5.112 dom_reachability

DESCRIPTION LINKS

Origin [297]

Constraint

dom_reachability(SOURCE, FLOW_GRAPH, DOMINATOR_GRAPH, TRANSITIVE_CLOSURE_GRAPH)

Arguments

SOURCE	:	int
FLOW_GRAPH	:	collection(index−int, succ−svar)
DOMINATOR_GRAPH	:	collection(index−int, succ−sint)
TRANSITIVE_CLOSURE_GRAPH	:	collection(index−int, succ−svar)

Restrictions

SOURCE ≥ 1
SOURCE ≤ |FLOW_GRAPH|
required(FLOW_GRAPH, [index, succ])
FLOW_GRAPH.index ≥ 1
FLOW_GRAPH.index ≤ |FLOW_GRAPH|
FLOW_GRAPH.succ ≥ 1
FLOW_GRAPH.succ ≤ |FLOW_GRAPH|
distinct(FLOW_GRAPH, index)
required(DOMINATOR_GRAPH, [index, succ])
|DOMINATOR_GRAPH| = |FLOW_GRAPH|
DOMINATOR_GRAPH.index ≥ 1
DOMINATOR_GRAPH.index ≤ |DOMINATOR_GRAPH|
DOMINATOR_GRAPH.succ ≥ 1
DOMINATOR_GRAPH.succ ≤ |DOMINATOR_GRAPH|
distinct(DOMINATOR_GRAPH, index)
required(TRANSITIVE_CLOSURE_GRAPH, [index, succ])
|TRANSITIVE_CLOSURE_GRAPH| = |FLOW_GRAPH|
TRANSITIVE_CLOSURE_GRAPH.index ≥ 1
TRANSITIVE_CLOSURE_GRAPH.index ≤ |TRANSITIVE_CLOSURE_GRAPH|
TRANSITIVE_CLOSURE_GRAPH.succ ≥ 1
TRANSITIVE_CLOSURE_GRAPH.succ ≤ |TRANSITIVE_CLOSURE_GRAPH|
distinct(TRANSITIVE_CLOSURE_GRAPH, index)

Purpose

Let FLOW_GRAPH, DOMINATOR_GRAPH and TRANSITIVE_CLOSURE_GRAPH be three directed graphs respectively called the *flow graph*, the *dominance graph* and the *transitive closure graph* which all have the same vertices. In addition let SOURCE denote a vertex of the flow graph called the *source node* (not necessarily a vertex with no incoming arcs). The dom_reachability constraint holds if and only if the flow graph (and its source node) verifies:

- The dominance relation expressed by the dominance graph (i.e., if there is an arc (i, j) in the dominance graph then, within the flow graph, all the paths from the source node to j contain i; note that when there is no path from the source node to j then any node dominates j).
- The transitive relation expressed by the transitive closure graph (i.e., if there is an arc (i, j) in the transitive closure graph then there is also a path from i to j in the flow graph).

Example

$$\left(\begin{array}{l}1, \left\langle\begin{array}{ll}\texttt{index}-1 & \texttt{succ}-\{2\},\\ \texttt{index}-2 & \texttt{succ}-\{3,4\},\\ \texttt{index}-3 & \texttt{succ}-\emptyset,\\ \texttt{index}-4 & \texttt{succ}-\emptyset\end{array}\right\rangle,\\ \left\langle\begin{array}{ll}\texttt{index}-1 & \texttt{succ}-\{2,3,4\},\\ \texttt{index}-2 & \texttt{succ}-\{3,4\},\\ \texttt{index}-3 & \texttt{succ}-\emptyset,\\ \texttt{index}-4 & \texttt{succ}-\emptyset\end{array}\right\rangle,\\ \left\langle\begin{array}{ll}\texttt{index}-1 & \texttt{succ}-\{1,2,3,4\},\\ \texttt{index}-2 & \texttt{succ}-\{2,3,4\},\\ \texttt{index}-3 & \texttt{succ}-\{3\},\\ \texttt{index}-4 & \texttt{succ}-\{4\}\end{array}\right\rangle\end{array}\right)$$

The flow graph, the dominance graph and the transitive closure graph corresponding to the second, third and fourth arguments of the dom_reachability constraint are respectively depicted by parts (A), (B) and (C) of Figure 5.226. The dom_reachability holds since the following conditions hold.

- The dominance relation expressed by the dominance graph is verified:
  - Since $(1, 2)$ belongs to the dominance graph all the paths from 1 to 2 in the flow graph pass through 1.
  - Since $(1, 3)$ belongs to the dominance graph all the paths from 1 to 3 in the flow graph pass through 1.
  - Since $(1, 4)$ belongs to the dominance graph all the paths from 1 to 4 in the flow graph pass through 1.
  - Since $(2, 3)$ belongs to the dominance graph all the paths from 1 to 3 in the flow graph pass through 2.
  - Since $(2, 4)$ belongs to the dominance graph all the paths from 1 to 4 in the flow graph pass through 2.
- The graph depicted by the fourth argument of the dom_reachability constraint (i.e., TRANSITIVE_CLOSURE_GRAPH) is the transitive closure of the graph depicted by the second argument (i.e., FLOW_GRAPH).

Typical $|\text{FLOW\_GRAPH}| > 2$

Symmetries

- Items of FLOW_GRAPH are permutable.
- Items of DOMINATOR_GRAPH are permutable.
- Items of TRANSITIVE_CLOSURE_GRAPH are permutable.

Usage The dom_reachability constraint was introduced in order to solve reachability problems (e.g., disjoint paths, simple path with mandatory nodes).

Remark Within the name dom_reachability, dom stands for *domination*. In the context of path problems SOURCE refers to the start of the path we want to build.

Algorithm It was shown in [295] that, finding out wether a dom_reachability constraint has a solution or not is NP-hard. This was achieved by reduction to *disjoint paths* problem [162].

The first implementation [296] of the dom_reachability constraint was done in **Mozart** [109]. Later on, a second implemention [295] was done in **Gecode** [340]. Both implementations consist of the following two parts:

- Algorithms [329] for maintaining the lower bound of the transitive closure graph.
- Algorithms for maintaining the upper bound of the transitive closure graph, while respecting the dominance constraints [170].

See also **common keyword:** path, path_from_to *(path)*.

Keywords **combinatorial object:** path.

constraint arguments: constraint involving set variables.

constraint type: predefined constraint, graph constraint.

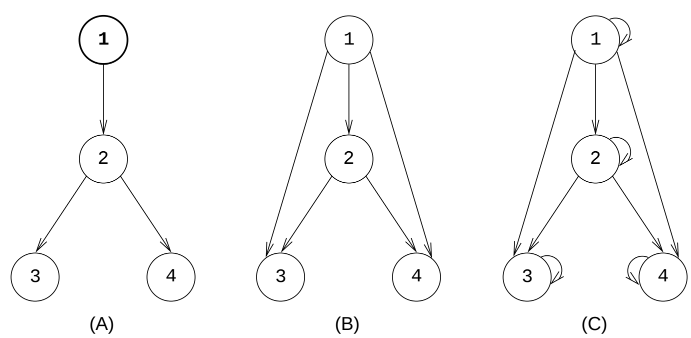

Figure 5.226: (A) Flow graph, (B) dominance graph and (C) transitive closure graph of the **Example** slot (taken from [295, page 40])