

## 5.67 cond\_lex\_greater

	DESCRIPTION	LINKS	AUTOMATON
<b>Origin</b>	Inspired by [398].		
<b>Constraint</b>	<code>cond_lex_greater(VECTOR1, VECTOR2, PREFERENCE_TABLE)</code>		
<b>Type</b>	TUPLE_OF_VALS : <code>collection(val-int)</code>		
<b>Arguments</b>	VECTOR1 : <code>collection(var-dvar)</code> VECTOR2 : <code>collection(var-dvar)</code> PREFERENCE_TABLE : <code>collection(tuple - TUPLE_OF_VALS)</code>		
<b>Restrictions</b>	<code>required(TUPLE_OF_VALS, val)</code> <code>required(VECTOR1, var)</code> <code>required(VECTOR2, var)</code> <code> VECTOR1  =  VECTOR2 </code> <code> VECTOR1  =  TUPLE_OF_VALS </code> <code>required(PREFERENCE_TABLE, tuple)</code> <code>same_size(PREFERENCE_TABLE, tuple)</code> <code>distinct(PREFERENCE_TABLE, [])</code> <code>in_relation(VECTOR1, PREFERENCE_TABLE)</code> <code>in_relation(VECTOR2, PREFERENCE_TABLE)</code>		
<b>Purpose</b>	<div style="border: 1px solid pink; padding: 5px;">           VECTOR1 and VECTOR2 are both assigned to the <math>I^{th}</math> and <math>J^{th}</math> items of the collection PREFERENCE_TABLE such that <math>I &gt; J</math>.         </div>		
<b>Example</b>	<div style="border: 1px solid blue; padding: 10px; display: inline-block;"> <math display="block">\left( \begin{array}{l} \langle 0, 0 \rangle, \\ \langle 1, 0 \rangle, \\ \text{tuple} - \langle 1, 0 \rangle, \\ \langle \text{tuple} - \langle 0, 1 \rangle, \\ \text{tuple} - \langle 0, 0 \rangle, \rangle \\ \text{tuple} - \langle 1, 1 \rangle \end{array} \right)</math> </div> <p>The <code>cond_lex_greater</code> constraint holds since VECTOR1 and VECTOR2 are respectively assigned to the third and first items of the collection PREFERENCE_TABLE.</p>		
<b>Typical</b>	<code> TUPLE_OF_VALS  &gt; 1</code> <code> VECTOR1  &gt; 1</code> <code> VECTOR2  &gt; 1</code> <code> PREFERENCE_TABLE  &gt; 1</code>		

**Symmetries**

- Items of VECTOR1, VECTOR2 and PREFERENCE\_TABLE.tuple are [permutable](#) (*same permutation used*).
- All occurrences of two distinct tuples of values in VECTOR1, VECTOR2 or PREFERENCE\_TABLE.tuple can be [swapped](#); all occurrences of a tuple of values in VECTOR1, VECTOR2 or PREFERENCE\_TABLE.tuple can be [renamed](#) to any unused tuple of values.

**Usage**

See [cond\\_lex\\_cost](#).

**See also**

**common keyword:** [cond\\_lex\\_cost](#), [cond\\_lex\\_greatereq](#), [cond\\_lex\\_less](#), [cond\\_lex\\_lesseq](#) (*preferences*), [lex\\_greater](#) (*lexicographic order*).

**implies:** [cond\\_lex\\_greatereq](#).

**Keywords**

**characteristic of a constraint:** [vector](#), [automaton](#), [automaton without counters](#), [reified automaton constraint](#).

**constraint network structure:** [Berge-acyclic constraint network](#).

**constraint type:** [order constraint](#).

**filtering:** [arc-consistency](#).

**modelling:** [preferences](#).

**symmetry:** [lexicographic order](#).

**Automaton**

Figure 5.139 depicts the automaton associated with the preference table of the `cond_lex_greater` constraint given in the example. Let  $VAR1_k$  and  $VAR2_k$  respectively be the var attributes of the  $k^{th}$  items of the `VECTOR1` and the `VECTOR2` collections. Figure 5.140 depicts the reformulation of the `cond_lex_greater` constraint. This reformulation uses:

- Two occurrences of the automaton depicted by Figure 5.139 for computing the positions I and J within the preference table corresponding to `VECTOR1` and `VECTOR2`.
- The binary constraint  $I > J$ .

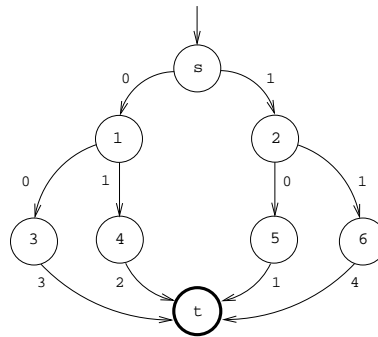


Figure 5.139: Automaton associated with the preference table of the `cond_lex_greater` constraint given in the example

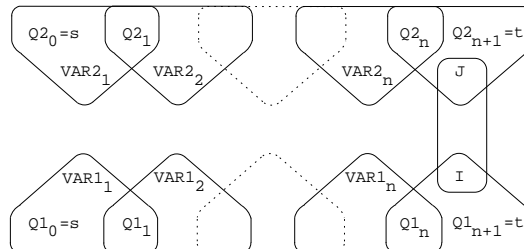


Figure 5.140: Hypergraph of the reformulation corresponding to the `cond_lex_greater` constraint: it uses two occurrences of the automaton of Figure 5.139 and the constraint  $I > J$

20060430

661