

5.151 `imply`

	DESCRIPTION	LINKS	AUTOMATON
Origin	Logic		
Constraint	<code>imply(VAR, VARIABLES)</code>		
Synonyms	<code>rel</code> , <code>ifthen</code> .		
Arguments	VAR : <code>dvar</code> VARIABLES : <code>collection(var-dvar)</code>		
Restrictions	$VAR \geq 0$ $VAR \leq 1$ $ VARIABLES = 2$ <code>required(VARIABLES, var)</code> $VARIABLES.var \geq 0$ $VARIABLES.var \leq 1$		
Purpose	Let VARIABLES be a collection of 0-1 variables VAR_1, VAR_2 . Enforce $VAR = (VAR_1 \Rightarrow VAR_2)$.		
Example	<pre>(1, <0, 0>) (1, <0, 1>) (0, <1, 0>) (1, <1, 1>)</pre>		
Symmetry	All occurrences of 0 in VAR and in VARIABLES.var can be set to 1.		
Systems	<code>reifiedLeftImp</code> in Choco , <code>rel</code> in Gecode , <code>ifthenbool</code> in JaCoP , <code>#=></code> in SICStus .		
See also	common keyword: <code>and</code> , <code>equivalent</code> , <code>nand</code> , <code>nor</code> , <code>or</code> , <code>xor</code> (<i>Boolean constraint</i>).		
Keywords	characteristic of a constraint: <code>automaton</code> , <code>automaton without counters</code> , <code>reified automaton constraint</code> . constraint network structure: Berge-acyclic constraint network. constraint type: Boolean constraint. filtering: arc-consistency.		

Automaton

Figure 5.305 depicts the automaton associated with the `imply` constraint. To the first argument `VAR` of the `imply` constraint corresponds the first signature variable. To each variable VAR_i of the second argument `VARIABLES` of the `imply` constraint corresponds the next signature variable. There is no signature constraint.

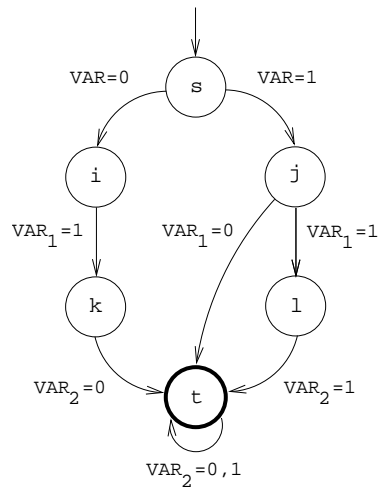


Figure 5.305: Automaton of the `imply` constraint

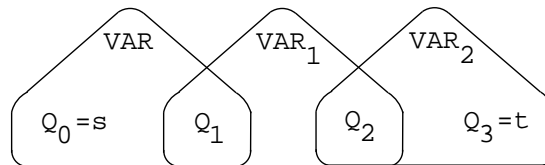


Figure 5.306: Hypergraph of the reformulation corresponding to the automaton of the `imply` constraint