLEY for which the feature value is VALUE.

**Purpose**

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression \(>(= | >)^*(< | =)^*<\).

Assume that the occurrence of the pattern VALLEY starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**

\[
\begin{align*}
2,\langle 1,3,7,4,3,6,6,5,5,3,3,2,6,5,5,7 \rangle, \\
\langle 0,0,0,0,0,0,0,0,0,1,0,0,0,0,0 \rangle
\end{align*}
\]

Figure 3.1074 provides an example where the POS_MIN_MIN_VALLEY
(2, [1, 3, 7, 4, 3, 6, 6, 5, 5, 3, 3, 2, 6, 5, 5, 7], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0])

**Typical**

- \(|\text{VARIABLES}| > 2\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Figure 3.1074: Illustrating the POS_MIN_MIN_VALLEY constraint of the Example slot
Automaton

Similar to the automaton of the MIN_MIN_VALLEY constraint but use the decoration table 2.33.
3.510  **POS_MIN_MIN_ZIGZAG**

**DESCRIPTION**

**Origin**
Based on constraint MIN_MIN_ZIGZAG.

**Constraint**

\[
\text{POS_MIN_MIN_ZIGZAG}(\text{VALUE}, \text{VARIABLES}, \text{FOUND})
\]

**Arguments**

- **VALUE** : dvar
- **VARIABLES** : collection(var − dvar)
- **FOUND** : collection(var − dvar)

**Restrictions**

\[
sv \leq 3 \lor rv \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} \geq \text{minv} \\
\text{VALUE} = +\infty \lor \text{VALUE} \leq \text{maxv} - 1 \\
\text{required}(	ext{VARIABLES}, \text{var}) \\
\text{required}(	ext{FOUND}, \text{var})
\]

where

\[
\text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \\
sv = |\text{VARIABLES}| \\
rv = \text{range}(\text{VARIABLES}.\text{var})
\]

The constraint MIN_MIN_ZIGZAG(VALUE, VARIABLES) holds. In addition, **FOUND** is a collection of 0/1 variables where the value 1 indicates the position of the **found** letter in those occurrences of the pattern **ZIGZAG** for which the feature value is **VALUE**.

The position of the **found** letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern **ZIGZAG** is the maximal subsequence which matches the regular expression \((<>)^+ (< | <>) | (>><>)(> | >><>)\).

Assume that the occurrence of the pattern **ZIGZAG** starts at position \(i\) and ends at position \(j\). The feature **MIN** computes the minimum of the values from index \(i + 1\) to index \(j\).

**Purpose**

Figure 3.1075 provides an example where the **POS_MIN_MIN_ZIGZAG** constraint holds.

**Example**

\[
\left( 1, \{4, 1, 3, 1, 4, 6, 1, 5, 2, 7, 2, 3, 1, 6, 1\}, \{0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0\} \right)
\]

Typical

- \(|\text{VARIABLES}| > 3\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Arg. properties**

- **Functional dependency**: **VALUE** determined by **VARIABLES**.
- **Functional dependency**: **FOUND** determined by **VARIABLES**.
Figure 3.1075: Illustrating the POS_MIN_MIN_ZIGZAG constraint of the Example slot
Automaton

Similar to the automaton of the MIN_MIN_ZIGZAG constraint but use the decoration table 2.33.
### Constraint

**Constraint**

\[
\text{POS\_MIN\_SURF\_BUMP\_ON\_DECREASING\_SEQUENCE}(\text{VALUE}, \text{VARIABLES}, \text{FOUND})
\]

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var - dvar)`
- **FOUND**: `collection(var - dvar)`

**Restrictions**

- \(sv \leq 5 \lor rv \leq 2 \Rightarrow \text{VALUE} = +\infty\)
- \(\text{VALUE} \geq 3 \times \text{minv} + 30\)
- \(\text{VALUE} = +\infty \lor \text{VALUE} \leq 3 \times \text{maxv} - 3\)
- \(\text{required}(\text{VARIABLES}, \text{var})\)
- \(\text{required}(\text{FOUND}, \text{var})\)

where

- \(\text{minv} = \text{minval}(\text{VARIABLES}.\text{var})\)
- \(\text{rv} = \text{range}(\text{VARIABLES}.\text{var})\)
- \(sv = |\text{VARIABLES}|\)
- \(\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})\)

**Purpose**

The constraint \(\text{MIN\_SURF\_BUMP\_ON\_DECREASING\_SEQUENCE}(\text{VALUE}, \text{VARIABLES})\) holds. In addition, **FOUND** is a collection of 0/1 variables where the value 1 indicates the position of the **found** letter in those occurrences of the pattern **BUMP\_ON\_DECREASING\_SEQUENCE** for which the feature value is **VALUE**.

The position of the **found** letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern **BUMP\_ON\_DECREASING\_SEQUENCE** is the subsequence which matches the regular expression **><>>**.

Assume that the occurrence of the pattern **BUMP\_ON\_DECREASING\_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **SURF** computes the sum of the values from index \(i + 2\) to index \(j\).

**Example**

\[
\left( 11, \langle 7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, -3, 3 \rangle, \langle 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0 \rangle \right)
\]

Figure 3.1076 provides an example where the **POS\_MIN\_SURF\_BUMP\_ON\_DECREASING\_SEQUENCE** \((11, [7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, -3, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0])\) constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 5\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 2\)
Figure 3.1076: Illustrating the POS_MIN_SURF_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton  Similar to the automaton of the MIN_SURF_BUMP_ON_DECREASING_SEQUENCE constraint but use the decoration table 2.33.
POS_MIN_SURF_BUMP_ON_DECREASING_SEQUENCE

2175
3.512 POS_MIN_SURF_DECREASING

**Origin**
Based on constraint **MIN_SURF_DECREASING**.

**Constraint**

```plaintext
POS_MIN_SURF_DECREASING(VALUE, VARIABLES, FOUND)
```

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var-dvar)`
- **FOUND**: `collection(var-dvar)`

**Restrictions**

```plaintext
sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = +∞
VALUE ≥ 2 * minv + 1
VALUE = +∞ ∨ VALUE ≤ 2 * maxv - 1
required(VARIABLES.var)
required(FOUND, var)
```

where

- `minv = minval(VARIABLES.var)`
- `maxv = maxval(VARIABLES.var)`
- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

The constraint `MIN_SURF_DECREASING(VALUE, VARIABLES)` holds. In addition, **FOUND** is a collection of 0/1 variables where the value 1 indicates the position of the **found** letter in those occurrences of the pattern **DECREASING** for which the feature value is **VALUE**.

**Purpose**

The position of the **found** letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern **DECREASING** is the subsequence which matches the regular expression `>`.

Assume that the occurrence of the pattern **DECREASING** starts at position `i` and ends at position `j`. The feature **SURF** computes the sum of the values from index `i` to index `j + 1`.

**Example**

```plaintext
(4, 3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4) .
```

Figure 3.1077 provides an example where the `POS_MIN_SURF_DECREASING` constraint holds.

**Typical**

```plaintext
|VARIABLES| > 1
range(VARIABLES.var) > 1
```
Figure 3.1077: Illustrating the POS_MIN_SURF_DECREASING constraint of the Example slot

Arg. properties
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_SURF_DECREASING constraint but use the decoration table 2.33.
3.513  POS_MIN_SURF_DECREASING_SEQUENCE

**Description**

Based on constraint `MIN_SURF_DECREASING_SEQUENCE`.

**Arguments**

- `VALUE` : `dvar`
- `VARIABLES` : `collection(var−dvar)`
- `FOUND` : `collection(var−dvar)`

**Restrictions**

- `sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = +∞`
- `rv = 2 ⇒ VALUE ≥ 2 * minv + 1`
- `rv ≥ 3 ⇒ VALUE ≥ min(2 * minv + 1, sv * (minv + 1))`
- `rv = 2 ⇒ VALUE = +∞ ∨ VALUE ≤ 2 * maxv − 1`
- `rv ≥ 3 ⇒ VALUE = +∞ ∨ VALUE ≤ max(2 * maxv − 1, sv * (maxv − 1))`
- `required(VARIABLES, var)`
- `required(FOUND, var)`

where

- `minv = minval(VARIABLES.var)`
- `rv = range(VARIABLES.var)`
- `sv = |VARIABLES|`
- `maxv = maxval(VARIABLES.var)`

**Purpose**

The constraint `MIN_SURF_DECREASING_SEQUENCE(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `DECREASING_SEQUENCE` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `DECREASING_SEQUENCE` is the maximal subsequence which matches the regular expression `>(>|=)*>|>`.

Assume that the occurrence of the pattern `DECREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i` to index `j + 1`.

**Example**

```
6, (3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 4, 6, 4, 4),
'(0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
```

Figure 3.1078 provides an example where the `POS_MIN_SURF_DECREASING_SEQUENCE` (6, [3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 4, 6, 4, 4], [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.
Figure 3.1078: Illustrating the POS_MIN_SURF_DECREASING_SEQUENCE constraint of the Example slot

Typical

$|\text{VARIABLES}| > 1$

$\text{range(VALUES.var)} > 1$

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MIN\_SURF\_DECREASING\_SEQUENCE} constraint but use the decoration table \texttt{2.33}.
3.514 POS_MIN_SURF_DECREASING_TERRACE

DESCRIPTION

Origin
Based on constraint MIN_SURF_DECREASING_TERRACE.

Constraint
POS_MIN_SURF_DECREASING_TERRACE(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var–dvar)
FOUND : collection(var–dvar)

Restrictions
sv ≤ 3 ∨ rv ≤ 2 ⇒ VALUE = +∞
VALUE ≥ min(2 * (minv + 1) | (sv – 2) * (minv + 1) |
VALUE = +∞ ∨ VALUE ≤ max(2 * (maxv – 1), (sv – 2) * (maxv – 1))
required(VARIABLES, var)
required(FOUND, var)

where
minv = minval(VARIABLES.var)
sv = |VARIABLES|
maxv = maxval(VARIABLES.var)
rv = range(VARIABLES.var)

The constraint MIN_SURF_DECREASING_TERRACE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern DECREASING_TERRACE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern DECREASING_TERRACE is the maximal subsequence which matches the regular expression >+=>.

Assume that the occurrence of the pattern DECREASING_TERRACE starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 1 to index j.

Example

Figure 3.1079 provides an example where the POS_MIN_SURF_DECREASING_TERRACE (4, [6, 4, 4, 5, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3], [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

Purpose

Typical

|VARIABLES| > 3
range(VARIABLES.var) > 2
Figure 3.1079: Illustrating the POS_MIN_SURF_DECREASING_TERRACE constraint of the Example slot

Arg. properties
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_SURF_DECREASING_TERRACE constraint but use the decoration table 2.33.
3.515  POS_MIN_SURF_DIP_ON_INCREASING_SEQUENCE

**DESCRIPTION**

- **Origin**: Based on constraint MIN_SURF_DIP_ON_INCREASING_SEQUENCE.
- **Constraint**: POS_MIN_SURF_DIP_ON_INCREASING_SEQUENCE(VALUE, VARIABLES, FOUND)

**Arguments**

- VALUE : dvar
- VARIABLES : collection(var=dvar)
- FOUND : collection(var=dvar)

**Restrictions**

\[
\begin{align*}
sv & \leq 5 \lor rv \leq 2 \Rightarrow VALUE = +\infty \\
VALUE & \geq 3 \cdot \text{minv} + 3 \delta \\
VALUE & = +\infty \lor VALUE \leq 3 \cdot \text{maxv} - 3 \\
\text{required}(VARIABLES, \text{var}) \\
\text{required}(\text{FOUND}, \text{var}) \\
\text{where} \\
\text{minv} &= \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{maxv} &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
sv &= |\text{VARIABLES}| \\
rv &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

The constraint \textit{MIN_SURF_DIP_ON_INCREASING_SEQUENCE(VALUE, VARIABLES)} holds. In addition, \text{FOUND} is a collection of 0/1 variables where the value 1 indicates the position of the \text{found} letter in those occurrences of the pattern \textit{DIP_ON_INCREASING_SEQUENCE} for which the feature value is \textit{VALUE}.

The position of the \text{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \textit{DIP_ON_INCREASING_SEQUENCE} is the subsequence which matches the regular expression <<><<.

Assume that the occurrence of the pattern \textit{DIP_ON_INCREASING_SEQUENCE} starts at position \(i\) and ends at position \(j\). The feature \text{SURF} computes the sum of the values from index \(i + 2\) to index \(j\).

**Purpose**

**Example**

\[
\begin{pmatrix}
9, 1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4) \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0)
\end{pmatrix}
\]

Figure 3.1080 provides an example where the \text{POS_MIN_SURF_DIP_ON_INCREASING_SEQUENCE} (9, [1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 5 \\
\text{range}(\text{VARIABLES}.\text{var}) > 2
\]
Figure 3.1080: Illustrating the POS_MIN_SURF_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot

Arg. properties

• Functional dependency: VALUE determined by VARIABLES.
• Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the `MIN_SURF_DIP_ON_INCREASING_SEQUENCE` constraint but use the decoration table 2.33.
POS_MIN_SURF_DIP_ON_INCREASING_SEQUENCE

2191
3.516 POS_MIN_SURF_GORGE

<table>
<thead>
<tr>
<th>Description</th>
<th>Automaton</th>
</tr>
</thead>
</table>

**Origin**
Based on constraint MIN_SURF_GORGE.

**Constraint**
POS_MIN_SURF_GORGE(VALUE, VARIABLES, FOUND)

**Arguments**
- VALUE: dvar
- VARIABLES: collection(var−dvar)
- FOUND: collection(var−dvar)

**Restrictions**
- sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = +∞
- rv = 2 ⇒ VALUE ≥ min
- rv ≥ 3 ⇒ VALUE ≥ min(minv, (sv − 2) ∗ (minv + 1) − 1)∗
- rv = 2 ⇒ VALUE = +∞ ∨ VALUE ≤ maxv + 1
- rv ≥ 3 ⇒
  - VALUE = +∞ ∨ VALUE ≤ max(maxv − 1, (sv − 2) ∗ (maxv − 1) − 1)
  - required(VARIABLES, var)
  - required(FOUND, var)

where
- minv = minval(VARIABLES.var)
- rv = range(VARIABLES.var)
- sv = |VARIABLES|
- maxv = maxval(VARIABLES.var)

**Purpose**
The constraint MIN_SURF_GORGE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern GORGE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression (> | > (= | >)* >) (< | < (= | >)* <).

Assume that the occurrence of the pattern GORGE starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 1 to index j.

**Example**

\[
\begin{pmatrix}
5, 1, 7, 3, 4, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7, 7 \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0
\end{pmatrix}
\]

Figure 3.1081 provides an example where the POS_MIN_SURF_GORGE (5, [1, 7, 3, 4, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]) constraint holds.

**Typical**
- |VARIABLES| > 2
- range(VARIABLES.var) > 1
Figure 3.1081: Illustrating the POS_MIN_SURF_GORGE constraint of the Example slot

Arg. properties

- **Functional dependency**: VALUE determined by VARIABLES.
- **Functional dependency**: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN\_SURF\_GORGE constraint but use the decoration table 2.33.
3.517 POS_MIN_SURF_INCREASING

Origin
Based on constraint MIN_SURF_INCREASING.

Constraint
POS_MIN_SURF_INCREASING(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var-dvar)
FOUND : collection(var-dvar)

Restrictions
sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = +∞
VALUE ≥ 2 * minv + 1
VALUE = +∞ ∨ VALUE ≤ 2 * maxv − 1
required(VARIABLES, var)
required(FOUND, var)
where
minv = minval(VARIABLES.var)
maxv = maxval(VARIABLES.var)
sv = |VARIABLES|
rv = range(VARIABLES.var)

Purpose
The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern INCREASING is the subsequence which matches the regular expression <.
Assume that the occurrence of the pattern INCREASING starts at position i and ends at position j. The feature SURF computes the sum of the values from index i to index j + 1.

Example
Figure 3.1082 provides an example where the POS_MIN_SURF_INCREASING (4, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3], [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]) constraint holds.

Typical
|VARIABLES| > 1
range(VARIABLES.var) > 1
Figure 3.1082: Illustrating the POS_MIN_SURF_INCREASING constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_SURF_INCREASING constraint but use the decoration table 2.33.
### 3.518 POS_MIN_SURF_INCREASING_SEQUENCE

**Description**

Based on constraint MIN_SURF_INCREASING_SEQUENCE.

**Constraint**

POS_MIN_SURF_INCREASING_SEQUENCE(VALUE, VARIABLES, FOUND)

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)
- **FOUND**: collection(var−dvar)

**Restrictions**

\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = +\infty \]
\[ rv = 2 \Rightarrow VALUE \geq 2 * \text{minv} + 1 \]
\[ rv \geq 3 \Rightarrow VALUE \geq \min(2 * \text{minv} + 1, sv \cdot (\text{minv} + 1)) \]
\[ rv = 2 \Rightarrow VALUE = +\infty \lor VALUE \leq 2 * \text{maxv} - 1 \]
\[ rv \geq 3 \Rightarrow VALUE = +\infty \lor VALUE \leq \max(2 * \text{maxv} - 1, sv \cdot (\text{maxv} - 1)) \]

**Purpose**

The constraint MIN_SURF_INCREASING_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern INCREASING_SEQUENCE for which the feature value is VALUE.

- The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.
- An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression \(< (< | =)^* < | <.\)

**Example**

\[
\begin{pmatrix}
4, [4, 3, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3], \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 3.1083 provides an example where the POS_MIN_SURF_INCREASING_SEQUENCE (4, [4, 3, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.
Figure 3.1083: Illustrating the POS_MIN_SURF_INCREASING_SEQUENCE constraint of the Example slot

Typical

\[ |\text{VARIABLES}| > 1 \]
\[ \text{range(}\text{VARIABLES}.\text{var}) > 1 \]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_SURF_INCREASING_SEQUENCE constraint but use the decoration table 2.33.
POS_MIN_SURF_INCREASING_SEQUENCE
3.519 POS_MIN_SURF_INCREASING_TERRACE

### Description

#### Origin
Based on constraint MIN_SURF_INCREASING_TERRACE.

#### Constraint
POS_MIN_SURF_INCREASING_TERRACE(VALUE, VARIABLES, FOUND)

#### Arguments
- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)
- **FOUND**: collection(var−dvar)

#### Restrictions

\[
\text{sv} \leq 3 \lor \text{rv} \leq 2 \Rightarrow \text{VALUE} = +\infty \\
\text{VALUE} \geq \min(2 \ast (\text{minv} + 1) \lor (\text{sv} - 2) \ast (\text{minv} + 1)) \\
\text{VALUE} = +\infty \lor \text{VALUE} \leq \max(2 \ast (\text{maxv} - 1), (\text{sv} - 2) \ast (\text{maxv} - 1))
\]

required(VARIABLES, var)
required(FOUND, var)

where
- \( \text{minv} = \minval(VARIABLES, \text{var}) \)
- \( \text{sv} = |\text{VARIABLES}| \)
- \( \text{maxv} = \maxval(VARIABLES, \text{var}) \)
- \( \text{rv} = \text{range}(VARIABLES, \text{var}) \)

The constraint MIN_SURF_INCREASING_TERRACE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern INCREASING_TERRACE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression \(<=+<\).

Assume that the occurrence of the pattern INCREASING_TERRACE starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i+1 \) to index \( j \).

#### Purpose

The constraint POS_MIN_SURF_INCREASING_TERRACE holds.

#### Example

\[ \begin{pmatrix} 9, \{1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4\}, \\
\{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0\} \end{pmatrix} \]

Figure 3.1084 provides an example where the POS_MIN_SURF_INCREASING_TERRACE constraint holds.

#### Typical

\[ |\text{VARIABLES}| > 3 \]
\[ \text{range}(\text{VARIABLES, var}) > 2 \]
Figure 3.1084: Illustrating the POS_MIN_SURF_INCREASING_TERRACE constraint of the Example slot

**Arg. properties**

- **Functional dependency:** VALUE determined by VARIABLES.
- **Functional dependency:** FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_SURF_INCREASING_TERRACE constraint but use the decoration table 2.33.
POS_MIN_SURF_INCREASING_TERRACE 2207
3.520 POS_MIN_SURF_INFLEXION

**DESCRIPTION**

Based on constraint MIN_SURF_INFLEXION.

**AUTOMATON**

- \(< (< | =) > | > (> | =)^* <\>

**Constraint**

POS_MIN_SURF_INFLEXION(VALUE, VARIABLES, FOUND)

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)
- **FOUND**: collection(var−dvar)

**Restrictions**

\[ sv \leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = +\infty \]
\[ \text{VALUE} \geq \min(\minv, (sv - 2) \cdot \minv) \]
\[ \text{VALUE} = +\infty \lor \text{VALUE} \leq \max(\maxv, (sv - 2) \cdot \maxv) \]
\[ \text{required}(\text{VARIABLES}.\text{var}) \]
\[ \text{required}(\text{FOUND}.\text{var}) \]

where

\[ \text{minv} = \minval(\text{VARIABLES}.\text{var}) \]
\[ \text{sv} = |\text{VARIABLES}| \]
\[ \text{maxv} = \maxval(\text{VARIABLES}.\text{var}) \]
\[ \text{rv} = \text{range}(\text{VARIABLES}.\text{var}) \]

The constraint MIN_SURF_INFLEXION(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern INFLEXION for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression \(< (< | =) > | > (> | =)^* <\>.

Assume that the occurrence of the pattern INFLEXION starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

\[
\begin{pmatrix}
1, (1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4), \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0)
\end{pmatrix}
\]

Figure 3.1085 provides an example where the POS_MIN_SURF_INFLEXION (1,[1,2,6,4,4,3,5,2,5,1,5,3,3,4,4],[0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0]) constraint holds.

**Purpose**

**Typical**

\[ |\text{VARIABLES}| > 2 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]
Figure 3.1085: Illustrating the POS_MIN_SURF_INFLEXION constraint of the **Example** slot

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_SURF_INFLEXION constraint but use the decoration table 2.33.
### 3.521 POS_MIN_SURF_PEAK

**DESCRIPTION**

**Origin**
Based on constraint MIN_SURF_PEAK.

**Constraint**
POS_MIN_SURF_PEAK(VALUE, VARIABLES, FOUND)

**Arguments**
- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)
- **FOUND**: collection(var−dvar)

**Restrictions**

- \(sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = +\infty\)
- \(VALUE \geq \min(\text{minv} + 10, (sv - 2) \times (\text{minv} + 1))\)
- \(VALUE = +\infty \lor VALUE \leq \max(\text{maxv}, (sv - 2) \times \text{maxv})\)
- required(VARIABLES, var)
- required(FOUND, var)

where
- \(\text{minv} = \text{minval}(\text{VARIABLES}.\text{var})\)
- \(sv = |\text{VARIABLES}|\)
- \(\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})\)
- \(rv = \text{range}(\text{VARIABLES}.\text{var})\)

The constraint MIN_SURF_PEAK(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern PEAK for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression \(<(= | <)^* (> | =)^* >\).

Assume that the occurrence of the pattern PEAK starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Purpose**

**Example**

(9, [7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 1], [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0])

Figure 3.1086 provides an example where the POS_MIN_SURF_PEAK (9, [7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 1], [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]) constraint holds.

**Typical**
- \(|\text{VARIABLES}| > 2\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Arg. properties**
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Figure 3.1086: Illustrating the POS\_MIN\_SURF\_PEAK constraint of the Example slot
Automaton Similar to the automaton of the MIN_SURF_PEAK constraint but use the decoration table 2.33.
3.522  POS_MIN_SURF.PLAIN

DESCRIPTION

Origin
Based on constraint MIN_SURF.PLAIN.

Constraint
POS_MIN_SURF.PLAIN(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions
sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = +\infty
VALUE \geq \min(minv, (sv - 2) + minv)
VALUE = +\infty \lor VALUE \leq \max(maxv - 1, (sv - 2) \ast (maxv - 1))
required(VARIABLES, var)
required(FOUND, var)
where
minv = minval(VARIABLES.var)
sv = |VARIABLES|
maxv = maxval(VARIABLES.var)
rv = range(VARIABLES.var)

The constraint MIN_SURF.PLAIN(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern PLAIN for which the feature value is VALUE.

Purpose
The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression > = * <.
Assume that the occurrence of the pattern PLAIN starts at position i and ends at position j. The feature SURF computes the sum of the values from index i + 1 to index j.

Example
\[
\begin{pmatrix}
4, (2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3), \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 3.1087 provides an example where the POS_MIN_SURF.PLAIN (4, [2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3], [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

Typical
|VARIABLES| > 2
range(VARIABLES.var) > 1

Arg. properties
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Figure 3.1087: Illustrating the POS_MIN_SURF_PLAIN constraint of the Example slot
Automaton  Similar to the automaton of the MIN_SURF_PLAIN constraint but use the decoration table 2.33.
3.523 POS_MIN_SURF_PLATEAU

### Description

The constraint $\text{POS\_MIN\_SURF\_PLATEAU}(\text{VALUE}, \text{VARIABLES}, \text{FOUND})$ holds. In addition, $\text{FOUND}$ is a collection of 0/1 variables where the value 1 indicates the position of the $\text{found}$ letter in those occurrences of the pattern $\text{PLATEAU}$ for which the feature value is $\text{VALUE}$.

The position of the $\text{found}$ letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern $\text{PLATEAU}$ is the maximal subsequence which matches the regular expression $\langle = \ast \rangle$.

Assume that the occurrence of the pattern $\text{PLATEAU}$ starts at position $i$ and ends at position $j$. The feature $\text{SURF}$ computes the sum of the values from index $i+1$ to index $j$.

### Example

Figure 3.1088 provides an example where the $\text{POS\_MIN\_SURF\_PLATEAU}(3, [7, 5, 2, 3, 1, 2, 4, 3, 4, 5, 5, 2, 2, 5], [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])$ constraint holds.

### Typical

$|\text{VARIABLES}| > 2$

$\text{range}(\text{VARIABLES}.\text{var}) > 1$
Figure 3.1088: Illustrating the POS_MIN_SURF_PLATEAU constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \textit{MIN\_SURF\_PLATEAU} constraint but use the decoration table \textbf{2.33}. 
POS_MIN_SURF_PLATEAU

2223
3.524  POS_MIN_SURF_PROPER_PLAIN

**Description**

Based on constraint MIN_SURF_PROPER_PLAIN.

**Constraint**

POS_MIN_SURF_PROPER_PLAIN(VALUE, VARIABLES, FOUND)

**Arguments**

- VALUE : dvar
- VARIABLES : collection(var−dvar)
- FOUND : collection(var−dvar)

**Restrictions**

\[
sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = +\infty \\
VALUE \geq \min(2 + \miny, (sv - 2) + \miny) \\
VALUE = +\infty \lor VALUE \leq \max(2 + (maxy - 1), (sv - 2) + (maxy - 1)) \\
required(VARIABLES, var) \\
required(FOUND, var)
\]

where

\[
\miny = \minval(VARIABLES.var) \\
sv = |VARIABLES| \\
maxy = \maxval(VARIABLES.var) \\
rv = range(VARIABLES.var)
\]

The constraint MIN_SURF_PROPER_PLAIN(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern PROPER_PLAIN for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern PROPER_PLAIN is the maximal subsequence which matches the regular expression \(> = + <\).

Assume that the occurrence of the pattern PROPER_PLAIN starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

\[
(8, [2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3, 5], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
\]

Figure 3.1089 provides an example where the POS_MIN_SURF_PROPER_PLAIN (8, [2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3, 5], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

**Typical**

\[|VARIABLES| > 3 \]
\[range(VARIABLES.var) > 1\]
Figure 3.1089: Illustrating the POS_MIN_SURF_PROPER_PLAIN constraint of the Example slot

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_SURF_PROPERPlain constraint but use the decoration table 2.33.
### 3.525 POS_MIN_SURF_PROPER_PLATEAU

**Origin**
Based on constraint `MIN_SURF_PROPER_PLATEAU`.

**Constraint**

\[
\text{POS_MIN_SURF_PROPER_PLATEAU}(\text{VALUE}, \text{VARIABLES}, \text{FOUND})
\]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection(var−dvar)} \\
\text{FOUND} & : \text{collection(var−dvar)}
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
sv \leq 3 \lor rv \leq 1 & \Rightarrow VALUE = +\infty \\
VALUE \geq \min(2 \ast (\text{minv} + 1) \ast (sv - 2) \ast (\text{minv} + 1) \ast 2) & \\
VALUE = +\infty \lor VALUE \leq \max(2 \ast \text{maxv}, (sv - 2) \ast \text{maxv}) & \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{required}(\text{FOUND}, \text{var}) & \\
\text{where} & \\
\text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) & \\
sv = |\text{VARIABLES}| & \\
\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) & \\
rv = \text{range}(\text{VARIABLES}.\text{var}) &
\end{align*}
\]

The constraint `MIN_SURF_PROPER_PLATEAU(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `PROPER_PLATEAU` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `PROPER_PLATEAU` is the maximal subsequence which matches the regular expression `<=+=>`.

Assume that the occurrence of the pattern `PROPER_PLATEAU` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i + 1` to index `j`.

**Example**

\[
\begin{pmatrix}
6, \langle 7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3 \rangle, \\
\langle 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 \rangle
\end{pmatrix}
\]

Figure 3.1090 provides an example where the `POS_MIN_SURF_PROPER_PLATEAU (6, [7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3], [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])` constraint holds.

**Purpose**

The constraint `MIN_SURF_PROPER_PLATEAU(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `PROPER_PLATEAU` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `PROPER_PLATEAU` is the maximal subsequence which matches the regular expression `<=+=>`.

Assume that the occurrence of the pattern `PROPER_PLATEAU` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i + 1` to index `j`.

**Typical**

\[
|\text{VARIABLES}| > 3 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]
Figure 3.1090: Illustrating the POS_MIN_SURF_PROPER_PLATEAU constraint of the Example slot

Arg. properties
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MIN\_SURF\_PROPER\_PLATEAU} constraint but use the decoration table \texttt{2.33}.
3.526  **POS_MIN_SURF_STEADY**

### DESCRIPTION

**Origin**
Based on constraint `MIN_SURF_STEADY`.

**Constraint**

```
POS_MIN_SURF_STEADY(VALUE, VARIABLES, FOUND)
```

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var=dvar)`
- **FOUND**: `collection(var=dvar)`

**Restrictions**

- `sv ≤ 1 ⇒ VALUE = +∞`
- `VALUE ≥ 2 + minv`
- `VALUE = +∞ ∨ VALUE ≤ 2 + maxv`
- `required(VARIABLES, var)`
- `required(FOUND, var)`

where

- `minv = minval(VARIABLES.var)`
- `maxv = maxval(VARIABLES.var)`
- `sv = |VARIABLES|`

The constraint `MIN_SURF_STEADY(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `STEADY` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `STEADY` is the subsequence which matches the regular expression `=`

Assume that the occurrence of the pattern `STEADY` starts at position `i` and ends at position `j`. The feature `SURF` computes the sum of the values from index `i` to index `j + 1`.

### Automaton

**Purpose**

Figure 3.1091 provides an example where the `POS_MIN_SURF_STEADY` constraint holds.

**Typical**

`|VARIABLES| > 1`

**Arg. properties**

- Functional dependency: `VALUE` determined by `VARIABLES`.
- Functional dependency: `FOUND` determined by `VARIABLES`. 

---

### Example

```
(2, (1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6), )
```

```
(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
```
Figure 3.1091: Illustrating the POS_MIN_SURF_STEADY constraint of the Example slot
Automaton

Similar to the automaton of the MIN_SURF_STEADY constraint but use the decoration table 2.33.
3.5.27 **POS_MIN_SURF_STEADY_SEQUENCE**

**Description**

Based on constraint **MIN_SURF_STEADY_SEQUENCE**.

**Constraint**

\[
\text{POS_MIN_SURF_STEADY_SEQUENCE}(\text{VALUE, VARIABLES, FOUND})
\]

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)
- **FOUND**: collection(var−dvar)

**Restrictions**

- \(sv \leq 1 \Rightarrow \text{VALUE} = +\infty\)
- \(rv = 1 \Rightarrow \text{VALUE} \geq sv \cdot \minv\)
- \(rv \geq 2 \Rightarrow \text{VALUE} \geq \min(2 \cdot \minv, sv \cdot \minv)\)
- \(rv = 1 \Rightarrow \text{VALUE} = +\infty \lor \text{VALUE} \leq sv \cdot \maxv\)
- \(rv \geq 2 \Rightarrow \text{VALUE} = +\infty \lor \text{VALUE} \leq \max(2 \cdot \maxv, sv \cdot \maxv)\)

**Purpose**

The constraint **MIN_SURF_STEADY_SEQUENCE(VALUE, VARIABLES)** holds. In addition, **FOUND** is a collection of 0/1 variables where the value 1 indicates the position of the **found** letter in those occurrences of the pattern **STEADY_SEQUENCE** for which the feature value is **VALUE**.

The position of the **found** letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern **STEADY_SEQUENCE** is the maximal subsequence which matches the regular expression \(\cdot^*\).

Assume that the occurrence of the pattern **STEADY_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **SURF** computes the sum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
\begin{pmatrix}
2, (3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1), \\
0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0
\end{pmatrix}
\]

Figure 3.1092 provides an example where the **POS_MIN_SURF_STEADY_SEQUENCE** (2, [2, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1], [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]) constraint holds.
Figure 3.1092: Illustrating the POS_MIN_SURF_STEADY_SEQUENCE constraint of the Example slot

**Typical**

\[ |\text{VARIABLES} | > 1 \]

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_SURF_STEADY_SEQUENCE constraint but use the decoration table 2.33.
POS_MIN_SURF_STEADY_SEQUENCE

2239
3.528  POS_MIN_SURF_STRICTLY_DECREASING_SEQUENCE

**DESCRIPTION**

Based on constraint **MIN_SURF_STRICTLY_DECREASING_SEQUENCE**.

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var−dvar)`
- **FOUND**: `collection(var−dvar)`

**Restrictions**

\[
\begin{align*}
sv \leq 1 \lor rv \leq 1 & \Rightarrow VALUE = +\infty \\
\minv < 0 & \Rightarrow VALUE \geq \ell_1 \ast \minv + \lfloor \ell_1 \ast (\ell_1 - 1)/2 \rfloor \\
\minv \geq 0 & \Rightarrow VALUE \geq 2 \ast \minv + 1 \\
\maxv > 0 & \Rightarrow VALUE = +\infty \lor VALUE \leq \ell_2 \ast \maxv - \lfloor \ell_2 \ast (\ell_2 - 1)/2 \rfloor \\
\maxv \leq 0 & \Rightarrow VALUE = +\infty \lor VALUE \leq 2 \ast \maxv - 1
\end{align*}
\]

The constraint **MIN_SURF_STRICTLY_DECREASING_SEQUENCE(VALUE, VARIABLES)** holds. In addition, **FOUND** is a collection of 0/1 variables where the value 1 indicates the position of the **found** letter in those occurrences of the pattern **STRICTLY_DECREASING_SEQUENCE** for which the feature value is **VALUE**.

**Purpose**

The position of the **found** letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** is the **maximal** subsequence which matches the regular expression \( ^+ \). Assume that the occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** starts at position \( i \) and ends at position \( j \). The feature **SURF** computes the sum of the values from index \( i \) to index \( j + 1 \).

**Example**

\[
\left( 7, \{4, 4, 6, 4, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3\}, \{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\} \right)
\]

Figure 3.1093 provides an example where the **POS_MIN_SURF_STRICTLY_DECREASING_SEQUENCE** \((7, \{4, 4, 6, 4, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3\}, \{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\})** constraint holds.
Figure 3.1093: Illustrating the POS_MIN_SURF STRICTLY DECREASING SEQUENCE constraint of the Example slot

Typical

\[ |\text{VARIABLES}| > 1 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the \texttt{MIN\_SURF\_STRICTLY\_DECREASING\_SEQUENCE} constraint but use the decoration table \texttt{2.33}. 
3.529 POS\_MIN\_SURF\_STRICTLY\_INCREASING\_SEQUENCE

**DESCRIPTION**

Based on constraint MIN\_SURF\_STRICTLY\_INCREASING\_SEQUENCE.

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} - \text{dvar}) \\
\text{FOUND} & : \text{collection}(\text{var} - \text{dvar})
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 1 \lor r_v \leq 1 \Rightarrow \text{VALUE} = +\infty \\
\minv & < 0 \Rightarrow \text{VALUE} \geq \ell_1 \ast \minv + [\ell_1 \ast (\ell_1 - 1)/2] \ast x \\
\minv & \geq 0 \Rightarrow \text{VALUE} \geq 2 \ast \minv + 1 \\
\maxv & > 0 \Rightarrow \text{VALUE} = +\infty \lor \text{VALUE} \leq \ell_2 \ast \maxv - [\ell_2 \ast (\ell_2 - 1)/2] \\
\maxv & \leq 0 \Rightarrow \text{VALUE} = +\infty \lor \text{VALUE} \leq 2 \ast \maxv - 1
\end{align*}
\]

required(VARIABLES, var)

\[\text{required(FOUND, var)}\]

where

\[
\begin{align*}
\text{minv} & = \text{minval}(\text{VARIABLES, var}) \\
\text{rv} & = \text{range}(\text{VARIABLES, var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\ell_1 & = \text{min}(|\text{sv, rv}, |\text{minv}|) \\
\ell_2 & = \text{min}(|\text{sv, rv}, |\text{maxv}|) \\
\text{maxv} & = \text{maxval}(\text{VARIABLES, var})
\end{align*}
\]

The constraint MIN\_SURF\_STRICTLY\_INCREASING\_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern STRICTLY\_INCREASING\_SEQUENCE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern STRICTLY\_INCREASING\_SEQUENCE is the maximal subsequence which matches the regular expression \(<^{+}\).

Assume that the occurrence of the pattern STRICTLY\_INCREASING\_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
\begin{pmatrix}
6, 4, 3, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3 \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 3.1094 provides an example where the POS\_MIN\_SURF\_STRICTLY\_INCREASING\_SEQUENCE (6, [4, 3, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.
Figure 3.1094: Illustrating the POS_MIN_SURF STRICTLY_INCREASING_SEQUENCE constraint of the Example slot

Typical

\[ |\text{VARIABLES}| > 1 \]
\[ \text{range(\text{VARIABLES}.var)} > 1 \]

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the $\text{MIN}_\text{SURF}_\text{STRICTLY}_\text{INCREASING}_\text{SEQUENCE}$ constraint but use the decoration table \ref{tab:2.33}.
### 3.530 POS_MIN_SURF_SUMMIT

**Origin**
Based on constraint MIN_SURF_SUMMIT.

**Constraint**
POS_MIN_SURF_SUMMIT(VALUE, VARIABLES, FOUND)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var−dvar)
- FOUND : collection(var−dvar)

**Restrictions**
\[ sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = +\infty \]
\[ rv = 2 \Rightarrow VALUE = minv + 10 \]
\[ rv \geq 3 \Rightarrow VALUE \geq min(minv + 18, (sv - 2) \times (minv + 1) + 12) \]
\[ rv = 2 \Rightarrow VALUE = +\infty \lor VALUE \leq maxv \]
\[ rv \geq 3 \Rightarrow VALUE = +\infty \lor VALUE \leq max(maxv, (sv - 2) \times (maxv - 1) + 1) \]

required(VARIABLES, var)
required(FOUND, var)

where
\[ minv = \text{minval}(\text{VARIABLES}.\text{var}) \]
\[ rv = \text{range}(\text{VARIABLES}.\text{var}) \]
\[ sv = |\text{VARIABLES}| \]
\[ maxv = \text{maxval}(\text{VARIABLES}.\text{var}) \]

**Purpose**
The constraint MIN_SURF_SUMMIT(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern SUMMIT for which the feature value is VALUE.
The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression \(< | < (= | <)^* <(>) | > (= | >)^* >)\).

Assume that the occurrence of the pattern SUMMIT starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**
\[
(3, 7, 1, 5, 4, 4, 3, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1),
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0)
\]

Figure 3.1095 provides an example where the POS_MIN_SURF_SUMMIT \((3, [7, 1, 5, 4, 4, 3, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0])\) constraint holds.

**Typical**
\[ |\text{VARIABLES}| > 2 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]
Figure 3.1095: Illustrating the POS_MIN_SURF_SUMMIT constraint of the Example slot

**Arg. properties**

- **Functional dependency:** VALUE determined by VARIABLES.
- **Functional dependency:** FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_SURF_SUMMIT constraint but use the decoration table 2.33.
3.531  POS_MIN_SURF_VALLEY

**DESCRIPTION**

**Origin**
Based on constraint MIN_SURF_VALLEY.

**Constraint**
POS_MIN_SURF_VALLEY(VALUE, VARIABLES, FOUND)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var−dvar)
- FOUND : collection(var−dvar)

**Restrictions**
- \( sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = +\infty \)
- \( VALUE \geq \min(minv, (sv - 2) + minv) \)
- \( VALUE = +\infty \lor VALUE \leq \max(maxv - 1, (sv - 2) + (maxv - 1)) \)
- \( \text{required}(\text{VARIABLES}, \text{var}) \)
- \( \text{required}(\text{FOUND}, \text{var}) \)

where
- \( minv = \text{minval}(\text{VARIABLES}, \text{var}) \)
- \( sv = |\text{VARIABLES}| \)
- \( maxv = \text{maxval}(\text{VARIABLES}, \text{var}) \)
- \( rv = \text{range}(\text{VARIABLES}, \text{var}) \)

The constraint \( \text{MIN\_SURF\_VALLEY}(\text{VALUE}, \text{VARIABLES}) \) holds. In addition, \( \text{FOUND} \) is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern \( \text{VALLEY} \) for which the feature value is \( \text{VALUE} \).

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern \( \text{VALLEY} \) is the maximal subsequence which matches the regular expression \( > (= | >)^* (< | =)^* < \).

Assume that the occurrence of the pattern \( \text{VALLEY} \) starts at position \( i \) and ends at position \( j \). The feature \( \text{SURF} \) computes the sum of the values from index \( i + 1 \) to index \( j \).

**Example**
\[
(7, \langle 1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 5 \rangle, \langle 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 \rangle)
\]

Figure 3.1096 provides an example where the \( \text{POS\_MIN\_SURF\_VALLEY} \) \((7, [1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 5], [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])\) constraint holds.

**Typical**
- \(|\text{VARIABLES}| > 2\)
- \(\text{range}(\text{VARIABLES}, \text{var}) > 1\)

**Arg. properties**
- Functional dependency: \( \text{VALUE} \) determined by \( \text{VARIABLES} \).
- Functional dependency: \( \text{FOUND} \) determined by \( \text{VARIABLES} \).
Figure 3.1096: Illustrating the POS_MIN_SURF_VALLEY constraint of the Example slot
Automaton

Similar to the automaton of the MIN_SURF_VALLEY constraint but use the decoration table 2.33.
3.532  POS_MIN_SURF_ZIGZAG

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
<tbody>
<tr>
<td>Origin</td>
<td>Based on constraint MIN_SURF_ZIGZAG.</td>
</tr>
<tr>
<td>Constraint</td>
<td>POS_MIN_SURF_ZIGZAG(VALUE, VARIABLES, FOUND)</td>
</tr>
<tr>
<td>Arguments</td>
<td>VALUE : dvar</td>
</tr>
<tr>
<td></td>
<td>VARIABLES : collection(var−dvar)</td>
</tr>
<tr>
<td></td>
<td>FOUND : collection(var−dvar)</td>
</tr>
</tbody>
</table>

Restrictions

\[
\begin{align*}
sv \leq 3 \lor rv \leq 1 & \Rightarrow VALUE = +\infty \\
VALUE \geq & \min \left( 2 \times \text{minv} + 10, \right. \\
& \left. \frac{sv - 1}{2} \times \text{minv} + \frac{sv - 2}{2} \times (\text{minv} + 1) \right) \\
VALUE \leq & \max \left( 2 \times \text{maxv} - 1, \right. \\
& \left. \frac{sv - 1}{2} \times \text{maxv} + \frac{sv - 2}{2} \times (\text{maxv} - 1) \right) \\
\text{required} & (\text{VARIABLES}, \text{var}) \\
\text{required} & (\text{FOUND}, \text{var}) \\
\text{where} & \\
\text{minv} & = \text{minval}(\text{VARIABLES}, \text{var}) \\
sv & = |\text{VARIABLES}| \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}, \text{var}) \\
rv & = \text{range}(\text{VARIABLES}, \text{var}) \\
\end{align*}
\]

Purpose

The constraint MIN_SURF_ZIGZAG(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern ZIGZAG for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression \((<>)^+ (< | <> ) | (> <)^+ (> | > <))\). Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i+1\) to index \(j\).

Example

\[
\begin{pmatrix}
5, \langle 4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1 \rangle, \\
\{0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\}
\end{pmatrix}
\]

Figure 3.1097 provides an example where the POS_MIN_SURF_ZIGZAG (5, [4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1], [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

Typical

\[|\text{VARIABLES}| > 3 \]

\[\text{range}(\text{VARIABLES}, \text{var}) > 1\]
Figure 3.1097: Illustrating the POS_MIN_SURF_ZIGZAG constraint of the Example slot

Arg. properties
- **Functional dependency**: VALUE determined by VARIABLES.
- **Functional dependency**: FOUND determined by VARIABLES.
Automaton

Similar to the automaton of the MIN_SURF_ZIGZAG constraint but use the decoration table 2.33.
POS_MIN_SURF_ZIGZAG 2259
3.533  POS_MIN_WIDTH_DECREASING_SEQUENCE

**DESCRIPTION**

Based on constraint MIN_WIDTH_DECREASING_SEQUENCE.

**Constraint**

POS_MIN_WIDTH_DECREASING_SEQUENCE(VALUE, VARIABLES, FOUND)

**Arguments**

VALUE : dvar  
VARIABLES : collection(var,-dvar)  
FOUND : collection(var,-dvar)

**Restrictions**

\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = sv + 1 \]
\[ VALUE \geq 2 \]
\[ rv = 2 \Rightarrow VALUE = sv + 1 \lor VALUE \leq 2 \]
\[ rv \geq 3 \Rightarrow VALUE = sv + 1 \lor VALUE \leq sv \]
\[ \text{required}(\text{VARIABLES}, \text{var}) \]
\[ \text{required}(\text{FOUND}, \text{var}) \]
where
\[ sv = |\text{VARIABLES}| \]
\[ rv = \text{range}(\text{VARIABLES}.\text{var}) \]

The constraint MIN_WIDTH_DECREASING_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern DECREASING_SEQUENCE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression \( (> \mid | =)^* (> \mid >) \).

Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i + 2 \).

**Example**

\[
\left( 2, \{3, 4, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4\}, \{0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\} \right)
\]

Figure 3.1098 provides an example where the POS_MIN_WIDTH_DECREASING_SEQUENCE (2, [3, 4, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4], [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 1 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Figure 3.1098: Illustrating the POS_MIN_WIDTH_DECREASING_SEQUENCE constraint of the Example slot
| Automaton | Similar to the automaton of the `MIN_WIDTH_DECREASING_SEQUENCE` constraint but use the decoration table 2.33. |
3.534  POS_MIN_WIDTH_DECREASING_TERRACE

### Description

#### Origin

Based on constraint MIN_WIDTH_DECREASING_TERRACE.

#### Constraint

\[
\text{POS_MIN_WIDTH_DECREASING_TERRACE}(\text{VALUE}, \text{VARIABLES}, \text{FOUND})
\]

#### Arguments

- VALUE : dvar
- VARIABLES : collection(var – dvar)
- FOUND : collection(var – dvar)

#### Restrictions

\[
\begin{align*}
\text{sv} & \leq 3 \lor \text{rv} \leq 2 \Rightarrow \text{VALUE} = \text{sv} + 1 \\
\text{VALUE} & \geq 2 \\
\text{VALUE} & = \text{sv} + 1 \lor \text{VALUE} \leq \text{sv} - 2 \\
\text{required} & (\text{VARIABLES}, \text{var}) \\
\text{required} & (\text{FOUND}, \text{var})
\end{align*}
\]

where

\[
\begin{align*}
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

The constraint MIN_WIDTH_DECREASING_TERRACE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern DECREASING_TERRACE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern DECREASING_TERRACE is the maximal subsequence which matches the regular expression \( > = + > \).

Assume that the occurrence of the pattern DECREASING_TERRACE starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

#### Purpose

The constraint MIN_WIDTH_DECREASING_TERRACE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern DECREASING_TERRACE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern DECREASING_TERRACE is the maximal subsequence which matches the regular expression \( > = + > \).

Assume that the occurrence of the pattern DECREASING_TERRACE starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

#### Example

\[
\left( 2, \langle 6, 4, 4, 4, 5, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3 \rangle, \langle 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0 \rangle \right)
\]

Figure 3.1099 provides an example where the POS_MIN_WIDTH_DECREASING_TERRACE (2, [6, 4, 4, 4, 5, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3], [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0]) constraint holds.

#### Typical

- \(|\text{VARIABLES}| > 3\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 2\)

#### Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Figure 3.1099: Illustrating the POS_MIN_WIDTH_DECREASING_TERRACE constraint of the Example slot
Automaton

Similar to the automaton of the `MIN_WIDTH_DECREASING_TERRACE` constraint but use the decoration table 2.33.
POS_MIN_WIDTH_DECSTRING_TERRACE
3.535 POS_MIN_WIDTH_GORGE

**DESCRIPTION**

Based on constraint MIN_WIDTH_GORGE.

**AUTOMATON**

\[(> | > (= | >)* >(< | < (= | <)* <)\]

**Origin**

Based on constraint MIN_WIDTH_GORGE.

**Constraint**

\[
\text{POS}_n \text{MIN}_n \text{WIDTH}_n \text{GORGE}_n(\text{VALUE}, \text{VARIABLES}, \text{FOUND})
\]

**Arguments**

- \(\text{VALUE} : \text{dvar}\)
- \(\text{VARIABLES} : \text{collection(var-dvar)}\)
- \(\text{FOUND} : \text{collection(var-dvar)}\)

**Restraints**

\[
\begin{align*}
\text{sv} \leq 2 \lor \text{rv} \leq 1 & \Rightarrow \text{VALUE} = \text{sv} + 1 \\
\text{VALUE} \geq 1 & \\
\text{rv} = 2 & \Rightarrow \text{VALUE} = \text{sv} + 1 \lor \text{VALUE} \leq 1 \\
\text{rv} \geq 3 & \Rightarrow \text{VALUE} = \text{sv} + 1 \lor \text{VALUE} \leq \text{sv} - 2 \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{required}(\text{FOUND}, \text{var}) & \\
\text{where} & \\
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

The constraint MIN_WIDTH_GORGE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern GORGE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression \((> | > (= | >)* >)(< | < (= | <)* <)\).

Assume that the occurrence of the pattern GORGE starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

**Purpose**

- **Typical**
  - \(|\text{VARIABLES}| > 2\)
  - \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.

**Example**

\[
\begin{pmatrix}
1, (1, 7, 3, 4, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7),
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0)
\end{pmatrix}
\]

Figure 3.1100 provides an example where the POS_MIN_WIDTH_GORGE (1, [1, 7, 3, 4, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]) constraint holds.
Figure 3.1100: Illustrating the POS_MIN_WIDTH_GORGE constraint of the Example slot
Automaton

Similar to the automaton of the MIN_WIDTH_GORGE constraint but use the decoration table 2.33.
3.536  

**POS_MIN_WIDTH_INCREASING_SEQUENCE**

**Description**

Based on constraint `MIN_WIDTH_INCREASING_SEQUENCE`.

**Constraint**

`POS_MIN_WIDTH_INCREASING_SEQUENCE(VALUE, VARIABLES, FOUND)`

**Arguments**

- `VALUE` : `dvar`
- `VARIABLES` : `collection(var-dvar)`
- `FOUND` : `collection(var-dvar)`

**Restrictions**

- `sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = sv + 1`
- `VALUE \geq 2`
- `rv = 2 \Rightarrow VALUE = sv + 1 \lor VALUE \leq 2v`
- `rv \geq 3 \Rightarrow VALUE = sv + 1 \lor VALUE \leq sv$
- `required(VARIABLES, var)`
- `required(FOUND, var)`

where

- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

The constraint `MIN_WIDTH_INCREASING_SEQUENCE(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `INCREASING_SEQUENCE` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern `INCREASING_SEQUENCE` is the maximal subsequence which matches the regular expression `< (< | =)^* < | <`.

Assume that the occurrence of the pattern `INCREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j - i + 2`.

**Example**

```latex
\begin{array}{c}
2, (4, 3, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3),
\end{array}
\begin{array}{c}
\{0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0\},
\end{array}
```

Figure 3.1101 provides an example where the `POS_MIN_WIDTH_INCREASING_SEQUENCE` (2, [4, 3, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3], [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]) constraint holds.

**Typical**

- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`

**Arg. properties**

- Functional dependency: `VALUE` determined by `VARIABLES`.
- Functional dependency: `FOUND` determined by `VARIABLES`.  

**Purpose**

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern `INCREASING_SEQUENCE` is the maximal subsequence which matches the regular expression `< (< | =)^* < | <`.

Assume that the occurrence of the pattern `INCREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j - i + 2`.
Figure 3.1101: Illustrating the POS_MIN_WIDTH_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Similar to the automaton of the MIN_WIDTH_INCREASING_SEQUENCE constraint but use the decoration table 2.33.
3.537 POS_MIN_WIDTH_INCREASING_TERRACE

**DESCRIPTION**  

**AUTOMATON**

**Origin**  
Based on constraint MIN_WIDTH_INCREASING_TERRACE.

**Constraint**  
POS_MIN_WIDTH_INCREASING_TERRACE(VALUE, VARIABLES, FOUND)

**Arguments**  
- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)
- **FOUND**: collection(var−dvar)

**Restrictions**  
\[ sv \leq 3 \lor rv \leq 2 \Rightarrow VALUE = sv + 1 \]  
\[ VALUE \geq 2 \]  
\[ VALUE = sv + 1 \lor VALUE \leq sv - 2 \]  
\[ \text{required}(\text{VARIABLES}, \text{var}) \]  
\[ \text{required}(\text{FOUND}, \text{var}) \]  
where  
\[ sv = |\text{VARIABLES}| \]  
\[ rv = \text{range}(\text{VARIABLES}.\text{var}) \]

The constraint MIN_WIDTH_INCREASING_TERRACE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern INCREASING_TERRACE for which the feature value is VALUE.

**Purpose**

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression \(<-=+<\). Assume that the occurrence of the pattern INCREASING_TERRACE starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

**Example**

\[
\begin{align*}
&2,(1,3,3,3,2,5,6,4,4,2,3,3,3,4,4), \\
&\{0,0,0,0,0,0,1,0,0,0,0,0\}
\end{align*}
\]

Figure 3.1102 provides an example where the POS_MIN_WIDTH_INCREASING_TERRACE (2, [1, 3, 3, 3, 2, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4], [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

**Typical**

- **|VARIABLES| > 3**
- **range(VARIABLES.var) > 2**

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Figure 3.1102: Illustrating the POS_MIN_WIDTH_INCREASING_TERRACE constraint of the Example slot
Automaton

Similar to the automaton of the \texttt{MIN\_WIDTH\_INCREASING\_TERRACE} constraint but use the decoration table \texttt{2.33}. 
3.538  POS_MIN_WIDTH_INFLEXION

Origin  Based on constraint MIN_WIDTH_INFLEXION.

Constraint  POS_MIN_WIDTH_INFLEXION(VALUE, VARIABLES, FOUND)

Arguments  
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions  
sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = sv + 1
VALUE ≥ 1
VALUE = sv + 1 ∨ VALUE ≤ sv − 2
required(VARIABLES, var)
required(FOUND, var)
where
sv = |VARIABLES|
rv = range(VARIABLES.var)

Purpose  
The constraint MIN_WIDTH_INFLEXION(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern INFLEXION for which the feature value is VALUE. The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression < (< | =)> | > (>| =)> <.
Assume that the occurrence of the pattern INFLEXION starts at position i and ends at position j. The feature WIDTH computes the value j − i.

Example  
\[
\begin{pmatrix}
1, (1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4), \\
0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 3.1103 provides an example where the POS_MIN_WIDTH_INFLEXION (1, [1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4], [0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0]) constraint holds.

Typical  
|VARIABLES| > 2
range(VARIABLES.var) > 1

Arg. properties  
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Figure 3.1103: Illustrating the POS_MIN_WIDTH_INFLEXION constraint of the Example slot
Automaton

Similar to the automaton of the MIN_WIDTH_INFLEXION constraint but use the decoration table 2.33.
3.539  POS_MIN_WIDTH_PEAK

### Description

Based on constraint MIN_WIDTH_PEAK.

#### Constraint

\[
\text{POS_MIN_WIDTH_PEAK(VALUE, VARIABLES, FOUND)}
\]

#### Arguments

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)
- **FOUND**: collection(var−dvar)

#### Restrictions

\[
s v \leq 2 \lor r v \leq 1 \Rightarrow \text{VALUE} = s v + 1
\]

\[
\text{VALUE} \geq 1
\]

\[
\text{VALUE} = s v + 1 \lor \text{VALUE} \leq s v - 2
\]

**required(VARIABLES, var)**

**required(FOUND, var)**

where

\[
s v = |\text{VARIABLES}|
\]

\[
r v = \text{range}(\text{VARIABLES}.\text{var})
\]

The constraint \(\text{MIN_WIDTH_PEAK(VALUE, VARIABLES)}\) holds. In addition, **FOUND** is a collection of 0/1 variables where the value 1 indicates the position of the **found** letter in those occurrences of the pattern **PEAK** for which the feature value is **VALUE**.

The position of the **found** letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern **PEAK** is the maximal subsequence which matches the regular expression \(< (= | <)^* (> | =)^* >\).

Assume that the occurrence of the pattern **PEAK** starts at position \(i\) and ends at position \(j\). The feature **WIDTH** computes the value \(j - i\).

#### Example

\[
(2, 7, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 1),
(0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
\]

Figure 3.1104 provides an example where the **POS_MIN_WIDTH_PEAK** (2, [7, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 1], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

#### Typical

\[
|\text{VARIABLES}| > 2
\]

\[
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

#### Arg. properties

- **Functional dependency**: **VALUE** determined by **VARIABLES**.
- **Functional dependency**: **FOUND** determined by **VARIABLES**.
Figure 3.1104: Illustrating the POS_MIN_WIDTH_PEAK constraint of the Example slot
Automaton

Similar to the automaton of the MIN_WIDTH_PEAK constraint but use the decoration table 2.33.
3.540 POS_MIN_WIDTH_plain

Origin
Based on constraint MIN_WIDTH_plain.

Constraint
POS_MIN_WIDTH_plain(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions
\[ sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = sv + 1 \]
VALUE \geq 1
VALUE = sv + 1 \lor VALUE \leq sv - 2 \]
required(VARIABLES, var)
required(FOUND, var)
where
sv = |VARIABLES|
rv = range(VARIABLES.var)

Purpose
The constraint MIN_WIDTH_plain(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern plain for which the feature value is VALUE.
The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

Example
(1, (2, 3, 6, 5, 7, 6, 4, 5, 4, 3, 6, 6, 3), (0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0))

Typical
|VARIABLES| > 2
range(VARIABLES.var) > 1

Arg. properties
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Figure 3.1105: Illustrating the `POS_MIN_WIDTH_PLAIN` constraint of the Example slot
Automaton

Similar to the automaton of the MIN_WIDTHPLAIN constraint but use the decoration table 2.33.
3.541 POS_MIN_WIDTH_PLATEAU

**DESCRIPTION**

**Origin**
Based on constraint MIN_WIDTH_PLATEAU.

**Constraint**
POS_MIN_WIDTH_PLATEAU(VALUE, VARIABLES, FOUND)

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALUE</td>
<td>dvar</td>
</tr>
<tr>
<td>VARIABLES</td>
<td>collection(var−dvar)</td>
</tr>
<tr>
<td>FOUND</td>
<td>collection(var−dvar)</td>
</tr>
</tbody>
</table>

**Restrictions**

\[
\begin{align*}
sv & \leq 2 \lor rv \leq 1 \Rightarrow VALUE = sv + 1 \\
VALUE & \geq 1 \\
VALUE = sv + 1 \lor VALUE \leq sv - 2 & \neq 0 \\
required(VARIABLES, var) & \\
required(FOUND, var) & \\
\text{where} & \\
sv &= |VARIABLES| \\
rv &= \text{range}(VARIABLES.var)
\end{align*}
\]

**Purpose**

The constraint MIN_WIDTH_PLATEAU(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern PLATEAU for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression \( < = ? > \).

Assume that the occurrence of the pattern PLATEAU starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

**Example**

\[
\begin{pmatrix}
3, (1, 3, 3, 5, 5, 5, 2, 4, 4, 4, 3, 3, 1, 5, 5), \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
\end{pmatrix}
\]

Figure 3.1106 provides an example where the POS_MIN_WIDTH_PLATEAU (3, [1, 3, 3, 5, 5, 5, 2, 4, 4, 4, 3, 3, 1, 5, 5], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

**Typical**

<table>
<thead>
<tr>
<th>Variable Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>(</td>
<td>VARIABLES</td>
</tr>
<tr>
<td>range(VARIABLES.var) &gt; 1</td>
<td></td>
</tr>
</tbody>
</table>

**Arg. properties**

- **Functional dependency**: VALUE determined by VARIABLES.
- **Functional dependency**: FOUND determined by VARIABLES.
Figure 3.1106: Illustrating the POS_MIN_WIDTH_PLATEAU constraint of the Example slot
Automaton

Similar to the automaton of the MIN_WIDTH_PLATEAU constraint but use the decoration table 2.33.
POS_MIN_WIDTH_PLATEAU 2295
Based on constraint \( \text{MIN_WIDTH_PROPER_PLAIN} \).

**Arguments**
- \( \text{VALUE} : \text{dvar} \)
- \( \text{VARIABLES} : \text{collection}(\text{var} - \text{dvar}) \)
- \( \text{FOUND} : \text{collection}(\text{var} - \text{dvar}) \)

**Restrictions**
- \[ \text{sv} \leq 3 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = \text{sv} + 1 \]
- \( \text{VALUE} \geq 2 \)
- \( \text{VALUE} = \text{sv} + 1 \lor \text{VALUE} \leq \text{sv} - 2 \text{dvar} \)
- \( \text{required}(\text{VARIABLES}, \text{var}) \)
- \( \text{required}(\text{FOUND}, \text{var}) \)

where
- \( \text{sv} = |\text{VARIABLES}| \)
- \( \text{rv} = \text{range}(\text{VARIABLES}.\text{var}) \)

The constraint \( \text{MIN_WIDTH_PROPER_PLAIN}(\text{VALUE}, \text{VARIABLES}) \) holds. In addition, \( \text{FOUND} \) is a collection of 0/1 variables where the value 1 indicates the position of the \text{found} letter in those occurrences of the pattern \text{PROPER_PLAIN} for which the feature value is \text{VALUE}.

The position of the \text{found} letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern \text{PROPER_PLAIN} is the maximal subsequence which matches the regular expression \( > =+< \).

Assume that the occurrence of the pattern \text{PROPER_PLAIN} starts at position \( i \) and ends at position \( j \). The feature \text{WIDTH} computes the value \( j - i \).

**Example**

\[
\begin{pmatrix}
2, (2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5),
0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 3.1107 provides an example where the \( \text{POS_MIN_WIDTH_PROPER_PLAIN} \) constraint holds.

**Typical**
- \( |\text{VARIABLES}| > 3 \)
- \( \text{range}(\text{VARIABLES}.\text{var}) > 1 \)

**Arg. properties**
- Functional dependency: \text{VALUE} determined by \text{VARIABLES}.
- Functional dependency: \text{FOUND} determined by \text{VARIABLES}.
Figure 3.1107: Illustrating the POS_MIN_WIDTH_PROPERPLAIN constraint of the Example slot
Automaton

Similar to the automaton of the `MIN_WIDTH_PROPERPLAIN` constraint but use the decoration table 2.33.
3.543 POS_MIN_WIDTH_PROPER_PLATEAU

**Origin**
Based on constraint `MIN_WIDTH_PROPER_PLATEAU`.

**Constraint**
`POS_MIN_WIDTH_PROPER_PLATEAU(VALUE, VARIABLES, FOUND)`

**Arguments**
- `VALUE : dvar`
- `VARIABLES : collection(var−dvar)`
- `FOUND : collection(var−dvar)`

**Restrictions**
- `sv ≤ 3 ∨ rv ≤ 1 ⇒ VALUE = sv + 1`
- `VALUE ≥ 2`
- `VALUE = sv + 1 ∨ VALUE ≤ sv − 2v`
- `required(VARIABLES, var)`
- `required(FOUND, var)`

where
- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

The constraint `MIN_WIDTH_PROPER_PLATEAU(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern `PROPER_PLATEAU` for which the feature value is `VALUE`.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `PROPER_PLATEAU` is the maximal subsequence which matches the regular expression `<=+>`. Assume that the occurrence of the pattern `PROPER_PLATEAU` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j − i`.

**Purpose**

**Example**

```
( 2, (7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3),
  (0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0)
)
```

Figure 3.1108 provides an example where the `POS_MIN_WIDTH_PROPER_PLATEAU` constraint holds.

**Typical**
- `|VARIABLES| > 3`
- `range(VARIABLES.var) > 1`

**Arg. properties**
- Functional dependency: `VALUE` determined by `VARIABLES`.
- Functional dependency: `FOUND` determined by `VARIABLES`.
Figure 3.1108: Illustrating the POS_MIN_WIDTH_PROPER_PLATEAU constraint of the Example slot
Automaton

Similar to the automaton of the MIN_WIDTH_PROPER_PLATEAU constraint but use the decoration table 2.33.
3.544  POS_MIN_WIDTH_STEADY_SEQUENCE

DESCRIPTION  AUTOMATON

Origin
Based on constraint MIN_WIDTH_STEADY_SEQUENCE.

Constraint
POS_MIN_WIDTH_STEADY_SEQUENCE(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions

\[
\begin{align*}
sv \leq 1 & \Rightarrow VALUE = sv + 1 \\
rv = 1 & \Rightarrow VALUE \geq sv \\
rv \geq 2 & \Rightarrow VALUE \geq 2 \\
VALUE = sv + 1 & \lor VALUE \leq sv \\
\text{required}(VARIABLES, var) & \\
\text{required}(FOUND, var)
\end{align*}
\]

where

\[
\begin{align*}
rv &= \text{range}(VARIABLES, var) \\
sv &= |VARIABLES|
\end{align*}
\]

Purpose
The constraint MIN_WIDTH_STEADY_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern STEADY_SEQUENCE for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern STEADY_SEQUENCE is the maximal subsequence which matches the regular expression \(+\).

Assume that the occurrence of the pattern STEADY_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i + 2\).

Example

\[
\begin{pmatrix}
2, (3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1), \\
0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0
\end{pmatrix}
\]

Figure 3.1109 provides an example where the POS_MIN_WIDTH_STEADY_SEQUENCE (2, [3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1], [0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0]) constraint holds.

Typical

\(|VARIABLES| > 1\)

Arg. properties

- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Figure 3.1109: Illustrating the POS_MIN_WIDTH_STEADY_SEQUENCE constraint of the Example slot
Automaton

Similar to the automaton of the $MIN\_WIDTH\_STEADY\_SEQUENCE$ constraint but use the decoration table 2.33.
### Description

**Origin**
Based on constraint `MIN_WIDTH_STRICTLY_DECREASING_SEQUENCE`.

**Constraint**
`POS_MIN_WIDTH_STRICTLY_DECREASING_SEQUENCE(VALUE, VARIABLES, FOUND)`

**Arguments**
- **VALUE**: `dvar`
- **VARIABLES**: `collection(var−dvar)`
- **FOUND**: `collection(var−dvar)`

**Restrictions**
- $sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = sv + 1$
- $VALUE \geq 2$
- $VALUE = sv + 1 \lor VALUE \leq \min(sv, rv)$
- `required(VARIABLES.var)`
- `required(FOUND.var)`

where
- $sv = |VARIABLES|$
- $rv = \text{range}(VARIABLES.var)$

The constraint `MIN_WIDTH_STRICTLY_DECREASING_SEQUENCE(VALUE, VARIABLES)` holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern `STRICTLY_DECREASING_SEQUENCE` for which the feature value is VALUE.

The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

An occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` is the maximal subsequence which matches the regular expression $>^+$. Assume that the occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` starts at position $i$ and ends at position $j$. The feature WIDTH computes the value $j - i + 2$.

### Purpose

**Example**

Figure 3.1110 provides an example where the `POS_MIN_WIDTH_STRICTLY_DECREASING_SEQUENCE` constraint holds.

<table>
<thead>
<tr>
<th>VARIABLES</th>
<th>&gt; 1</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>range(VARIABLES.var)</code></td>
<td>&gt; 1</td>
</tr>
</tbody>
</table>

**Arg. properties**
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Figure 3.110: Illustrating the POS_MIN_WIDTH.Strictly_Decreasing_Sequence constraint of the Example slot
Automaton

Similar to the automaton of the $\text{MIN\_WIDTH\_STRICLY\_DECREASING\_SEQUENCE}$ constraint but use the decoration table 2.33.
3.546  POS_MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE  

**DESCRIPTION**  

**Origin**  
Based on constraint MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE.

**Constraint**  
POS_MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES, FOUND)

**Arguments**  
\[\begin{align*} 
\text{VALUE} : & \ dvar \\
\text{VARIABLES} : & \ \text{collection}(\text{var} - \text{dvar}) \\
\text{FOUND} : & \ \text{collection}(\text{var} - \text{dvar}) 
\end{align*}\]

**Restrictions**  
\[\begin{align*} 
\text{sv} \leq 1 \lor \text{rv} \leq 1 & \Rightarrow \text{VALUE} = \text{sv} + 1 \\
\text{VALUE} \geq 2 & \\
\text{VALUE} = \text{sv} + 1 \lor \text{VALUE} \leq \min(\text{sv}, \text{rv}) & \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{required}(\text{FOUND}, \text{var}) & \\
\text{where} & \\
\text{sv} = |\text{VARIABLES}| & \\
\text{rv} = \text{range}(\text{VARIABLES}.\text{var}) & \\
\end{align*}\]

The constraint MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES) holds. In addition, FOUND is a collection of 0/1 variables where the value 1 indicates the position of the found letter in those occurrences of the pattern STRICTLY_INCREASING_SEQUENCE for which the feature value is VALUE. The position of the found letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression \(<^+\). Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i + 2\).

**Example**  
\[\left( \begin{array}{c} 
2, (4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3), \\
(0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) 
\end{array} \right)\]

Figure 3.1111 provides an example where the POS_MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE (2, [4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3], [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]) constraint holds.

**Typical**  
\[|\text{VARIABLES}| > 1\]
\[\text{range}(\text{VARIABLES}.\text{var}) > 1\]

**Arg. properties**  
- **Functional dependency:** VALUE determined by VARIABLES.
- **Functional dependency:** FOUND determined by VARIABLES.
Figure 3.1111: Illustrating the POS_MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE constraint of the Example slot
Automaton Similar to the automaton of the `MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE` constraint but use the decoration table 2.33.
3.547  **POS_MIN_WIDTH_SUMMIT**

**Origin**
Based on constraint **MIN_WIDTH_SUMMIT**.

**Constraint**

\[
\text{POS_MIN_WIDTH_SUMMIT(VALE, VARIABLES, FOUND)}
\]

**Arguments**

VALUE : dvar  
VARIABLES : collection(var−dvar)  
FOUND : collection(var−dvar)

**Restrictions**

\[
\begin{align*}
sv &\leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = sv + 1 \\
\text{VALUE} &\geq 1 \\
rv &\geq 2 \Rightarrow \text{VALUE} = sv + 1 \lor \text{VALUE} \leq 10 \\
rv &\geq 3 \Rightarrow \text{VALUE} = sv + 1 \lor \text{VALUE} \leq sv - 2 \\
\text{required}(\text{VARIABLES}, \text{var}) \\
\text{required}(\text{FOUND}, \text{var})
\end{align*}
\]

where

\[
\begin{align*}
sv &= |\text{VARIABLES}| \\
rv &= \text{range}(\text{VARIABLES.var})
\end{align*}
\]

The constraint **MIN_WIDTH_SUMMIT(VALE, VARIABLES)** holds. In addition, **FOUND** is a collection of 0/1 variables where the value 1 indicates the position of the **found** letter in those occurrences of the pattern **SUMMIT** for which the feature value is **VALUE**. The position of the **found** letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete. An occurrence of the pattern **SUMMIT** is the **maximal** subsequence which matches the regular expression \( (< | < ( = | <)^* < | > | > ( = | >)^* > ) \). Assume that the occurrence of the pattern **SUMMIT** starts at position \( i \) and ends at position \( j \). The feature **WIDTH** computes the value \( j - i \).

**Purpose**

**Example**

\[
\begin{pmatrix} 
1, 7, 1, 5, 4, 4, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1, 0 \\
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0
\end{pmatrix}
\]

Figure 3.1112 provides an example where the **POS_MIN_WIDTH_SUMMIT** (1, [7, 1, 5, 4, 4, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]) constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| &> 2 \\
\text{range}(\text{VARIABLES.var}) &> 1
\end{align*}
\]

**Arg. properties**

- Functional dependency: **VALUE** determined by **VARIABLES**.
- Functional dependency: **FOUND** determined by **VARIABLES**.
Figure 3.1112: Illustrating the POS_MIN_WIDTH_SUMMIT constraint of the Example slot
Automaton

Similar to the automaton of the *MIN_WIDTH_SUMMIT* constraint but use the decoration table 2.33.
### 3.548  POS_MIN_WIDTH_VALLEY

**DESCRIPTION**

Based on constraint `MIN_WIDTH_VALLEY`.

**Constraint**

`POS_MIN_WIDTH_VALLEY(VALUE, VARIABLES, FOUND)`

**Arguments**

- `VALUE`: `dvar`
- `VARIABLES`: `collection(var−dvar)`
- `FOUND`: `collection(var−dvar)`

**Restrictions**

- `sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = sv + 1`
- `VALUE ≥ 1`
- `VALUE = sv + 1 ∨ VALUE ≤ sv − 2`
- `required(VARIABLES, var)`
- `required(FOUND, var)`

where

- `sv = |VARIABLES|`
- `rv = range(VARIABLES.var)`

The constraint `MIN_WIDTH_VALLEY(VALUE, VARIABLES)` holds. In addition, `FOUND` is a collection of 0/1 variables where the value 1 indicates the position of the `found` letter in those occurrences of the pattern `VALLEY` for which the feature value is `VALUE`.

The position of the `found` letter in an occurrence of a pattern is the first position where the occurrence of pattern is identified, even if the pattern is not complete.

**Purpose**

An occurrence of the pattern `VALLEY` is the *maximal* subsequence which matches the regular expression `>(= | >)*(< | |)= <`. Assume that the occurrence of the pattern `VALLEY` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j − i`.

**Example**

```
  2,1,3,7,4,3,6,6,5,3,3,2,6,5,5,7,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0
```

Figure 3.1113 provides an example where the `POS_MIN_WIDTH_VALLEY (2, [1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 7, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])` constraint holds.

**Typical**

- `|VARIABLES| > 2`
- `range(VARIABLES.var) > 1`

**Arg. properties**

- **Functional dependency**: `VALUE` determined by `VARIABLES`.
- **Functional dependency**: `FOUND` determined by `VARIABLES`.
Figure 3.1113: Illustrating the POS_MIN_WIDTH_VALLEY constraint of the Example slot
Automaton

Similar to the automaton of the `MIN_WIDTH_VALLEY` constraint but use the decoration table 2.33.
3.549  POS_MIN_WIDTH_ZIGZAG

DESCRIPTION

Origin
Based on constraint MIN_WIDTH_ZIGZAG.

Constraint
POS_MIN_WIDTH_ZIGZAG(VALUE, VARIABLES, FOUND)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)
FOUND : collection(var−dvar)

Restrictions
\( sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = sv + 1 \)
\( VALUE \geq 2 \)
\( VALUE = sv + 1 \lor VALUE \leq sv - 2 \)
\( required(VARIABLES, var) \)
\( required(FOUND, var) \)

where
\( sv = |VARIABLES| \)
\( rv = range(VARIABLES.var) \)

Purpose
An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression \((<>)^+(<=|<>|)(<>)^+(>=|<>))\.

Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

Example

\[
\begin{pmatrix}
2, 4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1 \\
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
\end{pmatrix}
\]

Figure 3.1114 provides an example where the POS_MIN_WIDTH_ZIGZAG constraint holds.

Typical
\(|VARIABLES| > 3\)
\(range(VARIABLES.var) > 1\)

Arg. properties
- Functional dependency: VALUE determined by VARIABLES.
- Functional dependency: FOUND determined by VARIABLES.
Figure 3.1114: Illustrating the POS_MIN_WIDTH_ZIGZAG constraint of the Example slot
Automaton  

Similar to the automaton of the `MIN_WIDTH_ZIGZAG` constraint but use the decoration table 2.33.
**3.550 SUM_HEIGHT_DECREASING_TERRACE**

**Description**

Origin: Based on the `DECREASING_TERRACE` pattern.

Constraint: 

```
SUM_HEIGHT_DECREASING_TERRACE(VALUE, VARIABLES)
```

Arguments:

- `VALUE` : `dvar`
- `VARIABLES` : `collection(var—dvar)`
Restrictions

\( sv \leq 3 \vee rv \leq 2 \Rightarrow VALUE = 0 \)

\[
VALUE \geq \min_{q \in [lb1,ub1]} \min \left\{ 0, \sum \left( \prod \left( \min \left( 1, \max \left( 0, rv - 2 \right) \right) \right) \right) \right\}
\]

\[
VALUE \leq \max_{q \in [lb2,ub2]} \max \left\{ 0, \sum \left( \prod \left( \min \left( 1, \max \left( 0, rv - 2 \right) \right) \right) \right) \right\}
\]

required(VARIABLES, var)

where

\( sv = |VARIABLES| \)

\( rv = \text{range}(VARIABLES, \text{var}) \)

\( np1 = \max \left\{ 0, \sum \left( \min \left( 0, \min(rv, \minv + 1) - \left( (\max(1, sv - sv \ mod \ 2) - 2 * q) / 2 * q \right) \right) \right) \right\} \)

\( np2 = \max \left\{ 0, \sum \left( \min \left( 0, \min(rv, \maxv + 1) - \left( (\max(1, sv - sv \ mod \ 2) - 2 * q) / 2 * q \right) \right) \right) \right\} \)

\( \maxv = \maxval(VARIABLES, \text{var}) \)

\( \minv = \minval(VARIABLES, \text{var}) \)

\( lb1 = \min \left\{ \sum \left( \max(1, sv - sv \ mod \ 2) / \min \left( \max(1, sv - sv \ mod \ 2), \max(1, (\min(\minv + 1, rv) - 2) * 2 + 2)\right) \right)\right\} \)

\( ub1 = \lceil sv / 4 \rceil + 1 \)

\( lb2 = \min \left\{ \sum \left( \max(1, sv - sv \ mod \ 2) / \min \left( \max(1, sv - sv \ mod \ 2), \max(1, (\min(\maxv + 1, rv) - 2) * 2 + 2)\right) \right)\right\} \)

\( ub2 = \lceil sv / 4 \rceil + 1 \)
VALUE is the sum of all minimum values in each occurrence of the DECREASING_TERRACE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern DECREASING_TERRACE is the maximal subsequence which matches the regular expression \( > = + > \).

Assume that the occurrence of the pattern DECREASING_TERRACE starts at position \( i \) and ends at position \( j \). The feature MIN computes the minimum of the values from index \( i + 1 \) to index \( j \).

**Purpose**

**Example**

\((6, (6, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3))\)

---

Figure 3.1115 provides an example where the SUM_HEIGHT_DECREASING_TERRACE (6, [6, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3, 3]) constraint holds.

---

**Typical**

\(|\text{VARIABLES}| > 3\)

\(\text{range}(\text{VARIABLES}.\text{var}) > 2\)

**Arg. properties**

**Functional dependency**: VALUE determined by VARIABLES.
Figures 3.1116 and 3.1117 respectively depict the automaton associated with the constraint SUM_HEIGHT_DECREASING_TERRACE and its simplified form.

Figure 3.1116: Automaton for the SUM_HEIGHT_DECREASING_TERRACE constraint obtained by applying decoration Table 2.35 to the seed transducer of the DECREASING_TERRACE pattern where default is 0

Figure 3.1117: Automaton for the SUM_HEIGHT_DECREASING_TERRACE constraint obtained by applying decoration Table 2.37 to the seed transducer of the DECREASING_TERRACE pattern where default is 0
Table 3.142: Glue matrix for the SUM_HEIGHT_DECREASING_TERRACE constraint defined as the composition of the DECREASING_TERRACE pattern, the feature MIN, and the aggregator sum. Cells of the glue matrix are coloured with the colour of the constituent to which they are related.
SUM_HEIGHT_DECREASING_TERRACE

2333
Origin

Based on the `INCREASING_TERRACE` pattern.

Constraint

`SUM_HEIGHT_INCREASING_TERRACE(VALUE, VARIABLES)`

Arguments

```
VALUE    : dvar
VARIABLES : collection(var–dvar)
```
Restrictions

\[
sv \leq 3 \land rv \leq 2 \Rightarrow VALUE = 0
\]

\[
\text{VALUE} \geq \min_{q \in \{1b1, ub1\}} \min \left( \begin{array}{c}
0, \\
0, \\
\min \left( 0, \min(rv, [minv] + 1) - \right)
\end{array} \right)
\]

\[
\text{VALUE} \leq \max_{q \in \{1b2, ub2\}} \max \left( \begin{array}{c}
0, \\
0, \\
\min \left( 0, \min(rv, [maxv] + 1) - \right)
\end{array} \right)
\]

\[
\text{required(VARIABLES, var)}
\]

\[
sv = |\text{VARIABLES}|
\]

\[
rv = \text{range(VARIABLES, var)}
\]

\[
np1 = \max \left( \begin{array}{c}
0, \\
0, \\
\min \left( 0, \min(rv, [minv] + 1) - \right)
\end{array} \right)
\]

\[
np2 = \max \left( \begin{array}{c}
0, \\
0, \\
\min \left( 0, \min(rv, [maxv] + 1) - \right)
\end{array} \right)
\]

\[
maxv = \text{maxval(VARIABLES, var)}
\]

\[
minv = \text{minval(VARIABLES, var)}
\]

\[
lb1 = \min \left( \begin{array}{c}
|sv/4| + 1,
\end{array} \right)
\]

\[
ub1 = |sv/4| + 1
\]

\[
lb2 = \min \left( \begin{array}{c}
|sv/4| + 1,
\end{array} \right)
\]

\[
ub2 = |sv/4| + 1
\]
VALUE is the sum of all minimum values in each occurrence of the INCREASING_TERRACE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression \(<=+<\).

Assume that the occurrence of the pattern INCREASING_TERRACE starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i+1\) to index \(j\).

**Example**

\((8, (1, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4))\)

Figure 3.1118 provides an example where the SUM_HEIGHT_INCREASING_TERRACE \((8, [1, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 3, 4, 4])\) constraint holds.

Figure 3.1118: Illustrating the SUM_HEIGHT_INCREASING_TERRACE constraint of the Example slot

**Typical**

\(|\text{VARIABLES}| > 3\)

\(\text{range}(\text{VARIABLES}.\text{var}) > 2\)

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 3.1119 and 3.1120 respectively depict the automaton associated with the constraint `SUM_HEIGHT_INCREASING_TERRACE` and its simplified form.

\[
\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow +\infty \\
R &\leftarrow \text{default} \\
\end{align*}
\]

\[
\begin{align*}
\{ D \leftarrow \min(D, \text{VAR}_i) \} \\
\{ R \leftarrow R + \min(D, \text{VAR}_i) \} \\
\end{align*}
\]

Figure 3.1119: Automaton for the `SUM_HEIGHT_INCREASING_TERRACE` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `INCREASING_TERRACE` pattern where `default` is 0.

\[
\begin{align*}
R &\leftarrow \text{default} \\
\end{align*}
\]

\[
\begin{align*}
\{ R \leftarrow R + \text{VAR}_i \} \\
\end{align*}
\]

Figure 3.1120: Automaton for the `SUM_HEIGHT_INCREASING_TERRACE` constraint obtained by applying decoration Table 2.37 to the seed transducer of the `INCREASING_TERRACE` pattern where `default` is 0.
Table 3.143: Glue matrix for the `SUM_HEIGHT_INCREASING_TERRACE` constraint defined as the composition of the `INCREASING_TERRACE` pattern, the feature `MIN`, and the aggregator `sum`. Cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### 3.552 SUM_HEIGHT_PLAIN

**Origin**
Based on the **PLAIN** pattern.

**Constraint**
\[ \text{SUM_HEIGHT_PLAIN}(\text{VALUE}, \text{VARIABLES}) \]

**Arguments**
- VALUE : `dvar`
- VARIABLES : `collection(var-dvar)`

**Restrictions**
- \[ sv \leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = 0 \]
- \[ \text{VALUE} = 0 \lor \text{VALUE} \geq \min(\text{minv}, \text{minv} \cdot \text{np}) \]
- \[ \text{VALUE} = 0 \lor \text{VALUE} \leq \max(\text{maxv} - 1, (\text{maxv} - 1) \cdot \text{np}) \]

**Purpose**
An occurrence of the pattern **PLAIN** is the maximal subsequence which matches the regular expression \(>^*<_\).
Assume that the occurrence of the pattern **PLAIN** starts at position \(i\) and ends at position \(j\). The feature \(\text{MIN}\) computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**
\[(12, (2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3))\]

Figure 3.1121 provides an example where the **SUM_HEIGHT_PLAIN** \((12, [2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3])\) constraint holds.

**Typical**
- \(|\text{VARIABLES}| > 2\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Symmetry**
Items of \(\text{VARIABLES}\) can be reversed.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.1121: Illustrating the `SUM_HEIGHT_PLAIN` constraint of the `Example` slot
Automaton

Figures 3.1122 and 3.1123 respectively depict the automaton associated with the constraint SUM_HEIGHTPLAIN and its simplified form.

Figure 3.1122: Automaton for the SUM_HEIGHTPLAIN constraint obtained by applying decoration Table 2.35 to the seed transducer of the PLAIN pattern where default is 0

Figure 3.1123: Automaton for the SUM_HEIGHTPLAIN constraint obtained by applying decoration Table 2.37 to the seed transducer of the PLAIN pattern where default is 0
Table 3.144: Glue matrix for the SUM, HEIGHT, PLAIN constraint defined as the composition of the PLAIN pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\overrightarrow{c} + \overrightarrow{c}$</td>
<td>$\overrightarrow{c} + \overrightarrow{c}$</td>
<td>$\overrightarrow{c} + \overrightarrow{c}$</td>
</tr>
<tr>
<td>r</td>
<td>$\overrightarrow{c} + \overrightarrow{c}$</td>
<td>$\min(\overrightarrow{d}, \overrightarrow{d}, \text{VAR}_i)$</td>
<td>$\min(\overrightarrow{d}, \overrightarrow{d}, \text{VAR}_i)$</td>
</tr>
<tr>
<td>t</td>
<td>$\overrightarrow{c} + \overrightarrow{c}$</td>
<td>$\min(\overrightarrow{d}, \overrightarrow{d}, \text{VAR}_i)$</td>
<td>$\min(\overrightarrow{d}, \overrightarrow{d}, \text{VAR}_i)$</td>
</tr>
</tbody>
</table>
3.553 SUM_HEIGHT_PLATEAU

Origin
Based on the PLATEAU pattern.

Constraint
SUM_HEIGHT_PLATEAU(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
\[ sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = 0 \]
\[ VALUE = 0 \lor VALUE \geq \min(minv + 18, (minv + 1) * np) \]
\[ VALUE = 0 \lor VALUE \leq \max(maxv, maxv * np) \]

where
\[ sv = |VARIABLES| \]
\[ np = \max(0, \left\lfloor (sv - 1)/2 \right\rfloor) \]
\[ minv = \minval(VARIABLES.var) \]
\[ maxv = \maxval(VARIABLES.var) \]
\[ rv = \range(VARIABLES.var) \]

VALUE is the sum of all minimum values in each occurrence of the PLATEAU pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression \(<=*\). Assume that the occurrence of the pattern PLATEAU starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i + 1 to index j.

Example
\((12, [7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5])\)

Figure 3.1124 provides an example where the SUM_HEIGHT_PLATEAU \((12, [7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5])\) constraint holds.

Typical
\[ |VARIABLES| > 2 \]
\[ \range(VARIABLES.var) > 1 \]

Symmetry
Items of VARIABLES can be reversed.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 3.1124: Illustrating the SUM_HEIGHT_PLATEAU constraint of the Example slot
Automaton

Figures 3.1125 and 3.1126 respectively depict the automaton associated with the constraint SUM_HEIGHT_PLATEAU and its simplified form.

Figure 3.1125: Automaton for the SUM_HEIGHT_PLATEAU constraint obtained by applying decoration Table 2.35 to the seed transducer of the PLATEAU pattern where default is 0

Figure 3.1126: Automaton for the SUM_HEIGHT_PLATEAU constraint obtained by applying decoration Table 2.37 to the seed transducer of the PLATEAU pattern where default is 0
Table 3.145: Glue matrix for the $\text{SUM_HEIGHT_PLATEAU}$ constraint defined as the composition of the $\text{PLATEAU}$ pattern, the feature $\text{MIN}$, and the aggregator $\text{sum}$; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\vec{c} + \vec{c}$</td>
<td>$\vec{c} + \vec{c}$</td>
<td>$\vec{c} + \vec{c}$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\vec{c} + \vec{c}$</td>
<td>$\min(\vec{D}, \vec{D}, \text{VAR}_i)$</td>
<td>$\min(\vec{D}, \vec{D}, \text{VAR}_i)$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\vec{c} + \vec{c}$</td>
<td>$\min(\vec{D}, \vec{D}, \text{VAR}_i)$</td>
<td>$\min(\vec{D}, \vec{D}, \text{VAR}_i)$</td>
</tr>
</tbody>
</table>
3.554 SUM_HEIGHT_PROPER Plain

**DESCRIPTION**

**Origin**
Based on the PROPER Plain pattern.

**Constraint**
SUM_HEIGHT_PROPER Plain\((\text{VALUE}, \text{VARIABLES})\)

**Arguments**
\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} - \text{dvar})
\end{align*}
\]

**Restrictions**
\[
\begin{align*}
\text{sv} \leq 3 \lor \text{rv} \leq 1 & \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} = 0 \lor \text{VALUE} \geq \min(\text{minv}, \text{minv} \times \text{np}) & \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} = 0 \lor \text{VALUE} \leq \max(\text{maxv} - 1, (\text{maxv} - 1) \times \text{np}) & \text{required}(\text{VARIABLES}, \text{var}) \\
\text{where} & \\
\text{sv} & = |\text{VARIABLES}| \\
\text{np} & = \max(0, \lfloor (\text{sv} - 1)/3 \rfloor) \\
\text{minv} & = \minval(\text{VARIABLES}.\text{var}) \\
\text{maxv} & = \maxval(\text{VARIABLES}.\text{var}) \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**
VALUE is the sum of all minimum values in each occurrence of the PROPER Plain pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern PROPER Plain is the maximal subsequence which matches the regular expression \(\geq^{+}<_{\text{var}}\).

Assume that the occurrence of the pattern PROPER Plain starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**
\[
(12, (2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3, 5))
\]

Figure 3.1127 provides an example where the SUM_HEIGHT_PROPERPlain \((12, [2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 5])\) constraint holds.

**Typical**
\[
|\text{VARIABLES}| > 3 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.1127: Illustrating the SUM_HEIGHT_PROPER_PLAIN constraint of the Example slot
Figures 3.1128 and 3.1129 respectively depict the automaton associated with the constraint `SUM_HEIGHT_PROPER_PLAIN` and its simplified form.

Figure 3.1128: Automaton for the `SUM_HEIGHT_PROPER_PLAIN` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `PROPER_PLAIN` pattern where `default` is 0.

Figure 3.1129: Automaton for the `SUM_HEIGHT_PROPER_PLAIN` constraint obtained by applying decoration Table 2.37 to the seed transducer of the `PROPER_PLAIN` pattern where `default` is 0.
Table 3.146: Glue matrix for the \texttt{SUM\_HEIGHT\_PROPER\_PLAIN} constraint defined as the composition of the \texttt{PROPER\_PLAIN} pattern, the feature \texttt{MIN}, and the aggregator \texttt{sum}: cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>(s)</th>
<th>(r)</th>
<th>(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
</tr>
<tr>
<td>(r)</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\text{min}(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_i))</td>
</tr>
<tr>
<td>(t)</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\text{min}(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_i))</td>
<td>(\text{min}(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_i))</td>
</tr>
</tbody>
</table>
3.555 \textbf{SUM\_HEIGHT\_PROPER\_PLATEAU}

\begin{itemize}
  \item \textbf{Origin} Based on the \textit{PROPER\_PLATEAU} pattern.
  \item \textbf{Constraint} \texttt{SUM\_HEIGHT\_PROPER\_PLATEAU(VALUE, VARIABLES)}
  \item \textbf{Arguments}
    \begin{itemize}
      \item \texttt{VALUE} : \textit{dvar}
      \item \texttt{VARIABLES} : \textit{collection(var-dvar)}
    \end{itemize}
  \item \textbf{Restrictions}
    \begin{align*}
      sv & \leq 3 \lor rv \leq 1 \Rightarrow VALUE = 0 \\
      VALUE & = 0 \lor VALUE \geq \min (\minv + 1, (\minv + 1) \cdot np) \\
      VALUE & = 0 \lor VALUE \leq \max (\maxv, \maxv \cdot np) \texttt{required}(\text{VARIABLES}, \text{var})
    \end{align*}
    where
    \begin{align*}
      sv & = |\text{VARIABLES}| \\
      np & = \max (0, (sv - 1)/3) \\
      \minv & = \text{\texttt{minval(}\text{VARIABLES}.\text{var}\text{)}} \\
      \maxv & = \text{\texttt{maxval(}\text{VARIABLES}.\text{var}\text{)}} \\
      rv & = \text{\texttt{range(}\text{VARIABLES}.\text{var}\text{)}}
    \end{align*}
  \item \textbf{Purpose}
    \begin{itemize}
      \item VALUE is the sum of all minimum values in each occurrence of the \textit{PROPER\_PLATEAU} pattern in the time-series given by the \textit{VARIABLES} collection. If the pattern does not occur, VALUE takes the default value 0.
      \item An occurrence of the pattern \textit{PROPER\_PLATEAU} is the \textit{maximal} subsequence which matches the regular expression \texttt{< =+ >}.
      \item Assume that the occurrence of the pattern \textit{PROPER\_PLATEAU} starts at position \(i\) and ends at position \(j\). The feature \texttt{MIN} computes the minimum of the values from index \(i + 1\) to index \(j\).
    \end{itemize}
  \item \textbf{Example}
    \begin{itemize}
      \item \(12, (7, 1, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3)\)
    \end{itemize}
    \begin{itemize}
      \item Figure 3.1130 provides an example where the \texttt{SUM\_HEIGHT\_PROPER\_PLATEAU} \((12, [7, 1, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3])\) constraint holds.
    \end{itemize}
  \item \textbf{Typical}
    \begin{itemize}
      \item \(|\text{VARIABLES}| > 3 \land \text{range(}\text{VARIABLES}.\text{var}\text{)} > 1\)
    \end{itemize}
  \item \textbf{Symmetry}
    \begin{itemize}
      \item Items of \textit{VARIABLES} can be reversed.
    \end{itemize}
  \item \textbf{Arg. properties}
    \begin{itemize}
      \item \textit{Functional dependency}: VALUE determined by \textit{VARIABLES}.
    \end{itemize}
\end{itemize}
Figure 3.1130: Illustrating the `SUM_HEIGHT_PROPER_PLATEAU` constraint of the Example slot
Automaton

Figures 3.1131 and 3.1132 respectively depict the automaton associated with the constraint
SUM_HEIGHT_PROPER_PLATEAU and its simplified form.

Figure 3.1131: Automaton for the SUM_HEIGHT_PROPER_PLATEAU constraint obtained by applying decoration Table 2.35 to the seed transducer of the PROPER_PLATEAU pattern where default is 0

Figure 3.1132: Automaton for the SUM_HEIGHT_PROPER_PLATEAU constraint obtained by applying decoration Table 2.37 to the seed transducer of the PROPER_PLATEAU pattern where default is 0
### Table 3.147: Glue matrix for the `SUM_HEIGHT_PROPER_PLATEAU` constraint defined as the composition of the `PROPER_PLATEAU` pattern, the feature `MIN`, and the aggregator `sum`. Cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{C}$</td>
</tr>
<tr>
<td>r</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\min(\overline{D}, \overline{D}, \text{VAR}_i)$</td>
</tr>
<tr>
<td>t</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\min(\overline{D}, \overline{D}, \text{VAR}_i)$</td>
<td>$\min(\overline{D}, \overline{D}, \text{VAR}_i)$</td>
</tr>
</tbody>
</table>
3.556  SUM_HEIGHT_STEADY

**Origin**
Based on the STEADY pattern.

**Constraint**

```
SUM_HEIGHT_STEADY(VALUE, VARIABLES)
```

**Arguments**

```
VALUE : dvar
VARIABLES : collection(var−dvar)
```

**Restrictions**

```
sv ≤ 1 ⇒ VALUE = 0
rv = 1 ⇒ VALUE ≥ minv + np
rv ≥ 2 ⇒ VALUE = 0 ∨ VALUE ≥ min(minv, minv + np)
rv = 1 ⇒ VALUE ≤ maxv + np
rv ≥ 2 ⇒ VALUE = 0 ∨ VALUE ≤ max(maxv, maxv + np)
```

where

```
sv = |VARIABLES|
rv = range(VARIABLES.var)
np = max(0, sv − 1)
minv = minval(VARIABLES.var)
maxv = maxval(VARIABLES.var)
```

**Purpose**

VALUE is the sum of all minimum values in each occurrence of the STEADY pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern STEADY is the subsequence which matches the regular expression =.

Assume that the occurrence of the pattern STEADY starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i$ to index $j + 1$.

Value is the sum of all minimum values in each occurrence of the STEADY pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

**Example**

```
(30, (1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6))
```

Figure 3.1133 provides an example where the SUM_HEIGHT_STEADY (30, [1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6]) constraint holds.

**Typical**

```
|VARIABLES| > 1
```

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.1133: Illustrating the SUM_HEIGHT_STEADY constraint of the **Example** slot
Automaton

Figures 3.1134 and 3.1135 respectively depict the automaton associated with the constraint SUM_HEIGHT_STEADY and its simplified form.

\[
\begin{align*}
\{ C &\leftarrow \text{default} \\
D &\leftarrow +\infty \\
R &\leftarrow \text{default} \}
\end{align*}
\]

\[
\{ D &\leftarrow +\infty \\
R &\leftarrow R + \min(\min(D, \text{VAR}_i), \text{VAR}_{i+1}) \}
\]

Figure 3.1134: Automaton for the SUM_HEIGHT_STEADY constraint obtained by applying decoration Table 2.35 to the seed transducer of the STEADY pattern where default is 0

\[
\{ R &\leftarrow \text{default} \}
\]

\[
\{ R &\leftarrow R + \text{VAR}_i \}
\]

Figure 3.1135: Automaton for the SUM_HEIGHT_STEADY constraint obtained by applying decoration Table 2.37 to the seed transducer of the STEADY pattern where default is 0

Table 3.148: Glue matrix for the SUM_HEIGHT_STEADY constraint defined as the composition of the STEADY pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### 3.557 SUM_HEIGHT_STEADY_SEQUENCE

#### Description

Based on the STEADY_SEQUENCE pattern.

#### Constraint

**SUM_HEIGHT_STEADY_SEQUENCE(VALUE, VARIABLES)**

#### Arguments

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

#### Restrictions

\[
\begin{align*}
sv & \leq 1 \Rightarrow \text{VALUE} = 0 \\
rv & = 1 \Rightarrow \text{VALUE} = 0 \lor \text{VALUE} \geq \text{minv} \\
rv & \geq 2 \land \text{minv} = -1 \Rightarrow \text{VALUE} = 0 \lor \text{VALUE} \geq -1 \cdot \lfloor (sv - 2)/3 \rfloor - 1 \\
rv & \geq 2 \land \text{minv} \neq -1 \land sv \leq 4 \Rightarrow \\
& \quad \lor \left( \text{VALUE} = 0, \right. \\
& \quad \left. \lor \right) \left( \text{VALUE} \geq \min(\text{minv}, \text{minv} \cdot \lfloor (np + 1)/2 \rfloor + (\text{minv} + 1) \cdot \lfloor np/2 \rfloor) \right) \\
rv & \geq 2 \land \text{minv} \neq -1 \land sv \geq 5 \Rightarrow \\
& \quad \lor \left( \text{VALUE} = 0, \right. \\
& \quad \left. \lor \right) \left( \text{VALUE} \geq \min \left( \text{minv}, \text{minv} \cdot \lfloor (np + 1)/2 \rfloor + (\text{minv} + 1) \cdot \lfloor np/2 \rfloor, \right. \right. \\
& \quad \left. \left. (np + 1) \mod 2 \cdot sv \mod 2 \right) \right) \\
rv & = 1 \Rightarrow \text{VALUE} = 0 \lor \text{VALUE} \leq \text{maxv} \\
rv & \geq 2 \land \text{maxv} = 1 \Rightarrow \text{VALUE} = 0 \lor \text{VALUE} \leq [(sv - 2)/3] + 1 \\
rv & \geq 2 \land \text{maxv} \neq 1 \land sv \leq 4 \Rightarrow \\
& \quad \lor \left( \text{VALUE} = 0, \right. \\
& \quad \left. \lor \right) \left( \text{VALUE} \leq \max(\text{maxv}, \text{maxv} \cdot \lfloor (np + 1)/2 \rfloor + (\text{maxv} - 1) \cdot \lfloor np/2 \rfloor) \right) \\
rv & \geq 2 \land \text{maxv} \neq 1 \land sv \geq 5 \Rightarrow \\
& \quad \lor \left( \text{VALUE} = 0, \right. \\
& \quad \left. \lor \right) \left( \text{VALUE} \leq \max \left( \sum \left( \maxv, \frac{(np + 1)/2,}{(maxv - 1) \cdot \lfloor np/2 \rfloor,}{(np + 1) \mod 2 \cdot sv \mod 2} \right) \right) \right)
\end{align*}
\]

where

- **sv** = |VARIABLES|
- **rv** = range(VARIABLES.var)
- **np** = |sv/2|
- **minv** = minval(VARIABLES.var)
- **maxv** = maxval(VARIABLES.var)

```plaintext

required(VARIABLES, var)
```

---

**Note:** The text includes a table and a diagram that are not fully transcribed here. The diagram appears to illustrate the relationships and constraints described in the text.
**Purpose**

VALUE is the sum of all minimum values in each occurrence of the STEADY_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern STEADY_SEQUENCE is the maximal subsequence which matches the regular expression $=+$. Assume that the occurrence of the pattern STEADY_SEQUENCE starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i$ to index $j + 1$.

**Example**

$$(13, \langle 3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1 \rangle)$$

Figure 3.1136 provides an example where the SUM_HEIGHT_STENCILSEQUENCE $(13, [3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1])$ constraint holds.

Figure 3.1136: Illustrating the SUM_HEIGHT_STENCILSEQUENCE constraint of the Example slot

**Typical**

$|\text{VARIABLES}| > 1$

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 3.1137 and 3.1138 respectively depict the automaton associated with the constraint `SUM_HEIGHT_STEADY_SEQUENCE` and its simplified form.

```
\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow +\infty \\
R &\leftarrow \text{default}
\end{align*}
```

![Automaton Diagram](image)

Figure 3.1137: Automaton for the `SUM_HEIGHT_STEADY_SEQUENCE` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `STEADY_SEQUENCE` pattern where `default` is 0

```
\begin{align*}
C &\leftarrow \min(\min(D, \text VAR_i), \text VAR_{i+1}) \\
D &\leftarrow +\infty
\end{align*}
```

![Automaton Diagram](image)

Figure 3.1138: Automaton for the `SUM_HEIGHT_STEADY_SEQUENCE` constraint obtained by applying decoration Table 2.37 to the seed transducer of the `STEADY_SEQUENCE` pattern where `default` is 0
Table 3.149: Glue matrix for the SUM_HEIGHT_STEADY_SEQUENCE constraint defined as the composition of the STEADY_SEQUENCE pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.558  SUM_MAX_BUMP_ON_DECREASING_SEQUENCE

**Origin**  
Based on the BUMP_ON_DECREASING_SEQUENCE pattern.

**Constraint**  
SUM_MAX_BUMP_ON_DECREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**  
VALUE : dvar  
VARIABLES : collection(var−dvar)

**Restrictions**  
sv ≤ 5 ∨ rv ≤ 2 ⇒ VALUE = 0  
VALUE = 0 ∨ VALUE ≥ min(minv + 2, (minv + 2) * np)  
VALUE = 0 ∨ VALUE ≤ max(maxv, maxv + np) 
required(VARIABLES, var)  
where 
sv = |VARIABLES|  
np = max(0, |(sv - 3)/3|)  
minv =minval(VARIABLES.var)  
maxv =maxval(VARIABLES.var)  
rv =range(VARIABLES.var)

**Purpose**  
VALUE is the sum of all maximum values in each occurrence of the BUMP_ON_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.  
An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence which matches the regular expression >><<.  
Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i + 2 to index j.

**Example**  
(11, ⟨7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3⟩)

Figure 3.1139 provides an example where the SUM_MAX_BUMP_ON_DECREASING_SEQUENCE (11, ⟨7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3⟩) constraint holds.

**Typical**  
|VARIABLES| > 5  
range(VARIABLES.var) > 2

**Arg. properties**  
Functional dependency: VALUE determined by VARIABLES.
Figure 3.1139: Illustrating the SUM_MAX_BUMP_ON_DECREASING_SEQUENCE constraint of the Example slot
Figures 3.1140 and 3.1141 respectively depict the automaton associated with the constraint SUM_MAX_BUMP_ON_DECREASING_SEQUENCE and its simplified form.

Figure 3.1140: Automaton for the SUM_MAX_BUMP_ON_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the BUMP_ON_DECREASING_SEQUENCE pattern where default is 0.
Figure 3.1141: Automaton for the \texttt{SUM\_MAX\_BUMP\_ON\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.27 to the seed transducer of the \texttt{BUMP\_ON\_DECREASING\_SEQUENCE} pattern where \texttt{default} is 0
3.559 SUM_MAX_DECREASING

**DESCRIPTION AUTOMATON**

**Origin**
Based on the **DECREASING** pattern.

**Constraint**
**SUM_MAX_DECREASING(VALUE, VARIABLES)**

**Arguments**
VALUE : dvar
VARIABLES : collection(var-dvar)

**Restrictions**
\[
\begin{align*}
\text{sv} & \leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \\
0, \Sigma & \left( \prod \left( \begin{array}{c}
\text{max} \left( 0, \text{min} \left( 1, \text{np} - 1 \right) \right), \\
\text{sv} \text{ mod } \text{np} \times \text{q}, \\
\text{minv} + 1 + \text{np} - 1 \\
\text{max} \left( 0, \text{min} \left( 1, 2 - \text{np} \right) \right), \\
\text{minv} + 1, \\
\text{q} - 1 \\
\end{array} \right) \\
\right), \quad \text{0} \\
\end{align*}
\]

\[
\begin{align*}
\text{VALUE} & \geq \min_{q \in [\text{lb}1, \text{ub}1]} \text{min} \\
0, \Sigma & \left( \prod \left( \begin{array}{c}
\text{max} \left( 0, \text{min} \left( 1, \text{np} - 1 \right) \right), \\
\text{sv} \text{ mod } \text{np} \times \text{q}, \\
\text{maxv} - \text{np} + 1 \\
\text{max} \left( 0, \text{min} \left( 1, 2 - \text{np} \right) \right), \\
\text{maxv}, \\
\text{q} - 1 \\
\end{array} \right) \\
\right), \quad \text{0} \\
\end{align*}
\]

\text{required}(\text{VARIABLES, var})

where
\[
\begin{align*}
\text{sv} & = |\text{VARIABLES}| \\
\text{np} & = |\text{sv}|/2 \\
\text{maxv} & = \maxval(\text{VARIABLES.var}) \\
\text{minv} & = \minval(\text{VARIABLES.var}) \\
\text{rv} & = \range(\text{VARIABLES.var}) \\
\text{lb}1 & = \min \left( |\text{sv}|/2 + 1, \\
\sum \left( \begin{array}{c}
|\text{sv}|/\min(\text{min}(\text{sv}.\text{rv}), |\text{minv}| + 1) |, \\
\text{min}(1, \text{sv} \text{ mod } \text{min}(\text{min}(\text{sv}.\text{rv}), |\text{minv}| + 1)) \\
\end{array} \right) \\
\right) \\
\text{ub}1 & = |\text{sv}|/2 + 1 \\
\text{lb}2 & = \min \left( |\text{sv}|/2 + 1, \\
\sum \left( \begin{array}{c}
|\text{sv}|/\min(\text{min}(\text{sv}.\text{rv}), |\text{maxv}| + 1) |, \\
\text{min}(1, \text{sv} \text{ mod } \text{min}(\text{min}(\text{sv}.\text{rv}), |\text{maxv}| + 1)) \\
\end{array} \right) \\
\right) \\
\text{ub}2 & = |\text{sv}|/2 + 1
\end{align*}
\]
Purpose

VALUE is the sum of all maximum values in each occurrence of the DECREASING pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern DECREASING is the subsequence which matches the regular expression >.

Assume that the occurrence of the pattern DECREASING starts at position \( i \) and ends at position \( j \). The feature MAX computes the maximum of the values from index \( i \) to index \( j + 1 \).

Example

\[(23, (3, 4, 2, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))\]

Figure 3.1142 provides an example where the SUM_MAX_DECREASING \((23, [3, 4, 2, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4])\) constraint holds.

Figure 3.1142: Illustrating the SUM_MAX_DECREASING constraint of the Example slot

Typical

\[|\text{VARIABLES}| > 1\]

range(\text{VARIABLES}.var) > 1

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figures 3.1143 and 3.1144 respectively depict the automaton associated with the constraint `SUM_MAX_DECREASING` and its simplified form.

Figure 3.1143: Automaton for the `SUM_MAX_DECREASING` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `DECREASING` pattern where `default` is 0

Figure 3.1144: Automaton for the `SUM_MAX_DECREASING` constraint obtained by applying decoration Table 2.37 to the seed transducer of the `DECREASING` pattern where `default` is 0

Table 3.150: Glue matrix for the `SUM_MAX_DECREASING` constraint defined as the composition of the `DECREASING` pattern, the feature `MAX`, and the aggregator `sum`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.560  SUM_MAX_DECREASING_SEQUENCE

**Origin**
Based on the DECREASING_SEQUENCE pattern.

**Constraint**
SUM_MAX_DECREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var–dvar)

**Restrictions**
- \( sv \leq 1 \vee rv \leq 1 \Rightarrow VALUE = 0 \)
- \( VALUE = 0 \vee VALUE \geq \min(minv + 1, (minv + 1) \cdot np) \)
- \( VALUE = 0 \vee VALUE \leq \max(maxv, maxv \cdot np) \)

**Purpose**
VALUE is the sum of all maximum values in each occurrence of the DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression \( > (>|=)^{\ast} > | > \).

Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature MAX computes the maximum of the values from index \( i \) to index \( j + 1 \).

**Example**
\( (16, (3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4)) \)

Figure 3.1145 provides an example where the SUM_MAX_DECREASING_SEQUENCE \( (16, [3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4]) \) constraint holds.

**Typical**
- \(|VARIABLES| > 1\)
- \(\text{range}(\text{VARIABLES}.\var) > 1\)

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.1145: Illustrating the SUM_MAX_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 3.1146 and 3.1147 respectively depict the automaton associated with the constraint SUM_MAX_DECREASING_SEQUENCE and its simplified form.

\[
\begin{align*}
\{ & C \leftarrow \text{default} \\
& D \leftarrow -\infty \\
& R \leftarrow \text{default} \} \\
\leq s & \leq \\
\{ & C \leftarrow \text{default} \\
& D \leftarrow -\infty \\
& R \leftarrow R + C \} \\
\leq t & > \\
\} & \\
\{ & D \leftarrow \max(D, \text{VAR}_{i+1}) \} \\
\geq t & < \\
\} \leq \\
\{ & R \leftarrow \text{default} \\
\leq s & \leq \\
\} < \\
\{ & R \leftarrow R + \text{VAR}_{i} \} \\
\} = \\
\geq t & >
\end{align*}
\]

Figure 3.1146: Automaton for the SUM_MAX_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the DECREASING_SEQUENCE pattern where \text{default} is 0

\[
\begin{align*}
\{ & R \leftarrow \text{default} \\
\leq s & \leq \\
\} < \\
\{ & R \leftarrow R + \text{VAR}_{i} \} \\
\} = \\
\geq t & >
\end{align*}
\]

Figure 3.1147: Automaton for the SUM_MAX_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.37 to the seed transducer of the DECREASING_SEQUENCE pattern where \text{default} is 0
Table 3.151: Glue matrix for the SUM_MAX_DECREASING_SEQUENCE constraint defined as the composition of the DECREASING_SEQUENCE pattern, the feature MAX, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
</tr>
<tr>
<td>t</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\max(\overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}, \overrightarrow{VAR})$</td>
</tr>
</tbody>
</table>
### 3.561 SUM_MAX_DIP_ON_INCREASING_SEQUENCE

**Origin**
Based on the DIP_ON_INCREASING_SEQUENCE pattern.

**Constraint**

\[
\text{SUM_MAX_DIP_ON_INCREASING_SEQUENCE}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- \text{VALUE} : dvar
- \text{VARIABLES} : collection(var−dvar)

**Restrictions**

\[
\begin{align*}
sv \leq 5 & \vee rv \leq 2 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} = 0 & \vee \text{VALUE} \geq \min(\text{minv} + 2, (\text{minv} + 2) \cdot \text{np}) \\
\text{VALUE} = 0 & \vee \text{VALUE} \leq \max(\text{maxv}, \text{maxv} \cdot \text{np})
\end{align*}
\]

where

\[
\begin{align*}
sv &= |\text{VARIABLES}| \\
np &= \max(0, \lfloor (sv - 3)/3 \rfloor) \\
\text{minv} &= \text{minval}(\text{VARIABLES}\cdot\text{var}) \\
\text{maxv} &= \text{maxval}(\text{VARIABLES}\cdot\text{var}) \\
rv &= \text{range}(\text{VARIABLES}\cdot\text{var})
\end{align*}
\]

**Purpose**

\text{VALUE} is the sum of all maximum values in each occurrence of the DIP_ON_INCREASING_SEQUENCE pattern in the time-series given by the \text{VARIABLES} collection. If the pattern does not occur, \text{VALUE} takes the default value 0.

An occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence which matches the regular expression \text{<<><<}. Assume that the occurrence of the pattern DIP_ON_INCREASING_SEQUENCE starts at position \text{i} and ends at position \text{j}. The feature MAX computes the maximum of the values from index \text{i} + 2 to index \text{j}.

**Example**

(11, (1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4))

Figure 3.1148 provides an example where the SUM_MAX_DIP_ON_INCREASING_SEQUENCE (11, [1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4]) constraint holds.

**Typical**

\[
\begin{align*}
|\text{VARIABLES}| &> 5 \\
\text{range}(\text{VARIABLES}\cdot\text{var}) &> 2
\end{align*}
\]

**Arg. properties**

Functional dependency: \text{VALUE} determined by \text{VARIABLES}.
Figure 3.1148: Illustrating the SUM_MAX_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 3.1149 and 3.1150 respectively depict the automaton associated with the constraint \texttt{SUM}\_\texttt{MAX}\_\texttt{DIP}\_\texttt{ON}\_\texttt{INCREASING}\_\texttt{SEQUENCE} and its simplified form.

\begin{figure}[h]
\centering
\begin{tikzpicture}
\node[state, initial, accepting, xscale=1.5, yscale=1.5] (s) {$\geq s$};
\node[state, accepting, right of=s] (t) {$< t$};
\node[state, below of=s] (v) {$< v$};
\node[state, below of=v] (u) {$> u$};
\node[state, below of=v] (d) {$< r$};
\node[xshift=-2cm, yshift=-2cm] (c) {$C \leftarrow \text{default}$};
\node[xshift=-2cm, yshift=-3cm] (d1) {$D \leftarrow -\infty$};
\node[xshift=-2cm, yshift=-4cm] (r) {$R \leftarrow \text{default}$};
\draw[->] (s) edge node {$\geq$} (v);
\draw[->] (v) edge node {$< v$} (u);
\draw[->] (u) edge node {$> u$} (c);
\draw[->] (c) edge node {$\geq$} (s);
\draw[->] (s) edge node {$<$} (d);
\draw[->] (d) edge node {$< r$} (t);
\draw[->] (t) edge node {$< t$} (r);
\draw[->, dashed] (s) edge node {$< r$} (d);
\draw[->, dashed] (d) edge node {$< r$} (t);
\draw[->, dashed] (s) edge node {$< r$} (r);
\draw[->, dashed] (d) edge node {$< r$} (s);
\draw[->, dashed] (u) edge node {$< r$} (d);
\draw[->, dashed] (d) edge node {$< r$} (u);
\draw[->, dashed] (u) edge node {$< r$} (r);
\draw[->, dashed] (d) edge node {$< r$} (u);
\draw[->, dashed] (r) edge node {$< r$} (t);
\draw[->, dashed] (t) edge node {$< r$} (s);
\draw[->, dashed] (t) edge node {$< r$} (d);
\draw[->, dashed] (d) edge node {$< r$} (t);
\draw[->, dashed] (s) edge node {$< r$} (t);
\end{tikzpicture}
\caption{Automaton for the \texttt{SUM}\_\texttt{MAX}\_\texttt{DIP}\_\texttt{ON}\_\texttt{INCREASING}\_\texttt{SEQUENCE} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{DIP}\_\texttt{ON}\_\texttt{INCREASING}\_\texttt{SEQUENCE} pattern where \texttt{default} is 0}
\end{figure}
Figure 3.1150: Automaton for the SUM_MAX_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.25 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern where default is 0
3.562  SUM_MAX_INCREASING

DESCRIPTION

Based on the INCREASING pattern.

Constraint

SUM_MAX_INCREASING(VALUE, VARIABLES)

Arguments

VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions

\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \]

\[ VALUE \geq \min_{q \in [lb1,ub1]} \min \left( \sum_{0}^{q} \prod_{\left( \begin{array}{l}
\prod_{np-1, \minv+1, np-1, np-2]/2} \\
\max(0, \min(1, np-1)), sv \mod np \times q, \\
\minv +1 + np -1 \\
\max(0, \min(1, 2-np)), \minv +1, \\
q -1 \\
(np-1) \times maxv-\\
\prod_{np-1, np-2]/2} \\
\max(0, \min(1, np-1)), sv \mod np \times q, \\
maxv - np +1 \\
\min(0, \min(1, 2-np)), \maxv, \\
q -1 \\
\end{array} \right) \right) \]

\[ VALUE \leq \max_{q \in [lb2,ub2]} \max \left( \sum_{0}^{q} \prod_{\left( \begin{array}{l}
\prod_{np-1, \minv+1, np-1, np-2]/2} \\
\max(0, \min(1, np-1)), sv \mod np \times q, \\
\minv +1 + np -1 \\
\max(0, \min(1, 2-np)), \minv +1, \\
q -1 \\
(np-1) \times maxv-\\
\prod_{np-1, np-2]/2} \\
\max(0, \min(1, np-1)), sv \mod np \times q, \\
maxv - np +1 \\
\min(0, \min(1, 2-np)), \maxv, \\
q -1 \\
\end{array} \right) \right) \]

required(VARIABLES, var)

where

\[ sv = |VAR| \]
\[ np = |sv| \]
\[ maxv = \maxval(VAR) \]
\[ minv = \minval(VAR) \]
\[ rv = \range(VAR) \]

\[ lb1 = \min \left( \sum_{0}^{[sv/2]+1} \left( \begin{array}{l}
\min(1, sv \mod \min(\min(sv, rv), \minv+1)), \\
\min(\min(sv, rv), \minv+1)) \end{array} \right) \right) \]
\[ ub1 = [sv/2]+1 \]
\[ lb2 = \min \left( \sum_{0}^{[sv/2]+1} \left( \begin{array}{l}
\min(1, sv \mod \min(\min(sv, rv), \maxv+1)), \\
\min(\min(sv, rv), \maxv+1)) \end{array} \right) \right) \]
\[ ub2 = [sv/2]+1 \]
**Purpose**

VALUE is the sum of all maximum values in each occurrence of the INCREASING pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern INCREASING is the subsequence which matches the regular expression `<`. Assume that the occurrence of the pattern INCREASING starts at position $i$ and ends at position $j$. The feature MAX computes the maximum of the values from index $i$ to index $j + 1$.

**Example**

$\langle 21, 4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3 \rangle$

Figure 3.1151 provides an example where the SUM_MAX_INCREASING constraint holds.

Figure 3.1151: Illustrating the SUM_MAX_INCREASING constraint of the Example slot

**Typical**

$|\text{VARIABLES}| > 1$

range(\text{VARIABLES}.\text{var}) > 1$

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figures 3.1152 and 3.1153 respectively depict the automaton associated with the constraint `SUM_MAX_INCREASING` and its simplified form.

**Figure 3.1152:** Automaton for the `SUM_MAX_INCREASING` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `INCREASING` pattern where default is 0

**Figure 3.1153:** Automaton for the `SUM_MAX_INCREASING` constraint obtained by applying decoration Table 2.38 to the seed transducer of the `INCREASING` pattern where default is 0

**Table 3.152:** Glue matrix for the `SUM_MAX_INCREASING` constraint defined as the composition of the `INCREASING` pattern, the feature `MAX`, and the aggregator `sum`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
SUM_MAX_INCREASING

2383
### 3.563 SUM_MAX_INCREASING_SEQUENCE

**Description**

Based on the INCREASING_SEQUENCE pattern.

**Constraint**

\[
\text{SUM_MAX_INCREASING_SEQUENCE}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var–dvar)`

**Restrictions**

\[
sv \leq 1 \lor rv \leq 1 \Rightarrow \text{VALUE} = 0
\]

\[
\text{VALUE} = 0 \lor \text{VALUE} \geq \min(\text{minv} + 1, (\text{minv} + 1) \cdot \text{np})
\]

\[
\text{VALUE} = 0 \lor \text{VALUE} \leq \max(\text{maxv}, \text{maxv} \cdot \text{np})
\]

**Purpose**

\[
\text{VALUE} \text{ is the sum of all maximum values in each occurrence of the INCREASINGSEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.}
\]

An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression \(<(=|\geq)\ast<(\geq|\geq)\ast\). Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i\) to index \(j + 1\).

**Example**

\[
(14, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))
\]

Figure 3.1154 provides an example where the SUM_MAX_INCREASING_SEQUENCE \((14, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3])\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1
\]

\[
|\text{range}(\text{VARIABLES}.\text{var})| > 1
\]

**Arg. properties**

Functional dependency: \(\text{VALUE}\) determined by \(\text{VARIABLES}\).
Figure 3.1154: Illustrating the SUM_MAX_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 3.1155 and 3.1156 respectively depict the automaton associated with the constraint SUM_MAX_INCREASING.SEQUENCE and its simplified form.

\[
\begin{align*}
\{ & C \leftarrow \text{default} \\
& D \leftarrow -\infty \\
& R \leftarrow \text{default} \} \\
\geq s & \\
\begin{cases}
C \leftarrow \text{default} \\
D \leftarrow -\infty \\
R \leftarrow R + C
\end{cases} & \begin{cases}
C \leftarrow \max(D, \text{VAR}_{i}), \text{VAR}_{i+1} \\
D \leftarrow -\infty
\end{cases} \\
\leq t & \\
{\{ D \leftarrow \max(D, \text{VAR}_{i+1}) \}} & \{ C \leftarrow \max(C, \max(D, \text{VAR}_{i+1})) \}
\end{align*}
\]

Figure 3.1155: Automaton for the SUM_MAX_INCREASING.SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the INCREASING.SEQUENCE pattern where default is 0

\[
\begin{align*}
\{ & C \leftarrow \text{default} \\
& R \leftarrow \text{default} \} \\
\geq s & \\
\begin{cases}
C \leftarrow \text{default} \\
R \leftarrow R + C
\end{cases} & \{ C \leftarrow \max(\text{VAR}_{i}), \text{VAR}_{i+1} \} \\
\leq t & \\
{\{ C \leftarrow \max(C, \text{VAR}_{i+1}) \}}
\end{align*}
\]

Figure 3.1156: Automaton for the SUM_MAX_INCREASING.SEQUENCE constraint obtained by applying decoration Table 2.24 to the seed transducer of the INCREASING.SEQUENCE pattern where default is 0
Table 3.153: Glue matrix for the SUM_MAX_INCREASING_SEQUENCE constraint defined as the composition of the INCREASING_SEQUENCE pattern, the feature MAX, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.564 SUM_MAX_INFLEXION

**Origin**
Based on the INFLEXION pattern.

**Constraint**
SUM_MAX_INFLEXION(VALUE, VARIABLES)

**Arguments**
VALUE : dvar  
VARIABLES : collection(var−dvar)

**Restrictions**

\[
sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = 0
\]

\[
V \left( VALUE \geq \min (\text{minv, minv} \cdot |(np + 1)/2 + (\text{minv} + 1) \cdot |np/2|^{2})\right)
\]

\[
V \left( VALUE \leq \max (\text{maxv, maxv} \cdot |(np + 1)/2 + (\text{maxv} - 1) \cdot |np/2|^{4})\right)
\]

required(VARIABLES, var)

where

\[
sv = |\text{VARIABLES}|
\]
\[
np = \max (0, sv - 2)
\]
\[
\text{minv} = \text{minval}(\text{VARIABLES}.\text{var})
\]
\[
\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})
\]
\[
rv = \text{range}(\text{VARIABLES}.\text{var})
\]

**Purpose**

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression \(<(>|=|)*>|>(>|=|)*<\).

Assume that the occurrence of the pattern INFLEXION starts at position \(i\) and ends at position \(j\). The feature MAX computes the maximum of the values from index \(i + 1\) to index \(j\).

**Example**

\((31, [1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4])\)

Figure 3.1157 provides an example where the SUM_MAX_INFLEXION \((31, [1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4])\) constraint holds.

**Typical**

|VARIABLES| > 2  
range(VARIABLES.var) > 1

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.1157: Illustrating the SUM_MAX_INFLEXION constraint of the Example slot
Automaton

Figures 3.1158 and 3.1159 respectively depict the automaton associated with the constraint `SUM_MAX_INFLEXION` and its simplified form.

![Automaton Diagram](image)

Figure 3.1158: Automaton for the `SUM_MAX_INFLEXION` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `INFLEXION` pattern where `default` is 0 (transition `r → t` has the same accumulators updates as transition `t → r`)

![Automaton Diagram](image)

Figure 3.1159: Automaton for the `SUM_MAX_INFLEXION` constraint obtained by applying decoration Table 2.25 to the seed transducer of the `INFLEXION` pattern where `default` is 0 (transition `r → t` has the same accumulators updates as transition `t → r`
3.565 **SUM\_MAX\_PEAK**

**DESCRIPTION**

**Origin**

Based on the **PEAK** pattern.

**Constraint**

SUM\_MAX\_PEAK(VALUE, VARIABLES)

**Arguments**

VALUE : dvar  
VARIABLES : collection(var\_dvar)

**Restrictions**

\[ sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = 0 \]
\[ VALUE = 0 \lor VALUE \geq \min(\text{minv} + 18, (\text{minv} + 1) \times np) \]
\[ VALUE = 0 \lor VALUE \leq \max(\text{maxv}, \text{maxv} \times np) \]

(required)(VARIABLES, var)

where

\[ sv = |VARIABLES| \]
\[ np = \max(0, \lfloor (sv - 1)/2 \rfloor) \]
\[ \text{minv} = \text{minval}(VARIABLES.var) \]
\[ \text{maxv} = \text{maxval}(VARIABLES.var) \]
\[ rv = \text{range}(VARIABLES.var) \]

**Purpose**

An occurrence of the pattern **PEAK** is the maximal subsequence which matches the regular expression \(< (= | <)\ast (> | =)\ast >\).

Assume that the occurrence of the pattern **PEAK** starts at position \(i\) and ends at position \(j\). The feature **MAX** computes the maximum of the values from index \(i + 1\) to index \(j\).

**Example**

\((14, (7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 1))\)

Figure 3.1160 provides an example where the SUM\_MAX\_PEAK constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 2 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: **VALUE** determined by VARIABLES.
Figure 3.1160: Illustrating the SUM_MAX_PEAK constraint of the Example slot
Automaton

Figures 3.1161 and 3.1162 respectively depict the automaton associated with the constraint SUM_MAX_PEAK and its simplified form.

Figure 3.1161: Automaton for the SUM_MAX_PEAK constraint obtained by applying decoration Table 2.35 to the seed transducer of the PEAK pattern where default is 0

Figure 3.1162: Automaton for the SUM_MAX_PEAK constraint obtained by applying decoration Table 2.37 to the seed transducer of the PEAK pattern where default is 0
Table 3.154: Glue matrix for the SUM_MAX_PEAK constraint defined as the composition of the PEAK pattern, the feature MAX, and the aggregator sum: cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>(s)</th>
<th>(r)</th>
<th>(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
</tr>
<tr>
<td>(r)</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\max(\overrightarrow{D}, \overrightarrow{D}, \text{VAR}_i))</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}, \text{VAR}_i))</td>
</tr>
<tr>
<td>(t)</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
<td>(\max(\overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}, \text{VAR}_i))</td>
<td>(\overrightarrow{C} + \overrightarrow{C})</td>
</tr>
</tbody>
</table>
**3.566 SUM_MAX STRICTLY DECREASING SEQUENCE**

**Origin**
Based on the **STRICTLY DECREASING SEQUENCE** pattern.

**Constraint**
`SUM_MAX STRICTLY DECREASING SEQUENCE(VALUE, VARIABLES)`

**Arguments**
- `VALUE` : dvar
- `VARIABLES` : collection(var−dvar)

**Restrictions**
- `sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = 0`
- `VALUE = 0 ∨ VALUE ≥ min(minv + 1, (minv + 1) * np)`
- `VALUE = 0 ∨ VALUE ≤ max(maxv, maxv + np)`

where

- `sv = |VARIABLES|`
- `np = |sv/2|`
- `minv = minval(VARIABLES.var)`
- `maxv = maxval(VARIABLES.var)`
- `rv = range(VARIABLES.var)`

**Purpose**
 `VALUE` is the sum of all maximum values in each occurrence of the **STRICTLY DECREASING SEQUENCE** pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, `VALUE` takes the default value 0.

An occurrence of the pattern **STRICTLY DECREASING SEQUENCE** is the maximal sub-sequence which matches the regular expression `>`.

Assume that the occurrence of the pattern **STRICTLY DECREASING SEQUENCE** starts at position `i` and ends at position `j`. The feature `MAX` computes the maximum of the values from index `i` to index `j + 1`.

**Example**

`(16, 4, 4, 6, 4, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3)`

Figure 3.1163 provides an example where the `SUM_MAX STRICTLY DECREASING SEQUENCE (16, [4, 4, 6, 4, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3])` constraint holds.

**Typical**
- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`

**Arg. properties**
Functional dependency: `VALUE` determined by `VARIABLES`. 
Figure 3.1163: Illustrating the SUM_MAX STRICTLY DECREASING SEQUENCE constraint of the Example slot
Figures 3.1164 and 3.1165 respectively depict the automaton associated with the constraint \textsc{SUM\_MAX\_STRICTLY\_DECREASING\_SEQUENCE} and its simplified form.

![Automaton Diagram](image)

Figure 3.1164: Automaton for the \textsc{SUM\_MAX\_STRICTLY\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \textsc{STRICTLY\_DECREASING\_SEQUENCE} pattern where \texttt{default} is 0

![Automaton Diagram](image)

Figure 3.1165: Automaton for the \textsc{SUM\_MAX\_STRICTLY\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.37 to the seed transducer of the \textsc{STRICTLY\_DECREASING\_SEQUENCE} pattern where \texttt{default} is 0
Table 3.155: Glue matrix for the `SUM_MAX_STRICTLY_DECREASING_SEQUENCE` constraint defined as the composition of the `STRICTLY_DECREASING_SEQUENCE` pattern, the feature `MAX`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>( s )</th>
<th>( r )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>( \overrightarrow{C} + \overrightarrow{C} )</td>
<td>( \overrightarrow{C} + \overrightarrow{C} )</td>
</tr>
<tr>
<td>( r )</td>
<td>( \overrightarrow{C} + \overrightarrow{C} )</td>
<td>( \max(\overrightarrow{C}, \overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}, \overrightarrow{VAR}) )</td>
</tr>
</tbody>
</table>
3.567 SUM_MAX STRICTLY_INCREASING_SEQUENCE

**DESCRIPTION**

Based on the STRICTLY_INCREASING_SEQUENCE pattern.

**Constraint**

SUM_MAX STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**

- VALUE : dvar
- VARIABLES : collection(var−dvar)

**Restrictions**

\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \]
\[ VALUE = 0 \lor VALUE \geq \min(minv + 1, (minv + 1) \times np) \]
\[ VALUE = 0 \lor VALUE \leq \max(maxv, maxv \times np) \]

required(VARIABLES, var)

where

\[ sv = |VARIABLES| \]
\[ np = \lfloor sv/2 \rfloor \]
\[ minv = \text{minval}(VARIABLES.var) \]
\[ maxv = \text{maxval}(VARIABLES.var) \]
\[ rv = \text{range}(VARIABLES.var) \]

**Purpose**

VALUE is the sum of all maximum values in each occurrence of the STRICTLY_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal sub-sequence which matches the regular expression `<+`.

Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at position i and ends at position j. The feature MAX computes the maximum of the values from index i to index j + 1.

**Example**

\[ (14, (4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3)) \]

Figure 3.1166 provides an example where the SUM_MAX STRICTLY_INCREASING_SEQUENCE \((14, [4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3])\) constraint holds.

**Typical**

- |VARIABLES| > 1
- \[ \text{range}(VARIABLES.var) > 1 \]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.1166: Illustrating the `SUM_MAX_STRICTLY_INCREASING_SEQUENCE` constraint of the `Example` slot
Figures 3.1167 and 3.1168 respectively depict the automaton associated with the constraint `SUM_MAX STRICTLY INCREASING_SEQUENCE` and its simplified form.

**Figure 3.1167:** Automaton for the `SUM_MAX STRICTLY INCREASING_SEQUENCE` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `STRICTLY INCREASING_SEQUENCE` pattern where `default` is 0

**Figure 3.1168:** Automaton for the `SUM_MAX STRICTLY INCREASING_SEQUENCE` constraint obtained by applying decoration Table 2.24 to the seed transducer of the `STRICTLY INCREASING_SEQUENCE` pattern where `default` is 0
Table 3.156: Glue matrix for the `SUM_MAX STRICTLY_INCREASING_SEQUENCE` constraint defined as the composition of the `STRICTLY_INCREASING_SEQUENCE` pattern, the feature `MAX`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.568 SUM_MAX_SUMMIT

**Description**

Based on the SUMMIT pattern.

**Constraint**

\[
\text{SUM_MAX_SUMMIT}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection(var−dvar)}
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & = 0 \lor \text{VALUE} \geq \min(\text{minv} + 1, (\text{minv} + 1) \times \text{np}) \\
\text{VALUE} & = 0 \lor \text{VALUE} \leq \max(\text{maxv}, \text{maxv} \times \text{np}) \\
\text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where

\[
\begin{align*}
\text{sv} & = |\text{VARIABLES}| \\
\text{np} & = \max(0, \lfloor (\text{sv} - 1)/2 \rfloor) \\
\text{minv} & = \text{minval}(\text{VARIABLES}, \text{var}) \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}, \text{var}) \\
\text{rv} & = \text{range}(\text{VARIABLES}, \text{var})
\end{align*}
\]

**Purpose**

\[
\begin{align*}
\text{VALUE} & \text{ is the sum of all maximum values in each occurrence of the SUMMIT pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.} \\
\text{An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression } (\langle | < (\langle | <)^* \langle | > (\langle | >)^* \rangle) \\
\text{Assume that the occurrence of the pattern SUMMIT starts at position } i \text{ and ends at position } j. \text{ The feature MAX computes the maximum of the values from index } i + 1 \text{ to index } j.
\end{align*}
\]

**Example**

\[
(12, [7, 1, 5, 4, 3, 3, 4, 6, 2, 3, 4, 2, 3, 1])
\]

Figure 3.1169 provides an example where the SUM_MAX_SUMMIT \((12, [7, 1, 5, 4, 3, 3, 4, 6, 2, 3, 4, 2, 3, 1])\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 2 \\
\text{range}(\text{VARIABLES}, \text{var}) > 1
\]

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.1169: Illustrating the SUM_MAX_SUMMIT constraint of the Example slot
Automaton

Figures 3.1170 and 3.1171 respectively depict the automaton associated with the constraint SUM_MAX_SUMMIT and its simplified form.

Figure 3.1170: Automaton for the SUM_MAX_SUMMIT constraint obtained by applying decoration Table 2.35 to the seed transducer of the SUMMIT pattern where default is 0 (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$)

Table 3.157: Glue matrix for the SUM_MAX_SUMMIT constraint defined as the composition of the SUMMIT pattern, the feature MAX, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 3.1171: Automaton for the SUM_MAX_SUMMIT constraint obtained by applying decoration Table 2.37 to the seed transducer of the SUMMIT pattern where default is 0 (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$)
3.569 SUM_MAX_ZIGZAG

### DESCRIPTION

**Origin**
Based on the ZIGZAG pattern.

**Constraint**

\[
\text{SUM_MAX_ZIGZAG(VALUE, VARIABLES)}
\]

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

### Restrictions

- \( sv \leq 3 \lor rv \leq 1 \Rightarrow \text{VALUE} = 0 \)
- \[
\text{VALUE} \geq \min_{q \in [0, qub]} \min \left( 0, \sum \left( \Pi \left( \left( \frac{(sv - 3 + q)/4}{\minv + 1}, q \cdot (\minv + 2) \right), \min \left( 1, \max \left( \maxv - \minv - 1, 0 \right) \right), \max \left( -1, \min \left( 0, q \cdot (\minv + 12) \right) \right), \min \left( \left( \frac{(sv - 3 + q)/4}{\minv + 3} \right) \mod 3 \right) \right) \right) \right)
\]
- \( rv = 2 \Rightarrow \text{VALUE} = 0 \lor \text{VALUE} \leq \max(\maxv, \maxv \cdot np1) \)
- \( rv \geq 3 \Rightarrow \text{VALUE} = 0 \lor \text{VALUE} \leq \max(\maxv, \max(\maxv \cdot np1, \maxv \cdot np2)) \)

**required**(VARIABLES, var)

where

- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)
- \( np1 = \lfloor sv/4 \rfloor \)
- \( np2 = \max(0, (sv - 1)/3) \)
- \( \minv = \text{minval}(\text{VARIABLES}.\text{var}) \)
- \( \maxv = \text{maxval}(\text{VARIABLES}.\text{var}) \)
- \( qub = \min(1, \max(\maxv - \minv - 1, 0)) \cdot np2 \)

**Purpose**

\( \text{VALUE} \) is the sum of all maximum values in each occurrence of the ZIGZAG pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, \( \text{VALUE} \) takes the default value 0.

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression \((<>^+)\langle (< | <>)* (>| >)\rangle^+\langle |<>\rangle\rangle^+\langle |><>\rangle\rangle)

Assume that the occurrence of the pattern ZIGZAG starts at position \( i \) and ends at position \( j \). The feature MAX computes the maximum of the values from index \( i + 1 \) to index \( j \).

### Example

\[
(16, \langle 4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1 \rangle)
\]

Figure 3.1172 provides an example where the SUM_MAX_ZIGZAG (16, [4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1]) constraint holds.
**Figure 3.1172**: Illustrating the **SUM_MAX_ZIGZAG** constraint of the **Example** slot

**Typical**

\[ |\text{VARIABLES}| > 3 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

**Symmetry**

Items of **VARIABLES** can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by **VARIABLES**.
Automaton  
Figures 3.1173 and 3.1174 respectively depict the automaton associated with the constraint SUM_MAX_ZIGZAG and its simplified form.
Figure 3.1173: Automaton for the SUM_MAX_ZIGZAG constraint obtained by applying decoration Table 2.35 to the seed transducer of the ZIGZAG pattern where default is 0; (1) missing transitions from a, b, c, d, e, f to s are labelled by =; (2) on transitions from b, c, e, f to s the accumulator D is reset to its initial value; (3) on transitions from c, f to s the accumulator R is updated wrt C and the accumulator C is reset to its initial value.
Figure 3.1174: Automaton for the \textit{SUM\_MAX\_ZIGZAG} constraint obtained by applying decoration Table 2.23 to the seed transducer of the \textit{ZIGZAG} pattern where \texttt{default} is 0; missing transitions from \(a, b, c, d, e, f\) to \(s\) are labelled by \(=\); (2) on transitions from \(b, c, e, f\) to \(s\) the accumulator \(D\) is reset to its initial value; (3) on transitions from \(c, f\) to \(s\) the accumulator \(R\) is updated wrt \(C\) and the accumulator \(C\) is reset to its initial value.
Table 3.158: Glue matrix for the SUM-MAX-ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature MAX, and the aggregator SUM. Cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th>s</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\bar{c} + \bar{c}$</td>
<td>$\bar{c} + \bar{c}$</td>
<td>$\bar{c} + \bar{c}$</td>
<td>$\bar{c} + \bar{c}$</td>
<td>$\bar{c} + \bar{c}$</td>
<td>$\bar{c} + \bar{c}$</td>
</tr>
<tr>
<td>a</td>
<td>$\bar{c} + \bar{c}$</td>
<td>$\bar{c} + \bar{c}$</td>
<td>$\bar{c} + \bar{c}$</td>
<td>$\max(\bar{c}, \bar{b}, \bar{D}, \text{VAR}_i)$</td>
<td>$\bar{c} + \bar{c}$</td>
<td>$\max(\bar{b}, \bar{D}, \text{VAR}_i)$</td>
</tr>
<tr>
<td>b</td>
<td>$\bar{c} + \bar{c}$</td>
<td>$\bar{c} + \bar{c}$</td>
<td>$\max(\bar{D}, \bar{D}, \text{VAR}_i)$</td>
<td>$\bar{c} + \bar{c}$</td>
<td>$\max(\bar{D}, \bar{D}, \text{VAR}_i)$</td>
<td>$\bar{c} + \bar{c}$</td>
</tr>
<tr>
<td>c</td>
<td>$\bar{c} + \bar{c}$</td>
<td>$\max(\bar{c}, \bar{D}, \bar{D}, \text{VAR}_i)$</td>
<td>$\bar{c} + \bar{c}$</td>
<td>$\max(\bar{c}, \bar{D}, \bar{D}, \text{VAR}_i)$</td>
<td>$\bar{c} + \bar{c}$</td>
<td>$\max(\bar{c}, \bar{D}, \bar{D}, \text{VAR}_i)$</td>
</tr>
<tr>
<td>d</td>
<td>$\bar{c} + \bar{c}$</td>
<td>$\max(\bar{D}, \bar{D}, \text{VAR}_i)$</td>
<td>$\bar{c} + \bar{c}$</td>
<td>$\bar{c} + \bar{c}$</td>
<td>$\bar{c} + \bar{c}$</td>
<td>$\max(\bar{D}, \bar{D}, \text{VAR}_i)$</td>
</tr>
<tr>
<td>e</td>
<td>$\bar{c} + \bar{c}$</td>
<td>$\max(\bar{c}, \bar{D}, \bar{D}, \text{VAR}_i)$</td>
<td>$\bar{c} + \bar{c}$</td>
<td>$\max(\bar{c}, \bar{D}, \bar{D}, \text{VAR}_i)$</td>
<td>$\bar{c} + \bar{c}$</td>
<td>$\max(\bar{c}, \bar{D}, \bar{D}, \text{VAR}_i)$</td>
</tr>
<tr>
<td>f</td>
<td>$\bar{c} + \bar{c}$</td>
<td>$\max(\bar{c}, \bar{D}, \bar{D}, \text{VAR}_i)$</td>
<td>$\bar{c} + \bar{c}$</td>
<td>$\max(\bar{c}, \bar{D}, \bar{D}, \text{VAR}_i)$</td>
<td>$\bar{c} + \bar{c}$</td>
<td>$\max(\bar{c}, \bar{D}, \bar{D}, \text{VAR}_i)$</td>
</tr>
</tbody>
</table>
3.570 **SUM_MIN_BUMP_ON_DECREASING_SEQUENCE**

**DESCRIPTION**

Origin

Based on the BUMP_ON_DECREASING_SEQUENCE pattern.

Constraint

\[
\text{SUM_MIN_BUMP_ON_DECREASING_SEQUENCE}(\text{VALUE, VARIABLES})
\]

Arguments

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} - \text{dvar})
\end{align*}
\]

Restrictions

- \( sv \leq 5 \lor rv \leq 2 \Rightarrow VALUE = 0 \)
- \( VALUE = 0 \lor VALUE \geq \min(\text{minv}, \text{minv} \times np) \)
- \( VALUE = 0 \lor VALUE \leq \max(\text{maxv} - 2, (\text{maxv} - 2) \times np) \)
- \( \text{required}(\text{VARIABLES}, \text{var}) \)

where

\[
\begin{align*}
sv &= |\text{VARIABLES}| \\
np &= \max(0, \lfloor (sv - 3)/3 \rfloor) \\
\text{minv} &= \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{maxv} &= \text{maxval}(\text{VARIABLES}.\text{var}) \\
rv &= \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

VALUE is the sum of all minimum values in each occurrence of the BUMP_ON_DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE is the subsequence which matches the regular expression >>>. Assume that the occurrence of the pattern BUMP_ON_DECREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature \( \text{MIN} \) computes the minimum of the values from index \( i + 2 \) to index \( j \).

**Example**

\[
(7, (7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3))
\]

Figure 3.1175 provides an example where the SUM_MIN_BUMP_ON_DECREASING_SEQUENCE \((7, [7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3])\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 5 \\
\text{range}(\text{VARIABLES}.\text{var}) > 2
\]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.1175: Illustrating the `SUM_MIN_BUMP_ON_DECREASING_SEQUENCE` constraint of the Example slot
Figures 3.1176 and 3.1177 respectively depict the automaton associated with the constraint `SUM_MIN_BUMP_ON_DECREASING_SEQUENCE` and its simplified form.

Figure 3.1176: Automaton for the `SUM_MIN_BUMP_ON_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `BUMP_ON_DECREASING_SEQUENCE` pattern where `default` is 0.
Figure 3.177: Automaton for the \textsc{SUM\_MIN\_BUMP\_ON\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.25 to the seed transducer of the \textsc{BUMP\_ON\_DECREASING\_SEQUENCE} pattern where default is 0
**3.571 SUM_MIN_DECREASING**

**DESCRIPTION**

Based on the **DECREASING** pattern.

**AUTOMATON**

**Constraint**

\[ \text{SUM_MIN_DECREASING}(\text{VALUE}, \text{VARIABLES}) \]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection(var~dvar)}
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{VALUE} & \geq \min_{q \in [lb1, ub1]} \min \\
\text{VALUE} & \leq \max_{q \in [lb2, ub2]} \max \\
\end{align*}
\]

\[
\begin{align*}
\text{VALUE} & = 0 \\
\text{VALUE} & \geq 0, \sum q * \left( \prod \left( \begin{array}{c}
\text{np - 1, minv,} \\
\text{np - 1,} \\
\text{np - 1,} \\
\text{maxv - 1,}
\end{array} \right) \right) \\
\text{VALUE} & \leq \max_{q \in [lb2, ub2]} \max \\
\text{VALUE} & \geq -65
\end{align*}
\]

where

\[
\begin{align*}
\text{sv} & = |\text{VARIABLES}| \\
\text{np} & = |\text{sv/q}| \\
\text{maxv} & = \max(\text{VARIABLES}.\text{var}) \\
\text{minv} & = \min(\text{VARIABLES}.\text{var}) \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var}) \\
\text{lb1} & = \min \left( \left\lfloor \frac{|\text{sv}/2| + 1}{\sum \left( \\
\text{min}(1, \text{sv mod min (min (sv, rv), |minv|))} \right) + 1} \right\rfloor \right) \\
\text{ub1} & = |\text{sv}/2| + 1 \\
\text{lb2} & = \min \left( \left\lfloor \frac{|\text{sv}/2| + 1}{\sum \left( \\
\text{min}(1, \text{sv mod min (min (sv, rv), |maxv|))} \right) + 1} \right\rfloor \right) \\
\text{ub2} & = |\text{sv}/2| + 1
\end{align*}
\]
**Purpose**

VALUE is the sum of all minimum values in each occurrence of the `DECREASING` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern `DECREASING` is the subsequence which matches the regular expression `>`. Assume that the occurrence of the pattern `DECREASING` starts at position $i$ and ends at position $j$. The feature `MIN` computes the minimum of the values from index $i$ to index $j + 1$.

**Example**

\[(14, \langle 3, 4, 2, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4 \rangle)\]

Figure 3.1178 provides an example where the `SUM_MIN_DECREASING` constraint holds.

**Typical**

| `VARIABLES` | $|VARIABLES| > 1$  
| range(`VARIABLES.var`) | $> 1$ |

**Arg. properties**

Functional dependency: VALUE determined by `VARIABLES`. 

---

Figure 3.1178: Illustrating the `SUM_MIN_DECREASING` constraint of the **Example** slot.
Automaton

Figures 3.1179 and 3.1180 respectively depict the automaton associated with the constraint `SUM_MIN_DECREASING` and its simplified form.

\[
\begin{align*}
\{ & C \leftarrow \text{default} \\
& D \leftarrow +\infty \\
& R \leftarrow \text{default} \} \\
\{ & D \leftarrow +\infty \\
& R \leftarrow R + \min(\min(O, \text{VAR}_i), \text{VAR}_{i+1}) \}
\end{align*}
\]

Figure 3.1179: Automaton for the `SUM_MIN_DECREASING` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `DECREASING` pattern where \text{default} is 0

\[
\begin{align*}
\{ & R \leftarrow \text{default} \} \\
\{ & R \leftarrow R + \min(\text{VAR}_i, \text{VAR}_{i+1}) \}
\end{align*}
\]

Figure 3.1180: Automaton for the `SUM_MIN_DECREASING` constraint obtained by applying decoration Table 2.38 to the seed transducer of the `DECREASING` pattern where \text{default} is 0

Table 3.159: Glue matrix for the `SUM_MIN_DECREASING` constraint defined as the composition of the `DECREASING` pattern, the feature `MIN`, and the aggregator `sum`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### 3.572 SUM_MIN_DECREASING_SEQUENCE

**Description**

Based on the `DECREASING_SEQUENCE` pattern.

**Constraint**

`SUM_MIN_DECREASING_SEQUENCE(VALUE, VARIABLES)`

**Arguments**

- `VALUE` : `dvar`
- `VARIABLES` : `collection(var−dvar)`

**Restrictions**

\[
sv ≤ 1 \lor rv ≤ 1 \Rightarrow VALUE = 0
\]

\[
VALUE = 0 \lor VALUE \geq \min(\minv, \minv \times np)\]

\[
VALUE = 0 \lor VALUE \leq \max(\maxv - 1, (\maxv - 1) \times np)\]

**Purpose**

VALUE is the sum of all minimum values in each occurrence of the `DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern `DECREASING_SEQUENCE` is the maximal subsequence which matches the regular expression `>|=|^>`.

Assume that the occurrence of the pattern `DECREASING_SEQUENCE` starts at position \(i\) and ends at position \(j\). The feature `MIN` computes the minimum of the values from index \(i\) to index \(j + 1\).

**Example**

\[ (7, (3, 4, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4)) \]

Figure 3.1181 provides an example where the `SUM_MIN_DECREASING_SEQUENCE` constraint holds.

**Typical**

- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`

**Arg. properties**

Functional dependency: VALUE determined by `VARIABLES`.
Figure 3.1181: Illustrating the SUM_MIN_DECREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 3.1182 and 3.1183 respectively depict the automaton associated with the constraint \( \text{SUM\_MIN\_DECREASING\_SEQUENCE} \) and its simplified form.

\[
\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow +\infty \\
R &\leftarrow \text{default}
\end{align*}
\]

\[
\begin{array}{c}
\leq s \\
\{ C \leftarrow \text{default} \\
D &\leftarrow +\infty \\
R &\leftarrow R + C \}
\end{array}
\]

\[
\begin{array}{c}
\leq s \\
\{ C \leftarrow \text{default} \\
D &\leftarrow +\infty \\
R &\leftarrow R + C \}
\end{array}
\]

\[
\begin{array}{c}
\geq t \\
\{ C \leftarrow \text{default} \\
D &\leftarrow +\infty \\
R &\leftarrow R + C \}
\end{array}
\]

Figure 3.1182: Automaton for the \( \text{SUM\_MIN\_DECREASING\_SEQUENCE} \) constraint obtained by applying decoration Table 2.35 to the seed transducer of the \( \text{DECREASING\_SEQUENCE} \) pattern where \( \text{default} \) is 0.

\[
\begin{align*}
C &\leftarrow \text{default} \\
R &\leftarrow \text{default}
\end{align*}
\]

\[
\begin{array}{c}
\leq s \\
\{ C \leftarrow \text{default} \\
R &\leftarrow R + C \}
\end{array}
\]

\[
\begin{array}{c}
\leq s \\
\{ C \leftarrow \text{default} \\
R &\leftarrow R + C \}
\end{array}
\]

\[
\begin{array}{c}
\geq t \\
\{ C \leftarrow \text{default} \\
R &\leftarrow R + C \}
\end{array}
\]

Figure 3.1183: Automaton for the \( \text{SUM\_MIN\_DECREASING\_SEQUENCE} \) constraint obtained by applying decoration Table 2.24 to the seed transducer of the \( \text{DECREASING\_SEQUENCE} \) pattern where \( \text{default} \) is 0.
Table 3.160: Glue matrix for the SUM_MIN_DECREASING_SEQUENCE constraint defined as the composition of the DECREASING_SEQUENCE pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.573 SUM_MIN_DIP_ON_INCREASING_SEQUENCE

Slice < < > < <

**DESCRIPTION**

Based on the **DIP_ON_INCREASING_SEQUENCE** pattern.

**Constraint**

**SUM_MIN_DIP_ON_INCREASING_SEQUENCE**(VALUE, VARIABLES)

**Arguments**

| **VALUE** : dvar |
| **VARIABLES** : collection(var−dvar) |

**Restrictions**

\[
sv \leq 5 \lor rv \leq 2 \Rightarrow VALUE = 0 \\
VALUE = 0 \lor VALUE \geq \min(\text{min}_{V},\text{min}_{N} \times np) \\
VALUE = 0 \lor VALUE \leq \max(\text{max}_{V} - 2, (\text{max}_{V} - 2) \times np) \\
\text{required}(\text{VARIABLES},\text{var})
\]

where

\[
sv = |\text{VARIABLES}| \\
np = \max(0, (sv - 3)/3) \\
\text{min}_{V} = \min_{V}(\text{VARIABLES}.var) \\
\text{max}_{V} = \max_{V}(\text{VARIABLES}.var) \\
rv = \text{range}(\text{VARIABLES}.var)
\]

**Purpose**

VALUE is the sum of all minimum values in each occurrence of the **DIP_ON_INCREASING_SEQUENCE** pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern **DIP_ON_INCREASING_SEQUENCE** is the subsequence which matches the regular expression <<><>. Assume that the occurrence of the pattern **DIP_ON_INCREASING_SEQUENCE** starts at position \( i \) and ends at position \( j \). The feature **MIN** computes the minimum of the values from index \( i+2 \) to index \( j \).

**Example**

\[(3, (1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4))\]

Figure 3.1184 provides an example where the **SUM_MIN_DIP_ON_INCREASING_SEQUENCE** (3, [1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4]) constraint holds.

**Typical**

| **VARIABLES** | > 5 |
| **range**(VARIABLES.var) | > 2 |

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.1184: Illustrating the SUM_MIN_DIP_ON_INCREASING_SEQUENCE constraint of the Example slot
Figures 3.1185 and 3.1186 respectively depict the automaton associated with the constraint \texttt{SUM\_MIN\_DIP\_ON\_INCREASING\_SEQUENCE} and its simplified form.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{automaton.png}
\caption{Automaton for the \texttt{SUM\_MIN\_DIP\_ON\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{DIP\_ON\_INCREASING\_SEQUENCE} pattern where \texttt{default} is 0}
\end{figure}
Figure 3.1186: Automaton for the \texttt{SUM\_MIN\_DIP\_ON\_INCREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.27 to the seed transducer of the \texttt{DIP\_ON\_INCREASING\_SEQUENCE} pattern where \texttt{default} is 0
### 3.574 SUM_MIN_GORGE

**Origin**
Based on the GORGE pattern.

**Constraint**
\[ \text{SUM_MIN_GORGE}(\text{VALUE}, \text{VARIABLES}) \]

**Arguments**
- **VALUE**: \( \text{dvar} \)
- **VARIABLES**: \( \text{collection(\text{var}-\text{dvar})} \)

**Restrictions**
- \( \text{sv} \leq 2 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \)
- \( \text{VALUE} = 0 \lor \text{VALUE} \geq \min(\text{minv} \cdot \text{minv} + \text{np} \cdot \text{np}) \)
- \( \text{VALUE} = 0 \lor \text{VALUE} \leq \max(\maxv - 1, (\maxv - 1) * \text{np} * \text{np}) \)

**Purpose**
An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression \( (> | > (= | >)*) > (< | < (= | <)) <) \). Assume that the occurrence of the pattern GORGE starts at position \( i \) and ends at position \( j \). The feature MIN computes the minimum of the values from index \( i + 1 \) to index \( j \).

**Example**
\[ (12, (1, 7, 3, 4, 4, 5, 4, 2, 2, 6, 5, 6, 5, 7)) \]

Figure 3.1187 provides an example where the SUM_MIN_GORGE (12, [1, 7, 3, 4, 4, 5, 4, 2, 2, 6, 5, 6, 5, 7]) constraint holds.

**Typical**
- \( |\text{VARIABLES}| > 2 \)
- \( \text{range}(\text{VARIABLES} \cdot \text{var}) > 1 \)

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.1187: Illustrating the SUM_MIN_GORGE constraint of the Example slot
Automaton

Figures 3.1188 and 3.1189 respectively depict the automaton associated with the constraint SUM_MIN_GORGE and its simplified form.

Figure 3.1188: Automaton for the SUM_MIN_GORGE constraint obtained by applying decoration Table 2.35 to the seed transducer of the GORGE pattern where default is 0 (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$)

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
<th>$t$</th>
<th>$u$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\overline{c} + \overline{c}$</td>
<td>$\overline{c} + \overline{c}$</td>
<td>$\overline{c} + \overline{c}$</td>
<td>$\overline{c} + \overline{c}$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\overline{c} + \overline{c}$</td>
<td>$\min(\overline{D}, \overline{D}, VAR_i)$</td>
<td>$\overline{c} + \overline{c}$</td>
<td>$\overline{c} + \overline{c}$</td>
</tr>
<tr>
<td>$t$</td>
<td>$\overline{c} + \overline{c}$</td>
<td>$\min(\overline{c}, \overline{D}, \overline{D}, VAR_i)$</td>
<td>$\overline{c} + \overline{c}$</td>
<td>$\min(\overline{c}, \overline{D}, \overline{D}, VAR_i)$</td>
</tr>
<tr>
<td>$u$</td>
<td>$\overline{c} + \overline{c}$</td>
<td>$\overline{c} + \overline{c}$</td>
<td>$\min(\overline{c}, \overline{D}, \overline{D}, VAR_i)$</td>
<td>$\overline{c} + \overline{c}$</td>
</tr>
</tbody>
</table>

Table 3.161: Glue matrix for the SUM_MIN_GORGE constraint defined as the composition of the GORGE pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 3.1189: Automaton for the SUM_MIN_GORGE constraint obtained by applying decoration Table 2.37 to the seed transducer of the GORGE pattern where default is 0.
### 3.575 SUM_MIN_INCREMENTING

**DESCRIPTION AUTOMATON**

**Origin**
Based on the INCREASING pattern.

**Constraint**

\[ \text{SUM}_\text{MIN}_\text{INCREMENTING} (\text{VALUE}, \text{VARIABLES}) \]

**Arguments**

- \( \text{VALUE} : \text{dvar} \)
- \( \text{VARIABLES} : \text{collection(} \text{var} - \text{dvar} \text{)} \)

**Restrictions**

\[
sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0
\]

\[
\text{VALUE} \geq \min_{q \in [lb1, ub1]} \min \left( 0, \sum \Pi \left( \left( q \times \left( \sum \left( \left( (np - 1) \times \text{minv}, \max (0, \min (1, np - 1)) \right) \right)/2 \right) \right), \right) \right)
\]

\[
\text{VALUE} \leq \max_{q \in [lb2, ub2]} \max \left( 0, \sum \Pi \left( \left( q \times \left( \sum \left( \left( (np - 1, \text{maxv} - 1), \max (0, \min (1, np - 1)) \right) \right)/2 \right) \right), \right) \right)
\]

**required\((\text{VARIABLES, var})\) where**

- \( sv = |\text{VARIABLES}| \)
- \( np = |sv/q| \)
- \( \text{maxv} = \text{maxval}(\text{VARIABLES, var}) \)
- \( \text{minv} = \text{minval}(\text{VARIABLES, var}) \)
- \( \text{rv} = \text{range}(\text{VARIABLES, var}) \)
- \( lb1 = \min \left( \left( |sv/2| + 1, \sum \left( \left( |sv/\min(\min(\text{sv, rv}), |\text{minv}| + 1)) \right), \min(1, sv \mod \min(\min(\text{sv, rv}), |\text{minv}| + 1)) \right) \right) \)
- \( ub1 = |sv/2| + 1 \)
- \( lb2 = \min \left( \left( |sv/2| + 1, \sum \left( \left( |sv/\min(\min(\text{sv, rv}), \text{maxv} + 1)) \right), \min(1, sv \mod \min(\min(\text{sv, rv}), \text{maxv} + 1)) \right) \right) \)
- \( ub2 = |sv/2| + 1 \)

\[ \text{VALUE} \geq -65 \]

\[ \text{VALUE} \geq -41 \]
VALUE is the sum of all minimum values in each occurrence of the INCREASING pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern INCREASING is the subsequence which matches the regular expression <.
Assume that the occurrence of the pattern INCREASING starts at position $i$ and ends at position $j$. The feature MIN computes the minimum of the values from index $i$ to index $j + 1$.

Example

$\langle 12, 4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3 \rangle$

Figure 3.1190 provides an example where the SUM_MIN_INCREASING constraint holds.

Figure 3.1190: Illustrating the SUM_MIN_INCREASING constraint of the Example slot

Typical

$|\text{VARIABLES}| > 1$

range(VARIABLES.var) > 1

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figures 3.1191 and 3.1192 respectively depict the automaton associated with the constraint `SUM_MIN_INCREASING` and its simplified form.

Figure 3.1191: Automaton for the `SUM_MIN_INCREASING` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `INCREASING` pattern where `default` is 0

Figure 3.1192: Automaton for the `SUM_MIN_INCREASING` constraint obtained by applying decoration Table 2.37 to the seed transducer of the `INCREASING` pattern where `default` is 0

Table 3.162: Glue matrix for the `SUM_MIN_INCREASING` constraint defined as the composition of the `INCREASING` pattern, the feature `MIN`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
SUM_MIN_INCREASING

2437
3.576  SUM_MIN_INCREASING_SEQUENCE

**DESCRIPTION**

Origin
Based on the INCREASING_SEQUENCE pattern.

Constraint
SUM_MIN_INCREASING_SEQUENCE(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \]
\[ VALUE = 0 \lor VALUE \geq \min(\text{minv}, n \cdot \text{np}) \]
\[ VALUE = 0 \lor VALUE \leq \max(\text{maxv} - 1, (\text{maxv} - 1) \cdot n \cdot \text{np}) \]

where
\[ sv = |\text{VARIABLES}| \]
\[ np = \lfloor sv/2 \rfloor \]
\[ \text{minv} = \minval(\text{VARIABLES}.\text{var}) \]
\[ \text{maxv} = \maxval(\text{VARIABLES}.\text{var}) \]
\[ rv = \text{range}(\text{VARIABLES}.\text{var}) \]

VALUE is the sum of all minimum values in each occurrence of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression \(< (< | =)^* (< | <)\).

Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i\) to index \(j + 1\).

**Example**

\( (5, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3)) \)

Figure 3.1193 provides an example where the SUM_MIN_INCREASING_SEQUENCE \((5, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3])\) constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 1 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.1193: Illustrating the `SUM_MIN_INCREASING_SEQUENCE` constraint of the `Example` slot
Automaton

Figures 3.1194 and 3.1195 respectively depict the automaton associated with the constraint \( \text{SUM\_MIN\_INCREASING\_SEQUENCE} \) and its simplified form.

\[
\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow +\infty \\
R &\leftarrow \text{default}
\end{align*}
\]

\[
\begin{align*}
\{ C &\leftarrow \text{default} \\
D &\leftarrow +\infty \\
R &\leftarrow R + C \} \\
\{ D &\leftarrow \min(D, \text{VAR}_i+1) \} \\
\{ C &\leftarrow \min(C, \min(D, \text{VAR}_i+1)) \}
\end{align*}
\]

Figure 3.1194: Automaton for the \( \text{SUM\_MIN\_INCREASING\_SEQUENCE} \) constraint obtained by applying decoration Table 2.35 to the seed transducer of the \( \text{INCREASING\_SEQUENCE} \) pattern where default is 0.

\[
\begin{align*}
\{ R &\leftarrow \text{default} \} \\
\{ R &\leftarrow R + \text{VAR}_i \}
\end{align*}
\]

Figure 3.1195: Automaton for the \( \text{SUM\_MIN\_INCREASING\_SEQUENCE} \) constraint obtained by applying decoration Table 2.37 to the seed transducer of the \( \text{INCREASING\_SEQUENCE} \) pattern where default is 0.
Table 3.163: Glue matrix for the \texttt{SUM\_MIN\_INCREASING\_SEQUENCE} constraint defined as the composition of the \texttt{INCREASING\_SEQUENCE} pattern, the feature \texttt{MIN}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### 3.577 SUM_MIN_INFLEXION

#### Origin
Based on the **INFLEXION** pattern.

#### Constraint
```
SUM_MIN_INFLEXION(VALUE, VARIABLES)
```

#### Arguments
```
VALUE : dvar
VARIABLES : collection(var−dvar)
```

#### Restrictions
```
sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = 0
V
VALUE ≥ min(minv), minv * [(np + 1)/2] + (minv + 1) * [np/2] 2
VALUE = 0,
V
VALUE ≤ max(maxv), maxv * [(np + 1)/2] + (maxv − 1) * [np/2] 4
required(VARIABLES, var)
```

where
```
sv = |VARIABLES|
np = max(0, sv − 2)
minv =minval(VARIABLES.var)
maxv =maxval(VARIABLES.var)
rv =range(VARIABLES.var)
```

VALUE is the sum of all minimum values in each occurrence of the **INFLEXION** pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

#### Purpose
An occurrence of the pattern **INFLEXION** is the maximal subsequence which matches the regular expression `<( <| =)| > | (>| =)| > <`. Assume that the occurrence of the pattern **INFLEXION** starts at position i and ends at position j. The feature **MIN** computes the minimum of the values from index i + 1 to index j.

#### Example
```
(26, (1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 4, 4))
```

Figure 3.1196 provides an example where the **SUM_MIN_INFLEXION** `(26,[1, 2, 6, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 4, 4])` constraint holds.

#### Typical
```
|VARIABLES| > 2
range(VARIABLES.var) > 1
```

#### Symmetry
Items of VARIABLES can be reversed.

#### Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 3.1196: Illustrating the SUM_MIN_INFLEXION constraint of the Example slot
Figures 3.1197 and 3.1198 respectively depict the automaton associated with the constraint 
**SUM_MIN_INFLEXION** and its simplified form.

**Figure 3.1197:** Automaton for the **SUM_MIN_INFLEXION** constraint obtained by applying decoration Table 2.35 to the seed transducer of the **INFLEXION** pattern where *default* is 0 (transition $r \rightarrow t$ has the same accumulators updates as transition $t \rightarrow r$)

**Figure 3.1198:** Automaton for the **SUM_MIN_INFLEXION** constraint obtained by applying decoration Table 2.25 to the seed transducer of the **INFLEXION** pattern where *default* is 0 (transition $r \rightarrow t$ has the same accumulators updates as transition $t \rightarrow r$)
**3.578 SUM_MIN строго убывающая последовательность**

**Описание**

Используется в описании **STRICTLY_Decreasing_sequence**.

**Ограничение**

гарантируется

**Аргументы**

VALUE: dvar

VARIABLES: collection(var–dvar)

**Ограничения**

sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = 0

VALUE = 0 ∨ VALUE ≥ min(minv, minv + np)

VALUE = 0 ∨ VALUE ≤ max(maxv - 1, (maxv - 1) + np)

**Цель**

VALUE является суммой всех минимальных значений каждой вхождения паттерна **STRICTLY_Decreasing_sequence** в последовательность, заданную коллекцией VARIABLEs. Если паттерн не входит, VALUE принимает значение по умолчанию 0.

An occurrence of the pattern **STRICTLY_Decreasing_sequence** is the maximal subsequence which matches the regular expression >. Assume that the occurrence of the pattern **STRICTLY_Decreasing_sequence** starts at position i and ends at position j. The feature MIN computes the minimum of the values from index i to index j + 1.

**Пример**

(6, (4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3))

Figure 3.119 provides an example where the **SUM_MIN STRICTLY_Decreasing_sequence** (6, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3]) constraint holds.

**Обычно**

|VARIABLES| > 1

**Предикаты аргументов**

**Функциональная зависимость:** VALUE определяется коллекцией VARIABLEs.

**Алгоритм**
Figure 3.1199: Illustrating the SUM_MIN.Strictly.DECREASING.SEQUENCE constraint of the Example slot
Automaton

Figures 3.1200 and 3.1201 respectively depict the automaton associated with the constraint \text{SUM\_MIN\_STRICTLY\_DECREASING\_SEQUENCE} and its simplified form.

\[
\begin{align*}
\{ & C \leftarrow \text{default} \\
 & D \leftarrow +\infty \\
 & R \leftarrow \text{default} \}
\end{align*}
\]

\[
\leq s
\]

\[
\begin{align*}
\{ & C \leftarrow \text{default} \\
 & D \leftarrow +\infty \\
 & R \leftarrow R + C \}
\end{align*}
\]

\[
\begin{align*}
\{ & C \leftarrow \min(D, \text{VAR}_i), \text{VAR}_{i+1} \}
\end{align*}
\]

\[
> r
\]

\[
\begin{align*}
\{ & C \leftarrow \min(C, \text{VAR}_{i+1}) \\
 & D \leftarrow +\infty
\}
\end{align*}
\]

\[
\leq
\]

\[
\begin{align*}
\{ & C \leftarrow \text{default} \\
 & R \leftarrow \text{default} \}
\end{align*}
\]

\[
\leq s
\]

\[
\begin{align*}
\{ & C \leftarrow \text{default} \\
 & R \leftarrow R + C \}
\end{align*}
\]

\[
\begin{align*}
\{ & C \leftarrow \text{default} \\
 & D \leftarrow +\infty \\
 & R \leftarrow R + C \}
\end{align*}
\]

\[
\begin{align*}
\{ & C \leftarrow \min(C, \text{VAR}_{i+1}) \\
 & D \leftarrow +\infty
\}
\end{align*}
\]

\[
> r
\]

\[
\begin{align*}
\{ & C \leftarrow \min(C, \text{VAR}_{i+1}) \\
 & D \leftarrow +\infty
\}
\end{align*}
\]

Figure 3.1200: Automaton for the \text{SUM\_MIN\_STRICTLY\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \text{STRICTLY\_DECREASING\_SEQUENCE} pattern where \text{default} is 0

Figure 3.1201: Automaton for the \text{SUM\_MIN\_STRICTLY\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.24 to the seed transducer of the \text{STRICTLY\_DECREASING\_SEQUENCE} pattern where \text{default} is 0
Table 3.164: Glue matrix for the $\text{SUM}_\text{MIN}_\text{STRICTLY DECREASING SEQUENCE}$ constraint defined as the composition of the $\text{STRICTLY DECREASING SEQUENCE}$ pattern, the feature $\text{MIN}$, and the aggregator $\text{sum}$; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### 3.579 SUM_MIN STRICTLY_INCREASING_SEQUENCE

**Description**

Based on the \textsc{strictly increasing sequence} pattern.

**Constraint**

\textsc{sum\_min\_strictly\_increasing\_sequence}(\text{VALUE}, \text{VARIABLES})

**Arguments**

\begin{itemize}
  \item \text{VALUE} : dvar
  \item \text{VARIABLES} : collection(var–dvar)
\end{itemize}

**Restrictions**

\begin{align*}
  \text{sv} & \leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \\
  \text{VALUE} & = 0 \lor \text{VALUE} \geq \min(\text{minv}, \text{minv} \times \text{np}) \\
  \text{VALUE} & = 0 \lor \text{VALUE} \leq \max(\text{maxv} - 1, (\text{maxv} - 1) \times \text{np}) \\
\end{align*}

where

\begin{itemize}
  \item \text{sv} = |\text{VARIABLES}|
  \item \text{np} = \lfloor \text{sv}/2 \rfloor
  \item \text{minv} = \text{minval}(\text{VARIABLES}.\text{var})
  \item \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})
  \item \text{rv} = \text{range}(\text{VARIABLES}.\text{var})
\end{itemize}

**Purpose**

\text{VALUE} is the sum of all minimum values in each occurrence of the \textsc{strictly increasing sequence} pattern in the time-series given by the \text{VARIABLES} collection. If the pattern does not occur, \text{VALUE} takes the default value 0.

An occurrence of the pattern \textsc{strictly increasing sequence} is the maximal sub-sequence which matches the regular expression \texttt{<+}. Assume that the occurrence of the pattern \textsc{strictly increasing sequence} starts at position \textit{i} and ends at position \textit{j}. The feature \textsc{min} computes the minimum of the values from index \textit{i} to index \textit{j} + 1.

**Example**

\begin{itemize}
  \item \langle 5, (4, 3, 5, 5, 2, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3) \rangle
\end{itemize}

Figure 3.1202 provides an example where the \textsc{sum\_min\_strictly\_increasing\_sequence} (5, [4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3]) constraint holds.

**Typical**

\begin{itemize}
  \item |\text{VARIABLES}| > 1
  \item \text{range}(\text{VARIABLES}.\text{var}) > 1
\end{itemize}

**Arg. properties**

Functional dependency: \text{VALUE} determined by \text{VARIABLES}. 
Figure 3.1202: Illustrating the SUM_MIN_STRICTLY_INCREASING_SEQUENCE constraint of the Example slot
Figures 3.1203 and 3.1204 respectively depict the automaton associated with the constraint SUM_MIN_STRICTLY_INCREASING_SEQUENCE and its simplified form.

Figure 3.1203: Automaton for the SUM_MIN_STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern where default is 0

Figure 3.1204: Automaton for the SUM_MIN_STRICTLY_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.37 to the seed transducer of the STRICTLY_INCREASING_SEQUENCE pattern where default is 0
Table 3.165: Glue matrix for the `SUM_MIN_STRICTLY_INCREASING_SEQUENCE` constraint defined as the composition of the `STRICTLY_INCREASING_SEQUENCE` pattern, the feature `MIN`, and the aggregator `sum`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
</tr>
<tr>
<td>r</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\min(\overrightarrow{C}, \overrightarrow{C}, \overrightarrow{D}, \overrightarrow{D}, \text{VAR}_{i})$</td>
</tr>
</tbody>
</table>
3.580  SUM_MIN_VALLEY

**DESCRIPTION**

**Origin**
Based on the VALLEY pattern.

**Constraint**
SUM_MIN_VALLEY(VALUE, VARIABLES)

**Arguments**
VALUE : dvar
VARIABLES : collection(var−dvar)

**Restrictions**

\[
sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = 0
\]

\[
VALUE = 0 \lor VALUE \geq \min(minV, minv \star np)\]

\[
VALUE = 0 \lor VALUE \leq \max(maxv - 1, (maxv - 1) \star np)\]

required(VARIABLES, var)

where

\[
sv = |VARIABLES|
\]

\[
np = \max(0, \lfloor (sv - 1)/2 \rfloor)
\]

\[
minV = \minval(VARIABLES.var)
\]

\[
maxv = \maxval(VARIABLES.var)
\]

\[
rv = \range(VARIABLES.var)
\]

**Purpose**

VALUE is the sum of all minimum values in each occurrence of the VALLEY pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression \((= | >)\star (< | =)\star <\).

Assume that the occurrence of the pattern VALLEY starts at position \(i\) and ends at position \(j\). The feature \(MIN\) computes the minimum of the values from index \(i + 1\) to index \(j\).

**Example**

\((10, (1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 7))\)

Figure 3.1205 provides an example where the SUM_MIN_VALLEY (10, [1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 7]) constraint holds.

**Typical**

\(|VARIABLES| > 2\)

\(\range(VARIABLES.var) > 1\)

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.1205: Illustrating the SUM_MIN_VALLEY constraint of the Example slot
Automaton

Figures 3.1206 and 3.1207 respectively depict the automaton associated with the constraint SUM_MIN_VALLEY and its simplified form.

Figure 3.1206: Automaton for the SUM_MIN_VALLEY constraint obtained by applying decoration Table 2.35 to the seed transducer of the VALLEY pattern where default is 0

Figure 3.1207: Automaton for the SUM_MIN_VALLEY constraint obtained by applying decoration Table 2.37 to the seed transducer of the VALLEY pattern where default is 0
Table 3.166: Glue matrix for the SUM_MIN_VALLEY constraint defined as the composition of the VALLEY pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\overline{C} + \overline{C})</td>
</tr>
<tr>
<td>r</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\min(\overline{D}, \overline{D}, \text{VAR}_i))</td>
<td>(\min(\overline{C}, \overline{D}, \overline{D}, \text{VAR}_i))</td>
</tr>
<tr>
<td>t</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\min(\overline{C}, \overline{D}, \overline{D}, \text{VAR}_i))</td>
<td>(\overline{C} + \overline{C})</td>
</tr>
</tbody>
</table>
### 3.581 SUM_MIN_ZIGZAG

**Description**

- **Origin**: Based on the ZIGZAG pattern.

- **Constraint**: SUM_MIN_ZIGZAG(VALUE, VARIABLES)

- **Arguments**:
  - **VALUE**: dvar
  - **VARIABLES**: collection(var−dvar)

- **Restrictions**

  \[
  \begin{align*}
  sv \leq 3 \text{ } &\lor\text{ } rv \leq 1 \Rightarrow VALUE = 0 \\
  rv = 2 \Rightarrow VALUE = 0 \text{ } &\lor\text{ } VALUE \geq \min(\text{minv}, \text{minv} \times np1) \\
  rv \geq 3 \Rightarrow VALUE = 0 \text{ } &\lor\text{ } VALUE \geq \min(\text{minv}, \min(\text{minv} \times np1, \text{minv} \times np2)) \\
  \end{align*}
  \]

  \[
  VALUE \leq \max_{q \in [0,qub]} \max
  \begin{bmatrix}
  0, \sum (\prod (\min (1, \max (\text{maxv} - \text{minv} - 1, 0)), \text{maxv} - q * (\text{maxv} - 2)), \text{maxv}) \\
  \prod (\min (\text{maxv} - q * (\text{maxv} - 3)) \mod 3, \text{maxv}) \\
  \end{bmatrix}
  \]

- **Purpose**

  VALUE is the sum of all minimum values in each occurrence of the ZIGZAG pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

  An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression \((<>)^+ (< | <> | > >)^+ (> | <>)\).

  Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature MIN computes the minimum of the values from index \(i + 1\) to index \(j\).

- **Example**

  \[(3, (4, 1, 3, 1, 4, 6, 1, 5, 3, 2, 3, 1, 6, 1))\]

  Figure 3.1208 provides an example where the SUM_MIN_ZIGZAG (3, [4, 1, 3, 1, 4, 6, 1, 5, 3, 2, 3, 1, 6, 1]) constraint holds.
Figure 3.208: Illustrating the SUM_MIN_ZIGZAG constraint of the Example slot

Typical

- $|\text{VARIABLES}| > 3$
- $\text{range}(\text{VARIABLES}.\text{var}) > 1$

Symmetry

Items of VARIABLES can be reversed.

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 3.1209 and 3.1210 respectively depict the automaton associated with the constraint SUM_MIN_ZIGZAG and its simplified form.
Figure 3.1209: Automaton for the SUM_MIN_ZIGZAG constraint obtained by applying decoration Table 2.35 to the seed transducer of the ZIGZAG pattern where default is 0; (1) missing transitions from a, b, c, d, e, f to s are labelled by \(=\); (2) on transitions from b, c, e, f to s the accumulator D is reset to its initial value; (3) on transitions from c, f to s the accumulator R is updated wrt C and the accumulator C is reset to its initial value.
Figure 3.1210: Automaton for the SUM_MIN_ZIGZAG constraint obtained by applying decoration Table 2.23 to the seed transducer of the ZIGZAG pattern where default is 0; missing transitions from a, b, c, d, e, f to s are labelled by =; (2) on transitions from b, c, e, f to s the accumulator D is reset to its initial value; (3) on transitions from c, f to s the accumulator R is updated wrt C and the accumulator C is reset to its initial value.
Table 3.167: Glue matrix for the SUM_MIN_ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature MIN, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.582 SUM_RANGE_DECREASING

DESCRIPTION

Origin
Based on the DECREASING pattern.

Constraint
SUM_RANGE_DECREASING(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \]
\[ VALUE \geq 0 \]
\[ VALUE \leq \lfloor sv/2 \rfloor \times (rv - 1) \]
\[ \text{required}(\text{VARIABLES}, \text{var}) \]
where
\[ rv = \text{range}(\text{VARIABLES}.\text{var}) \]
\[ sv = |\text{VARIABLES}| \]

VALUE is the sum of the differences between the largest and smallest value in each occurrence of the DECREASING pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

Purpose
An occurrence of the pattern DECREASING is the subsequence which matches the regular expression \( > \).
Assume that the occurrence of the pattern DECREASING starts at position \( i \) and ends at position \( j \). The feature RANGE computes the range of the values from index \( i \) to index \( j+1 \).

Example
\((9, (3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4))\)

Figure 3.1211 provides an example where the SUM_RANGE_DECREASING (9, [3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4]) constraint holds.

Typical
\[ |\text{VARIABLES}| > 1 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

Symmetry
One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 3.1211: Illustrating the SUM_RANGE_DECREASING constraint of the Example slot
Figures 3.1212 and 3.1213 respectively depict the automaton associated with the constraint SUM_RANGE_DECREASING and its simplified form.

Figure 3.1212: Automaton for the SUM_RANGE_DECREASING constraint obtained by applying decoration Table 2.46 to the seed transducer of the DECREASING pattern where default is 0

Figure 3.1213: Automaton for the SUM_RANGE_DECREASING constraint obtained by applying decoration Table 2.44 to the seed transducer of the DECREASING pattern where default is 0
3.583 SUM_RANGE_DECREASING_SEQUENCE

**DESCRIPTION**

**Origin**
Based on the DECREASING_SEQUENCE pattern.

**Constraint**

\[
\text{SUM\_RANGE\_DECREASING\_SEQUENCE(}\text{VALUE, VARIABLES})
\]

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**

\[
s_v \leq 1 \lor rv \leq 1 \Rightarrow \text{VALUE} = 0
\]

\[
\text{VALUE} \geq 0
\]

\[
\text{VALUE} \leq \lfloor s_v/2 \rfloor \cdot (rv - 1) \cdot \text{required(VARIABLES, var)}
\]

where

\[
rv = \text{range(VARIABLES.var)}
\]

\[
s_v = |\text{VARIABLES}|\]

**Purpose**

- **VALUE** is the sum of the differences between the largest and smallest value in each occurrence of the DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, **VALUE** takes the default value 0.

- An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression \( >(>||=)^*>|\).

- Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature RANGE computes the range of the values from index \(i\) to index \(j + 1\).

**Example**

\[(9, (3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 4, 4))\]

Figure 3.1214 provides an example where the SUM_RANGE_DECREASING_SEQUENCE \((9, [3, 4, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 4, 4])\) constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 1\)
- \(\text{range(VARIABLES.var)} > 1\)

**Symmetry**

- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
Figure 3.1214: Illustrating the SUM\_RANGE\_DECREASING\_SEQUENCE constraint of the Example slot
Automaton

Figures 3.1215 and 3.1216 respectively depict the automaton associated with the constraint `SUM_RANGE_DECREASING_SEQUENCE` and its simplified form.

**Figure 3.1215:** Automaton for the `SUM_RANGE_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 2.46 to the seed transducer of the `DECREASING_SEQUENCE` pattern where `default` is 0

**Figure 3.1216:** Automaton for the `SUM_RANGE_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 2.44 to the seed transducer of the `DECREASING_SEQUENCE` pattern where `default` is 0
3.584  SUM_RANGE_INCREASING

DESCRIPTION  AUTOMATON

Origin
Based on the INCREASING pattern.

Constraint
SUM_RANGE_INCREASING(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = 0
VALUE ≥ 0
VALUE ≤ ⌊sv/2⌋ *(rv − 1)0
determined required(VARIABLES, var)
where
rv =range(VARIABLES, var)
sv = |VARIABLES|

VALUE is the sum of the differences between the largest and smallest value in each occurrence of the INCREASING pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

Purpose
An occurrence of the pattern INCREASING is the subsequence which matches the regular expression <.
Assume that the occurrence of the pattern INCREASING starts at position i and ends at position j. The feature RANGE computes the range of the values from index i to index j + 1.

Example
(9, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))

Figure 3.1217 provides an example where the SUM_RANGE_INCREASING (9, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3]) constraint holds.

Typical
|VARIABLES| > 1
range(VARIABLES, var) > 1

Symmetry
One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 3.1217: Illustrating the SUM\_RANGE\_INCREASING constraint of the **Example** slot
Automaton

Figures 3.1218 and 3.1219 respectively depict the automaton associated with the constraint SUM_RANGE_INCREASING and its simplified form.

\[
\begin{align*}
&\{ C \leftarrow \text{default} \\
&\quad H \leftarrow \text{VAR}_i \\
&\quad R \leftarrow \text{default} \}
\end{align*}
\]

\[
\begin{align*}
&\{ H \leftarrow \text{VAR}_i + 1 \\
&\quad R \leftarrow R + |H - \text{VAR}_i + 1| \}
\end{align*}
\]

Figure 3.1218: Automaton for the SUM_RANGE_INCREASING constraint obtained by applying decoration Table 2.46 to the seed transducer of the INCREASING pattern where default is 0.

\[
\begin{align*}
&\{ R \leftarrow \text{default} \}
\end{align*}
\]

\[
\begin{align*}
&\{ R \leftarrow R + \max(0, \text{VAR}_{i+1} - \text{VAR}_i) \}
\end{align*}
\]

Figure 3.1219: Automaton for the SUM_RANGE_INCREASING constraint obtained by applying decoration Table 2.45 to the seed transducer of the INCREASING pattern where default is 0; \( R_i - R_{i-2} - X_i + X_{i-2} \geq 0 \) and \( R_i - R_{i-1} - X_i + X_{i-1} \geq 0 \) are linear invariants.
3.585 SUM_RANGE_INCREASING_SEQUENCE

**DESCRIPTION**

Based on the INCREASING_SEQUENCE pattern.

**Constraint**

SUM_RANGE_INCREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**

VALUE : dvar
VARIABLES : collection(var - dvar)

**Restrictions**

\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \]
\[ VALUE \geq 0 \]
\[ VALUE \leq \lfloor sv/2 \rfloor \times (rv - 1) \]
\[ \text{required}(\text{VARIABLES}, \text{var}) \]

where

\[ rv = \text{range}(\text{VARIABLES}, \text{var}) \]
\[ sv = |\text{VARIABLES}| \]

**Purpose**

VALUE is the sum of the differences between the largest and smallest value in each occurrence of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression \(< (< | =)*) < | <\).

Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature RANGE computes the range of the values from index \(i\) to index \(j + 1\).

**Example**

\((9, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))\)

Figure 3.120 provides an example where the SUM_RANGE_INCREASING_SEQUENCE (9, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3]) constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 1 \]
\[ \text{range}(\text{VARIABLES}, \text{var}) > 1 \]

**Symmetry**

One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.1220: Illustrating the SUM_RANGE_INCREASING_SEQUENCE constraint of the Example slot
Figures 3.1221 and 3.1222 respectively depict the automaton associated with the constraint SUM\_RANGE\_INCREMENTING\_SEQUENCE and its simplified form.

\[
\begin{align*}
\left\{ \begin{array}{l}
C \leftarrow \text{default} \\
H \leftarrow \text{VAR}_1 \\
R \leftarrow \text{default}
\end{array} \right. & \quad \geq s \\
& \quad \{ H \leftarrow \text{VAR}_{i+1} \} \\
\left\{ \begin{array}{l}
C \leftarrow \text{default} \\
H \leftarrow \text{VAR}_{i+1} \\
R \leftarrow R + C
\end{array} \right. & \quad > \\
& \quad \{ C \leftarrow |H - \text{VAR}_{i+1}| \} \\
= & \quad \leq t \\
& \quad \{ C \leftarrow |H - \text{VAR}_{i+1}| \}
\end{align*}
\]

Figure 3.1221: Automaton for the SUM\_RANGE\_INCREMENTING\_SEQUENCE constraint obtained by applying decoration Table 2.46 to the seed transducer of the INCREASING\_SEQUENCE pattern where default is 0

\[
\begin{align*}
\{ R \leftarrow \text{default} \} & \quad \geq s \\
& \quad \geq \\
& \quad \{ R \leftarrow R + \max(0, \text{VAR}_{i+1} - \text{VAR}_i) \} \\
= & \quad \leq t \\
& \quad \{ R \leftarrow R + \max(0, \text{VAR}_{i+1} - \text{VAR}_i) \}
\end{align*}
\]

Figure 3.1222: Automaton for the SUM\_RANGE\_INCREMENTING\_SEQUENCE constraint obtained by applying decoration Table 2.45 to the seed transducer of the INCREASING\_SEQUENCE pattern where default is 0
### Description

**Origin**
Based on the `SUM_RANGE_STRICTLY_DECREASING_SEQUENCE` pattern.

**Constraint**

```plaintext
SUM_RANGE_STRICTLY_DECREASING_SEQUENCE(VALUE, VARIABLES)
```

**Arguments**

<table>
<thead>
<tr>
<th>VALUE</th>
<th>: dvar</th>
</tr>
</thead>
<tbody>
<tr>
<td>VARIABLES</td>
<td>: collection(var=dvar)</td>
</tr>
</tbody>
</table>

**Restrictions**

- \( sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \)
- \( VALUE \geq 0 \)
- \( VALUE \leq \lfloor sv/2 \rfloor \times (rv - 1) \)

\( ^\text{required}(\text{VARIABLES}, \text{var}) \)

where

\( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

\( sv = |\text{VARIABLES}| \)

**Purpose**
An occurrence of the pattern `SUM_RANGE_STRICTLY_DECREASING_SEQUENCE` is the maximal subsequence which matches the regular expression \( >^+ \).

Assume that the occurrence of the pattern `SUM_RANGE_STRICTLY_DECREASING_SEQUENCE` starts at position \( i \) and ends at position \( j \). The feature `RANGE` computes the range of the values from index \( i \) to index \( j + 1 \).

**Example**

\( (10, (4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3)) \)

Figure 3.1223 provides an example where the `SUM_RANGE_STRICTLY_DECREASING_SEQUENCE` \( (10, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3]) \) constraint holds.

**Typical**

\( |\text{VARIABLES}| > 1 \)

\( \text{range}(\text{VARIABLES}.\text{var}) > 1 \)

**Symmetry**
One and the same constant can be added to the `VAR` attribute of all items of `VARIABLES`.

**Arg. properties**

Functional dependency: `VALUE` determined by `VARIABLES`. 
Figure 3.1223: Illustrating the \texttt{SUM\_RANGE\_STRICLTY\_DECREASING\_SEQUENCE} constraint of the \texttt{Example} slot
Automaton

Figures 3.1224 and 3.1225 respectively depict the automaton associated with the constraint SUM_RANGE_STRICTLY_DECREASING_SEQUENCE and its simplified form.

Figure 3.1224: Automaton for the SUM RANGE STRICTLY DECREASING SEQUENCE constraint obtained by applying decoration Table 2.46 to the seed transducer of the STRICTLY DECREASING SEQUENCE pattern where default is 0

Figure 3.1225: Automaton for the SUM RANGE STRICTLY DECREASING SEQUENCE constraint obtained by applying decoration Table 2.44 to the seed transducer of the STRICTLY DECREASING SEQUENCE pattern where default is 0
SUM_RANGE_STRICTLY_DECREASING_SEQUENCE

2483
3.587 **SUM\_RANGE\_STRICTLY\_INCREASING\_SEQUENCE**

### Description

**Origin**
Based on the **STRICTLY\_INCREASING\_SEQUENCE** pattern.

**Constraint**

\[
\text{SUM\_RANGE\_STRICTLY\_INCREASING\_SEQUENCE}(\text{VALUE, VARIABLES})
\]

**Arguments**

- **VALUE**: `dvar`
- **VARIABLES**: `collection(var - dvar)`

**Restrictions**

\[
\begin{align*}
\text{sv} \leq 1 \lor \text{rv} \leq 1 & \implies \text{VALUE} = 0 \\
\text{VALUE} & \geq 0 \\
\text{VALUE} & \leq \lfloor \text{sv}/2 \rfloor \times (\text{rv} - 1) \\
\text{required(} \text{VARIABLES, var} )
\end{align*}
\]

where

- \( \text{rv} = \text{range(} \text{VARIABLES, var} ) \)
- \( \text{sv} = |\text{VARIABLES}| \)

**Purpose**

An occurrence of the pattern **STRICTLY\_INCREASING\_SEQUENCE** is the **maximal** sub-sequence which matches the regular expression `<+`. Assume that the occurrence of the pattern **STRICTLY\_INCREASING\_SEQUENCE** starts at position \( i \) and ends at position \( j \). The feature **RANGE** computes the range of the values from index \( i \) to index \( j + 1 \).

**Example**

\((9, (4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3))\)

Figure 3.1226 provides an example where the **SUM\_RANGE\_STRICTLY\_INCREASING\_SEQUENCE** constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1, \text{range}(\text{VARIABLES, var}) > 1
\]

**Symmetry**
One and the same constant can be added to the `var` attribute of all items of **VARIABLES**.

**Arg. properties**

Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 3.1226: Illustrating the `SUM_RANGE STRICTLY_INCREASING_SEQUENCE` constraint of the `Example` slot
Figures 3.1227 and 3.1228 respectively depict the automaton associated with the constraint `SUM_RANGE_STRICTLY_INCREASING_SEQUENCE` and its simplified form.

```
\{ C \leftarrow \text{default} \\
H \leftarrow \text{VAR}_i \\
R \leftarrow \text{default} \}
```

Figure 3.1227: Automaton for the `SUM_RANGE_STRICTLY_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 2.46 to the seed transducer of the `STRICTLY_INCREASING_SEQUENCE` pattern where `default` is 0

```
\{ R \leftarrow \text{default} \}
```

Figure 3.1228: Automaton for the `SUM_RANGE_STRICTLY_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 2.45 to the seed transducer of the `STRICTLY_INCREASING_SEQUENCE` pattern where `default` is 0
SUM_RANGE_STRICTLY_INCREASING_SEQUENCE 2487
3.588 SUM_SURF_BUMP_ON_DECREASING_SEQUENCE

### Description

**Origin**
Based on the \texttt{BUMP\_ON\_DECREASING\_SEQUENCE} pattern.

**Constraint**

\[
\text{SUM\_SURF\_BUMP\_ON\_DECREASING\_SEQUENCE}((\text{VALUE}, \text{VARIABLES}))
\]

**Arguments**

- \texttt{VALUE} : \texttt{dvar}
- \texttt{VARIABLES} : \texttt{collection(var-dvar)}

**Restrictions**

\[
\begin{align*}
sv & \leq 5 \lor rv \leq 2 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} = 0 & \lor \text{VALUE} \geq \min(3 \times \text{minv} + 3, (3 \times \text{minv} + 3) \times np) \\
\text{VALUE} = 0 & \lor \text{VALUE} \leq \max(3 \times \text{maxv} - 3, (3 \times \text{maxv} - 3) \times np) \\
\text{VALUE} & = 0 \lor \text{VALUE} \leq \min(3 \times \text{minv} + 3, (3 \times \text{minv} + 3) \times np)
\end{align*}
\]

where

- \texttt{sv} = |\text{VARIABLES}|
- \texttt{np} = \max(0, [(\text{sv} - 3)/3])
- \texttt{minv} = \text{minval}(\text{VARIABLES}.\text{var})
- \texttt{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})
- \texttt{rv} = \text{range}(\text{VARIABLES}.\text{var})

**Purpose**

\texttt{VALUE} is the sum of the surface of occurrences of the \texttt{BUMP\_ON\_DECREASING\_SEQUENCE} pattern in the time-series given by the \texttt{VARIABLES} collection. If the pattern does not occur, \texttt{VALUE} takes the default value 0.

An occurrence of the pattern \texttt{BUMP\_ON\_DECREASING\_SEQUENCE} is the subsequence which matches the regular expression \texttt{>>><>}. Assume that the occurrence of the pattern \texttt{BUMP\_ON\_DECREASING\_SEQUENCE} starts at position \texttt{i} and ends at position \texttt{j}. The feature \texttt{SURF} computes the sum of the values from index \texttt{i + 2} to index \texttt{j}.

**Example**

\[
(27, [7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3])
\]

Figure 3.129 provides an example where the \texttt{SUM\_SURF\_BUMP\_ON\_DECREASING\_SEQUENCE} (27, [7, 6, 5, 6, 5, 4, 1, 4, 7, 5, 4, 2, 5, 4, 3, 3]) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 5 \\
\text{range}(\text{VARIABLES}.\text{var}) > 2
\]

**Arg. properties**

Functional dependency: \texttt{VALUE} determined by \texttt{VARIABLES}.
Figure 3.1229: Illustrating the \texttt{SUM\_SURF\_BUMP\_ON\_DECREASING\_SEQUENCE} constraint of the \textbf{Example} slot
Figures 3.1230 and 3.1231 respectively depict the automaton associated with the constraint `SUM_SURF_BUMP_ON_DECREASING_SEQUENCE` and its simplified form.

Figure 3.1230: Automaton for the `SUM_SURF_BUMP_ON_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `BUMP_ON_DECREASING_SEQUENCE` pattern where `default` is 0
Figure 3.1231: Automaton for the \textsc{sum_surf_bump_on_decreasing_sequence} constraint obtained by applying decoration Table 2.25 to the seed transducer of the \textsc{bump_on_decreasing_sequence} pattern where \texttt{default} is 0
SUM_SURF_DECREASING

### Description
Based on the **DECREASING** pattern.

### Constraint
**SUM_SURF_DECREASING(VALUE, VARIABLES)**

#### Arguments
- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

#### Restrictions
- $sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0$
- $VALUE \geq \min_{q \in [lb1, ub1]} \min$
- $VALUE \leq \max_{q \in [lb2, ub2]} \max$

**required**(VARIABLES, var)

where
- $sv = |VARIABLES|$
- $np = \lfloor sv/q \rfloor$
- $maxv = \maxval$(VARIABLES.var)
- $minv = \minval$(VARIABLES.var)
- $rv = \range$(VARIABLES.var)
- $lb1 = \min \left( \frac{sv}{2} + 1, \sum \left( \frac{sv/\min(\min sv, rv), \lfloor \min v \rfloor + 1)}{\min(1, sv mod \min sv, rv, \lfloor \min v \rfloor + 1)} \right) \right)$
- $ub1 = \lfloor sv/2 \rfloor + 1$
- $lb2 = \min \left( \frac{sv}{2} + 1, \sum \left( \frac{sv/\min(\min sv, rv), \max v + 1)}{\min(1, sv mod \min sv, rv, \lfloor \max v \rfloor + 1)} \right) \right)$
- $ub2 = \lfloor sv/2 \rfloor + 1$
**Purpose**

An occurrence of the pattern **DECREASING** is the subsequence which matches the regular expression `>`. Assume that the occurrence of the pattern **DECREASING** starts at position $i$ and ends at position $j$. The feature **SURF** computes the sum of the values from index $i$ to index $j+1$.

**Example**

$$(37, [3, 4, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4])$$

Figure 3.1232 provides an example where the **SUM_SURF_DECREASING** $(37, [3, 4, 2, 5, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4])$ constraint holds.

![Graph showing feature values](image)

Figure 3.1232: Illustrating the **SUM_SURF_DECREASING** constraint of the **Example** slot

**Typical**

$|\text{VARIABLES}| > 1$

\[\text{range}(\text{VARIABLES}.\text{var}) > 1]\n
**Arg. properties**

Functional dependency: **VALUE** determined by **VARIABLES**.
Automaton

Figures 3.1233 and 3.1234 respectively depict the automaton associated with the constraint SUM_SURF_DECREASING and its simplified form.

\[
\begin{align*}
\{ & C \gets \text{default} \\
& D \gets 0 \\
& R \gets \text{default} \} \\
\{ & D \gets 0 \\
& R \gets R + D + \text{VAR}_i + \text{VAR}_{i+1} \} \quad \overset{\geq}{\rightarrow} \quad S \\
\end{align*}
\]

\[
\begin{align*}
\{ & R \gets \text{default} \} \\
\{ & R \gets R + \text{VAR}_i + \text{VAR}_{i+1} \} \quad \overset{\geq}{\rightarrow} \quad S \\
\end{align*}
\]

Figure 3.1233: Automaton for the SUM_SURF_DECREASING constraint obtained by applying decoration Table 2.35 to the seed transducer of the DECREASING pattern where default is 0.

Figure 3.1234: Automaton for the SUM_SURF_DECREASING constraint obtained by applying decoration Table 2.38 to the seed transducer of the DECREASING pattern where default is 0.

Table 3.168: Glue matrix for the SUM_SURF_DECREASING constraint defined as the composition of the DECREASING pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
SUM_SURF_DECREASING 2495
3.590  SUM_SURF_DECREASING_SEQUENCE

DESCRIPTION AUTOMATON

Origin
Based on the DECREASING_SEQUENCE pattern.

Constraint
SUM_SURF_DECREASING_SEQUENCE(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = 0
rv = 2 ⇒ VALUE = 0 ∨ VALUE ≥ min(2 * minv + 1, (2 * minv + 1) * np)
rv ≥ 3 ⇒

VALUE = 0,

VALUE ≥ min(2 * minv + 1, (2 * minv + 1) * np + min(0, sv mod 2 * (minv + 2)))

rv = 2 ⇒ VALUE = 0 ∨ VALUE ≤ max(2 * maxv − 1, (2 * maxv − 1) * np)
rv ≥ 3 ⇒

VALUE = 0,

VALUE ≤ max(2 * maxv − 1, (2 * maxv − 1) * np + max(0, sv mod 2 * (maxv − 2)))

required (VARIABLES, var).

where
sv = |VARIABLES|
np = ⌊sv/2⌋
minv = minval (VARIABLES.var)
maxv = maxval (VARIABLES.var)
rv = range (VARIABLES.var)

Purpose
VALUE is the sum of the surface of occurrences of the DECREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern DECREASING_SEQUENCE is the maximal subsequence which matches the regular expression > (> | =)* > | >.

Assume that the occurrence of the pattern DECREASING_SEQUENCE starts at position i and ends at position j. The feature SURF computes the sum of the values from index i to index j + 1.

Example
(34, ⟨3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4⟩)

Figure 3.1235 provides an example where the SUM_SURF_DECREASING_SEQUENCE (34, [3, 4, 2, 2, 5, 6, 6, 4, 4, 3, 1, 1, 4, 6, 4, 4]) constraint holds.
**Figure 3.1235:** Illustrating the `SUM_SURF_DECREASING_SEQUENCE` constraint of the `Example` slot

**Typical**

- `|VARIABLES| > 1`
- `range(VARIABLES.var) > 1`

**Arg. properties**

- Functional dependency: VALUE determined by VARIABLES.
Figures 3.1236 and 3.1237 respectively depict the automaton associated with the constraint SUM_SURF_DECREASING_SEQUENCE and its simplified form.

Figure 3.1236: Automaton for the SUM_SURF_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the DECREASING_SEQUENCE pattern where default is 0

Figure 3.1237: Automaton for the SUM_SURF_DECREASING_SEQUENCE constraint obtained by applying decoration Table 2.29 to the seed transducer of the DECREASING_SEQUENCE pattern where default is 0
Table 3.169: Glue matrix for the SUM_SURF_DECREASING_SEQUENCE constraint defined as the composition of the DECREASING_SEQUENCE pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.591 SUM_SURF_DECREASING_TERRACE

**DESCRIPTION**

Based on the DECREASING_TERRACE pattern.

**Constraint**

\[ SUM\_SURF\_DECREASING\_TERRACE(VALUE, VARIABLES) \]

**Arguments**

\[
\begin{align*}
VALUE & : dvar \\
VARIABLES & : collection(var-dvar)
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
sv \leq 3 \lor rv \leq 2 & \Rightarrow VALUE = 0 \\
VALUE = 0 \lor VALUE \geq \min(2 + (\minv + 1) \times (sv - 2) + (\minv + 1) \times x) \\
VALUE = 0 \lor VALUE \leq \max(2 + (\maxv - 1) \times (sv - 2) + (\maxv - 1) \times y) \\
\end{align*}
\]

where

\[
\begin{align*}
\minv & = \minval(VARIABLES.var) \\
\maxv & = \maxval(VARIABLES.var) \\
sv & = |VARIABLES| \\
rv & = \text{range}(VARIABLES.var)
\end{align*}
\]

**Purpose**

VALUE is the sum of the surface of occurrences of the DECREASING_TERRACE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern DECREASING_TERRACE is the maximal subsequence which matches the regular expression \(\geq \)\(^+\)\(\geq\).

Assume that the occurrence of the pattern DECREASING_TERRACE starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

\((12, (6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3))\)

Figure 3.1238 provides an example where the SUM_SURF_DECREASING_TERRACE\((12, [6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3])\) constraint holds.

**Typical**

\[ |VARIABLES| > 3 \]
\[ \text{range}(VARIABLES.var) > 2 \]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.1238: Illustrating the SUM_SURF_DECREASING_TERRACE constraint of the Example slot
Figures 3.1239 and 3.1240 respectively depict the automaton associated with the constraint \( \text{SUM_SURF_DECREASING_TERRACE} \) and its simplified form.

Figure 3.1239: Automaton for the \( \text{SUM_SURF_DECREASING_TERRACE} \) constraint obtained by applying decoration Table 2.35 to the seed transducer of the \( \text{DECREASING_TERRACE} \) pattern where \( \text{default} \) is 0

Figure 3.1240: Automaton for the \( \text{SUM_SURF_DECREASING_TERRACE} \) constraint obtained by applying decoration Table 2.28 to the seed transducer of the \( \text{DECREASING_TERRACE} \) pattern where \( \text{default} \) is 0
Table 3.170: Glue matrix for the `SUM_SURF_DECREASING_TERRACE` constraint defined as the composition of the `DECREASING_TERRACE` pattern, the feature `SURF`, and the aggregator sum: cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>C + C</td>
<td>C + C</td>
<td>C + C</td>
</tr>
<tr>
<td>r</td>
<td>C + C</td>
<td>C + C</td>
<td>D + D + VAR₁</td>
</tr>
<tr>
<td>t</td>
<td>C + C</td>
<td>D + D + VAR₁</td>
<td>D + D + VAR₁</td>
</tr>
</tbody>
</table>
3.592  SUM_SURF_DIP_ON_INCREASING_SEQUENCE

Origin
Based on the DIP_ON_INCREASING_SEQUENCE pattern.

Constraint
SUM_SURF_DIP_ON_INCREASING_SEQUENCE(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv ≤ 5 ∨ rv ≤ 2 ⇒ VALUE = 0
VALUE = 0 ∨ VALUE ≥ min(3 * minv + 3, (3 * minv + 3) * np)
VALUE = 0 ∨ VALUE ≤ max(3 * maxv − 3, (3 * maxv − 3) * np)
required(VARIABLES, var)
where
sv = |VARIABLES|
np = max(0, (sv−3)/3)
minv = minval(VARIABLES, var)
maxv = maxval(VARIABLES, var)
rv = range(VARIABLES, var)

Purpose
VALUE is the sum of the surface of occurrences of the DIP_ON_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern DIP_ON_INCREASING_SEQUENCE is the subsequence which matches the regular expression <\(<<><><\)>.<

Assume that the occurrence of the pattern DIP_ON_INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i+2\) to index \(j\).

Example
(19, (1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4))

Figure 3.1241 provides an example where the SUM_SURF_DIP_ON_INCREASING_SEQUENCE (19, [1, 2, 3, 2, 5, 6, 7, 4, 1, 3, 4, 6, 1, 2, 4, 4]) constraint holds.

Typical
|VARIABLES| > 5
range(VARIABLES, var) > 2

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 3.1241: Illustrating the `SUM_SURF_DIP_ON_INCREASING_SEQUENCE` constraint of the `Example` slot
Figures 3.1242 and 3.1243 respectively depict the automaton associated with the constraint `SUM_SURF_DIP_ON_INCREASING_SEQUENCE` and its simplified form.

Figure 3.1242: Automaton for the `SUM_SURF_DIP_ON_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `DIP_ON_INCREASING_SEQUENCE` pattern where `default` is 0
Figure 3.1243: Automaton for the SUM_SURF_DIP_ON_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.25 to the seed transducer of the DIP_ON_INCREASING_SEQUENCE pattern where default is 0
### 3.593 SUM_SURF_GORGE

**DESCRIPTION**

**AUTOMATON**

\[ (> | > (= | >)^* (> | < (= | <)^* <) \]

**Origin**

Based on the GORGE pattern.

**Constraint**

\[ \text{SUM}_\text{SURF}_\text{GORGE}(\text{VALUE}, \text{VARIABLES}) \]

**Arguments**

- \text{VALUE} : \text{dvar}
- \text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})

**Restrictions**

- \( sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = 0 \)
- \( rv = 2 \Rightarrow VALUE = 0 \lor VALUE \geq \min(\text{minv} \times, \text{minv} \times \text{np} \times) \)
- \( rv \geq 3 \Rightarrow VALUE = 0, \lor \)
  \( VALUE \geq \min(\text{minv} \times, \min((sv - 2) \times (\text{minv} + 1) - 1 \times, \text{minv} \times \text{np} \times)) \)
- \( rv = 2 \Rightarrow VALUE = 0 \lor VALUE \leq \max(\text{maxv} - 1, (\text{maxv} - 1) \times \text{np} \times) \)
- \( rv \geq 3 \Rightarrow VALUE = 0, \lor \)
  \( VALUE \leq \max(\text{maxv} - 1, \max((sv - 2) \times (\text{maxv} - 1) - 1, (\text{maxv} - 1) \times \text{np} \times)) \)

**Purpose**

An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression \( (> | > (= | >)^* (> | < (= | <)^* <) \). Assume that the occurrence of the pattern GORGE starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i + 1 \) to index \( j \).

**Example**

\( (25, [1, 7, 3, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7]) \)

Figure 3.1244 provides an example where the \text{SUM}_\text{SURF}_\text{GORGE} \{25, [1, 7, 3, 4, 5, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7]\} constraint holds.

**Typical**

- \( |\text{VARIABLES}| > 2 \)
- \( \text{range}(\text{VARIABLES}.\text{var}) > 1 \)

where

- \( sv = |\text{VARIABLES}| \)
- \( np = \max(0, [(sv - 1) / 2]) \)
- \( \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \)
- \( \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \)
- \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

\( \text{VALUE} \) is the sum of the surface of occurrences of the GORGE pattern in the time-series given by the \text{VARIABLES} collection. If the pattern does not occur, \text{VALUE} takes the default value 0.
Symmetry  
Items of VARIABLES can be reversed.

Arg. properties  
Functional dependency: VALUE determined by VARIABLES.
Figures 3.1245 and 3.1246 respectively depict the automaton associated with the constraint SUM_SURF_GORGE and its simplified form.

![Automaton Diagram]

Figure 3.1245: Automaton for the SUM_SURF_GORGE constraint obtained by applying decoration Table 2.35 to the seed transducer of the GORGE pattern where default is 0 (transition u \rightarrow r has the same accumulator update as transition r \rightarrow u).

<table>
<thead>
<tr>
<th>s</th>
<th>r</th>
<th>t</th>
<th>u</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\overline{C} + \overline{C})</td>
</tr>
<tr>
<td>r</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\overline{D} + \overline{D} + \overline{VAR_i})</td>
<td>(\overline{C} + \overline{D} + \overline{D} + \overline{VAR_i})</td>
</tr>
<tr>
<td>t</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\overline{C} + \overline{D} + \overline{D} + \overline{VAR_i})</td>
<td>(\overline{C} + \overline{C})</td>
</tr>
<tr>
<td>u</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\overline{C} + \overline{C})</td>
<td>(\overline{C} + \overline{D} + \overline{D} + \overline{VAR_i})</td>
</tr>
</tbody>
</table>

Table 3.171: Glue matrix for the SUM_SURF_GORGE constraint defined as the composition of the GORGE pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 3.1246: Automaton for the SUM_SURF_GORGE constraint obtained by applying decoration Table 2.25 to the seed transducer of the GORGE pattern where default is 0 (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$)
3.594  SUM_SURF_INCREASING

### Description

**Origin**

Based on the `INCREASING` pattern.

**Constraint**

\[ \text{SUM\_SURF\_INCREASING}(\text{VALUE}, \text{VARIABLES}) \]

**Arguments**

\[ \begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} \rightarrow \text{dvar})
\end{align*} \]

**Restrictions**

\[ \begin{align*}
\text{sv} & \leq 1 \lor \text{rv} \leq 1 \implies \text{VALUE} = 0 \\
\text{VALUE} & \geq \min_{q \in [1, \text{sv}]} \min \left( \begin{array}{c}
0, \sum \Pi \left( \text{q} \cdot \left( \sum \left( \frac{2 \cdot (\text{np} - 1) \cdot \text{minv}}{\text{np} \cdot \text{np}} \right) \right) + 1 \right), \\
\text{q} \cdot \left( \sum \left( \frac{2 \cdot (\text{np} - 1) \cdot \text{maxv}}{\text{np} \cdot \text{np}} \right) \right) - 1, \\
\text{q} \cdot \left( \sum \left( \frac{2 \cdot \text{minv}}{\text{np} \cdot \text{np}} \right) \right), \\
\text{q} \cdot \left( \sum \left( \frac{2 \cdot \text{maxv}}{\text{np} \cdot \text{np}} \right) \right) - 1, \\
\text{q} \cdot \left( \sum \left( \frac{2 \cdot \text{maxv}}{\text{np} \cdot \text{np}} \right) \right) - 1 \end{array} \right)
\end{align*} \]

\[ \text{VALUE} \leq \max_{q \in [1, \text{sv}]} \max \left( \begin{array}{c}
0, \sum \Pi \left( \text{q} \cdot \left( \sum \left( \frac{2 \cdot (\text{np} - 1) \cdot \text{minv}}{\text{np} \cdot \text{np}} \right) \right) + 1 \right), \\
\text{q} \cdot \left( \sum \left( \frac{2 \cdot (\text{np} - 1) \cdot \text{maxv}}{\text{np} \cdot \text{np}} \right) \right) - 1, \\
\text{q} \cdot \left( \sum \left( \frac{2 \cdot \text{minv}}{\text{np} \cdot \text{np}} \right) \right), \\
\text{q} \cdot \left( \sum \left( \frac{2 \cdot \text{maxv}}{\text{np} \cdot \text{np}} \right) \right) - 1, \\
\text{q} \cdot \left( \sum \left( \frac{2 \cdot \text{maxv}}{\text{np} \cdot \text{np}} \right) \right) - 1 \end{array} \right)
\]

\[ \text{required}(\text{VARIABLES}, \text{var}) \]

where

\[ \begin{align*}
\text{sv} & = |\text{VARIABLES}| \\
\text{np} & = \lfloor \text{sv}/2 \rfloor + 1 \\
\text{maxv} & = \maxval(\text{VARIABLES}, \text{var}) \\
\text{minv} & = \minval(\text{VARIABLES}, \text{var}) \\
\text{rv} & = \range(\text{VARIABLES}, \text{var}) \\
\text{lb1} & = \min \left( \left( \lfloor \text{sv}/2 \rfloor + 1, \sum \left( \frac{\text{sv}/\min(\text{minv}, \text{rv}) + 1]}{\min(1, \text{sv} \mod \min(\text{minv}, \text{rv}) + 1)} \right) \right) \right) \\
\text{ub1} & = \lfloor \text{sv}/2 \rfloor + 1 \\
\text{lb2} & = \min \left( \left( \lfloor \text{sv}/2 \rfloor + 1, \sum \left( \frac{\text{sv}/\min(\text{maxv}, \text{rv}) + 1]}{\min(1, \text{sv} \mod \min(\text{maxv}, \text{rv}) + 1)} \right) \right) \right) \\
\text{ub2} & = \lfloor \text{sv}/2 \rfloor + 1
\end{align*} \]
VALUE is the sum of the surface of occurrences of the INCREASING pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern INCREASING is the subsequence which matches the regular expression \(<\).

Assume that the occurrence of the pattern INCREASING starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i\) to index \(j + 1\).

**Example**

\((33, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))\)

Figure 3.1247 provides an example where the SUM_SURF_INCREASING \((33, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3])\) constraint holds.

Figure 3.1247: Illustrating the SUM_SURF_INCREASING constraint of the Example slot

**Typical**

\(|\text{VARIABLES}| > 1\)

\(\text{range} (\text{VARIABLES}.\text{var}) > 1\)

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 3.1248 and 3.1249 respectively depict the automaton associated with the constraint \texttt{SUM\_SURF\_INCREASING} and its simplified form.

\begin{equation}
\begin{cases}
C \leftarrow \text{default} \\
D \leftarrow 0 \\
R \leftarrow \text{default}
\end{cases}
\end{equation}

\begin{equation}
\begin{cases}
\{D \leftarrow 0 \} \\
\{R \leftarrow R + D + \text{VAR}_i + \text{VAR}_{i+1}\}
\end{cases}
\end{equation}

Figure 3.1248: Automaton for the \texttt{SUM\_SURF\_INCREASING} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{INCREASING} pattern where \texttt{default} is 0

\begin{equation}
\begin{cases}
\{R \leftarrow \text{default}\}
\end{cases}
\end{equation}

\begin{equation}
\begin{cases}
\{R \leftarrow R + \text{VAR}_i + \text{VAR}_{i+1}\}
\end{cases}
\end{equation}

Figure 3.1249: Automaton for the \texttt{SUM\_SURF\_INCREASING} constraint obtained by applying decoration Table 2.38 to the seed transducer of the \texttt{INCREASING} pattern where \texttt{default} is 0

Table 3.172: Glue matrix for the \texttt{SUM\_SURF\_INCREASING} constraint defined as the composition of the \texttt{INCREASING} pattern, the feature \texttt{SURF}, and the aggregator \texttt{sum}; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.595 SUM_SURF_INCREASING_SEQUENCE

**DESCRIPTION**

Origin

Based on the INCREASING_SEQUENCE pattern.

Constraint

SUM_SURF_INCREASING_SEQUENCE(VALUE, VARIABLES)

Arguments

VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions

\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \]
\[ rv = 2 \Rightarrow VALUE = 0 \lor VALUE \geq \min(2 \cdot \text{minv} + 1, (2 \cdot \text{minv} + 1) \cdot \text{np}) \]
\[ rv \geq 3 \Rightarrow \]
\[ VALUE = 0, \]
\[ \lor VALUE \geq \min \left( 2 \cdot \text{minv} + 1, (2 \cdot \text{minv} + 1) \cdot \text{np} + \min(0, sv \mod 2 \cdot (\text{minv} + 2)) \right) \]
\[ rv = 2 \Rightarrow VALUE = 0 \lor VALUE \leq \max(2 \cdot \text{maxv} - 1, (2 \cdot \text{maxv} - 1) \cdot \text{np}) \]
\[ rv \geq 3 \Rightarrow \]
\[ VALUE = 0, \]
\[ \lor VALUE \leq \max \left( 2 \cdot \text{maxv} - 3, (2 \cdot \text{maxv} - 1) \cdot \text{np} + \max(0, sv \mod 2 \cdot (\text{maxv} - 2)) \right) \]

required(VARIABLES, var)

where

\[ sv = |\text{VARIABLES}| \]
\[ np = \lfloor sv/2 \rfloor \]
\[ \text{minv} = \text{minval}(\text{VARIABLES}.\text{var}) \]
\[ \text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var}) \]
\[ rv = \text{range}(\text{VARIABLES}.\text{var}) \]

VALUE is the sum of the surface of occurrences of the INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression \(< (< | =)^* < | \)\.

Assume that the occurrence of the pattern INCREASING_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i \) to index \( j + 1 \).

Example

\((29, (4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3))\)

Figure 3.125 provides an example where the SUM_SURF_INCREASING_SEQUENCE (29, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3]) constraint holds.
Figure 3.1250: Illustrating the SUM_SURF_INCREASING_SEQUENCE constraint of the Example slot

**Typical**

\[ |VARIABLES| > 1 \]
\[ \text{range} (VARIABLES.var) > 1 \]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 3.1251 and 3.1252 respectively depict the automaton associated with the constraint SUM_SURF_INCREASING_SEQUENCE and its simplified form.

Figure 3.1251: Automaton for the SUM_SURF_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the INCREASING_SEQUENCE pattern where default is 0

Figure 3.1252: Automaton for the SUM_SURF_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.29 to the seed transducer of the INCREASING_SEQUENCE pattern where default is 0
Table 3.173: Glue matrix for the **SUM_SURF_INCREASING_SEQUENCE** constraint defined as the composition of the **INCREASING_SEQUENCE** pattern, the feature **SURF**, and the aggregator **sum**; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\vec{C} + \vec{C}$</td>
<td>$\vec{C} + \vec{C}$</td>
</tr>
<tr>
<td>t</td>
<td>$\vec{C} + \vec{C}$</td>
<td>$\vec{C} + \vec{C} + \vec{D} + \vec{D} + \text{VAR}_i$</td>
</tr>
</tbody>
</table>
3.596 SUM_SURF_INCREASING_TERRACE

**DESCRIPTION**

Origin

Based on the INCREASING_TERRACE pattern.

Constraint

SUM_SURF_INCREASING_TERRACE(VALUE, VARIABLES)

Arguments

VALUE : dvar
VARIABLES : collection(var–dvar)

Restrictions

\[ sv \leq 3 \lor rv \leq 2 \Rightarrow VALUE = 0 \]
\[ VALUE = 0 \lor VALUE \geq \text{min}(2 \cdot (\text{minv} + 1) \cdot (sv - 2) \cdot (\text{minv} + 1) + 2) \]
\[ VALUE = 0 \lor VALUE \leq \text{max}(2 \cdot (\text{maxv} - 1) \cdot (sv - 2) \cdot (\text{maxv} - 1) + 2) \]

where

\[ \text{minv} = \text{minval}(\text{VARIABLES} \cdot \text{var}) \]
\[ \text{maxv} = \text{maxval}(\text{VARIABLES} \cdot \text{var}) \]
\[ sv = |\text{VARIABLES}| \]
\[ rv = \text{range}(\text{VARIABLES} \cdot \text{var}) \]

VALUE is the sum of the surface of occurrences of the INCREASING_TERRACE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

Purpose

An occurrence of the pattern INCREASING_TERRACE is the maximal subsequence which matches the regular expression \(<=+<\).

Assume that the occurrence of the pattern INCREASING_TERRACE starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

Example

\[(19, [1, 3, 3, 3, 2, 5, 6, 4, 4, 2, 3, 3, 4, 4])\]

Figure 3.1253 provides an example where the SUM_SURF_INCREASING_TERRACE \((19, [1, 3, 3, 3, 2, 5, 6, 4, 4, 2, 3, 3, 4, 4])\) constraint holds.

Typical

\[ |\text{VARIABLES}| > 3 \]
\[ \text{range}(\text{VARIABLES} \cdot \text{var}) > 2 \]

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 3.1253: Illustrating the SUM_SURF_INCREASING_TERRACE constraint of the Example slot
Figures 3.1254 and 3.1255 respectively depict the automaton associated with the constraint SUM_SURF_INCREASING_TERRACE and its simplified form.

\[ \begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow \text{default} \\
\end{align*} \]

\[ \geq s \]

\[ \begin{align*}
R + C &> \ \\
\end{align*} \]

\[ < \]

\[ \begin{align*}
{\{ D \leftarrow D + \text{VAR}_i \}} \\
{\{ R \leftarrow R + D + \text{VAR}_i \}} \\
\end{align*} \]

Figure 3.1254: Automaton for the SUM_SURF_INCREASING_TERRACE constraint obtained by applying decoration Table 2.35 to the seed transducer of the INCREASING_TERRACE pattern where default is 0

\[ \begin{align*}
D &\leftarrow 0 \\
R &\leftarrow \text{default} \\
\end{align*} \]

\[ \geq s \]

\[ \begin{align*}
R &> \ \\
\end{align*} \]

\[ < \]

\[ \begin{align*}
{\{ D \leftarrow D + \text{VAR}_i \}} \\
\end{align*} \]

\[ \begin{align*}
{\{ D \leftarrow D + \text{VAR}_i \}} \\
\end{align*} \]

\[ \begin{align*}
{\{ D \leftarrow 0 \}} \\
\end{align*} \]

\[ \begin{align*}
{\{ R \leftarrow R + D + \text{VAR}_i \}} \\
\end{align*} \]

Figure 3.1255: Automaton for the SUM_SURF_INCREASING_TERRACE constraint obtained by applying decoration Table 2.28 to the seed transducer of the INCREASING_TERRACE pattern where default is 0
Table 3.174: Glue matrix for the SUM_SURF_INCREASING_TERRACE constraint defined as the composition of the INCREASING_TERRACE pattern, the feature SURF, and the aggregator sum: cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>C + C</td>
<td>C + C</td>
<td>C + C</td>
</tr>
<tr>
<td>r</td>
<td>C + C</td>
<td>C + C</td>
<td>D + D + VAR_i</td>
</tr>
<tr>
<td>t</td>
<td>C + C</td>
<td>D + D + VAR_i</td>
<td>D + D + VAR_i</td>
</tr>
</tbody>
</table>
### DESCRIPTION

**Origin**
Based on the INFLEXION pattern.

**Constraint**
```
SUM_SURF_INFLEXION(VALUE, VARIABLES)
```

**Arguments**
```
VALUE : dvar
VARIABLES : collection(var−dvar)
```

**Restrictions**
```
sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = 0
VALUE = 0 \lor VALUE \geq \min(minv_1, (sv - 2) * minv_2)
VALUE = 0 \lor VALUE \leq \max(maxv_3, (sv - 2) * maxv_4)
```

where
```
minv = \minval(VARIABLES, var)
maxv = \maxval(VARIABLES, var)
sv = |VARIABLES|
rv = \range(VARIABLES, var)
```

**Purpose**
VALUE is the sum of the surface of occurrences of the INFLEXION pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression `< (< | =)* > | > (>|=)* >`.

Assume that the occurrence of the pattern INFLEXION starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 1$ to index $j$.

**Example**
```
(49, (1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4))
```

Figure 3.1256 provides an example where the SUM_SURF_INFLEXION ($49, [1, 2, 6, 4, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4]$) constraint holds.

**Typical**
```
|VARIABLES| > 2
range(VARIABLES, var) > 1
```

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.1256: Illustrating the SUM_SURF_INFLEXION constraint of the Example slot
Figures 3.1257 and 3.1258 respectively depict the automaton associated with the constraint SUM_SURF_INFLEXION and its simplified form.

Figure 3.1257: Automaton for the SUM_SURF_INFLEXION constraint obtained by applying decoration Table 2.35 to the seed transducer of the INFLEXION pattern where default is 0 (transition $r \rightarrow t$ has the same accumulators updates as transition $t \rightarrow r$)

Figure 3.1258: Automaton for the SUM_SURF_INFLEXION constraint obtained by applying decoration Table 2.25 to the seed transducer of the INFLEXION pattern where default is 0 (transition $r \rightarrow t$ has the same accumulators updates as transition $t \rightarrow r$)
SUM_SURF_INPLEXION

2527
3.598 SUM_SURF_PEAK

**DESCRIPTION**

**Origin**

Based on the PEAK pattern.

**Constraint**

\[ \text{SUM_SURF_PEAK}(\text{VALUE}, \text{VARIABLES}) \]

**Arguments**

\[
\begin{align*}
\text{VALUE} &: \text{dvar} \\
\text{VARIABLES} &: \text{collection(var-dvar)}
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
sv &\leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} &= 0 \lor \text{VALUE} \geq \min(\minv + 1, (sv - 2) \cdot (\minv + 1) \cdot rz) \\
\text{VALUE} &= 0 \lor \text{VALUE} \leq \max(\maxv \cdot rz, (sv - 2) \cdot \maxv \cdot rz) \\
\text{VALUE} &= 0 \lor \text{VALUE} \leq \minv + \maxv \cdot rz \\
\text{VALUE} &= 0 \lor \text{VALUE} \leq (11 - 2) \cdot (-3 + 1) \\
\text{VALUE} &= 0 \lor \text{VALUE} \leq 18 = (11 - 2) \cdot (-3 + 1)
\end{align*}
\]

where

\[
\begin{align*}
\minv &= \text{minval} (\text{VARIABLES}.\text{var}) \\
\maxv &= \text{maxval} (\text{VARIABLES}.\text{var}) \\
sv &= |\text{VARIABLES}| \\
rv &= \text{range} (\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression \(< (= | <) ^* (>| =) ^* >\). Assume that the occurrence of the pattern PEAK starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

\[
(32, (7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 1))
\]

Figure 3.1259 provides an example where the SUM_SURF_PEAK \((32, [7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 1])\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 2 \\
\text{range} (\text{VARIABLES}.\text{var}) > 1
\]

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.1259: Illustrating the **SUM_SURF_PEAK** constraint of the **Example** slot
Figures 3.1260 and 3.1261 respectively depict the automaton associated with the constraint SUM_SURF_PEAK and its simplified form.

Figure 3.1260: Automaton for the SUM_SURF_PEAK constraint obtained by applying decoration Table 2.35 to the seed transducer of the PEAK pattern where default is 0

Figure 3.1261: Automaton for the SUM_SURF_PEAK constraint obtained by applying decoration Table 2.25 to the seed transducer of the PEAK pattern where default is 0
Table 3.175: Glue matrix for the SUM_SURF_PEAK constraint defined as the composition of the PEAK pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{C}$</td>
</tr>
<tr>
<td>r</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{D} + \overline{D} + \text{VAR}_i$</td>
<td>$\overline{C} + \overline{D} + \overline{D} + \text{VAR}_i$</td>
</tr>
<tr>
<td>t</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{D} + \overline{D} + \text{VAR}_i$</td>
<td>$\overline{C} + \overline{C}$</td>
</tr>
</tbody>
</table>
### 3.599 SUM_SURF_PLAIN

**Description**

**Origin**

Based on the \textsc{plain} pattern.

**Constraint**

\[
\text{SUM\_SURF\_PLAIN}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection} (\text{var} - \text{dvar})
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
\text{sv} \leq 2 & \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & = 0 \lor \text{VALUE} \geq \min(\text{minv}, \text{minv} \times (\text{sv} - 2))^2 \\
\text{VALUE} & = 0 \lor \text{VALUE} \leq \max(\maxv - 1, (\maxv - 1) \times (\text{sv} - 2))
\end{align*}
\]

where

\[
\begin{align*}
\text{minv} & = \minval(\text{VARIABLES}.\text{var}) \\
\text{maxv} & = \maxval(\text{VARIABLES}.\text{var}) \\
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

An occurrence of the pattern \textsc{plain} is the maximal subsequence which matches the regular expression \texttt{\textbf{>}}\texttt{=}\texttt{*}\texttt{\textless{}}.

Assume that the occurrence of the pattern \textsc{plain} starts at position \(i\) and ends at position \(j\). The feature \textsc{surf} computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**

\[
(15, \langle 2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 6, 6, 3 \rangle)
\]

Figure 3.1262 provides an example where the \textsc{sum\_surf\_plain} \((15, [2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 6, 6, 3])\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 2 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Symmetry**

Items of \textsc{variables} can be reversed.

**Arg. properties**

Functional dependency: \textsc{value} determined by \textsc{variables}.
Figure 3.1262: Illustrating the SUM_SURF_PLAIN constraint of the Example slot
Figures 3.1263 and 3.1264 respectively depict the automaton associated with the constraint SUM_SURF_PLAIN and its simplified form.

**Figure 3.1263:** Automaton for the SUM_SURF_PLAIN constraint obtained by applying decoration Table 2.35 to the seed transducer of the PLAIN pattern where default is 0

**Figure 3.1264:** Automaton for the SUM_SURF_PLAIN constraint obtained by applying decoration Table 2.28 to the seed transducer of the PLAIN pattern where default is 0
Table 3.176: Glue matrix for the SUM_SURF_PLAIN constraint defined as the composition of the PLAIN pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.600  SUM_SURF_PLATEAU

**Origin**
Based on the PLATEAU pattern.

**Constraint**
SUM_SURF_PLATEAU(VALUE, VARIABLES)

**Arguments**
VALUE : dvar
VARIABLES : collection(var−dvar)

**Restrictions**

\[
\begin{align*}
sv \leq 2 \lor rv \leq 1 & \Rightarrow VALUE = 0 \\
VALUE = 0 \lor VALUE \geq \min(minv + 1, (sv - 2) \times (minv + 1) \times 2) \\
VALUE = 0 \lor VALUE \leq \max(maxv, (sv - 2) \times maxv) \\
\end{align*}
\]

required(VARIABLES, var)

where

\[
\begin{align*}
minv &= \text{minval}(\text{VARIABLES}\text{.var}) \\
maxv &= \text{maxval}(\text{VARIABLES}\text{.var}) \\
sv &= |\text{VARIABLES}| \\
rv &= \text{range}(\text{VARIABLES}\text{.var})
\end{align*}
\]

**Purpose**
VALUE is the sum of the surface of occurrences of the PLATEAU pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression <\=*>. Assume that the occurrence of the pattern PLATEAU starts at position \(i\) and ends at position \(j\). The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

**Example**
\[(17, [7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5])\]

Figure 3.1265 provides an example where the SUM_SURF_PLATEAU (17, [7, 5, 2, 3, 1, 2, 2, 4, 3, 3, 4, 5, 5, 2, 2, 5]) constraint holds.

**Typical**
\[
\begin{align*}
|\text{VARIABLES}| & > 2 \\
\text{range}(\text{VARIABLES}\text{.var}) & > 1
\end{align*}
\]

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.1265: Illustrating the SUM_SURF_PLATEAU constraint of the **Example** slot
Automaton

Figures 3.1266 and 3.1267 respectively depict the automaton associated with the constraint SUM_SURF_PLATEAU and its simplified form.

Figure 3.1266: Automaton for the SUM_SURF_PLATEAU constraint obtained by applying decoration Table 2.35 to the seed transducer of the PLATEAU pattern where default is 0

Figure 3.1267: Automaton for the SUM_SURF_PLATEAU constraint obtained by applying decoration Table 2.28 to the seed transducer of the PLATEAU pattern where default is 0
Table 3.177: Glue matrix for the `SUM_SURF_PLATEAU` constraint defined as the composition of the `PLATEAU` pattern, the feature `SURF`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
</tr>
<tr>
<td>r</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i$</td>
</tr>
<tr>
<td>t</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + \text{VAR}_i$</td>
</tr>
</tbody>
</table>
### 3.601 SUM_SURF_PROPER.PLAIN

**Origin**
Based on the PROPER.PLAIN pattern.

**Constraint**
SUM_SURF_PROPER.PLAIN(VALUE, VARIABLES)

**Arguments**
- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**
- $sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = 0$
- $VALUE = 0 \lor VALUE \geq \min(2 \ast \text{minv}_x, (sv - 2) \ast \text{minv}_y)$
- $VALUE = 0 \lor VALUE \leq \max(2 \ast (\maxv_x - 1), (sv - 2) \ast (\maxv_x - 1))$

where
- $\text{minv} = \text{minval}(\text{VARIABLES}.\text{var})$
- $\text{maxv} = \text{maxval}(\text{VARIABLES}.\text{var})$
- $sv = |\text{VARIABLES}|$
- $rv = \text{range}(\text{VARIABLES}.\text{var})$

**Purpose**
VALUE is the sum of the surface of occurrences of the PROPER.PLAIN pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern PROPER.PLAIN is the maximal subsequence which matches the regular expression $> = + <$.

Assume that the occurrence of the pattern PROPER.PLAIN starts at position $i$ and ends at position $j$. The feature SURF computes the sum of the values from index $i + 1$ to index $j$.

**Example**

$$(27, [2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3, 5])$$

Figure 3.1268 provides an example where the SUM_SURF_PROPER.PLAIN ($27, [2, 7, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3, 5]$) constraint holds.

**Typical**

$|\text{VARIABLES}| > 3$

$\text{range}(\text{VARIABLES}.\text{var}) > 1$

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.1268: Illustrating the \texttt{SUM}\_\texttt{SURF}\_\texttt{PROPER}\_\texttt{PLAIN} constraint of the \textbf{Example} slot
Automaton

Figures 3.1269 and 3.1270 respectively depict the automaton associated with the constraint `SUM_SURF_PROPER_PLAIN` and its simplified form.

Figure 3.1269: Automaton for the `SUM_SURF_PROPER_PLAIN` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `PROPER_PLAIN` pattern where `default` is 0.

Figure 3.1270: Automaton for the `SUM_SURF_PROPER_PLAIN` constraint obtained by applying decoration Table 2.28 to the seed transducer of the `PROPER_PLAIN` pattern where `default` is 0.
Table 3.178: Glue matrix for the SUM_SRF_PROPERPLAIN constraint defined as the composition of the PROPERPLAIN pattern, the feature SURF, and the aggregator sum: cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### 3.602 SUM_SURF_PROPER_PLATEAU

**DESCRIPTION**

- **Origin**: Based on the \textit{PROPER_PLATEAU} pattern.

- **Constraint**: \texttt{SUM\_SURF\_PROPER\_PLATEAU(VALUE, VARIABLES)}

- **Arguments**:
  
  - \texttt{VALUE} : dvar
  - \texttt{VARIABLES} : collection(var–dvar)

- **Restrictions**:
  
  \begin{align*}
  \text{sv} \leq 3 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \\
  \text{VALUE} = 0 \lor \text{VALUE} \geq \min(2 \ast (\minv + 1), (\text{sv} - 2) \ast (\minv + 1)) \\
  \text{VALUE} = 0 \lor \text{VALUE} \leq \max(2 \ast \maxv, (\text{sv} - 2) \ast \maxv) \\
  \end{align*}

  where
  
  \begin{align*}
  \minv &= \text{minval(VARIABLES.var)} \\
  \maxv &= \text{maxval(VARIABLES.var)} \\
  \text{sv} &= |\text{VARIABLES}| \\
  \text{rv} &= \text{range(VARIABLES.var)}
  \end{align*}

- **Purpose**

  VALUE is the sum of the surface of occurrences of the \textit{PROPER_PLATEAU} pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

  An occurrence of the pattern \textit{PROPER_PLATEAU} is the \textit{maximal} subsequence which matches the regular expression $< = + >$.

  Assume that the occurrence of the pattern \textit{PROPER_PLATEAU} starts at position $i$ and ends at position $j$. The feature \textit{SURF} computes the sum of the values from index $i + 1$ to index $j$.

- **Example**:

  \[(29, (7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3))\]

  Figure 3.1271 provides an example where the \texttt{SUM\_SURF\_PROPER\_PLATEAU (29, [7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3])} constraint holds.

- **Typical**:

  \begin{align*}
  |\text{VARIABLES}| &> 3 \\
  \text{range(VARIABLES.var)} &> 1
  \end{align*}

- **Symmetry**: Items of VARIABLES can be reversed.

- **Arg. properties**: Functional dependency: VALUE determined by VARIABLES.
Figure 3.1271: Illustrating the SUM_SURF_PROPER_PLATEAU constraint of the Example slot
Automaton

Figures 3.1272 and 3.1273 respectively depict the automaton associated with the constraint \texttt{SUM\_SURF\_PROPER\_PLATEAU} and its simplified form.

\begin{figure}[h]
\centering
\includegraphics[width=0.8\textwidth]{figure3.1272.png}
\caption{Automaton for the \texttt{SUM\_SURF\_PROPER\_PLATEAU} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{PROPER\_PLATEAU} pattern where \texttt{default} is 0.}
\end{figure}

\begin{figure}[h]
\centering
\includegraphics[width=0.8\textwidth]{figure3.1273.png}
\caption{Automaton for the \texttt{SUM\_SURF\_PROPER\_PLATEAU} constraint obtained by applying decoration Table 2.28 to the seed transducer of the \texttt{PROPER\_PLATEAU} pattern where \texttt{default} is 0.}
\end{figure}
<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{C}$</td>
</tr>
<tr>
<td>r</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{D} + \overline{D} + \text{VAR}_i$</td>
</tr>
<tr>
<td>t</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{D} + \overline{D} + \text{VAR}_i$</td>
<td>$\overline{D} + \overline{D} + \text{VAR}_i$</td>
</tr>
</tbody>
</table>

Table 3.179: Glue matrix for the SUM_SURF_PROPER_PLATEAU constraint defined as the composition of the PROPER_PLATEAU pattern, the feature SURF, and the aggregator sum: cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.603 \textsc{sum\_surf\_steady}

\begin{itemize}
  \item \textbf{DESCRIPTION}
  \begin{itemize}
    \item \textbf{Origin}: Based on the \textsc{steady} pattern.
    \item \textbf{Constraint}: \textsc{sum\_surf\_steady} \((\text{value, variables})\)
    \item \textbf{Arguments}:
      \begin{itemize}
        \item \textsc{value} : \texttt{dvar}
        \item \textsc{variables} : \texttt{collection(var-dvar)}
      \end{itemize}
    \item \textbf{Restrictions}:
      \begin{align*}
        & sv \leq 1 \Rightarrow \text{value} = 0 \\
        & rv = 1 \Rightarrow \text{value} \geq 2 + \text{minv} \times \text{np}
      \end{align*}
      where
      \begin{align*}
        sv &= |\text{variables}| \\
        np &= \max(0, sv - 1) \\
        \text{minv} &= \text{minval}(\text{variables}.) \\
        \text{maxv} &= \text{maxval}(\text{variables}.) \\
        rv &= \text{range}(\text{variables}.)
      \end{align*}
      \begin{align*}
        & rv \geq 2 \Rightarrow \text{value} = 0 \lor \text{value} \geq \min(2 \times \text{minv}, 2 \times \text{minv} \times \text{np}) \\
        & rv = 1 \Rightarrow \text{value} \leq 2 + \text{maxv} \times \text{np}
      \end{align*}
      \begin{align*}
        & rv \geq 2 \Rightarrow \text{value} = 0 \lor \text{value} \leq \max(2 \times \text{maxv}, 2 \times \text{maxv} \times \text{np})
      \end{align*}
  \end{itemize}
  \item \textbf{Purpose}:
    \begin{itemize}
      \item \textsc{value} is the sum of the surface of occurrences of the \textsc{steady} pattern in the time-series given by the \texttt{variables} collection. If the pattern does not occur, \textsc{value} takes the default value 0.
      \item An occurrence of the pattern \textsc{steady} is the subsequence which matches the regular expression \texttt{=}. Assume that the occurrence of the pattern \textsc{steady} starts at position \(i\) and ends at position \(j\). The feature \texttt{surf} computes the sum of the values from index \(i\) to index \(j + 1\).
    \end{itemize}
  \item \textbf{Example}:
    \begin{itemize}
      \item \texttt{(60, [1, 1, 7, 3, 3, 5, 5, 6, 5, 5, 7, 2, 6, 6])}
    \end{itemize}
  \item \textbf{Typical}:
    \begin{itemize}
      \item \(|\text{variables}| > 1\)
    \end{itemize}
  \item \textbf{Symmetry}:
    \begin{itemize}
      \item Items of \texttt{variables} can be \texttt{reversed}.
    \end{itemize}
  \item \textbf{Arg. properties}:
    \begin{itemize}
      \item Functional dependency: \textsc{value} determined by \texttt{variables}.
    \end{itemize}
\end{itemize}
Figure 3.1274: Illustrating the SUM_SURF_STEADY constraint of the Example slot
Automaton

Figures 3.1275 and 3.1276 respectively depict the automaton associated with the constraint `SUM_SURF_STEADY` and its simplified form.

\[
\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow \text{default}
\end{align*}
\]

\[
\begin{align*}
D &\leftarrow 0 \\
R &\leftarrow R + D + \text{VAR}_i + \text{VAR}_{i+1}
\end{align*}
\]

\[
R + C
\]

Figure 3.1275: Automaton for the `SUM_SURF_STEADY` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `STEADY` pattern where `default` is 0

\[
\begin{align*}
\{R &\leftarrow \text{default}\} \\
\{R &\leftarrow R + \text{VAR}_i + \text{VAR}_{i+1}\}
\end{align*}
\]

\[
R
\]

Figure 3.1276: Automaton for the `SUM_SURF_STEADY` constraint obtained by applying decoration Table 2.38 to the seed transducer of the `STEADY` pattern where `default` is 0

\[
\begin{array}{c}
\begin{array}{c}
\text{s} \\
\text{s} \rightarrow C + C
\end{array}
\end{array}
\]

Table 3.180: Glue matrix for the `SUM_SURF_STEADY` constraint defined as the composition of the `STEADY` pattern, the feature `SURF`, and the aggregator `sum`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.604  SUM_SURF_STEADY_SEQUENCE

OVERVIEW

DESCRIPTION

Based on the STEADY_SEQUENCE pattern.

Constraint

SUM_SURF_STEADY_SEQUENCE(VALUE, VARIABLES)

Arguments

VALUE : dvar
VARIABLES : collection(var–dvar)

Restrictions

sv ≤ 1 ⇒ VALUE = 0
rv = 1 ∧ sv ≥ 2 ⇒ VALUE ≥ minv * sv
rv ≥ 2 ⇒ VALUE = 0 ∨ VALUE ≥ min(2 * minv, minv * sv)
rv = 1 ∧ sv ≥ 2 ⇒ VALUE ≤ maxv * sv
rv ≥ 2 ⇒ VALUE = 0 ∨ VALUE ≤ max(2 * maxv, maxv * sv)

required(VARIABLES, var)

where

minv = minval(VARIABLES, var)
maxv = maxval(VARIABLES, var)
sv = |VARIABLES|
rv = range(VARIABLES, var)

VALUE is the sum of the surface of occurrences of the STEADY_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern STEADY_SEQUENCE is the maximal subsequence which matches the regular expression =+.

Assume that the occurrence of the pattern STEADY_SEQUENCE starts at position i and ends at position j. The feature SURF computes the sum of the values from index i to index j + 1.

Example

(31, (3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1))

Figure 3.1277 provides an example where the SUM_SURF_STEADY_SEQUENCE (31, [3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1]) constraint holds.

Typical

|VARIABLES| > 1

Symmetry

Items of VARIABLES can be reversed.

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 3.1277: Illustrating the SUM_SURF_STEADY_SEQUENCE constraint of the Example slot
Automaton

Figures 3.1278 and 3.1279 respectively depict the automaton associated with the constraint SUM_SURF_STEADY_SEQUENCE and its simplified form.

\[
C \leftarrow \text{default} \\
D \leftarrow 0 \\
R \leftarrow \text{default}
\]

\[
\Rightarrow
\]

\[
\begin{align*}
C & \leftarrow \text{default} \\
D & \leftarrow 0 \\
R & \leftarrow R + C
\end{align*}
\]

\[
\Rightarrow
\]

\[
\begin{align*}
C & \leftarrow \text{default} \\
D & \leftarrow 0
\end{align*}
\]

\[
\Rightarrow
\]

Figure 3.1278: Automaton for the SUM_SURF_STEADY_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the STEADY_SEQUENCE pattern where \(\text{default}\) is 0

\[
\begin{align*}
R & \leftarrow \text{default}
\end{align*}
\]

\[
\Rightarrow
\]

\[
\begin{align*}
R & \leftarrow R + \text{VAR}_i + \text{VAR}_{i+1}
\end{align*}
\]

\[
\Rightarrow
\]

\[
\begin{align*}
R & \leftarrow R + \text{VAR}_{i+1}
\end{align*}
\]

Figure 3.1279: Automaton for the SUM_SURF_STEADY_SEQUENCE constraint obtained by applying decoration Table 2.38 to the seed transducer of the STEADY_SEQUENCE pattern where \(\text{default}\) is 0
Table 3.181: Glue matrix for the `SUM_SURF_STEADY_SEQUENCE` constraint defined as the composition of the `STEADY_SEQUENCE` pattern, the feature `SURF`, and the aggregator `sum`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>$s$</th>
<th>$r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
</tr>
<tr>
<td>$r$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + \overrightarrow{VAR}$</td>
</tr>
</tbody>
</table>
3.605 **SUM_SURF_STRICTLY_DECREASING_SEQUENCE**

### DESCRIPTION

**Origin**
Based on the **STRICTLY_DECREASING_SEQUENCE** pattern.

**Constraint**

\[
\text{SUM_SURF_STRICTLY_DECREASING_SEQUENCE} (\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

\[
\begin{align*}
\text{VALUE} & : dvar \\
\text{VARIABLES} & : \text{collection}(\text{var} - \text{dvar})
\end{align*}
\]

**Restrictions**

\[
\begin{align*}
sv & \leq 1 \lor rv \leq 1 \implies \text{VALUE} = 0 \\
rv & = 2 \implies \text{VALUE} = 0 \lor \text{VALUE} \geq \min(2 \times \text{minv} + 1, (2 \times \text{minv} + 1) \times \text{np}) \\
rv & \geq 3 \\
& \quad \implies \text{VALUE} = 0, \\
& \quad \lor \text{VALUE} \geq \min \left( 2 \times \text{minv} + 1, \\
& \quad \quad \quad \quad \quad \left( \text{maxv} - 1 \right) - \text{maxv} + \min(0, sv \mod 2 \times (\text{minv} + 2)) \right) \\
rv & = 2 \implies \text{VALUE} = 0 \lor \text{VALUE} \leq \max(2 \times \text{maxv} - 1, (2 \times \text{maxv} - 1) \times \text{np}) \\
rv & \geq 3 \\
& \quad \implies \text{VALUE} = 0, \\
& \quad \lor \text{VALUE} \leq \max \left( 2 \times \text{maxv} - 1 \right) - \text{maxv} + \min(0, sv \mod 2 \times (\text{maxv} - 2)) \\
\end{align*}
\]

required\((\text{VARIABLES}, \text{var})\)

where

\[
\begin{align*}
sv & = |\text{VARIABLES}| \\
np & = \lfloor sv / 2 \rfloor \\
\text{minv} & = \text{minval}(\text{VARIABLES}.\text{var}) \\
\text{maxv} & = \text{maxval}(\text{VARIABLES}.\text{var}) \\
rv & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

\(\text{VALUE}\) is the sum of the surface of occurrences of the **STRICTLY_DECREASING_SEQUENCE** pattern in the time-series given by the \text{VARIABLES} collection. If the pattern does not occur, **VALUE** takes the default value 0.

An occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** is the maximal subsequence which matches the regular expression \(>^+\).

Assume that the occurrence of the pattern **STRICTLY_DECREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature \text{SURF} computes the sum of the values from index \(i\) to index \(j + 1\).

### Example

\((31, (4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3))\)

Figure 3.1280 provides an example where the **SUM_SURF_STRICTLY_DECREASING_SEQUENCE** \((31, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3])\) constraint holds.
Figure 3.1280: Illustrating the `SUM_SURF_STRICTLY_DECREASING_SEQUENCE` constraint of the **Example** slot

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES.var}) > 1
\]

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 3.1281 and 3.1282 respectively depict the automaton associated with the constraint `SUM_SURF_STRICTLY_DECREASING_SEQUENCE` and its simplified form.

Figure 3.1281: Automaton for the `SUM_SURF_STRICTLY_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `STRICTLY_DECREASING_SEQUENCE` pattern where `default` is 0

Figure 3.1282: Automaton for the `SUM_SURF_STRICTLY_DECREASING_SEQUENCE` constraint obtained by applying decoration Table 2.38 to the seed transducer of the `STRICTLY_DECREASING_SEQUENCE` pattern where `default` is 0
Table 3.182: Glue matrix for the SUM_SURF.Strictly_Decreasing_Sequence constraint defined as the composition of the Strictly_Decreasing_Sequence pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.606  SUM_SURF_STRICTLY_INCREASING_SEQUENCE

DESCRIPTION

Origin
Based on the STRICTLY_INCREASING_SEQUENCE pattern.

Constraint
SUM_SURF_STRICTLY_INCREASING_SEQUENCE(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv ≤ 1 ∨ rv ≤ 1 ⇒ VALUE = 0
rv = 2 ⇒ VALUE = 0 ∨ VALUE ≥ min(2 * minv + 1, (2 * minv + 1) * np)
rv ≥ 3 ⇒
   VALUE = 0,
   VALUE ≥ min(2 * minv + 1, (2 * minv + 1) * np + min(0, sv mod 2 * (minv + 2)))
rv = 2 ⇒ VALUE = 0 ∨ VALUE ≤ max(2 * maxv − 1, (2 * maxv − 1) * np)
rv ≥ 3 ⇒
   VALUE = 0,
   VALUE ≤ max(2 * maxv − 1, (2 * maxv − 1) * np + max(0, sv mod 2 * (maxv − 2)))
required(VARIABLES, var)

where
sv = |VARIABLES|
np = [sv/2]
minv = minval(VARIABLES.var)
maxv = maxval(VARIABLES.var)
rv = range(VARIABLES.var)

Purpose
VALUE is the sum of the surface of occurrences of the STRICTLY_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression <+.

Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at position i and ends at position j. The feature SURF computes the sum of the values from index i to index j + 1.

Example
(30, [4, 3, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3])

Figure 3.1283 provides an example where the SUM_SURF_STRICTLY_INCREASING_SEQUENCE (30, [4, 3, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3]) constraint holds.
Figure 3.1283: Illustrating the SUM_SURF_STRICTLY_INCREASING_SEQUENCE constraint of the Example slot

Typical

\[
\text{Future length} > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Automaton

Figures 3.1284 and 3.1285 respectively depict the automaton associated with the constraint `SUM_SURF STRICTLY_INCREASING_SEQUENCE` and its simplified form.

![Automaton Diagram](image)

Figure 3.1284: Automaton for the `SUM_SURF STRICTLY_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `STRICTLY_INCREASING_SEQUENCE` pattern where `default` is 0

![Automaton Diagram](image)

Figure 3.1285: Automaton for the `SUM_SURF STRICTLY_INCREASING_SEQUENCE` constraint obtained by applying decoration Table 2.38 to the seed transducer of the `STRICTLY_INCREASING_SEQUENCE` pattern where `default` is 0
Table 3.183: Glue matrix for the `SUM_SURF_STRICTLY_INCREASING_SEQUENCE` constraint defined as the composition of the `STRICTLY_INCREASING_SEQUENCE` pattern, the feature `SURF`, and the aggregator `sum`, cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>(\vec{C} + \vec{C})</td>
<td>(\vec{C} + \vec{C})</td>
</tr>
<tr>
<td>r</td>
<td>(\vec{C} + \vec{C})</td>
<td>(\vec{C} + \vec{C} + \vec{D} + \vec{D} + \text{VAR}_i)</td>
</tr>
</tbody>
</table>
3.607 SUM_SURF_SUMMIT

**Origin**
Based on the SUMMIT pattern.

**Constraint**
SUM_SURF_SUMMIT(VALUE, VARIABLES)

**Arguments**
VALUE : dvar
VARIABLES : collection(var–dvar)

**Restrictions**
\[ sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = 0 \]
\[ rv = 2 \Rightarrow VALUE = 0 \lor VALUE \geq \min(\minv + 1, (\minv + 1) \times np) \]
\[ rv \geq 3 \Rightarrow \]
\[ (VALUE = 0, VALUE \geq \min((\minv + 1) \times np, \min(\minv + 1, (sv - 2) \times (\minv + 1) + 1))) \]
\[ rv = 2 \Rightarrow VALUE = 0 \lor VALUE \leq \max(\maxv, \maxv \times np) \]
\[ rv \geq 3 \Rightarrow \]
\[ (VALUE = 0, VALUE \leq \max(\maxv \times np, \max(\sv - 2) \times (\maxv - 1) + 1)) \]

**Purpose**
VALUE is the sum of the surface of occurrences of the SUMMIT pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern SUMMIT is the maximal subsequence which matches the regular expression \( (< | (= | (<)^* | (>) | (> | (= | (>)^* |>) \).

Assume that the occurrence of the pattern SUMMIT starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i + 1 \) to index \( j \).

**Example**
\((23, (7, 1, 5, 4, 4, 3, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1))\)

Figure 3.1286 provides an example where the SUM_SURF_SUMMIT \((23, [7, 1, 5, 4, 4, 3, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1])\) constraint holds.

**Typical**
\(|\text{VARIABLES}| > 2\)
\(\text{range(VARIABLES.var)} > 1\)
Figure 3.1286: Illustrating the SUM_SURF_SUMMIT constraint of the Example slot

**Symmetry**

Items of VARIABLES can be reversed.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figures 3.1287 and 3.1288 respectively depict the automaton associated with the constraint SUM_SURF_SUMMIT and its simplified form.

![Automaton Diagram](image)

**Figure 3.1287**: Automaton for the SUM_SURF_SUMMIT constraint obtained by applying decoration Table 2.35 to the seed transducer of the SUMMIT pattern where default is 0 (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$)
Figure 3.1288: Automaton for the \texttt{SUM\_SURF\_SUMMIT} constraint obtained by applying decoration Table 2.25 to the seed transducer of the \texttt{SUMMIT} pattern where \texttt{default} is 0 (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$)
Table 3.184: Glue matrix for the SUM_SURF_SUMMIT constraint defined as the composition of the SUMMIT pattern, the feature SURF, and the aggregator sum: cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
<th>u</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\bar{C} + \bar{C}$</td>
<td>$\bar{C} + \bar{C}$</td>
<td>$\bar{C} + \bar{C}$</td>
<td>$\bar{C} + \bar{C}$</td>
</tr>
<tr>
<td>r</td>
<td>$\bar{C} + \bar{C}$</td>
<td>$\bar{D} + \bar{D} + \text{VAR}_i$</td>
<td>$\bar{C} + \bar{D} + \bar{D} + \text{VAR}_i$</td>
<td>$\bar{C} + \bar{C}$</td>
</tr>
<tr>
<td>t</td>
<td>$\bar{C} + \bar{C}$</td>
<td>$\bar{C} + \bar{D} + \bar{D} + \text{VAR}_i$</td>
<td>$\bar{C} + \bar{C}$</td>
<td>$\bar{C} + \bar{D} + \bar{D} + \text{VAR}_i$</td>
</tr>
<tr>
<td>u</td>
<td>$\bar{C} + \bar{C}$</td>
<td>$\bar{C} + \bar{C}$</td>
<td>$\bar{C} + \bar{D} + \bar{D} + \text{VAR}_i$</td>
<td>$\bar{C} + \bar{C}$</td>
</tr>
</tbody>
</table>
3.608 **SUM_SURF_VALLEY**

**DESCRIPTION**

**Origin**
Based on the VALLEY pattern.

**Constraint**
SUM_SURF_VALLEY(VALUE, VARIABLES)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var−dvar)

**Restrictions**

\[
\begin{align*}
sv & \leq 2 \land rv \leq 1 \Rightarrow VALUE = 0 \\
VALUE & = 0 \lor VALUE \geq \min\{\minv, \minv \cdot (sv - 2)\} \\
VALUE & = 0 \lor VALUE \leq \max\{\maxv - 1, (\maxv - 1) \cdot (sv - 2)\} \\
\text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where
- \( \minv = \minval(\text{VARIABLES}.\text{var}) \)
- \( \maxv = \maxval(\text{VARIABLES}.\text{var}) \)
- \( sv = |\text{VARIABLES}| \)
- \( rv = \text{range}(\text{VARIABLES}.\text{var}) \)

**Purpose**
An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression \( > (= | >) (< | =)* < \). Assume that the occurrence of the pattern VALLEY starts at position \( i \) and ends at position \( j \). The feature SURF computes the sum of the values from index \( i + 1 \) to index \( j \).

**Example**
(35, (1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 5, 7))

Figure 3.1289 provides an example where the SUM_SURF_VALLEY (35, [1, 3, 7, 4, 3, 6, 6, 5, 3, 3, 2, 6, 5, 5, 5, 7]) constraint holds.

**Typical**
- \(|\text{VARIABLES}| > 2 \)
- \( \text{range}(\text{VARIABLES}.\text{var}) > 1 \)

**Symmetry**
Items of VARIABLES can be reversed.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.1289: Illustrating the SUM_SURF_VALLEY constraint of the Example slot
Automaton

Figures 3.1290 and 3.1291 respectively depict the automaton associated with the constraint SUM_SURF_VALLEY and its simplified form.

Figure 3.1290: Automaton for the SUM_SURF_VALLEY constraint obtained by applying decoration Table 2.35 to the seed transducer of the VALLEY pattern where default is 0

Figure 3.1291: Automaton for the SUM_SURF_VALLEY constraint obtained by applying decoration Table 2.25 to the seed transducer of the VALLEY pattern where default is 0
Table 3.185: Glue matrix for the `SUM_SURF_VALLEY` constraint defined as the composition of the `VALLEY` pattern, the feature `SURF`, and the aggregator `sum`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.609  SUM_SURF_ZIGZAG

Origin
Based on the ZIGZAG pattern.

Constraint
SUM_SURF_ZIGZAG(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var–dvar)

Restrictions
\[ sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = 0 \]
\[ \lor \]
\[ VALUE \geq \min \left( 2 \cdot \minv + 1, 8 \right), \]
\[ VALUE = 0, \]
\[ \lor \]
\[ VALUE \leq \max \left( 2 \cdot \maxv - 1, 8 \right), \]
\[ \left\{ \left\lfloor (sv - 1)/2 \right\rfloor \cdot \maxv + \left\lfloor (sv - 2)/2 \right\rfloor \cdot (\maxv - 1) \right\} \]
required(VARIABLES, var)

where
\[ \minv = \minval(VARIABLES, var) \]
\[ \maxv = \maxval(VARIABLES, var) \]
\[ sv = |VARIABLES| \]
\[ rv = \text{range}(VARIABLES, var) \]

VALUE is the sum of the surface of occurrences of the ZIGZAG pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

Purpose
An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression \((<>)^+ (< | <> | (>><>)^+ (> | <>))\). Assume that the occurrence of the pattern ZIGZAG starts at position i and ends at position j. The feature SURF computes the sum of the values from index \(i + 1\) to index \(j\).

Example
\((33, (4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1))\)

Figure 3.1292 provides an example where the SUM_SURF_ZIGZAG (33, [4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1]) constraint holds.

Typical
\(|VARIABLES| > 3\]
\(\text{range}(VARIABLES, var) > 1\)

Symmetry
Items of VARIABLES can be reversed.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 3.1292: Illustrating the SUM_SURF_ZIGZAG constraint of the Example slot
Automaton

Figures 3.1293 and 3.1294 respectively depict the automaton associated with the constraint SUM_SURF_ZIGZAG and its simplified form.
Figure 3.1293: Automaton for the SUM_SURF_ZIGZAG constraint obtained by applying decoration Table 2.35 to the seed transducer of the ZIGZAG pattern where default is 0; (1) missing transitions from a, b, c, d, e, f to s are labelled by =; (2) on transitions from b, c, e, f to s the accumulator D is reset to its initial value; (3) on transitions from c, f to s the accumulator R is updated wrt C and the accumulator C is reset to its initial value.
Figure 3.1294: Automaton for the SUM_SURF_ZIGZAG constraint obtained by applying decoration Table 2.28 to the seed transducer of the ZIGZAG pattern where default is 0; (1) missing transitions from a, b, c, d, e, f to s are labelled by =; (2) on transitions from b, c, e, f to s the accumulator D is reset to its initial value
Table 3.186: Glue matrix for the SUMSURFZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature SURF, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
## 3.610 SUM\_WIDTH\_DECREASING\_SEQUENCE

**Origin** Based on the **DECREASING\_SEQUENCE** pattern.

**Constraint**

\[
\text{SUM\_WIDTH\_DECREASING\_SEQUENCE}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**

\[
\begin{align*}
sv & \leq 1 \lor rv & \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & = 0 \lor \text{VALUE} \geq 2 \\
rv & = 2 \Rightarrow \text{VALUE} = 2 \times np \\
rv & \geq 3 \Rightarrow \text{VALUE} & \leq sv \\
\text{required}(\text{VARIABLES}, \text{var})
\end{align*}
\]

where

\[
\begin{align*}
rv &= \text{range}(\text{VARIABLES}, \text{var}) \\
sv &= |\text{VARIABLES}| \\
np &= \lfloor sv / 2 \rfloor
\end{align*}
\]

**Purpose**

\(\text{VALUE}\) is the sum of the width of occurrences of the **DECREASING\_SEQUENCE** pattern in the time-series given by the **VARIABLES** collection. If the pattern does not occur, **VALUE** takes the default value 0.

An occurrence of the pattern **DECREASING\_SEQUENCE** is the maximal subsequence which matches the regular expression \(>(>|=)^*>|>\). Assume that the occurrence of the pattern **DECREASING\_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **WIDTH** computes the value \(j - i + 2\).

**Example**

\((9, (3, 4, 2, 3, 6, 4, 3, 1, 1, 4, 6, 4, 4))\)

Figure 3.1295 provides an example where the **SUM\_WIDTH\_DECREASING\_SEQUENCE** 
\((9, [3, 4, 2, 3, 6, 4, 3, 1, 1, 4, 6, 4, 4])\) constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}, \text{var}) > 1
\]

**Symmetry**

One and the same constant can be added to the var attribute of all items of **VARIABLES**.

**Arg. properties** Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 3.1295: Illustrating the \texttt{SUM\_WIDTH\_DECREASING\_SEQUENCE} constraint of the \textbf{Example} slot
Automaton

Figures 3.1296 and 3.1297 respectively depict the automaton associated with the constraint \text{SUM_WIDTH_DECREASING_SEQUENCE} and its simplified form.

\[
\begin{align*}
&\begin{cases}
C \leftarrow \text{default} \\
D \leftarrow 0 \\
R \leftarrow \text{default}
\end{cases} \\
\{ \leq s \} \\
&\begin{cases}
C \leftarrow \text{default} \\
D \leftarrow 0 \\
R \leftarrow R + C
\end{cases} \quad \begin{cases}
C \leftarrow D + 2 \\
D \leftarrow 0
\end{cases} \\
\{ \geq t \} \\
&\begin{cases}
D \leftarrow D + 1
\end{cases} \quad \begin{cases}
C \leftarrow C + D + 1
\end{cases}
\end{align*}
\]

Figure 3.1296: Automaton for the \text{SUM_WIDTH_DECREASING_SEQUENCE} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \text{DECREASING_SEQUENCE} pattern where \text{default} is 0.

\[
\begin{array}{|c|c|}
\hline
s & t \\
\hline
\text{c} & \text{c} + \text{c} \\
\text{c} & \text{c} + \text{c} + \text{d} + \text{d} + 1 \\
\text{c} + \text{c} & \text{c} + \text{c} + \text{d} + \text{d} + 1 \\
\hline
\end{array}
\]

Table 3.187: Glue matrix for the \text{SUM_WIDTH_DECREASING_SEQUENCE} constraint defined as the composition of the \text{DECREASING_SEQUENCE} pattern, the feature \text{WIDTH}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 3.1297: Automaton for the \textsc{sum\_width\_decreasing\_sequence} constraint obtained by applying decoration Table 2.29 to the seed transducer of the \textsc{decreasing\_sequence} pattern where \texttt{default} is 0; \( R_i - R_{i-1} \geq 0 \) and \(-R_i + R_{i-1} + D_{i-1} + 1 \geq 0 \) are linear invariants.
### 3.611 SUM_WIDTH_DECREASING_TERRACE

#### Origin
Based on the DECREASING_TERRACE pattern.

#### Constraint

$$\text{SUM_WIDTH_DECREASING_TERRACE}((\text{VALUE}, \text{VARIABLES}))$$

#### Arguments

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

#### Restrictions

\[
s v \leq 3 \lor r v \leq 2 \Rightarrow \text{VALUE} = 0  \\
\text{VALUE} = 0 \lor \text{VALUE} \geq 2  \\
\text{VALUE} \leq \max(0, s v - 2 v) \\
\text{required}(\text{VARIABLES}, \text{var}) \\
\text{where}  \\
s v = |\text{VARIABLES}|  \\
r v = \text{range}(\text{VARIABLES}.\text{var})
\]

#### Purpose

An occurrence of the pattern DECREASING_TERRACE is the maximal subsequence which matches the regular expression \( \geq^+ \geq \). Assume that the occurrence of the pattern DECREASING_TERRACE starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

#### Example

\( (4, (6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3)) \)

Figure 3.1298 provides an example where the SUM_WIDTH_DECREASING_TERRACE \( (4, [6, 4, 4, 4, 5, 2, 2, 1, 3, 3, 5, 4, 4, 3, 3]) \) constraint holds.

#### Typical

- \( |\text{VARIABLES}| > 3 \)
- \( \text{range}(\text{VARIABLES}.\text{var}) > 2 \)

#### Symmetry
One and the same constant can be added to the var attribute of all items of VARIABLES.

#### Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 3.1298: Illustrating the SUM_WIDTH_DECREASING_TERRACE constraint of the Example slot
Figures 3.1299 and 3.1300 respectively depict the automaton associated with the constraint SUM_WIDTH_DECREASING_TERRACE and its simplified form.

Figure 3.1299: Automaton for the SUM_WIDTH_DECREASING_TERRACE constraint obtained by applying decoration Table 2.35 to the seed transducer of the DECREASING_TERRACE pattern where default is 0

Table 3.188: Glue matrix for the SUM_WIDTH_DECREASING_TERRACE constraint defined as the composition of the DECREASING_TERRACE pattern, the feature WIDTH, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 3.1300: Automaton for the SUM_WIDTH_DECREASING_TERRACE constraint obtained by applying decoration Table 2.28 to the seed transducer of the DECREASING_TERRACE pattern where default is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + D_{i-1} + 1 \geq 0$ are linear invariants.
3.612  SUM_WIDTH_GORGE

DESCRIPTION

Origin
Based on the GORGE pattern.

Constraint
SUM_WIDTH_GORGE(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = 0
VALUE = 0 ∨ VALUE ≥ 1
rv = 2 ⇒ VALUE ≤ np
rv ≥ 3 ⇒ VALUE ≤ max(0, sv − 2)
required(VARIABLES, var)
where
rv = range(VARIABLES.var)
sv = |VARIABLES|
np = max(0, [(sv − 1)/2])

Purpose
An occurrence of the pattern GORGE is the maximal subsequence which matches the regular expression (> | > (= | >)* >)(< | < (= | <)* <).
Assume that the occurrence of the pattern GORGE starts at position i and ends at position j. The feature WIDTH computes the value j − i.

Example
(6, (1, 7, 3, 4, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7))

Figure 3.1301 provides an example where the SUM_WIDTH_GORGE
(6, [1, 7, 3, 4, 5, 4, 2, 2, 6, 5, 4, 6, 5, 7]) constraint holds.

Typical
|VARIABLES| > 2
range(VARIABLES.var) > 1

Symmetries
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 3.1301: Illustrating the \texttt{SUM\_WIDTH\_GORGE} constraint of the \textbf{Example} slot
Automaton

Figures 3.1302 and 3.1303 respectively depict the automaton associated with the constraint SUM_WIDTHGORGE and its simplified form.

Figure 3.1302: Automaton for the SUM_WIDTHGORGE constraint obtained by applying decoration Table 2.35 to the seed transducer of the GORGE pattern where default is 0 (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$)
Figure 3.1303: Automaton for the SUM_WIDTH_GORGE constraint obtained by applying decoration Table 2.25 to the seed transducer of the GORGE pattern where default is 0 (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$); $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + D_{i-1} + 1 \geq 0$ are linear invariants.
Table 3.189: Glue matrix for the `SUM_WIDTH_GORGE` constraint defined as the composition of the `GORGE` pattern, the feature `WIDTH`, and the aggregator `sum`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### 3.613 SUM_WIDTH_INCREASING_SEQUENCE

<table>
<thead>
<tr>
<th>➤</th>
<th>◀</th>
<th>DESCRIPTION</th>
<th>AUTOMATON</th>
</tr>
</thead>
</table>

**Origin**

Based on the **INCREASING_SEQUENCE** pattern.

**Constraint**

SUM_WIDTH_INCREASING_SEQUENCE(VALUE, VARIABLES)

**Arguments**

- **VALUE**: dvar
- **VARIABLES**: collection(var–dvar)

**Restrictions**

\[ sv \leq 1 \lor rv \leq 1 \Rightarrow VALUE = 0 \]
\[ VALUE = 0 \lor VALUE \geq 2 \]
\[ rv = 2 \Rightarrow VALUE \leq 2 \times np \]
\[ rv \geq 3 \Rightarrow VALUE \leq sv^2 \]

**Purpose**

An occurrence of the pattern **INCREASING_SEQUENCE** is the **maximal** subsequence which matches the regular expression \(< (< | =)^* | < \). Assume that the occurrence of the pattern **INCREASING_SEQUENCE** starts at position \(i\) and ends at position \(j\). The feature **WIDTH** computes the value \(j - i + 2\).

**Example**

\[(9, 4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3)\]

Figure 3.1304 provides an example where the **SUM_WIDTH_INCREASING_SEQUENCE** (9, [4, 3, 5, 5, 2, 1, 1, 3, 3, 4, 6, 6, 3, 1, 3, 3]) constraint holds.

**Typical**

\[ |\text{VARIABLES}| > 1 \]
\[ \text{range}(\text{VARIABLES}.\text{var}) > 1 \]

**Symmetry**

One and the same constant can be **added** to the \(\text{var}\) attribute of all items of \(\text{VARIABLES}\).

**Arg. properties**

Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 3.104: Illustrating the SUM_WIDTH_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 3.1305 and 3.1306 respectively depict the automaton associated with the constraint SUM_WIDTH_INCREASING_SEQUENCE and its simplified form.

\[
\begin{align*}
C &\leftarrow \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow \text{default}
\end{align*}
\]

\[
\begin{align*}
C &> \text{default} \\
D &\leftarrow 0 \\
R &\leftarrow R + C
\end{align*}
\]

\[
\begin{align*}
D &\leftarrow D + 1
\end{align*}
\]

\[
\begin{align*}
C &\leftarrow D + 2 \\
D &\leftarrow 0
\end{align*}
\]

\[
\begin{align*}
C &\leftarrow C + D + 1 \\
D &\leftarrow 0
\end{align*}
\]

Figure 3.1305: Automaton for the SUM_WIDTH_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the INCREASING_SEQUENCE pattern where default is 0

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{C}$</td>
</tr>
<tr>
<td>t</td>
<td>$\overline{C} + \overline{C}$</td>
<td>$\overline{C} + \overline{C} + \overline{D} + \overline{D} + 1$</td>
</tr>
</tbody>
</table>

Table 3.190: Glue matrix for the SUM_WIDTH_INCREASING_SEQUENCE constraint defined as the composition of the INCREASING_SEQUENCE pattern, the feature WIDTH, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 3.1306: Automaton for the SUM_WIDTH_INCREASING_SEQUENCE constraint obtained by applying decoration Table 2.29 to the seed transducer of the INCREASING_SEQUENCE pattern where default is 0; \( R_i - R_{i-1} \geq 0 \) and \(-R_i + R_{i-1} + D_{i-1} + 1 \geq 0\) are linear invariants.
### SUM_WIDTH_INCREASING_TERRACE

#### Description

- **Origin**: Based on the `INCREASING_TERRACE` pattern.

- **Constraint**: `SUM_WIDTH_INCREASING_TERRACE(VALUE, VARIABLES)`

- **Arguments**:
  - `VALUE : dvar`
  - `VARIABLES : collection(var−dvar)`

- **Restrictions**:
  - `sv ≤ 3 ∨ rv ≤ 2` $\Rightarrow$ `VALUE = 0`
  - `VALUE = 0` $\lor$ `VALUE ≥ 2`
  - `VALUE ≤ max(0, sv − 2k)`

#### Purpose

- **VALUE** is the sum of the width of occurrences of the `INCREASING_TERRACE` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, **VALUE** takes the default value 0.

- **An occurrence of the pattern `INCREASING_TERRACE` is the maximal subsequence which matches the regular expression `≤+≤`**.

- **Assume that the occurrence of the pattern `INCREASING_TERRACE` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value `j − i`**.

#### Example

```
(5, (1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 4, 4))
```

**Figure 3.1307** provides an example where the `SUM_WIDTH_INCREASING_TERRACE (5, [1, 3, 3, 3, 2, 5, 5, 6, 4, 4, 2, 3, 3, 4, 4])` constraint holds.

#### Typical

- `|VARIABLES| > 3`
- `range(VARIABLES.var) > 2`

#### Symmetry

- One and the same constant can be added to the `var` attribute of all items of `VARIABLES`.

#### Arg. properties

- Functional dependency: **VALUE** determined by `VARIABLES`. 
Figure 3.1307: Illustrating the `SUM_WIDTH_INCREASING_TERRACE` constraint of the Example slot
Automaton

Figures 3.1308 and 3.1309 respectively depict the automaton associated with the constraint `SUM_WIDTH_INCREASING_TERRACE` and its simplified form.

\[
\begin{cases}
C \leftarrow \text{default} \\
D \leftarrow 0 \\
R \leftarrow \text{default}
\end{cases}
\]

\[\geq \]

\[\geq\]

\[<\]

\[<\]

\[\{D \leftarrow D + 1\}\]

\[\{D \leftarrow D + 1\}\]

\[<\]

\[<\]

\[\{D \leftarrow 0\}

\[R \leftarrow R + D + 1\}

\]

Figure 3.1308: Automaton for the `SUM_WIDTH_INCREASING_TERRACE` constraint obtained by applying decoration Table 2.35 to the seed transducer of the `INCREASING_TERRACE` pattern where `default` is 0.

Table 3.191: Glue matrix for the `SUM_WIDTH_INCREASING_TERRACE` constraint defined as the composition of the `INCREASING_TERRACE` pattern, the feature `WIDTH`, and the aggregator `sum`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Figure 3.1309: Automaton for the SUM_WIDTH_INCREASING_TERRACE constraint obtained by applying decoration Table 2.28 to the seed transducer of the INCREASING_TERRACE pattern where default is 0; \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-1} + D_{i-1} + 1 \geq 0 \) are linear invariants.
3.615 SUM_WIDTH_INFLEXION

DESCRIPTION AUTOMATON

Origin
Based on the INFLEXION pattern.

Constraint
SUM_WIDTH_INFLEXION(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = 0
VALUE = 0 ∨ VALUE ≥ 1
VALUE ≤ max(0, sv − 2v)
required(VARIABLES, var)

where
sv = |VARIABLES|
rv = range(VARIABLES.var)

Purpose
An occurrence of the pattern INFLEXION is the maximal subsequence which matches the regular expression < (| | =)* > | | (>| |=)* <. Assume that the occurrence of the pattern INFLEXION starts at position i and ends at position j. The feature WIDTH computes the value j − i.

Example
(13, (1, 2, 6, 6, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4))

Figure 3.1310 provides an example where the SUM_WIDTH_INFLEXION (13, [1, 2, 6, 6, 4, 3, 5, 2, 5, 1, 5, 3, 3, 4, 4]) constraint holds.

Typical
|VARIABLES| > 2
range(VARIABLES.var) > 1

Symmetries
• Items of VARIABLES can be reversed.
• One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 3.1310: Illustrating the SUM_WIDTH_INFLEXION constraint of the \textbf{Example} slot
Figures 3.1311 and 3.1312 respectively depict the automaton associated with the constraint SUM_WIDTH_INFLEXION and its simplified form.

Figure 3.1311: Automaton for the SUM_WIDTH_INFLEXION constraint obtained by applying decoration Table 2.35 to the seed transducer of the INFLEXION pattern where \texttt{default} is 0 (transition \(r \rightarrow t\) has the same accumulators updates as transition \(t \rightarrow r\)).

\[
\begin{align*}
&\{ C \leftarrow \texttt{default} \} \quad \{ D \leftarrow 0 \} \quad \{ R \leftarrow \texttt{default} \} \\
&\{ D \leftarrow D + 1 \} \\
&\{ D \leftarrow 0 \} \\
&\{ R \leftarrow R + D + 1 \}
\end{align*}
\]

Figure 3.1312: Automaton for the SUM_WIDTH_INFLEXION constraint obtained by applying decoration Table 2.25 to the seed transducer of the INFLEXION pattern where \texttt{default} is 0 (transition \(r \rightarrow t\) has the same accumulators updates as transition \(t \rightarrow r\)); \(R_i - R_{i-1} \geq 0\) and \(-R_i + R_{i-1} + D_{i-1} + 1 \geq 0\) are linear invariants.
3.6.16 SUM_WIDTH_PEAK

DESCRIPTION AUTOMATON

Origin
Based on the PEAK pattern.

Constraint
SUM_WIDTH_PEAK(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = 0
VALUE = 0 ∨ VALUE ≥ 1
VALUE ≤ max(0, sv - 2\(k\))
required(VARIABLES, var)
where
sv = |VARIABLES|
rv = range(VARIABLES.var)

Purpose
An occurrence of the pattern PEAK is the maximal subsequence which matches the regular expression
\(< (= | <)^* (> | =)^* >\).
Assume that the occurrence of the pattern PEAK starts at position \(i\) and ends at position \(j\).
The feature WIDTH computes the value \(j - i\).

Example
(8, ⟨7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 3, 1⟩)

Figure 3.1313 provides an example where the SUM_WIDTH_PEAK
(8, [7, 5, 5, 1, 4, 5, 2, 2, 3, 5, 6, 2, 3, 3, 1]) constraint holds.

Typical
|VARIABLES| > 2
range(VARIABLES.var) > 1

Symmetries
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 3.1313: Illustrating the SUM_WIDTH_PEAK constraint of the Example slot
Automaton

Figures 3.1314 and 3.1315 respectively depict the automaton associated with the constraint SUM_WIDTH_PEAK and its simplified form.

Figure 3.1314: Automaton for the SUM_WIDTH_PEAK constraint obtained by applying decoration Table 2.35 to the seed transducer of the PEAK pattern where default is 0.

Figure 3.1315: Automaton for the SUM_WIDTH_PEAK constraint obtained by applying decoration Table 2.25 to the seed transducer of the PEAK pattern where default is 0; \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-1} + D_{i-1} + 1 \geq 0 \) are linear invariants.
Table 3.192: Glue matrix for the SUM_WIDTH_PEAK constraint defined as the composition of the PEAK pattern, the feature WIDTH, and the aggregator sum: cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.617 SUM_WIDTHPLAIN

**DESCRIPTION**

**Origin**

Based on the PLAIN pattern.

**Constraint**

SUM_WIDTHPLAIN(VALUE, VARIABLES)

**Arguments**

VALUE : dvar
VARIABLES : collection(var–dvar)

**Restrictions**

\[ sv \leq 2 \lor rv \leq 1 \Rightarrow VALUE = 0 \]
\[ VALUE = 0 \lor VALUE \geq 1 \]
\[ VALUE \leq \max(0, sv - 2x) \]
\[ \\
\]
where
\[ sv = |VARIABLES| \]
\[ rv = \text{range}(VARIABLES\text{var}) \]

VALUE is the sum of the width of occurrences of the PLAIN pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

**Purpose**

An occurrence of the pattern PLAIN is the maximal subsequence which matches the regular expression \( > = " < \).

Assume that the occurrence of the pattern PLAIN starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i \).

**Example**

\( (4, (2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3)) \)

Figure 3.1316 provides an example where the SUM_WIDTHPLAIN (4, [2, 3, 6, 5, 7, 6, 6, 4, 5, 5, 4, 3, 3, 6, 6, 3]) constraint holds.

**Typical**

\[ |VARIABLES| > 2 \]
\[ \text{range}(VARIABLES\text{var}) > 1 \]

**Symmetries**

- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.1316: Illustrating the SUM_WIDTHPLAIN constraint of the Example slot
Automaton

Figures 3.1317 and 3.1318 respectively depict the automaton associated with the constraint SUM_WIDTH_PLAIN and its simplified form.

\[
\begin{align*}
&= t \\
&\rightarrow \text{false} \\
&\rightarrow \text{true} \\
&\rightarrow \text{true} \\
&\rightarrow \text{false} \\
&\rightarrow \text{false} \\
&\rightarrow \text{true} \\
&\rightarrow \text{true} \\
&\rightarrow \text{false} \\
&\rightarrow \text{false} \\
&\rightarrow \text{true} \\
&\end{align*}
\]

Figure 3.1317: Automaton for the SUM_WIDTH_PLAIN constraint obtained by applying decoration Table 2.35 to the seed transducer of the PLAIN pattern where default is 0

\[
\begin{align*}
&= t \\
&\rightarrow \text{false} \\
&\rightarrow \text{true} \\
&\rightarrow \text{true} \\
&\rightarrow \text{false} \\
&\rightarrow \text{false} \\
&\rightarrow \text{true} \\
&\rightarrow \text{true} \\
&\rightarrow \text{false} \\
&\rightarrow \text{false} \\
&\rightarrow \text{true} \\
&\end{align*}
\]

Figure 3.1318: Automaton for the SUM_WIDTH_PLAIN constraint obtained by applying decoration Table 2.28 to the seed transducer of the PLAIN pattern where default is 0; \(R_i - R_{i-1} \geq 0\) and \(-R_i + R_{i-1} + D_{i-1} + 1 \geq 0\) are linear invariants.
Table 3.193: Glue matrix for the `SUM_WIDTH_PLAIN` constraint defined as the composition of the `PLAIN` pattern, the feature `WIDTH`, and the aggregator `sum`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\bar{C} + \bar{C}$</td>
<td>$\bar{C} + \bar{C}$</td>
<td>$\bar{C} + \bar{C}$</td>
</tr>
<tr>
<td>r</td>
<td>$\bar{C} + \bar{C}$</td>
<td>$\bar{D} + \bar{D} + 1$</td>
<td>$\bar{D} + \bar{D} + 1$</td>
</tr>
<tr>
<td>t</td>
<td>$\bar{C} + \bar{C}$</td>
<td>$\bar{D} + \bar{D} + 1$</td>
<td>$\bar{D} + \bar{D} + 1$</td>
</tr>
</tbody>
</table>
3.6.18  SUM_WIDTH_PLATEAU

**Origin**
Based on the PLATEAU pattern.

**Constraint**
```
SUM_WIDTH_PLATEAU(VALUE, VARIABLES)
```

**Arguments**
- `VALUE`: `dvar`
- `VARIABLES`: `collection(var−dvar)`

**Restrictions**
- `sv ≤ 2 \lor rv ≤ 1 \Rightarrow VALUE = 0`
- `VALUE = 0 \lor VALUE ≥ 1`
- `VALUE ≤ \max(0, sv − 2\nu)`
- `required(VARIABLES, var)`
  where
  - `sv = |VARIABLES|`
  - `rv = range(VARIABLES.var)`

**Purpose**
An occurrence of the pattern PLATEAU is the maximal subsequence which matches the regular expression `<="=">`. Assume that the occurrence of the pattern PLATEAU starts at position `i` and ends at position `j`. The feature WIDTH computes the value `j − i`.

**Example**
```
(7, ⟨1, 3, 3, 5, 5, 5, 2, 4, 4, 4, 3, 3, 1, 5, 5⟩)
```

Figure 3.1319 provides an example where the SUM_WIDTH_PLATEAU `(7, [1, 3, 3, 5, 5, 5, 2, 4, 4, 4, 3, 3, 1, 5, 5])` constraint holds.

**Typical**
- `|VARIABLES| > 2`
- `range(VARIABLES.var) > 1`

**Symmetries**
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
- Functional dependency: VALUE determined by VARIABLES.
Figure 3.1319: Illustrating the SUM_WIDTH_PLATEAU constraint of the Example slot
Figures 3.1320 and 3.1321 respectively depict the automaton associated with the constraint `SUM_WIDTH_PLATEAU` and its simplified form.

Figure 3.1320: Automaton for the `SUM_WIDTH_PLATEAU` constraint obtained by applying decoration Table 2.35 to the seed transducer of the PLATEAU pattern where `default` is 0.

Figure 3.1321: Automaton for the `SUM_WIDTH_PLATEAU` constraint obtained by applying decoration Table 2.28 to the seed transducer of the PLATEAU pattern where `default` is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + D_{i-1} + 1 \geq 0$ are linear invariants.
Table 3.194: Glue matrix for the SUM\_WIDTH\_PLATEAU constraint defined as the composition of the PLATEAU pattern, the feature WIDTH, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.619  SUM_WIDTH_PROPERPlain

Description

Origin
Based on the PROPERPlain pattern.

Constraint

\[
\text{SUM_WIDTH_PROPERPlain}(\text{VALUE, VARIABLES})
\]

Arguments

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} - \text{dvar})
\end{align*}
\]

Restrictions

\[
\begin{align*}
\text{sv} \leq 3 & \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} = 0 & \lor \text{VALUE} \geq 2 \\
\text{VALUE} & \leq \max(0, \text{sv} - 2) \\
\text{required}(\text{VARIABLES}, \text{var}) & \text{where} \\
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES}, \text{var})
\end{align*}
\]

Purpose

An occurrence of the pattern PROPERPlain is the maximal subsequence which matches the regular expression \(> = ^+ <\).

Assume that the occurrence of the pattern PROPERPlain starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

Example

\[
(7, (2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3, 5))
\]

Figure 3.1322 provides an example where the SUM_WIDTH_PROPERPlain \((7, [2, 7, 5, 5, 6, 3, 7, 4, 4, 5, 6, 5, 3, 3, 3, 5])\) constraint holds.

Typical

\[
\begin{align*}
|\text{VARIABLES}| & > 3 \\
\text{range}(\text{VARIABLES}, \text{var}) & > 1
\end{align*}
\]

Symmetries

- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 3.1322: Illustrating the SUM_WIDTH_PROPER_PLAIN constraint of the Example slot
Automaton

Figures 3.1323 and 3.1324 respectively depict the automaton associated with the constraint SUM_WIDTH_PROPER_PLAIN and its simplified form.

\[
\begin{align*}
&D \leftarrow \text{default} \\
&D_0 = 0 \\
&R \leftarrow \text{default} \\
\end{align*}
\]

\[
\begin{align*}
&C \leftarrow \text{default} \\
&D \leftarrow 0 \\
&R \leftarrow \text{default} \\
\end{align*}
\]

\[
\begin{align*}
&D \leftarrow \text{default} \\
&D_0 = 0 \\
\end{align*}
\]

\[
\begin{align*}
&D \leftarrow \text{default} \\
&D_0 = 0 \\
\end{align*}
\]

\[
\begin{align*}
&D \leftarrow \text{default} \\
&D_0 = 0 \\
\end{align*}
\]

Figure 3.1323: Automaton for the SUM_WIDTH_PROPER_PLAIN constraint obtained by applying decoration Table 2.35 to the seed transducer of the PROPER_PLAIN pattern where default is 0.

\[
\begin{align*}
&D \leftarrow \text{default} \\
&D_0 = 0 \\
\end{align*}
\]

\[
\begin{align*}
&D \leftarrow \text{default} \\
&D_0 = 0 \\
\end{align*}
\]

\[
\begin{align*}
&D \leftarrow \text{default} \\
&D_0 = 0 \\
\end{align*}
\]

\[
\begin{align*}
&D \leftarrow \text{default} \\
&D_0 = 0 \\
\end{align*}
\]

Figure 3.1324: Automaton for the SUM_WIDTH_PROPER_PLAIN constraint obtained by applying decoration Table 2.28 to the seed transducer of the PROPER_PLAIN pattern to the seed transducer of the PROPER_PLAIN pattern where default is 0; \( R_i - R_{i-1} \geq 0 \) and \(-R_i + R_{i-1} + D_{i-1} + 1 \geq 0\) are linear invariants.
Table 3.195: Glue matrix for the `SUM_WIDTH_PROPER_PLAIN` constraint defined as the composition of the `PROPER_PLAIN` pattern, the feature `WIDTH`, and the aggregator `sum`: cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
</tr>
<tr>
<td>r</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
</tr>
<tr>
<td>t</td>
<td>$\overrightarrow{C} + \overrightarrow{C}$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
<td>$\overrightarrow{D} + \overrightarrow{D} + 1$</td>
</tr>
</tbody>
</table>
3.620  SUM_WIDTH_PROPER_PLATEAU

**Origin**
Based on the PROPER_PLATEAU pattern.

**Constraint**
SUM_WIDTH_PROPER_PLATEAU(VALUE, VARIABLES)

**Arguments**
- VALUE : dvar
- VARIABLES : collection(var − dvar)

**Restrictions**
\[ sv \leq 3 \lor rv \leq 1 \Rightarrow VALUE = 0 \]
\[ VALUE = 0 \lor VALUE \geq 2 \]
\[ VALUE \leq \max(0, sv - 2) \]
\[ required(VARIABLES, var) \]
where
\[ sv = |VARIABLES| \]
\[ rv = \text{range}(VARIABLES.var) \]

**Purpose**
An occurrence of the pattern PROPER_PLATEAU is the maximal subsequence which matches the regular expression \(< = + >\).
Assume that the occurrence of the pattern PROPER_PLATEAU starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

**Example**
\((7, [7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3])\)

Figure 3.1325 provides an example where the SUM_WIDTH_PROPER_PLATEAU (7, [7, 1, 3, 3, 2, 5, 1, 4, 4, 3, 2, 3, 5, 5, 3]) constraint holds.

**Typical**
\[ |VARIABLES| > 3 \]
\[ \text{range}(VARIABLES.var) > 1 \]

**Symmetries**
- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**
Functional dependency: VALUE determined by VARIABLES.
Figure 3.1325: Illustrating the `SUM_WIDTH_PROPER_PLATEAU` constraint of the Example slot
Figures 3.1326 and 3.1327 respectively depict the automaton associated with the constraint \texttt{SUM\_WIDTH\_PROPER\_PLATEAU} and its simplified form.

Figure 3.1326: Automaton for the \texttt{SUM\_WIDTH\_PROPER\_PLATEAU} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{PROPER\_PLATEAU} pattern where default is 0.

Figure 3.1327: Automaton for the \texttt{SUM\_WIDTH\_PROPER\_PLATEAU} constraint obtained by applying decoration Table 2.28 to the seed transducer of the \texttt{PROPER\_PLATEAU} pattern where default is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + D_{i-1} + 1 \geq 0$ are linear invariants.
Table 3.196: Glue matrix for the \texttt{SUM\_WIDTH\_PROPER\_PLATEAU} constraint defined as the composition of the \texttt{PROPER\_PLATEAU} pattern, the feature \texttt{WIDTH}, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.621 SUM_WIDTH_STEADY_SEQUENCE

**Description**

Based on the STEADY_SEQUENCE pattern.

**Argument**

- **VALUE**: dvar
- **VARIABLES**: collection(var−dvar)

**Restrictions**

\[ sv \leq 1 \Rightarrow VALUE = 0 \]
\[ rv = 1 \Rightarrow VALUE \geq sv \]
\[ rv \geq 2 \Rightarrow VALUE = 0 \lor VALUE \geq 2 \]
\[ VALUE \leq sv \]
\[ required(VARIABLES, var) \]

where

\[ sv = |VARIABLES| \]
\[ rv = range(VARIABLES.var) \]

**Purpose**

VALUE is the sum of the width of occurrences of the STEADY_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern STEADY_SEQUENCE is the maximal subsequence which matches the regular expression \( =^+ \).

Assume that the occurrence of the pattern STEADY_SEQUENCE starts at position \( i \) and ends at position \( j \). The feature WIDTH computes the value \( j - i + 2 \).

**Example**

\( (11, (3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1)) \)

Figure 3.1328 provides an example where the SUM_WIDTH_STEADY_SEQUENCE (11, [3, 1, 1, 4, 5, 5, 6, 2, 2, 4, 4, 3, 2, 1, 1]) constraint holds.

**Typical**

\[ |VARIABLES| > 1 \]

**Symmetries**

- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.1328: Illustrating the `SUM_WIDTH_STEADY_SEQUENCE` constraint of the Example slot
Figures 3.1329 and 3.1330 respectively depict the automaton associated with the constraint SUM_WIDTH_STEADY_SEQUENCE and its simplified form.

Figure 3.1329: Automaton for the SUM_WIDTH_STEADY_SEQUENCE constraint obtained by applying decoration Table 2.35 to the seed transducer of the STEADY_SEQUENCE pattern where default is 0.

Figure 3.1330: Automaton for the SUM_WIDTH_STEADY_SEQUENCE constraint obtained by applying decoration Table 2.38 to the seed transducer of the STEADY_SEQUENCE pattern where default is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + 2 \geq 0$ are linear invariants.
Table 3.197: Glue matrix for the SUM_WIDTH_STEADY_SEQUENCE constraint defined as the composition of the STEADY_SEQUENCE pattern, the feature WIDTH, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
### 3.622 SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE

**Origin**
Based on the `STRICTLY_DECREASING_SEQUENCE` pattern.

**Constraint**

\[
\text{SUM}\_\text{WIDTH}\_\text{STRICTLY}\_\text{DECREASING}\_\text{SEQUENCE}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- `VALUE` : dvar
- `VARIABLES` : collection(var−dvar)

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 1 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & = 0 \lor \text{VALUE} \geq 2 \\
\text{rv} & = 2 \Rightarrow \text{VALUE} \leq \text{sv} - \text{sv} \mod 2 \\
\text{rv} & \geq 3 \Rightarrow \text{VALUE} \leq \text{sv} \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{where} & \\
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

- `VALUE` is the sum of the width of occurrences of the `STRICTLY_DECREASING_SEQUENCE` pattern in the time-series given by the `VARIABLES` collection. If the pattern does not occur, `VALUE` takes the default value 0.
- An occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` is the maximal subsequence which matches the regular expression `>\+`.
- Assume that the occurrence of the pattern `STRICTLY_DECREASING_SEQUENCE` starts at position `i` and ends at position `j`. The feature `WIDTH` computes the value \( j - i + 2 \).

**Example**

\[
(8, (4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 5, 2, 2, 4, 3))
\]

Figure 3.1331 provides an example where the `SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE (8, [4, 4, 6, 4, 1, 1, 3, 4, 4, 6, 6, 5, 2, 2, 4, 3])` constraint holds.

**Typical**

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

**Symmetry**

One and the same constant can be added to the `var` attribute of all items of `VARIABLES`.

**Arg. properties**

Functional dependency: `VALUE` determined by `VARIABLES`. 
Figure 3.1331: Illustrating the SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE constraint of the Example slot
Figures 3.1332 and 3.1333 respectively depict the automaton associated with the constraint \texttt{SUM\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE} and its simplified form.

Figure 3.1332: Automaton for the \texttt{SUM\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{STRICTLY\_DECREASING\_SEQUENCE} pattern where \texttt{default} is 0

Figure 3.1333: Automaton for the \texttt{SUM\_WIDTH\_STRICTLY\_DECREASING\_SEQUENCE} constraint obtained by applying decoration Table 2.38 to the seed transducer of the \texttt{STRICTLY\_DECREASING\_SEQUENCE} pattern where \texttt{default} is 0; \(-R_i + R_{i-1} + 2 \geq 0\) are linear invariants.
Table 3.198: Glue matrix for the `SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE` constraint defined as the composition of the `STRICTLY_DECREASING_SEQUENCE` pattern, the feature `WIDTH`, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>( s )</th>
<th>( r )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>( \overrightarrow{C} + \overrightarrow{C} )</td>
<td>( \overrightarrow{C} + \overrightarrow{C} )</td>
</tr>
<tr>
<td>( r )</td>
<td>( \overrightarrow{C} + \overrightarrow{C} )</td>
<td>( \overrightarrow{C} + \overrightarrow{C} + \overrightarrow{D} + \overrightarrow{D} + 1 )</td>
</tr>
</tbody>
</table>
3.623  SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE

Description

Based on the STRICTLY_INCREASING_SEQUENCE pattern.

Constraint

\[
\text{SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE}(\text{VALUE}, \text{VARIABLES})
\]

Arguments

\[
\begin{align*}
\text{VALUE} & : \text{dvar} \\
\text{VARIABLES} & : \text{collection}(\text{var} \text{− dvar})
\end{align*}
\]

Restrictions

\[
\begin{align*}
\text{sv} \leq 1 \land \text{rv} \leq 1 & \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} = 0 & \lor \text{VALUE} \geq 2 \\
\text{rv} = 2 & \Rightarrow \text{VALUE} \leq \text{sv} - \text{sv} \mod 2 \\
\text{rv} \geq 3 & \Rightarrow \text{VALUE} \leq \text{sv} \\
\text{required}(\text{VARIABLES}, \text{var}) & \\
\text{where} & \\
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

Purpose

VALUE is the sum of the width of occurrences of the STRICTLY_INCREASING_SEQUENCE pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern STRICTLY_INCREASING_SEQUENCE is the maximal subsequence which matches the regular expression \(<^+\).

Assume that the occurrence of the pattern STRICTLY_INCREASING_SEQUENCE starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i + 2\).

Example

\[
(10, (4, 3, 5, 5, 2, 1, 1, 2, 3, 4, 6, 6, 3, 1, 2, 3))
\]

Figure 3.1334 provides an example where the \text{SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE} constraint holds.

Typical

\[
|\text{VARIABLES}| > 1 \\
\text{range}(\text{VARIABLES}.\text{var}) > 1
\]

Symmetry

One and the same constant can be added to the \text{var} attribute of all items of VARIABLES.

Arg. properties

Functional dependency: VALUE determined by VARIABLES.
Figure 3.1334: Illustrating the SUM_WIDTH STRICTLY_INCREASING_SEQUENCE constraint of the Example slot
Automaton

Figures 3.1335 and 3.1336 respectively depict the automaton associated with the constraint \texttt{SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE} and its simplified form.

\begin{align*}
\{ & C \leftarrow \text{default} \cr & D \leftarrow 0 \cr & R \leftarrow \text{default} \}
\end{align*}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure3.1335}
\caption{Automaton for the \texttt{SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE} constraint obtained by applying decoration Table 2.35 to the seed transducer of the \texttt{STRICTLY_INCREASING_SEQUENCE} pattern where \texttt{default} is 0}
\end{figure}

\begin{align*}
\{ & C \leftarrow \text{default} \cr & D \leftarrow 0 \cr & R \leftarrow R + C \}
\end{align*}

\begin{align*}
\{ & C \leftarrow D + 2 \cr & D \leftarrow 0 \}
\end{align*}

\begin{align*}
\{ & C \leftarrow C + D + 1 \cr & D \leftarrow 0 \}
\end{align*}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure3.1336}
\caption{Automaton for the \texttt{SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE} constraint obtained by applying decoration Table 2.38 to the seed transducer of the \texttt{STRICTLY_INCREASING_SEQUENCE} pattern where \texttt{default} is 0; \( R_i - R_{i-1} \geq 0 \) and \(-R_i + R_{i-1} + 2 \geq 0\) are linear invariants.}
\end{figure}
Table 3.199: Glue matrix for the `SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE` constraint defined as the composition of the `STRICTLY_INCREASING_SEQUENCE` pattern, the feature `WIDTH`, and the aggregator `sum`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>s</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>C̄ + C̄</td>
<td>C̄ + C̄</td>
</tr>
<tr>
<td>r</td>
<td>C̄ + C̄</td>
<td>C̄ + C̄ + D̄ + D̄ + 1</td>
</tr>
</tbody>
</table>
3.624 **SUM_WIDTH_SUMMIT**

**DESCRIPTION**

**Origin**
Based on the **SUMMIT** pattern.

**Constraint**

\[
\text{SUM_WIDTH_SUMMIT}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

- \(\text{VALUE} : \text{dvar}\)
- \(\text{VARIABLES} : \text{collection}(\text{var} - \text{dvar})\)

**Restrictions**

\[
\begin{align*}
sv &\leq 2 \lor rv \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} = 0 &\lor \text{VALUE} \geq 1 \\
rv = 2 &\Rightarrow \text{VALUE} \leq np \\
rv \geq 3 &\Rightarrow \text{VALUE} \leq \max(0, rv - 2\alpha) \\
\text{required}(\text{VARIABLES}, \text{var}) &
\end{align*}
\]

where

- \(sv = |\text{VARIABLES}|\)
- \(rv = \text{range}(\text{VARIABLES}.\text{var})\)
- \(np = \max(0, \lfloor (sv - 1)/2 \rfloor)\)

**Purpose**

An occurrence of the pattern **SUMMIT** is the maximal subsequence which matches the regular expression \((< | < (= | <)^* <) (> | > (= | >)^* >)\).

Assume that the occurrence of the pattern **SUMMIT** starts at position \(i\) and ends at position \(j\). The feature **WIDTH** computes the value \(j - i\).

**Example**

\((6, (7, 1, 5, 4, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1))\)

Figure 3.1337 provides an example where the **SUM_WIDTH_SUMMIT** (6, [7, 1, 5, 4, 3, 4, 6, 6, 2, 3, 4, 2, 3, 1]) constraint holds.

**Typical**

- \(|\text{VARIABLES}| > 2\)
- \(\text{range}(\text{VARIABLES}.\text{var}) > 1\)

**Symmetries**

- Items of **VARIABLES** can be reversed.
- One and the same constant can be added to the \(\text{var}\) attribute of all items of **VARIABLES**.

**Arg. properties**

Functional dependency: **VALUE** determined by **VARIABLES**.
Figure 3.1337: Illustrating the SUM_WIDTH_SUMMIT constraint of the Example slot
Automaton

Figures 3.1338 and 3.1339 respectively depict the automaton associated with the constraint SUM_WIDTH_SUMMIT and its simplified form.

Figure 3.1338: Automaton for the SUM_WIDTH_SUMMIT constraint obtained by applying decoration Table 2.35 to the seed transducer of the SUMMIT pattern where default is 0 (transition $u \rightarrow r$ has the same accumulator update as transition $r \rightarrow u$)
Figure 3.1339: Automaton for the SUM_WIDTH_SUMMIT constraint obtained by applying decoration Table 2.25 to the seed transducer of the SUMMIT pattern where default is 0 (transition \( u \rightarrow r \) has the same accumulator update as transition \( r \rightarrow u \)); \( R_i - R_{i-1} \geq 0 \) and \( -R_i + R_{i-1} + D_{i-1} + 1 \geq 0 \) are linear invariants.
Table 3.200: Glue matrix for the SUM_WIDTH_SUMMIT constraint defined as the composition of the SUMMIT pattern, the feature WIDTH, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
3.625 SUM_WIDTH_VALLEY

DESCRIPTION AUTOMATON

Origin
Based on the VALLEY pattern.

Constraint
SUM_WIDTH_VALLEY(VALUE, VARIABLES)

Arguments
VALUE : dvar
VARIABLES : collection(var−dvar)

Restrictions
sv ≤ 2 ∨ rv ≤ 1 ⇒ VALUE = 0
VALUE = 0 ∨ VALUE ≥ 1
VALUE ≤ max(0, sv − 2x)
required(VARIABLES, var)
where
sv = |VARIABLES|
rv = range(VARIABLES var)

Purpose
VALUE is the sum of the width of occurrences of the VALLEY pattern in the time-series given by the VARIABLES collection. If the pattern does not occur, VALUE takes the default value 0.

An occurrence of the pattern VALLEY is the maximal subsequence which matches the regular expression > (= | >)∗ (< | =)∗ <.
Assume that the occurrence of the pattern VALLEY starts at position i and ends at position j. The feature WIDTH computes the value j − i.

Example
(9, (1, 3, 7, 4, 3, 6, 5, 3, 3, 2, 6, 5, 5, 7))

Figure 3.1340 provides an example where the SUM_WIDTH_VALLEY (9, [1, 3, 7, 4, 3, 6, 5, 3, 3, 2, 6, 5, 5, 7]) constraint holds.

Typical
|VARIABLES| > 2
range(VARIABLES var) > 1

Symmetries
• Items of VARIABLES can be reversed.
• One and the same constant can be added to the var attribute of all items of VARIABLES.

Arg. properties
Functional dependency: VALUE determined by VARIABLES.
Figure 3.1340: Illustrating the SUM_WIDTH_VALLEY constraint of the Example slot
Figures 3.1341 and 3.1342 respectively depict the automaton associated with the constraint SUM_WIDTH_VALLEY and its simplified form.

Figure 3.1341: Automaton for the SUM_WIDTH_VALLEY constraint obtained by applying decoration Table 2.35 to the seed transducer of the VALLEY pattern where default is 0

Figure 3.1342: Automaton for the SUM_WIDTH_VALLEY constraint obtained by applying decoration Table 2.25 to the seed transducer of the VALLEY pattern where default is 0; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + D_{i-1} + 1 \geq 0$ are linear invariants.
Table 3.201: Glue matrix for the `SUM_WIDTH_VALLEY` constraint defined as the composition of the `VALLEY` pattern, the feature `WIDTH`, and the aggregator `sum`; cells of the glue matrix are coloured with the colour of the constituent to which they are related.

<table>
<thead>
<tr>
<th></th>
<th>(s)</th>
<th>(r)</th>
<th>(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(s)</td>
<td>(\tilde{C} + \tilde{C})</td>
<td>(\tilde{C} + \tilde{C})</td>
<td>(\tilde{C} + \tilde{C})</td>
</tr>
<tr>
<td>(r)</td>
<td>(\tilde{C} + \tilde{C})</td>
<td>(\tilde{D} + \tilde{D} + 1)</td>
<td>(\tilde{C} + \tilde{D} + \tilde{D} + 1)</td>
</tr>
<tr>
<td>(t)</td>
<td>(\tilde{C} + \tilde{C})</td>
<td>(\tilde{C} + \tilde{D} + \tilde{D} + 1)</td>
<td>(\tilde{C} + \tilde{C})</td>
</tr>
</tbody>
</table>
### 3.626 SUM_WIDTH_ZIGZAG

**DESCRIPTION**

**Origin**
Based on the ZIGZAG pattern.

**Constraint**

\[
\text{SUM_WIDTH_ZIGZAG}(\text{VALUE}, \text{VARIABLES})
\]

**Arguments**

<table>
<thead>
<tr>
<th>VALUE</th>
<th>dvar</th>
</tr>
</thead>
<tbody>
<tr>
<td>VARIABLES</td>
<td>\text{collection(var-dvar)}</td>
</tr>
</tbody>
</table>

**Restrictions**

\[
\begin{align*}
\text{sv} & \leq 3 \lor \text{rv} \leq 1 \Rightarrow \text{VALUE} = 0 \\
\text{VALUE} & = 0 \lor \text{VALUE} \geq 2 \\
\text{VALUE} & \leq \max(0, \text{sv} - 2k) \\
\text{required}(\text{VARIABLES}, \text{var}) & \text{where} \\
\text{sv} & = |\text{VARIABLES}| \\
\text{rv} & = \text{range}(\text{VARIABLES}.\text{var})
\end{align*}
\]

**Purpose**

An occurrence of the pattern ZIGZAG is the maximal subsequence which matches the regular expression \((<>)^+ (<< | <> | >>)^+ (>> | <<)^+\). Assume that the occurrence of the pattern ZIGZAG starts at position \(i\) and ends at position \(j\). The feature WIDTH computes the value \(j - i\).

**Example**

\[(11, (4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1))\]

Figure 3.1343 provides an example where the SUM_WIDTH_ZIGZAG \((11, [4, 1, 3, 1, 4, 6, 1, 5, 5, 2, 7, 2, 3, 1, 6, 1])\) constraint holds.

**Typical**

| \(|\text{VARIABLES}| > 3) | range(\text{VARIABLES}.\text{var}) > 1 |

**Symmetries**

- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

**Arg. properties**

Functional dependency: VALUE determined by VARIABLES.
Figure 3.1343: Illustrating the SUM_WIDTH_ZIGZAG constraint of the Example slot.
Automaton

Figures 3.1344 and 3.1345 respectively depict the automaton associated with the constraint SUM_WIDTH_ZIGZAG and its simplified form.
Figure 3.1344: Automaton for the `SUM_WIDTH_ZIGZAG` constraint obtained by applying decoration Table 2.35 to the seed transducer of the ZIGZAG pattern where `default` is 0; (1) missing transitions from `a, b, c, d, e, f` to `s` are labelled by `=`; (2) on transitions from `b, c, e, f` to `s` the accumulator `D` is reset to its initial value; (3) on transitions from `c, f` to `s` the accumulator `R` is updated wrt `C` and the accumulator `C` is reset to its initial value.
Figure 3.1345: Automaton for the SUM_WIDTH_ZIGZAG constraint obtained by applying decoration Table 2.39 to the seed transducer of the ZIGZAG pattern where default is 0; missing transitions from $a, b, c, d, e, f$ to $s$ are labelled by $=$; $R_i - R_{i-1} \geq 0$ and $-R_i + R_{i-1} + 2 \geq 0$ are linear invariants.
Table 3.202: Glue matrix for the SUM_WIDTH_ZIGZAG constraint defined as the composition of the ZIGZAG pattern, the feature WIDTH, and the aggregator sum; cells of the glue matrix are coloured with the colour of the constituent to which they are related.
Appendix A

Electronic Constraint Catalogue

The file attached to each constraint is available from a link located on the leftmost upper corner of the first page of each constraint provided you are using Adobe Reader.

- The file `eval.pl` containing all utilities is available from this link.
- The file `src.pl` allowing to download all time series constraints in SICStus is available from the link.
- Finally the MiniZinc implementation of the time-series constraints is available from this link. It corresponds to a reformulation of the corresponding automata constraints.
Bibliography


Index
ALL_EQUAL_HEIGHT_DECREASING_TERRACE, 126
ALL_EQUAL_HEIGHT_INCREASING_TERRACE, 130
ALL_EQUAL_HEIGHT_PLAIN, 134
ALL_EQUAL_HEIGHT_PLATEAU, 138
ALL_EQUAL_HEIGHT_PROPER_PLAIN, 142
ALL_EQUAL_HEIGHT_PROPER_PLATEAU, 146
ALL_EQUAL_HEIGHT_STEADY, 150
ALL_EQUAL_HEIGHT_STEADY_SEQUENCE, 154
ALL_EQUAL_MAX_BUMP_ON_DECREASING_SEQUENCE, 158
ALL_EQUAL_MAX_DECREASING, 162
ALL_EQUAL_MAX_DECREASING_SEQUENCE, 166
ALL_EQUAL_MAX_DIP_ON_INCREASING_SEQUENCE, 170
ALL_EQUAL_MAX_INCREASING, 174
ALL_EQUAL_MAX_INCREASING_SEQUENCE, 178
ALL_EQUAL_MAX_INFLEXION, 182
ALL_EQUAL_MAX_PEAK, 184
ALL_EQUAL_MAX_STRICTLY_DECREASING_SEQUENCE, 188
ALL_EQUAL_MAX_STRICTLY_INCREASING_SEQUENCE, 192
ALL_EQUAL_MAX_SUMMIT, 196
ALL_EQUAL_MAX_ZIGZAG, 200
ALL_EQUAL_MIN_BUMP_ON_DECREASING_SEQUENCE, 204
ALL_EQUAL_MIN_DECREASING, 208
ALL_EQUAL_MIN_DECREASING_SEQUENCE, 212
ALL_EQUAL_MIN_DIP_ON_INCREASING_SEQUENCE, 216
ALL_EQUAL_MIN_GORGE, 220
ALL_EQUAL_MIN_INCREASING, 222
ALL_EQUAL_MIN_INCREASING_SEQUENCE, 226
ALL_EQUAL_MIN_INFLEXION, 230
ALL_EQUAL_MIN_STRICTLY_DECREASING_SEQUENCE, 234
ALL_EQUAL_MIN_STRICTLY_INCREASING_SEQUENCE, 238
ALL_EQUAL_MIN_VALLEY, 242
ALL_EQUAL_MIN_ZIGZAG, 246
ALL_EQUAL_RANGE_DECREASING, 250
ALL_EQUAL_RANGE_DECREASING_SEQUENCE, 254
ALL_EQUAL_RANGE_INCREASING, 258
ALL_EQUAL_RANGE_INCREASING_SEQUENCE, 262
ALL_EQUAL_RANGE_STRICTLY_DECREASING_SEQUENCE, 266
ALL_EQUAL_RANGE_STRICTLY_INCREASING_SEQUENCE, 270
ALL_EQUAL_SURF_BUMP_ON_DECREASING_SEQUENCE, 274
ALL_EQUAL_SURF_DECREASING, 278
ALL_EQUAL_SURF_DECREASING_SEQUENCE, 282
ALL_EQUAL_SURF_DECREASING_TERRACE, 286
ALL_EQUAL_SURF_DIP_ON_INCREASING_SEQUENCE, 290
ALL_EQUAL_SURF_GORGE, 294
ALL_EQUAL_SURF_INCREASING, 296
ALL_EQUAL_SURF_INCREASING_SEQUENCE, 300
ALL_EQUAL_SURF_INCREASING_TERRACE, 304
ALL_EQUAL_SURF_INFLEXION, 308
INDEX

ALL_EQUAL_SURF_PEAK, 312
ALL_EQUAL_SURF_PLAIN, 316
ALL_EQUAL_SURF_PLATEAU, 320
ALL_EQUAL_SURF_PROPER_PLAIN, 324
ALL_EQUAL_SURF_PROPER_PLATEAU, 328
ALL_EQUAL_SURF_STEADY, 332
ALL_EQUAL_SURF_STRICTLY_SEQUENCE, 336
ALL_EQUAL_SURF_STRICTLY_DECREASING_SEQUENCE, 340
ALL_EQUAL_SURF_STRICTLY_INCREASING_SEQUENCE, 344
ALL_EQUAL_SURF_SUMMIT, 348
ALL_EQUAL_SURF_VALLEY, 352
ALL_EQUAL_SURF_ZIGZAG, 356
ALL_EQUAL_WIDTH_DECREASING_SEQUENCE, 360
ALL_EQUAL_WIDTH_DECREASING_TERRACE, 364
ALL_EQUAL_WIDTH_GORGE, 368
ALL_EQUAL_WIDTH_INCREASING_SEQUENCE, 372
ALL_EQUAL_WIDTH_INCREASING_TERRACE, 376
ALL_EQUAL_WIDTH_INFLEXION, 380
ALL_EQUAL_WIDTH_PEAK, 382
ALL_EQUAL_WIDTH_PLAIN, 386
ALL_EQUAL_WIDTH_PLATEAU, 390
ALL_EQUAL_WIDTH_PROPER_PLAIN, 394
ALL_EQUAL_WIDTH_PROPER_PLATEAU, 398
ALL_EQUAL_WIDTH_STEADY_SEQUENCE, 402
ALL_EQUAL_WIDTH_STRICTLY_DECREASING_SEQUENCE, 406
ALL_EQUAL_WIDTH_STRICTLY_INCREASING_SEQUENCE, 410
ALL_EQUAL_WIDTH_SUMMIT, 414
ALL_EQUAL_WIDTH_VALLEY, 418
ALL_EQUAL_WIDTH_ZIGZAG, 422

D

DECREASING_HEIGHT_DECREASING_TERRACE, 426
DECREASING_HEIGHT_INCREASING_TERRACE, 430
DECREASING_HEIGHT_PLAIN, 434
DECREASING_HEIGHT_PLATEAU, 438
DECREASING_HEIGHT_PROPER_PLAIN, 442
DECREASING_HEIGHT_PROPER_PLATEAU, 446
DECREASING_HEIGHT_STEADY, 450
DECREASING_HEIGHT_STEADY_SEQUENCE, 454
DECREASING_MAX_BUMP_ON_DECREASING_SEQUENCE, 458
DECREASING_MAX_DECREASING, 462
DECREASING_MAX_DECREASING_SEQUENCE, 466
DECREASING_MAX_DIP_ON_INCREASING_SEQUENCE, 470
DECREASING_MAX_INCREASING, 474
DECREASING_MAX_INCREASING_SEQUENCE, 478
DECREASING_MAX_INFLEXION, 482
DECREASING_MAX_PEAK, 486
DECREASING_MAX_STRICTLY_DECREASING_SEQUENCE, 490
DECREASING_MAX_STRICTLY_INCREASING_SEQUENCE, 494
DECREASING_MAX_SUMMIT, 498
DECREASING_MAX_ZIGZAG, 502
DECREASING_MIN_BUMP_ON_DECREASING_SEQUENCE, 506
DECREASING_MIN_DECREASING, 510
DECREASING_MIN_DECREASING_SEQUENCE, 514
DECREASING_MIN_DIP_ON_INCREASING_SEQUENCE, 518
DECREASING_MIN_GORGE, 522
DECREASING_MIN_INCREASING, 526
DECREASING_MIN_INCREASING_SEQUENCE, 530
DECREASING_MIN_INFLEXION, 534
DECREASING_MIN_STRICTLY_DECREASING_SEQUENCE, 538
DECREASING_MIN_STRICTLY_INCREASING_SEQUENCE, 542
DECREASING_MIN_VALLEY, 546
DECREASING_MIN_ZIGZAG, 550
DECREASING_RANGE_DECREASING, 554
DECREASING_RANGE_DECREASING_SEQUENCE, 558
DECREASING_RANGE_INCREASING, 562
DECREASING_RANGE_INCREASING_SEQUENCE, 566
DECREASING_RANGE_STRICTLY_DECREASING_SEQUENCE, 570
DECREASING_RANGE_STRICTLY_INCREASING_SEQUENCE, 574
DECREASING_SURF_BUMP_ON_DECREASING_SEQUENCE, 578
DECREASING_SURF_DECREASING, 582
DECREASING_SURF_DECREASING_SEQUENCE, 586
DECREASING_SURF_DECREASING_TERRACE, 590
DECREASING_SURF_DIP_ON_INCREASING_SEQUENCE, 594
DECREASING_SURF_GORGE, 598
DECREASING_SURF_INCREASING, 602
DECREASING_SURF_INCREASING_SEQUENCE, 604
DECREASING_SURF_INCREASING_TERRACE, 608
DECREASING_SURF_INFLEXION, 612
DECREASING_SURF_PEAK, 616
DECREASING_SURF_PLAIN, 620
DECREASING_SURF_PLATEAU, 624
DECREASING_SURF_PROPER_PLAIN, 628
DECREASING_SURF_PROPER_PLATEAU, 632
DECREASING_SURF_STEADY, 636
DECREASING_SURF_STEADY_SEQUENCE, 640
DECREASING_SURF_STRICTLY_DECREASING_SEQUENCE, 644
DECREASING_SURF_STRICTLY_INCREASING_SEQUENCE, 648
DECREASING_SURF_SUMMIT, 652
DECREASING_SURF_VALLEY, 656
DECREASING_SURF_ZIGZAG, 660
DECREASING_WIDTH_DECREASING_SEQUENCE, 664
DECREASING_WIDTH_DECREASING_TERRACE, 668
DECREASING_WIDTH_GORGE, 672
DECREASING_WIDTH_INCREASING_SEQUENCE, 676
DECREASING_WIDTH_INCREASING_TERRACE, 680
DECREASING_WIDTH_INFLEXION, 684
DECREASING_WIDTH_PEAK, 688
DECREASING_WIDTH_PLAIN, 692
DECREASING_WIDTH_PLATEAU, 696
DECREASING_WIDTH_PROPER_PLAIN, 700
DECREASING_WIDTH_PROPER_PLATEAU, 704
DECREASING_WIDTH_STEADY_SEQUENCE, 708
DECREASING_WIDTH_STRICTLY_DECREASING_SEQUENCE, 710
DECREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE, 714
DECREASING_WIDTH_SUMMIT, 718
DECREASING_WIDTH_VALLEY, 722
DECREASING_WIDTH_ZIGZAG, 726
DISTINCT, 4

F

first, 5

I

in_attr, 3
in_list, 3
INCREASING_HEIGHT DECREASING_TERRACE, 730
INCREASING_HEIGHT_INCREASING_TERRACE, 734
INCREASING_HEIGHT_PLAIN, 738
INCREASING_HEIGHT_PLATEAU, 742
INCREASING_HEIGHT_PROPER_PLAIN, 746
INCREASING_HEIGHT_PROPER_PLATEAU, 750
INCREASING_HEIGHT_STEADY, 754
INCREASING_HEIGHT_STEADY_SEQUENCE, 758
INCREASING_MAX_BUMP_ON_DECREASING_SEQUENCE, 762
INCREASING_MAX_DECREASING, 766
INCREASING_MAX_DECREASING_SEQUENCE, 770
INCREASING_MAX_DIP_ON_INCREASING_SEQUENCE, 774
INCREASING_MAX_INCREASING, 778
INCREASING_MAX_INCREASING_SEQUENCE, 782
INCREASING_MAX_INFLEXION, 786
INCREASING_MAX_PEAK, 790
INCREASING_MAX_STRICTLY_DECREASING_SEQUENCE, 794
INCREASING_MAX_STRICTLY_INCREASING_SEQUENCE, 798
INCREASING_MAX_SUMMIT, 802
INCREASING_MAX_ZIGZAG, 806
INCREASING_MIN_BUMP_ON_DECREASING_SEQUENCE, 810
INCREASING_MIN_DECREASING, 814
INCREASING_MIN_DECREASING_SEQUENCE, 818
INCREASING_MIN_DIP_ON_INCREASING_SEQUENCE, 822
INCREASING_MIN_GORGE, 826
INCREASING_MIN_INCREASING, 830
INCREASING_MIN_INCREASING_SEQUENCE, 834
INCREASING_MIN_INFLEXION, 838
INCREASING_MIN_STRICTLY_DECREASING_SEQUENCE, 842
INCREASING_MIN_STRICTLY_INCREASING_SEQUENCE, 846
INCREASING_MIN_VALLEY, 850
INCREASING_MIN_ZIGZAG, 854
INCREASING_RANGE_DECREASING, 858
INCREASING_RANGE_DECREASING_SEQUENCE, 862
INCREASING_RANGE_INCREASING, 866
INCREASING_RANGE_INCREASING_SEQUENCE, 870
INCREASING_RANGE_STRICTLY_DECREASING_SEQUENCE, 874
INCREASING_RANGE_STRICTLY_INCREASING_SEQUENCE, 878
INCREASING_SEQ, 4
INCREASING_SURF_BUMP_ON_DECREASING_SEQUENCE, 882
INCREASING_SURF_DECREASING, 886
INCREASING_SURF_DECREASING_SEQUENCE, 888
INCREASING_SURF_DECREASING_TERRACE, 892
INCREASING_SURF_DIP_ON_INCREASING_SEQUENCE, 896
INCREASING_SURF_GORGE, 900
INCREASING_SURF_INCREASING, 904
INCREASING_SURF_INCREASING_SEQUENCE, 908
INCREASING_SURF_INCREASING_TERRACE, 912
INCREASING_SURF_INFLEXION, 916
INCREASING_SURF_PEAK, 920
INCREASING_SURF_PLAIN, 924
INCREASING_SURF_PLATEAU, 928
INCREASING_SURF_PROPER_PLAIN, 932
INCREASING_SURF_PROPER_PLATEAU, 936
INCREASING_SURF_STEADY, 940
INCREASING_SURF_STEADY_SEQUENCE, 944
INCREASING_SURF_STRICTLY_DECREASING_SEQUENCE, 948
INCREASING_SURF_STRICTLY_INCREASING_SEQUENCE, 952
INCREASING_SURF_SUMMIT, 956
INCREASING_SURF_VALLEY, 960
INCREASING_SURF_ZIGZAG, 964
INCREASING_WIDTH_DECREASING_SEQUENCE, 968
INCREASING_WIDTH_DECREASING_TERRACE, 972
INCREASING_WIDTH_GORGE, 976
INCREASING_WIDTH_INCREASING_SEQUENCE, 980
INCREASING_WIDTH_INCREASING_TERRACE, 984
INCREASING_WIDTH_INFLEXION, 988
INCREASING_WIDTH_PEAK, 992
INCREASING_WIDTH_PLAIN, 996
INCREASING_WIDTH_PLATEAU, 1000
INCREASING_WIDTH_PROPER_PLAIN, 1004
INCREASING_WIDTH_PROPER_PLATEAU, 1008
INCREASING_WIDTH_STEADY_SEQUENCE, 1012
INDEX

INCREASING_WIDTH_STRICTLY_DECREASING_SEQUENCE, 1014
INCREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE, 1018
INCREASING_WIDTH_SUMMIT, 1022
INCREASING_WIDTH_VALLEY, 1026
INCREASING_WIDTH_ZIGZAG, 1030

L

last, 5

M

MAX_HEIGHT_DECREASING_TERRACE, 1034, 1760
MAX_HEIGHT_INCREASING_TERRACE, 1038, 1764
MAX_HEIGHT_PLAIN, 1042, 1768
MAX_HEIGHT_PLATEAU, 1046, 1772
MAX_HEIGHT_PROPER_PLAIN, 1050, 1776
MAX_HEIGHT_PROPER_PLATEAU, 1054, 1780
MAX_HEIGHT_STEADY, 1058, 1784
MAX_HEIGHT_STEADY_SEQUENCE, 1062, 1788
MAX_MAX_BUMP_ON_DECREASING_SEQUENCE, 1066, 1792
MAX_MAX_DECREASING, 1070, 1796
MAX_MAX_DECREASING_SEQUENCE, 1074, 1800
MAX_MAX_DIP_ON_INCREASING_SEQUENCE, 1078, 1804
MAX_MAX_INCREASING, 1082, 1808
MAX_MAX_INCREASING_SEQUENCE, 1086, 1812
MAX_MAX_INFLEXION, 1090, 1816
MAX_MAX_PEAK, ii, 1094, 1820
MAX_MAX_STRICTLY_DECREASING_SEQUENCE, 1098, 1824
MAX_MAX_STRICTLY_INCREASING_SEQUENCE, 1102, 1828
MAX_MAX_SUMMIT, 1106, 1832
MAX_MAX_ZIGZAG, 1110, 1836
MAX_MIN_BUMP_ON_DECREASING_SEQUENCE, 1116, 1840
MAX_MIN_DECREASING, 1120, 1844
MAX_MIN_DECREASING_SEQUENCE, 1124, 1848
MAX_MIN_DIP_ON_INCREASING_SEQUENCE, 1128, 1852
MAX_MIN_GORGE, 1132, 1856
MAX_MIN_INCREASING, 1136, 1860
MAX_MIN_INCREASING_SEQUENCE, 1140, 1864
MAX_MIN_INFLEXION, 1144, 1868
MAX_MIN_STRICTLY_DECREASING_SEQUENCE, 1148, 1872
MAX_MIN_STRICTLY_INCREASING_SEQUENCE, 1152, 1876
MAX_MIN_VALLEY, 1156, 1880
MAX_MIN_ZIGZAG, 1160, 1884
MAX_RANGE_DECREASING, ii, 1166
INDEX

MAX_RANGE_DECREASING_SEQUENCE, 1170
MAX_RANGE_INCREASING, ii, 1174
MAX_RANGE_INCREASING_SEQUENCE, 1178
MAX_RANGE.Strictly_DECREASING_SEQUENCE, 1182
MAX_RANGE.Strictly_INCREASING_SEQUENCE, 1186
MAX_SURF_BUMP_ON_DECREASING_SEQUENCE, 1190, 1888
MAX_SURF_DECREASING, 1194, 1892
MAX_SURF_DECREASING_SEQUENCE, 1198, 1896
MAX_SURF_DECREASING_TERRACE, 1202, 1900
MAX_SURF_DIP_ON_INCREASING_SEQUENCE, 1206, 1904
MAX_SURF_GORGE, 1210, 1908
MAX_SURF_INCREASING, 1214, 1912
MAX_SURF_INCREASING_SEQUENCE, 1218, 1916
MAX_SURF_INCREASING_TERRACE, 1222, 1920
MAX_SURF_INFLEXION, 1226, 1924
MAX_SURF_PEAK, 1230, 1928
MAX_SURF_PLAIN, 1234, 1932
MAX_SURF_PLATEAU, 1238, 1936
MAX_SURF_PROPER_PLAIN, 1242, 1940
MAX_SURF_PROPER_PLATEAU, 1246, 1944
MAX_SURF_STEADY, 1250, 1948
MAX_SURF_STEADY_SEQUENCE, 1254, 1952
MAX_SURF.Strictly_DECREASING_SEQUENCE, 1258, 1956
MAX_SURF.Strictly_INCREASING_SEQUENCE, 1262, 1960
MAX_SURF_SUMMIT, 1266, 1964
MAX_SURF_VALLEY, 1270, 1968
MAX_SURF_ZIGZAG, 1274, 1972
MAX_WIDTH_DECREASING_SEQUENCE, 1280, 1976
MAX_WIDTH_DECREASING_TERRACE, 1284, 1980
MAX_WIDTH_GORGE, 1288, 1984
MAX_WIDTH_INCREASING_SEQUENCE, 1294, 1988
MAX_WIDTH_INCREASING_TERRACE, 1298, 1992
MAX_WIDTH_INFLEXION, 1302, 1996
MAX_WIDTH_PEAK, 1306, 2000
MAX_WIDTH_PLAIN, 1310, 2004
MAX_WIDTH_PLATEAU, 1314, 2008
MAX_WIDTH_PROPER_PLAIN, 1318, 2012
MAX_WIDTH_PROPER_PLATEAU, 1322, 2016
MAX_WIDTH_STEADY_SEQUENCE, 1326, 2020
MAX_WIDTH.Strictly_DECREASING_SEQUENCE, ii, 1330, 2024
MAX_WIDTH.Strictly_INCREASING_SEQUENCE, ii, 1334, 2028
MAX_WIDTH_SUMMIT, 1338, 2032
MAX_WIDTH_VALLEY, 1344, 2036
MAX_WIDTH_ZIGZAG, 1348, 2040
maxval, 6
MIN_HEIGHT_DECREASING_TERRACE, 1354, 2044
MIN_HEIGHT_INCREASING_TERRACE, 1358, 2048
MIN_HEIGHT_PLAIN, 1362, 2052
MIN_HEIGHT_PLATEAU, 1366, 2056
MIN_HEIGHT_PROPER_PLAIN, 1370, 2060
<table>
<thead>
<tr>
<th>INDEX</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIN_HEIGHT_PROPER_PLATEAU, 1374, 2064</td>
</tr>
<tr>
<td>MIN_HEIGHT_STEADY, 1378, 2068</td>
</tr>
<tr>
<td>MIN_HEIGHT_STEADY_SEQUENCE, 1382, 2072</td>
</tr>
<tr>
<td>MIN_MAX_BUMP_ON_DECREASING_SEQUENCE, 1386, 2076</td>
</tr>
<tr>
<td>MIN_MAX_DECREASING, 1390, 2080</td>
</tr>
<tr>
<td>MIN_MAX_DECREASING_SEQUENCE, 1394, 2084</td>
</tr>
<tr>
<td>MIN_MAX_DIP_ON_INCREASING_SEQUENCE, 1398, 2088</td>
</tr>
<tr>
<td>MIN_MAX_INCREASING, 1402, 2092</td>
</tr>
<tr>
<td>MIN_MAX_INCREASING_SEQUENCE, 1406, 2096</td>
</tr>
<tr>
<td>MIN_MAX_INFLEXION, 1410, 2100</td>
</tr>
<tr>
<td>MIN_MAX_PEAK, ii, 1414, 2104</td>
</tr>
<tr>
<td>MIN_MAX STRICTLY DECREASING_SEQUENCE, 1418, 2108</td>
</tr>
<tr>
<td>MIN_MAX STRICTLY INCREASING_SEQUENCE, 1422, 2112</td>
</tr>
<tr>
<td>MIN_MAX SUMMIT, 1426, 2116</td>
</tr>
<tr>
<td>MIN_MAX_ZIGZAG, 1430, 2120</td>
</tr>
<tr>
<td>MIN_MAX_BUMP_ON_DECREASING_SEQUENCE, 1436, 2124</td>
</tr>
<tr>
<td>MIN_MAX_Decreasing, 1440, 2128</td>
</tr>
<tr>
<td>MIN_MAX_Decreasing_SEQUENCE, 1444, 2132</td>
</tr>
<tr>
<td>MIN_MIN_BUMP_ON_DECREASING_SEQUENCE, 1450, 2136</td>
</tr>
<tr>
<td>MIN_MIN_BUMP, 1452, 2140</td>
</tr>
<tr>
<td>MIN_MIN_INCREASING, 1456, 2144</td>
</tr>
<tr>
<td>MIN_MIN_INCREASING_SEQUENCE, 1460, 2148</td>
</tr>
<tr>
<td>MIN_MIN_INFLEXION, 1464, 2152</td>
</tr>
<tr>
<td>MIN_MIN STRICTLY DECREASING_SEQUENCE, 1468, 2156</td>
</tr>
<tr>
<td>MIN_MIN STRICTLY INCREASING_SEQUENCE, 1472, 2160</td>
</tr>
<tr>
<td>MIN_MIN_VALLEY, 1476, 2164</td>
</tr>
<tr>
<td>MIN_MIN_ZIGZAG, 1480, 2168</td>
</tr>
<tr>
<td>MIN_RANGE_DEC creasing, 1486</td>
</tr>
<tr>
<td>MIN_RANGE_Decreasing_SEQUENCE, 1490</td>
</tr>
<tr>
<td>MIN_RANGE_INCREASING, 1494</td>
</tr>
<tr>
<td>MIN_RANGE_INCREASING_SEQUENCE, 1498</td>
</tr>
<tr>
<td>MIN_RANGE STRICTLY DECREASING_SEQUENCE, 1502</td>
</tr>
<tr>
<td>MIN_RANGE STRICTLY INCREASING_SEQUENCE, 1506</td>
</tr>
<tr>
<td>MIN_SURF_BUMP_ON_DECREASING_SEQUENCE, 1510, 2172</td>
</tr>
<tr>
<td>MIN_SURF_Decreasing, 1514, 2176</td>
</tr>
<tr>
<td>MIN_SURF_Decreasing_SEQUENCE, 1518, 2180</td>
</tr>
<tr>
<td>MIN_SURF_Decreasing_Terrace, 1522, 2184</td>
</tr>
<tr>
<td>MIN_SURF_DIP_ON_INCREASING_SEQUENCE, 1526, 2188</td>
</tr>
<tr>
<td>MIN_SURF_GORGE, 1530, 2192</td>
</tr>
<tr>
<td>MIN_SURF_INCREASING, 1534, 2196</td>
</tr>
<tr>
<td>MIN_SURF_INCREASING_SEQUENCE, 1538, 2200</td>
</tr>
<tr>
<td>MIN_SURF_INCREASING_Terrace, 1542, 2204</td>
</tr>
<tr>
<td>MIN_SURF_INFLEXION, 1546, 2208</td>
</tr>
<tr>
<td>MIN_SURF_PEAK, 1550, 2212</td>
</tr>
<tr>
<td>MIN_SURF_PLAIN, 1554, 2216</td>
</tr>
<tr>
<td>MIN_SURF_Plateau, 1558, 2220</td>
</tr>
<tr>
<td>MIN_SURF_PROPER_PLAIN, 1562, 2224</td>
</tr>
<tr>
<td>MIN_SURF_PROPER_Plateau, 1566, 2228</td>
</tr>
<tr>
<td>MIN_SURF_STEADY, 1570, 2232</td>
</tr>
<tr>
<td>MIN_SURF_STEADY_SEQUENCE, 1574, 2236</td>
</tr>
</tbody>
</table>
INDEX

MIN_SURF_STRICTLY_DECREASING_SEQUENCE, 1578, 2240
MIN_SURF_STRICTLY_INCREASING_SEQUENCE, 1582, 2244
MIN_SURF_SUMMIT, 1586, 2248
MIN_SURF_VALLEY, 1590, 2252
MIN_SURF_ZIGZAG, 1594, 2256
MIN_WIDTH_DECREASING_SEQUENCE, 1600, 2260
MIN_WIDTH_DECREASING_TERRACE, 1604, 2264
MIN_WIDTH_GORGE, 1608, 2268
MIN_WIDTH_INCREASING_SEQUENCE, 1612, 2272
MIN_WIDTH_INCREASING_TERRACE, 1616, 2276
MIN_WIDTH_INFLEXION, 1620, 2280
MIN_WIDTH_PEAk, 1624, 2284
MIN_WIDTH_PLAIN, ii, 1628, 2288
MIN_WIDTH_PLATEAU, ii, 1632, 2292
MIN_WIDTH_PROPER_PLAIN, 1636, 2296
MIN_WIDTH_PROPER_PLATEAU, 1640, 2300
MIN_WIDTH_STEADY_SEQUENCE, 1644, 2304
MIN_WIDTH_STRICTLY_DECREASING_SEQUENCE, 1648, 2308
MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE, 1652, 2312
MIN_WIDTH_SUMMIT, 1656, 2316
MIN_WIDTH_VALLEY, 1660, 2320
MIN_WIDTH_ZIGZAG, 1664, 2324
minval, 6

N

NB_BUMP_ON_DECREASING_SEQUENCE, ii, 1670
NB_DECREASING, 1674
NB_DECREASING_SEQUENCE, 1678
NB_DECREASING_TERRACE, 1682
NB_DIP_ON_INCREASING_SEQUENCE, ii, 1686
NB_GORGE, ii, 1690
NB_INCREASING, 1694
NB_INCREASING_SEQUENCE, 1698
NB_INCREASING_TERRACE, 1702
NB_INFLEXION, 1706
NB_PEAK, ii, 1710
NB_PLAIN, 1714
NB_PLATEAU, 1718
NB_PROPER_PLAIN, 1722
NB_PROPER_PLATEAU, 1726
NB_STEADY, 1730
NB_STEADY_SEQUENCE, 1734
NB_STRICTLY_DECREASING_SEQUENCE, 1738
NB_STRICTLY_INCREASING_SEQUENCE, 1742
NB_SUMMIT, ii, 1746
NB_VALLEY, ii, 1750
P

POS_MAX_HEIGHT_DECREASING_TERRACE, 1760
POS_MAX_HEIGHT_INCREASING_TERRACE, 1764
POS_MAX_HEIGHT.PLAIN, 1768
POS_MAX_HEIGHT_PLATEAU, 1772
POS_MAX_HEIGHT_PROPER.PLAIN, 1776
POS_MAX_HEIGHT_PROPER.PLATEAU, 1780
POS_MAX_HEIGHT_STEADY, 1784
POS_MAX_HEIGHT_STEADY_SEQUENCE, 1788
POS_MAX_MAX_BUMP_ON_DECREASING_SEQUENCE, 1792
POS_MAX_MAX_DECRESING, 1796
POS_MAX_MAX_DECRESING_SEQUENCE, 1800
POS_MAX_MAX_DIP_ON_INCREASING_SEQUENCE, 1804
POS_MAX_MAX_INCREASING, 1808
POS_MAX_MAX_INCREASING_SEQUENCE, 1812
POS_MAX_MAX_INFLEXION, 1816
POS_MAX_MAX_PEAK, 1820
POS_MAX_MAX_STRICTLY_DECRESING_SEQUENCE, 1824
POS_MAX_MAX_STRICTLY_INCREASING_SEQUENCE, 1828
POS_MAX_MAX_SUMMIT, 1832
POS_MAX_MAX_ZIGZAG, 1836
POS_MAX_MIN_BUMP_ON_DECREASING_SEQUENCE, 1840
POS_MAX_MIN_DecREASING, 1844
POS_MAX_MIN_DECRESING_SEQUENCE, 1848
POS_MAX_MIN_DIP_ON_INCREASING_SEQUENCE, 1852
POS_MAX_MIN_GORGE, 1856
POS_MAX_MIN_INCREASING, 1860
POS_MAX_MIN_INCREASING_SEQUENCE, 1864
POS_MAX_MIN_INFLEXION, 1868
POS_MAX_MIN_STRICTLY_DECRESING_SEQUENCE, 1872
POS_MAX_MIN_STRICTLY_INCREASING_SEQUENCE, 1876
POS_MAX_MIN_VALLEY, 1880
POS_MAX_MIN_ZIGZAG, 1884
POS_MAX_SURF_BUMP_ON_DECREASING_SEQUENCE, 1888
POS_MAX_SURF_DECRESING, 1892
POS_MAX_SURF_DECRESING_SEQUENCE, 1896
POS_MAX_SURF_DECRESING_TERRACE, 1900
POS_MAX_SURF_DIP_ON_INCREASING_SEQUENCE, 1904
POS_MAX_SURF_GORGE, 1908
POS_MAX_SURF_INCREASING, 1912
POS_MAX_SURF_INCREASING_SEQUENCE, 1916
POS_MAX_SURF_INCREASING_TERRACE, 1920

NB.ZIGZAG, iii, 1754
non_increasing_size, 4
nval, 6
POS_MIN_WIDTH_ZIGZAG, 2324
prod, 6

R
range, 6
require_at_least, 5
required, 4

S
same_size, 5
signature

AUTOMATON
ALL_EQUAL_HEIGHT_DECREASING_TERRACE, 126
ALL_EQUAL_HEIGHT_INCREASING_TERRACE, 130
ALL_EQUAL_HEIGHT_PLAIN, 134
ALL_EQUAL_HEIGHT_PLATEAU, 138
ALL_EQUAL_HEIGHT_PROPER_PLAIN, 142
ALL_EQUAL_HEIGHT_PROPER_PLATEAU, 146
ALL_EQUAL_HEIGHT_STEADY, 150
ALL_EQUAL_HEIGHT_STEADY_SEQUENCE, 154
ALL_EQUAL_MAX_BUMP_ON_DECREASING_SEQUENCE, 158
ALL_EQUAL_MAX_DECREASING, 162
ALL_EQUAL_MAX_DECREASING_SEQUENCE, 166
ALL_EQUAL_MAX_DIP_ON_INCREASING_SEQUENCE, 170
ALL_EQUAL_MAX_INCREASING, 174
ALL_EQUAL_MAX_INCREASING_SEQUENCE, 178
ALL_EQUAL_MAX_INFLEXION, 182
ALL_EQUAL_MAX_PEAK, 184
ALL_EQUAL_MAX.Strictly.%_DECREASING_SEQUENCE, 188
ALL_EQUAL_MAX.Strictly.%_INCREASING_SEQUENCE, 192
ALL_EQUAL_MAX_SUMMIT, 196
ALL_EQUAL_MAX_ZIGZAG, 200
ALL_EQUAL_MIN_BUMP_ON_DECREASING_SEQUENCE, 204
ALL_EQUAL_MIN_DECREASING, 208
ALL_EQUAL_MIN_DECREASING_SEQUENCE, 212
ALL_EQUAL_MIN_DIP_ON_INCREASING_SEQUENCE, 216
ALL_EQUAL_MIN_GORGE, 220
ALL_EQUAL_MIN_INCREASING, 222
ALL_EQUAL_MIN_INCREASING_SEQUENCE, 226
ALL_EQUAL_MIN_INFLEXION, 230
ALL_EQUAL_MIN.Strictly.%_DECREASING_SEQUENCE, 234
ALL_EQUAL_MIN.Strictly.%_INCREASING_SEQUENCE, 238
ALL_EQUAL_MIN_VALLEY, 242
ALL_EQUAL_MIN_ZIGZAG, 246
ALL_EQUAL_RANGE_DECREASING, 250
ALL_EQUAL_RANGE_DECREASING_SEQUENCE, 254
ALL_EQUAL_RANGE_INCREASING, 258
ALL_EQUAL_RANGE_INCREASING_SEQUENCE, 262
ALL_EQUAL_RANGE.StrictLY_DECREASING_SEQUENCE, 266
ALL_EQUAL_RANGE.StrictLY_INCREASING_SEQUENCE, 270
ALL_EQUAL_SURF_BUMP_ON_DECREASING_SEQUENCE, 274
ALL_EQUAL_SURF_DECREASING, 278
ALL_EQUAL_SURF_DECREASING_SEQUENCE, 282
ALL_EQUAL_SURF_DIP_ON_INCREASING_SEQUENCE, 290
ALL_EQUAL_SURF_GORGE, 294
ALL_EQUAL_SURF_INCREASING, 296
ALL_EQUAL_SURF_INCREASING_SEQUENCE, 300
ALL_EQUAL_SURF_INCREASING_TERRACE, 304
ALL_EQUAL_SURF_INFLEXION, 308
ALL_EQUAL_SURF_PEAK, 312
ALL_EQUAL_SURF_PLAIN, 316
ALL_EQUAL_SURF_PLATEAU, 320
ALL_EQUAL_SURF_PROPER_PLAIN, 324
ALL_EQUAL_SURF_PROPER_PLATEAU, 328
ALL_EQUAL_SURF_STEADY, 332
ALL_EQUAL_SURF_STEADY_SEQUENCE, 336
ALL_EQUAL_SURF_STRICTLY_DECREASING_SEQUENCE, 340
ALL_EQUAL_SURF_STRICTLY_INCREASING_SEQUENCE, 344
ALL_EQUAL_SURF_SUMMIT, 348
ALL_EQUAL_SURF_VALLEY, 352
ALL_EQUAL_SURF_ZIGZAG, 356
ALL_EQUAL_WIDTH_DECREASING_SEQUENCE, 360
ALL_EQUAL_WIDTH_DECREASING_TERRACE, 364
ALL_EQUAL_WIDTH_GORGE, 368
ALL_EQUAL_WIDTH_INCREASING_SEQUENCE, 372
ALL_EQUAL_WIDTH_INCREASING_TERRACE, 376
ALL_EQUAL_WIDTH_INFLEXION, 380
ALL_EQUAL_WIDTH_PEAK, 382
ALL_EQUAL_WIDTH_PLAIN, 386
ALL_EQUAL_WIDTH_PLATEAU, 390
ALL_EQUAL_WIDTH_PROPER_PLAIN, 394
ALL_EQUAL_WIDTH_PROPER_PLATEAU, 398
ALL_EQUAL_WIDTH_STEADY_SEQUENCE, 402
ALL_EQUAL_WIDTH_STRICTLY_DECREASING_SEQUENCE, 406
ALL_EQUAL_WIDTH_STRICTLY_INCREASING_SEQUENCE, 410
ALL_EQUAL_WIDTH_SUMMIT, 414
ALL_EQUAL_WIDTH_VALLEY, 418
ALL_EQUAL_WIDTH_ZIGZAG, 422
DECREASING_HEIGHT_DECREASING_TERRACE, 426
DECREASING_HEIGHT_INCREASING_TERRACE, 430
DECREASING_HEIGHT_PLAIN, 434
DECREASING_HEIGHT_PLATEAU, 438
DECREASING_HEIGHT_PROPER_PLAIN, 442
DECREASING_HEIGHT_PROPER_PLATEAU, 446
DECREASING_HEIGHT_STEADY, 450
DECREASING_HEIGHT_STEADY_SEQUENCE, 454
DECREASING_MAX_BUMP_ON_DECREASING_SEQUENCE, 458
DECREASING_MAX_DECREASING, 462
DECREASING_MAX_DECREASING_SEQUENCE, 466
DECREASING_MAX_DIP_ON_INCREASING_SEQUENCE, 470
DECREASING_MAX_INCREASING, 474
DECREASING_MAX_INCREMENTING_SEQUENCE, 478
DECREASING_MAX_INFLEXION, 482
DECREASING_MAX_PEAK, 486
DECREASING_MAX.Strictly_DECREASING_SEQUENCE, 490
DECREASING_MAX.Strictly_INCREASING_SEQUENCE, 494
DECREASING_MAX_SUMMIT, 498
DECREASING_MAX.ZIGZAG, 502
DECREASING_MIN_BUMP_ON_DECREASING_SEQUENCE, 506
DECREASING_MIN_DECREASING, 510
DECREASING_MIN_DECREASING_SEQUENCE, 514
DECREASING_MIN_DIP_ON_INCREASING_SEQUENCE, 518
DECREASING_MIN_GORGE, 522
DECREASING_MIN_INCREMENTING, 526
DECREASING_MIN_INCREMENTING_SEQUENCE, 530
DECREASING_MIN_INFLEXION, 534
DECREASING_MIN.Strictly_DECREASING_SEQUENCE, 538
DECREASING_MIN.Strictly_INCREASING_SEQUENCE, 542
DECREASING_MIN_Valley, 546
DECREASING_MIN.ZIGZAG, 550
DECREASING_RANGE_DECREASING, 554
DECREASING_RANGE_DECREASING_SEQUENCE, 558
DECREASING_RANGE_INCREASING, 562
DECREASING_RANGE_INCREMENTING_SEQUENCE, 566
DECREASING_RANGE.Strictly_DECREASING_SEQUENCE, 570
DECREASING_RANGE.Strictly_INCREASING_SEQUENCE, 574
DECREASING_SURF_BUMP_ON_DECREASING_SEQUENCE, 578
DECREASING_SURF_DECREASING, 582
DECREASING_SURF_DECREASING_SEQUENCE, 586
DECREASING_SURF_DECREASING_TERRACE, 590
DECREASING_SURF_DIP_ON_INCREASING_SEQUENCE, 594
DECREASING_SURF_GORGE, 598
DECREASING_SURF_INCREASING, 602
DECREASING_SURF_INCREASING_SEQUENCE, 604
DECREASING_SURF_INCREASING_TERRACE, 608
DECREASING_SURF_INFLEXION, 612
DECREASING_SURF_PEAK, 616
DECREASING_SURF_PLAIN, 620
DECREASING_SURF_PLATEAU, 624
DECREASING_SURF_PROPER_PLAIN, 628
DECREASING_SURF_PROPER_PLATEAU, 632
DECREASING_SURF_STEADY, 636
DECREASING_SURF_STEADY_SEQUENCE, 640
DECREASING_SURF_STRICTLY_DECRESING_SEQUENCE, 644
DECREASING_SURF_STRICTLY_INCREASING_SEQUENCE, 648
DECREASING_SURF_SUMMIT, 652
DECREASING_SURF_VALLEY, 656
DECREASING_SURF_ZIGZAG, 660
DECREASING_WIDTH_DECRESING_SEQUENCE, 664
DECREASING_WIDTH_DECRESING_TERRACE, 668
DECREASING_WIDTH_GORGE, 672
DECREASING_WIDTH_INCREASING_SEQUENCE, 676
DECREASING_WIDTH_INCREASING_TERRACE, 680
DECREASING_WIDTH_INFLEXION, 684
DECREASING_WIDTH_PEAK, 688
DECREASING_WIDTH_PLAIN, 692
DECREASING_WIDTH_PLATEAU, 696
DECREASING_WIDTH_PROPER_PLAIN, 700
DECREASING_WIDTH_PROPER_PLATEAU, 704
DECREASING_WIDTH_STEADY_SEQUENCE, 708
DECREASING_WIDTH_STRICTLY_DECRESING_SEQUENCE, 710
DECREASING_WIDTH_STRICTLY_INCREASING_SEQUENCE, 714
DECREASING_WIDTH_SUMMIT, 718
DECREASING_WIDTH_VALLEY, 722
DECREASING_WIDTH_ZIGZAG, 726
INCREASING_HEIGHT_DECRESING_TERRACE, 730
INCREASING_HEIGHT_INCREASING_TERRACE, 734
INCREASING_HEIGHT_PLAIN, 738
INCREASING_HEIGHT_PLATEAU, 742
INCREASING_HEIGHT_PROPER_PLAIN, 746
INCREASING_HEIGHT_PROPER_PLATEAU, 750
INCREASING_HEIGHT_STEADY, 754
INCREASING_HEIGHT_STEADY_SEQUENCE, 758
INCREASING_MAX_BUMP_ON_DECRESING_SEQUENCE, 762
INCREASING_MAX_DECRESING, 766
INCREASING_MAX_DECRESING_SEQUENCE, 770
INCREASING_MAX_DIP_ON_INCREASING_SEQUENCE, 774
INCREASING_MAX_INCREASING, 778
INCREASING_MAX_INCREASING_SEQUENCE, 782
INCREASING_MAX_INFLEXION, 786
INCREASING_MAX_PEAK, 790
INCREASING_MAX_STRICTLY_DECRESING_SEQUENCE, 794
INCREASING_MAX_STRICTLY_INCREASING_SEQUENCE, 798
INCREASING_MAX_SUMMIT, 802
INCREASING_MAX_ZIGZAG, 806
INCREASING_MIN_BUMP_ON_DECRESING_SEQUENCE, 810
INCREASING_MIN_DECRESING, 814
INCREASING_MIN_DECRESING_SEQUENCE, 818
INCREASING_MIN_DIP_ON_INCREASING_SEQUENCE, 822
INCREASING_MIN_GORGE, 826
INCREASING_MIN_INCREASING, 830
INCREASING_MIN_INCREASING_SEQUENCE, 834
INDEX

MAX_HEIGHT_DECREASING_TERRACE, 1034
MAX_HEIGHT_INCREASING_TERRACE, 1038
MAX_HEIGHT_PLAIN, 1042
MAX_HEIGHT_PLATEAU, 1046
MAX_HEIGHT_PROPER_PLAIN, 1050
MAX_HEIGHT_PROPER_PLATEAU, 1054
MAX_HEIGHT_STEADY, 1058
MAX_HEIGHT_STEADY_SEQUENCE, 1062
MAX_MAX_BUMP_ON_DECREASING_SEQUENCE, 1066
MAX_MAX_DECREASING, 1070
MAX_MAX_DECREASING_SEQUENCE, 1074
MAX_MAX_DIP_ON_INCREASING_SEQUENCE, 1078
MAX_MAX_INCREASING, 1082
MAX_MAX_INCREASING_SEQUENCE, 1086
MAX_MAX_INFLEXION, 1090
MAX_MAX_PEAK, 1094
MAX_MAX_STRICTLY_DECREASING_SEQUENCE, 1098
MAX_MAX_STRICTLY_INCREASING_SEQUENCE, 1102
MAX_MAX_SUMMIT, 1106
MAX_MAX_ZIGZAG, 1110
MAX_MIN_BUMP_ON_DECREASING_SEQUENCE, 1116
MAX_MIN_DECREASING, 1120
MAX_MIN_DECREASING_SEQUENCE, 1124
MAX_MIN_DIP_ON_INCREASING_SEQUENCE, 1128
MAX_MIN_GORGE, 1132
MAX_MIN_INCREASING, 1136
MAX_MIN_INCREASING_SEQUENCE, 1140
MAX_MIN_INFLEXION, 1144
MAX_MIN_STRICTLY_DECREASING_SEQUENCE, 1148
MAX_MIN_STRICTLY_INCREASING_SEQUENCE, 1152
MAX_MIN_VALLEY, 1156
MAX_MIN_ZIGZAG, 1160
MAX_RANGE_DECREASING, 1166
MAX_RANGE_DECREASING_SEQUENCE, 1170
MAX_RANGE_INCREASING, 1174
MAX_RANGE_INCREASING_SEQUENCE, 1178
MAX_RANGE_STRICTLY_DECREASING_SEQUENCE, 1182
MAX_RANGE_STRICTLY_INCREASING_SEQUENCE, 1186
MAX_SURF_BUMP_ON_DECREASING_SEQUENCE, 1190
MAX_SURF_DECREASING, 1194
MAX_SURF_DECREASING_SEQUENCE, 1198
MAX_SURF_DECREASING_TERRACE, 1202
MAX_SURF_DIP_ON_INCREASING_SEQUENCE, 1206
MAX_SURF_GORGE, 1210
MAX_SURF_INCREASING, 1214
MAX_SURF_INCREASING_SEQUENCE, 1218
MAX_SURF_INCREASING_TERRACE, 1222
MAX_SURF_INFLEXION, 1226
MAX_SURF_PEAK, 1230
MAX_SURF_PLAIN, 1234
INDEX

MIN_MIN_DIP_ON_INCREASING_SEQUENCE, 1448
MIN_MIN_GORGE, 1452
MIN_MIN_INCREASING, 1456
MIN_MIN_INCREASING_SEQUENCE, 1460
MIN_MIN_INFLEXION, 1464
MIN_MIN.Strictly_DECREASING_SEQUENCE, 1468
MIN_MIN.Strictly_INCREASING_SEQUENCE, 1472
MIN_MIN_VALLEY, 1476
MIN_MIN.ZIGZAG, 1480
MIN_RANGE_DECREASING, 1486
MIN_RANGE_DECREASING_SEQUENCE, 1490
MIN_RANGE_INCREASING, 1494
MIN_RANGE_INCREASING_SEQUENCE, 1498
MIN_RANGE.Strictly_DECREASING_SEQUENCE, 1502
MIN_RANGE.Strictly_INCREASING_SEQUENCE, 1506
MIN_SURF_BUMP_ON_DECREASING_SEQUENCE, 1510
MIN_SURF_DECREASING, 1514
MIN_SURF_DECREASING_SEQUENCE, 1518
MIN_SURF_DECREASING_TERRACE, 1522
MIN_SURF_DIP_ON_INCREASING_SEQUENCE, 1526
MIN_SURF_GORGE, 1530
MIN_SURF_INCREASING, 1534
MIN_SURF_INCREASING_SEQUENCE, 1538
MIN_SURF_INCREASING_TERRACE, 1542
MIN_SURF_INFLEXION, 1546
MIN_SURF_PEAK, 1550
MIN_SURF_PLAIN, 1554
MIN_SURF_PLATEAU, 1558
MIN_SURF_PROPER_PLAIN, 1562
MIN_SURF_PROPER_PLATEAU, 1566
MIN_SURF_STEADY, 1570
MIN_SURF_STEADY_SEQUENCE, 1574
MIN_SURF.Strictly_DECREASING_SEQUENCE, 1578
MIN_SURF.Strictly_INCREASING_SEQUENCE, 1582
MIN_SURF_SUMMIT, 1586
MIN_SURF.Valley, 1590
MIN_SURF.ZIGZAG, 1594
MIN_WIDTH_DECREASING_SEQUENCE, 1600
MIN_WIDTH_DECREASING_TERRACE, 1604
MIN_WIDTH_GORGE, 1608
MIN_WIDTH_INCREASING_SEQUENCE, 1612
MIN_WIDTH_INCREASING_TERRACE, 1616
MIN_WIDTH_INFLEXION, 1620
MIN_WIDTH_PEAK, 1624
MIN_WIDTH_PLAIN, 1628
MIN_WIDTH_PLATEAU, 1632
MIN_WIDTH_PROPER_PLAIN, 1636
MIN_WIDTH_PROPER_PLATEAU, 1640
MIN_WIDTH_STEADY_SEQUENCE, 1644
MIN_WIDTH.Strictly_DECREASING_SEQUENCE, 1648
MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE, 1652
MIN_WIDTH_SUMMIT, 1656
MIN_WIDTH_VALLEY, 1660
MIN_WIDTH_ZIGZAG, 1664
NB_BUMP_ON_DECREASING_SEQUENCE, 1670
NB_DECREASING, 1674
NB_DECREASING_SEQUENCE, 1678
NB_DECREASING_TERRACE, 1682
NB_DIP_ON_INCREASING_SEQUENCE, 1686
NB_GORGE, 1690
NB_INCREASING, 1694
NB_INCREASING_SEQUENCE, 1698
NB_INCREASING_TERRACE, 1702
NB_INFLEXION, 1706
NB_PEAK, 1710
NB_PLAIN, 1714
NB_PLATEAU, 1718
NB_PROPER_PLAIN, 1722
NB_PROPER_PLATEAU, 1726
NB_STEADY, 1730
NB_STEADY_SEQUENCE, 1734
NB_STRICTLY_DECREASING_SEQUENCE, 1738
NB_STRICTLY_INCREASING_SEQUENCE, 1742
NB_SUMMIT, 1746
NB_VALLEY, 1750
NB_ZIGZAG, 1754
POS_MAX_HEIGHT_DECREASING_TERRACE, 1760
POS_MAX_HEIGHT_INCREASING_TERRACE, 1764
POS_MAX_HEIGHT_PLAIN, 1768
POS_MAX_HEIGHT_PLATEAU, 1772
POS_MAX_HEIGHT_PROPER_PLAIN, 1776
POS_MAX_HEIGHT_PROPER_PLATEAU, 1780
POS_MAX_HEIGHT_STEADY, 1784
POS_MAX_HEIGHT_STEADY_SEQUENCE, 1788
POS_MAX_MAX_BUMP_ON_DECREASING_SEQUENCE, 1792
POS_MAX_MAX_DECREASING, 1796
POS_MAX_MAX_DECREASING_SEQUENCE, 1800
POS_MAX_MAX_DIP_ON_INCREASING_SEQUENCE, 1804
POS_MAX_MAX_INCREASING, 1808
POS_MAX_MAX_INCREASING_SEQUENCE, 1812
POS_MAX_MAX_INFLEXION, 1816
POS_MAX_MAX_PEAK, 1820
POS_MAX_MAX_STRICTLY_DECREASING_SEQUENCE, 1824
POS_MAX_MAX_STRICTLY_INCREASING_SEQUENCE, 1828
POS_MAX_MAX_SUMMIT, 1832
POS_MAX_MAX_ZIGZAG, 1836
POS_MAX_MIN_BUMP_ON_DECREASING_SEQUENCE, 1840
POS_MAX_MIN_DECREASING, 1844
POS_MAX_MIN_DECREASING_SEQUENCE, 1848
POS_MAX_MIN_DIP_ON_INCREASING_SEQUENCE, 1852
INDEX

POS_MAX_MIN_GORGE, 1856
POS_MAX_MIN_INCREASING, 1860
POS_MAX_MIN_INCREASING_SEQUENCE, 1864
POS_MAX_MIN_INFLEXION, 1868
POS_MAX_MIN_STRICTLY_DECREASING_SEQUENCE, 1872
POS_MAX_MIN_STRICTLY_INCREASING_SEQUENCE, 1876
POS_MAX_MIN_VALLEY, 1880
POS_MAX_MIN_ZIGZAG, 1884
POS_MAX_SURF_BUMP_ON_DECREASING_SEQUENCE, 1888
POS_MAX_SURF_DECREASING, 1892
POS_MAX_SURF_DECREASING_SEQUENCE, 1896
POS_MAX_SURF_DECREASING_TERRACE, 1900
POS_MAX_SURF_DIP_ON_INCREASING_SEQUENCE, 1904
POS_MAX_SURF_GORGE, 1908
POS_MAX_SURF_INCREASING, 1912
POS_MAX_SURF_INCREASING_SEQUENCE, 1916
POS_MAX_SURF_INCREASING_TERRACE, 1920
POS_MAX_SURF_INFLEXION, 1924
POS_MAX_SURF_PEAK, 1928
POS_MAX_SURFPLAIN, 1932
POS_MAX_SURF PLATEAU, 1936
POS_MAX_SURF PROPER_PLAIN, 1940
POS_MAX_SURF_PROPER_PLATEAU, 1944
POS_MAX_SURF_STEADY, 1948
POS_MAX_SURF_STEADY_SEQUENCE, 1952
POS_MAX_SURF_STRICTLY_DECREASING_SEQUENCE, 1956
POS_MAX_SURF_STRICTLY_INCREASING_SEQUENCE, 1960
POS_MAX_SURF_SUMMIT, 1964
POS_MAX_SURF_VALLEY, 1968
POS_MAX_SURF_ZIGZAG, 1972
POS_MAX_WIDTH_DECREASING_SEQUENCE, 1976
POS_MAX_WIDTH_DECREASING_TERRACE, 1980
POS_MAX_WIDTH_GORGE, 1984
POS_MAX_WIDTH_INCREASING_SEQUENCE, 1988
POS_MAX_WIDTH_INCREASING_TERRACE, 1992
POS_MAX_WIDTH_INFLEXION, 1996
POS_MAX_WIDTH_PEAK, 2000
POS_MAX_WIDTH_PLAIN, 2004
POS_MAX_WIDTH_PLATEAU, 2008
POS_MAX_WIDTH_PROPER_PLAIN, 2012
POS_MAX_WIDTH_PROPER_PLATEAU, 2016
POS_MAX_WIDTH_STEADY_SEQUENCE, 2020
POS_MAX_WIDTH_STRICTLY_DECREASING_SEQUENCE, 2024
POS_MAX_WIDTH_STRICTLY_INCREASING_SEQUENCE, 2028
POS_MAX_WIDTH_SUMMIT, 2032
POS_MAX_WIDTH_VALLEY, 2036
POS_MAX_WIDTH_ZIGZAG, 2040
POS_MIN_HEIGHT_DECREASING_TERRACE, 2044
POS_MIN_HEIGHT_INCREASING_TERRACE, 2048
POS_MIN_HEIGHT_PLAIN, 2052
INDEX

POS_MIN_SURF_ZIGZAG, 2256
POS_MIN_WIDTH_DECRESING_SEQUENCE, 2260
POS_MIN_WIDTH_DECRESING_TERRACE, 2264
POS_MIN_WIDTH_GORGE, 2268
POS_MIN_WIDTH_INCREASING_SEQUENCE, 2272
POS_MIN_WIDTH_INCREASING_TERRACE, 2276
POS_MIN_WIDTH_INFLEXION, 2280
POS_MIN_WIDTH_PEAK, 2284
POS_MIN_WIDTH_PLAIN, 2288
POS_MIN_WIDTH_PLATEAU, 2292
POS_MIN_WIDTH_PROPER_PLAIN, 2296
POS_MIN_WIDTH_PROPER_PLATEAU, 2300
POS_MIN_WIDTH_STEADY_SEQUENCE, 2304
POS_MIN_WIDTH_STRICTLY_DECRESING_SEQUENCE, 2308
POS_MIN_WIDTH_STRICTLY_INCREASING_SEQUENCE, 2312
POS_MIN_WIDTH_SUMMIT, 2316
POS_MIN_WIDTH_VALLEY, 2320
POS_MIN_WIDTH_ZIGZAG, 2324
SUM_HEIGHT_DECRESING_TERRACE, 2328
SUM_HEIGHT_INCREASING_TERRACE, 2334
SUM_HEIGHT_PLAIN, 2340
SUM_HEIGHT_PLATEAU, 2344
SUM_HEIGHT_PROPER_PLAIN, 2348
SUM_HEIGHT_PROPER_PLATEAU, 2352
SUM_HEIGHT_STEADY, 2356
SUM_HEIGHT_STEADY_SEQUENCE, 2360
SUM_MAX_BUMP_ON_DECRESING_SEQUENCE, 2364
SUM_MAX_DECRESING, 2368
SUM_MAX_DECRESING_SEQUENCE, 2372
SUM_MAX_DIP_ON_INCREASING_SEQUENCE, 2376
SUM_MAX_INCREASING, 2380
SUM_MAX_INCREASING_SEQUENCE, 2384
SUM_MAX_INFLEXION, 2388
SUM_MAX_PEAK, 2392
SUM_MAX_STRICTLY_DECRESING_SEQUENCE, 2396
SUM_MAX_STRICTLY_INCREASING_SEQUENCE, 2400
SUM_MAX_SUMMIT, 2404
SUM_MAX_ZIGZAG, 2408
SUM_MIN_BUMP_ON_DECRESING_SEQUENCE, 2414
SUM_MIN_DECRESING, 2418
SUM_MIN_DECRESING_SEQUENCE, 2422
SUM_MIN_DIP_ON_INCREASING_SEQUENCE, 2426
SUM_MIN_GORGE, 2430
SUM_MIN_INCREASING, 2434
SUM_MIN_INCREASING_SEQUENCE, 2438
SUM_MIN_INFLEXION, 2442
SUM_MIN_STRICTLY_DECRESING_SEQUENCE, 2446
SUM_MIN_STRICTLY_INCREASING_SEQUENCE, 2450
SUM_MIN_VALLEY, 2454
SUM_MIN_ZIGZAG, 2458
INDEX

SUM_RANGE_DECREASING, 2464
SUM_RANGE_DECREASING_SEQUENCE, 2468
SUM_RANGE_INCREASING, 2472
SUM_RANGE_INCREASING_SEQUENCE, 2476
SUM_RANGE_STRICTLY_DECREASING_SEQUENCE, 2480
SUM_RANGE_STRICTLY_INCREASING_SEQUENCE, 2484
SUM_SURF_BUMP_ON_DECREASING_SEQUENCE, 2488
SUM_SURF_DECREASING, 2492
SUM_SURF_DECREASING_SEQUENCE, 2496
SUM_SURF_DECREASING_TERRACE, 2500
SUM_SURF_DIP_ON_INCREASING_SEQUENCE, 2504
SUM_SURF_GORGE, 2508
SUM_SURF_INCREASING, 2512
SUM_SURF_INCREASING_SEQUENCE, 2516
SUM_SURF_INCREASING_TERRACE, 2520
SUM_SURF_INFLEXION, 2524
SUM_SURF_PEAK, 2528
SUM_SURF_PLAIN, 2532
SUM_SURF_PLATEAU, 2536
SUM_SURF_PROPER_PLAIN, 2540
SUM_SURF_PROPER_PLATEAU, 2544
SUM_SURF_STEADY, 2548
SUM_SURF_STEADY_SEQUENCE, 2552
SUM_SURF_STRICTLY_DECREASING_SEQUENCE, 2556
SUM_SURF_STRICTLY_INCREASING_SEQUENCE, 2560
SUM_SURF_SUMMIT, 2564
SUM_SURF_VALLEY, 2570
SUM_SURF_ZIGZAG, 2574
SUM_WIDTH_DECREASING_SEQUENCE, 2580
SUM_WIDTH_DECREASING_TERRACE, 2584
SUM_WIDTH_GORGE, 2588
SUM_WIDTH_INCREASING_SEQUENCE, 2594
SUM_WIDTH_INCREASING_TERRACE, 2598
SUM_WIDTH_INFLEXION, 2602
SUM_WIDTH_PEAK, 2606
SUM_WIDTH_PLAIN, 2610
SUM_WIDTH_PLATEAU, 2614
SUM_WIDTH_PROPER_PLAIN, 2618
SUM_WIDTH_PROPER_PLATEAU, 2622
SUM_WIDTH_STEADY_SEQUENCE, 2626
SUM_WIDTH_STRICTLY_DECREASING_SEQUENCE, 2630
SUM_WIDTH_STRICTLY_INCREASING_SEQUENCE, 2634
SUM_WIDTH_SUMMIT, 2638
SUM_WIDTH_VALLEY, 2644
SUM_WIDTH_ZIGZAG, 2648

sum, 5
SUM_HEIGHT_DECREASING_TERRACE, 2328
SUM_HEIGHT_INCREASING_TERRACE, 2334
SUM_HEIGHT_PLAIN, 2340
SUM_HEIGHT_PLATEAU, 2344
INDEX

SUM_HEIGHT_PROPER_PLAIN, 2348
SUM_HEIGHT_PROPER_PLATEAU, 2352
SUM_HEIGHT_STEADY, 2356
SUM_HEIGHT_STEADY_SEQUENCE, 2360
SUM_MAX_BUMP_ON_DECREASING_SEQUENCE, 2364
SUM_MAX_DECREASING, 2368
SUM_MAX_DECREASING_SEQUENCE, 2372
SUM_MAX_DIP_ON_INCREASING_SEQUENCE, 2376
SUM_MAX_INCREASING, 2380
SUM_MAX_INCREASING_SEQUENCE, 2384
SUM_MAX_INFLEXION, 2388
SUM_MAX_PEAK, 2392
SUM_MAX_STRICTLY_DECREASING_SEQUENCE, 2396
SUM_MAX_STRICTLY_INCREASING_SEQUENCE, 2400
SUM_MAX_SUMMIT, 2404
SUM_MAX_ZIGZAG, 2408
SUM_MIN_BUMP_ON_DECREASINGSEQUENCE, 2414
SUM_MIN_DECREASING, 2418
SUM_MIN_DECREASINGSEQUENCE, 2422
SUM_MIN_DIP_ON_INCREASINGSEQUENCE, 2426
SUM_MIN_GORGE, 2430
SUM_MIN_INCREASING, 2434
SUM_MIN_INCREASINGSEQUENCE, 2438
SUM_MIN_INFLEXION, 2442
SUM_MIN_STRICTLY_DECREASINGSEQUENCE, 2446
SUM_MIN_STRICTLY_INCREASINGSEQUENCE, 2450
SUM_MIN_VALLEY, 2454
SUM_MIN_ZIGZAG, 2458
SUM_RANGE_DECREASING, 2464
SUM_RANGE_DECREASINGSEQUENCE, 2468
SUM_RANGE_INCREASING, 2472
SUM_RANGE_INCREASINGSEQUENCE, 2476
SUM_RANGE_STRICTLY_DECREASINGSEQUENCE, 2480
SUM_RANGE_STRICTLY_INCREASINGSEQUENCE, 2484
SUM_SURF_BUMP_ON_DECREASINGSEQUENCE, 2488
SUM_SURF_DECREASING, 2492
SUM_SURF_DECREASINGSEQUENCE, 2496
SUM_SURF_DECREASING_TERRACE, 2500
SUM_SURF_DIP_ON_INCREASINGSEQUENCE, 2504
SUM_SURF_GORGE, 2508
SUM_SURF_INCREASING, 2512
SUM_SURF_INCREASINGSEQUENCE, 2516
SUM_SURF_INCREASING_TERRACE, 2520
SUM_SURF_INFLEXION, 2524
SUM_SURF_PEAK, 2528
SUM_SURF_PLAIN, 2532
SUM_SURF_PLATEAU, 2536
SUM_SURF_PROPER_PLAIN, 2540
SUM_SURF_PROPER_PLATEAU, 2544
SUM_SURF_STEADY, 2548
SUM_SURF_STEADY_SEQUENCE, 2552
SUM_SURF.Strictly_DECREASING_SEQUENCE, 2556
SUM_SURF.Strictly_INCREASING_SEQUENCE, 2560
SUM_SURF_SUMMIT, 2564
SUM_SURF_VALLEY, 2570
SUM_SURF_ZIGZAG, 2574
SUM_WIDTH_Decreasing_SEQUENCE, 2580
SUM_WIDTH_Decreasing_TERRACE, 2584
SUM_WIDTH_GORGE, 2588
SUM_WIDTH_Increasing_SEQUENCE, 2594
SUM_WIDTH_Increasing_TERRACE, 2598
SUM_WIDTH_INFLEXION, 2602
SUM_WIDTH_PEAK, 2606
SUM_WIDTH_PLAIN, 2610
SUM_WIDTH_PLATEAU, 2614
SUM_WIDTH_PROPER_PLAIN, 2618
SUM_WIDTH_PROPER_PLATEAU, 2622
SUM_WIDTH_STEADY_SEQUENCE, 2626
SUM_WIDTH_Strictly_Decreasing_SEQUENCE, 2630
SUM_WIDTH_Strictly_Increasing_SEQUENCE, 2634
SUM_WIDTH_SUMMIT, 2638
SUM_WIDTH_VALLEY, 2644
SUM_WIDTH_ZIGZAG, 2648