

5.410 two_orth_do_not_overlap

	DESCRIPTION	LINKS	GRAPH	AUTOMATON
Origin	Used for defining <code>diffn</code> .			
Constraint	<code>two_orth_do_not_overlap(ORTHOTOPE1, ORTHOTOPE2)</code>			
Type	ORTHOTOPE : <code>collection(ori-dvar, siz-dvar, end-dvar)</code>			
Arguments	ORTHOTOPE1 : ORTHOTOPE ORTHOTOPE2 : ORTHOTOPE			
Restrictions	$ ORTHOTOPE > 0$ <code>require_at_least(2, ORTHOTOPE, [ori, siz, end])</code> $ORTHOTOPE.siz \geq 0$ $ORTHOTOPE.ori \leq ORTHOTOPE.end$ $ ORTHOTOPE1 = ORTHOTOPE2 $ <code>orth_link_ori_siz_end(ORTHOTOPE1)</code> <code>orth_link_ori_siz_end(ORTHOTOPE2)</code>			
Purpose	For two orthotopes O_1 and O_2 enforce that there exists at least one dimension i such that the projections on i of O_1 and O_2 do not overlap.			
Example	$\left(\left(\langle \mathbf{ori} - 2 \ \mathbf{siz} - 2 \ \mathbf{end} - 4, \mathbf{ori} - 1 \ \mathbf{siz} - 3 \ \mathbf{end} - 4 \rangle, \right) \right. \\ \left. \left(\langle \mathbf{ori} - 4 \ \mathbf{siz} - 4 \ \mathbf{end} - 8, \mathbf{ori} - 3 \ \mathbf{siz} - 3 \ \mathbf{end} - 6 \rangle \right) \right)$ <p>Figure 5.777 represents the respective position of the two rectangles of the example. The coordinates of the leftmost lowest corner of each rectangle are stressed in bold. The <code>two_orth_do_not_overlap</code> constraint holds since the two rectangles do not overlap.</p>			
Typical	$ ORTHOTOPE > 1$			
Symmetries	<ul style="list-style-type: none"> Arguments are <code>permutable</code> w.r.t. permutation (ORTHOTOPE1, ORTHOTOPE2). Items of ORTHOTOPE1 and ORTHOTOPE2 are <code>permutable</code> (<i>same permutation used</i>). ORTHOTOPE1.siz can be <code>decreased</code> to any value ≥ 0. ORTHOTOPE2.siz can be <code>decreased</code> to any value ≥ 0. 			
Used in	<code>diffn</code> .			
See also	implied by: <code>two_orth_are_in_contact</code> .			
Keywords	characteristic of a constraint: automaton, automaton without counters, reified automaton constraint. constraint network structure: Berge-acyclic constraint network.			

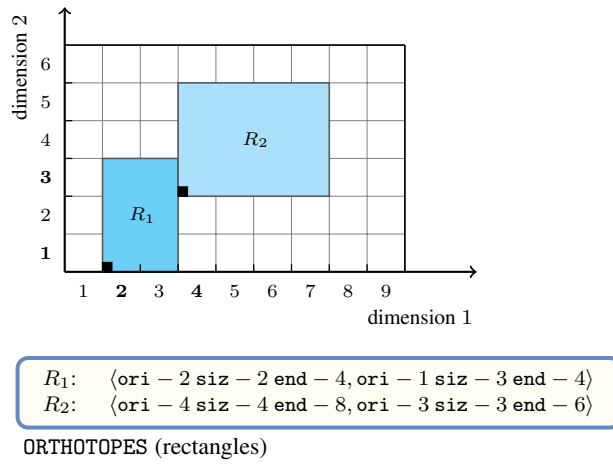


Figure 5.777: The two non overlapping rectangles of the **Example** slot

constraint type: logic.

filtering: arc-consistency, constructive disjunction.

final graph structure: bipartite, no loop.

geometry: geometrical constraint, non-overlapping, orthotope.

Arc input(s)	ORTHOTOPE1 ORTHOTOPE2
Arc generator	$\text{SYMMETRIC_PRODUCT}(=) \mapsto \text{collection}(\text{orthotope1}, \text{orthotope2})$
Arc arity	2
Arc constraint(s)	$\text{orthotope1.end} \leq \text{orthotope2.ori} \vee \text{orthotope1.siz} = 0$
Graph property(ies)	$\text{NARC} \geq 1$
Graph class	<ul style="list-style-type: none"> • BIPARTITE • NO_LOOP

Graph model

We build an initial graph where each arc corresponds to the fact that, either the projection of an **orthotope** on a given dimension is empty, either it is located before the projection in the same dimension of the other **orthotope**. Finally we ask that at least one arc constraint remains in the final graph.

Parts (A) and (B) of Figure 5.778 respectively show the initial and final graph associated with the **Example** slot. Since we use the **NARC** graph property, the unique arc of the final graph is stressed in bold. It corresponds to the fact that the projection in dimension 1 of the first **orthotope** is located before the projection in dimension 1 of the second **orthotope**. Therefore the two **orthotopes** do not overlap.

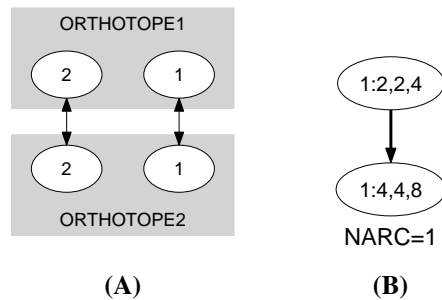


Figure 5.778: Initial and final graph of the `two_orth_do_not_overlap` constraint

Automaton

Figure 5.779 depicts the automaton associated with the `two_orth_do_not_overlap` constraint. Let $ORI1_i$, $SIZ1_i$ and $END1_i$ respectively be the `ori`, the `siz` and the `end` attributes of the i^{th} item of the `ORTHOTOPE1` collection. Let $ORI2_i$, $SIZ2_i$ and $END2_i$ respectively be the `ori`, the `siz` and the `end` attributes of the i^{th} item of the `ORTHOTOPE2` collection. To each sextuple $(ORI1_i, SIZ1_i, END1_i, ORI2_i, SIZ2_i, END2_i)$ corresponds a 0-1 signature variable S_i as well as the following signature constraint: $((SIZ1_i > 0) \wedge (SIZ2_i > 0) \wedge (END1_i > ORI2_i) \wedge (END2_i > ORI1_i)) \Leftrightarrow S_i$.

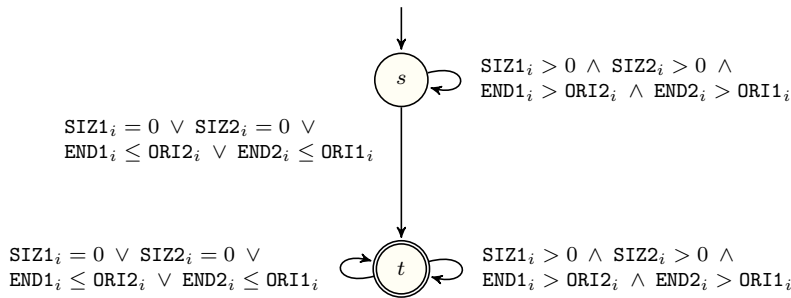


Figure 5.779: Automaton of the `two_orth_do_not_overlap` constraint

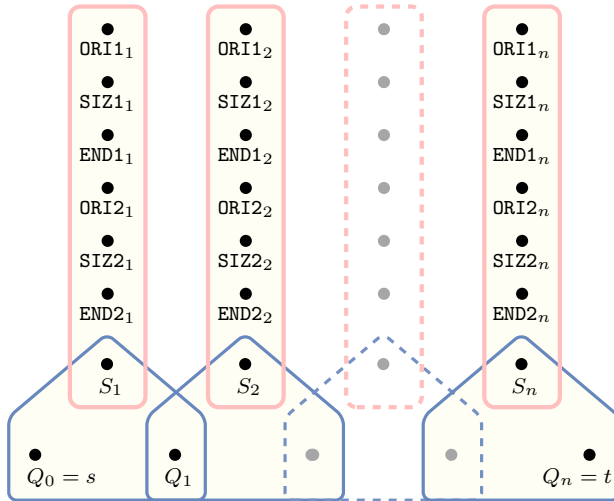


Figure 5.780: Hypergraph of the reformulation corresponding to the automaton of the `two_orth_do_not_overlap` constraint