5.405 tree_resource

DESCRIPTION LINKS GRAPH

Origin Derived from tree.

Constraint tree_resource(RESOURCE, TASK)

Arguments

RESOURCE	:	collection(id−int, nb_task−dvar)
TASK	:	collection(id−int, father−dvar, resource−dvar)

Restrictions

|RESOURCE| > 0
required(RESOURCE, [id, nb_task])
RESOURCE.id ≥ 1
RESOURCE.id ≤ |RESOURCE|
distinct(RESOURCE, id)
RESOURCE.nb_task ≥ 0
RESOURCE.nb_task ≤ |TASK|
required(TASK, [id, father, resource])
TASK.id > |RESOURCE|
TASK.id ≤ |RESOURCE| + |TASK|
distinct(TASK, id)
TASK.father ≥ 1
TASK.father ≤ |RESOURCE| + |TASK|
TASK.resource ≥ 1
TASK.resource ≤ |RESOURCE|

Purpose Cover a digraph G in such a way that each vertex belongs to one distinct tree. Each tree is made up from one *resource* vertex and several *task* vertices. The resource vertices correspond to the roots of the different trees. For each resource a domain variable nb_task indicates how many task-vertices belong to the corresponding tree. For each task a domain variable resource gives the identifier of the resource that can handle the task.

Example

$$\left(\begin{array}{l} \left\langle \begin{array}{lll} \mathtt{id}-1 & \mathtt{nb\_task}-4, \\ \mathtt{id}-2 & \mathtt{nb\_task}-0, \\ \mathtt{id}-3 & \mathtt{nb\_task}-1 \end{array} \right\rangle, \\ \begin{array}{llll} \mathtt{id}-4 & \mathtt{father}-8 & \mathtt{resource}-1, \end{array} \\ \left\langle \begin{array}{llll} \mathtt{id}-5 & \mathtt{father}-3 & \mathtt{resource}-3, \\ \mathtt{id}-6 & \mathtt{father}-8 & \mathtt{resource}-1, \\ \mathtt{id}-7 & \mathtt{father}-1 & \mathtt{resource}-1, \end{array} \right\rangle \\ \begin{array}{llll} \mathtt{id}-8 & \mathtt{father}-1 & \mathtt{resource}-1 \end{array} \end{array}\right)$$

The tree_resource constraint holds since the graph associated with the items of the RESOURCE and the TASK collections corresponds to 3 trees (i.e., |RESOURCE| = 3): each tree respectively involves the vertices $\{1, 4, 6, 7, 8\}$, $\{2\}$ and $\{3, 5\}$. They are depicted by Figure 5.768, where *resource* and *task* vertices are respectively coloured in blue and pink.

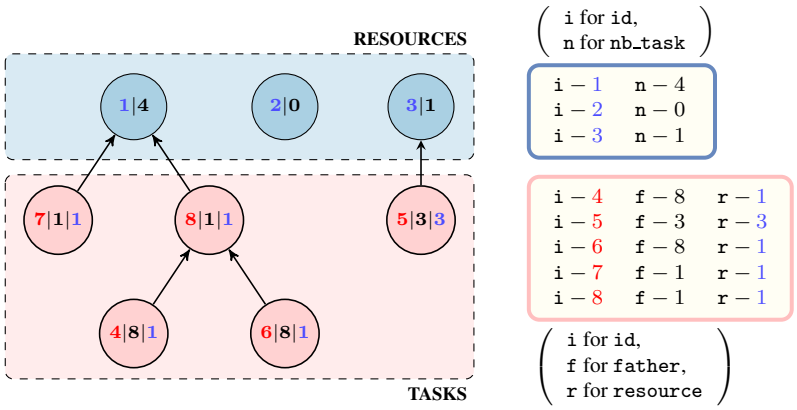

Figure 5.768: The three trees corresponding to the **Example** slot; each resource vertex (in blue) contains the information id|nb_task where nb_task is the number of tasks in the tree, while each task vertex (in pink) contains the information id|father|resource where father is the index of its father in the tree and resource is the index of the corresponding root task in the tree.

Typical

$|\texttt{RESOURCE}| > 0$
$|\texttt{TASK}| > |\texttt{RESOURCE}|$

Symmetries

- Items of RESOURCE are permutable.
- Items of TASK are permutable.

Reformulation

The tree_resource(RESOURCE, TASK) constraint can be expressed in term of a conjunction of one tree constraint, |TASK| element constraints and one global_cardinality constraint:

- The tree constraint expresses the fact that we have a well formed tree.
- The element constraint is used for expressing the link between the father attribute of an item of the TASK collection and its corresponding resource attribute.
- The global_cardinality constraint is used to link the resource attribute of the items of the TASK collection with the nb_task attribute of the items of the RESOURCE collection.

With respect to the **Example** slot we get the following conjunction of constraints:

$\texttt{tree}(3, \langle \texttt{index} - 1\ \texttt{succ} - 1,$
$\texttt{index} - 2\ \texttt{succ} - 2,$
$\texttt{index} - 3\ \texttt{succ} - 3,$
$\texttt{index} - 4\ \texttt{succ} - 8,$
$\texttt{index} - 5\ \texttt{succ} - 3,$
$\texttt{index} - 6\ \texttt{succ} - 8,$
$\texttt{index} - 7\ \texttt{succ} - 1,$
$\texttt{index} - 8\ \texttt{succ} - 1\rangle),$

$$\begin{array}{l}
\texttt{element}(8, \langle 1,2,3,1,3,1,1,1\rangle, 1),\\
\texttt{element}(3, \langle 1,2,3,1,3,1,1,1\rangle, 3),\\
\texttt{element}(8, \langle 1,2,3,1,3,1,1,1\rangle, 1),\\
\texttt{element}(1, \langle 1,2,3,1,3,1,1,1\rangle, 1),\\
\texttt{element}(1, \langle 1,2,3,1,3,1,1,1\rangle, 1),\\
\texttt{global_cardinality}(\langle 1,3,1,1,1\rangle,\\
\qquad\qquad\qquad\qquad \langle \texttt{val} - 1\, \texttt{noccurrence} - 4,\\
\qquad\qquad\qquad\qquad\ \ \texttt{val} - 2\, \texttt{noccurrence} - 0,\\
\qquad\qquad\qquad\qquad\ \ \texttt{val} - 3\, \texttt{noccurrence} - 1\rangle).
\end{array}$$

See also **root concept:** `tree`.

used in reformulation: `element`, `global_cardinality`, `tree`.

Keywords **characteristic of a constraint:** derived collection.

constraint type: graph constraint, resource constraint, graph partitioning constraint.

final graph structure: tree, connected component.

Derived Collection

$$\mathtt{col}\left(\begin{array}{l}\mathtt{RESOURCE\_TASK}-\mathtt{collection}\left(\begin{array}{l}\mathtt{index}-\mathtt{int},\\ \mathtt{succ}-\mathtt{dvar},\\ \mathtt{name}-\mathtt{dvar}\end{array}\right),\\ \left[\begin{array}{l}\mathtt{item}\left(\begin{array}{l}\mathtt{index}-\mathtt{RESOURCE.id},\\ \mathtt{succ}-\mathtt{RESOURCE.id},\\ \mathtt{name}-\mathtt{RESOURCE.id}\end{array}\right),\\ \mathtt{item}\left(\begin{array}{l}\mathtt{index}-\mathtt{TASK.id},\\ \mathtt{succ}-\mathtt{TASK.father},\\ \mathtt{name}-\mathtt{TASK.resource}\end{array}\right)\end{array}\right]\end{array}\right)$$

Arc input(s)	RESOURCE_TASK
Arc generator	$CLIQUE \mapsto \mathtt{collection}(\mathtt{resource\_task1}, \mathtt{resource\_task2})$
Arc arity	2
Arc constraint(s)	• resource_task1.succ = resource_task2.index • resource_task1.name = resource_task2.name
Graph property(ies)	• **MAX_NSCC** ≤ 1 • **NCC** $= \lvert\mathtt{RESOURCE}\rvert$ • **NVERTEX** $= \lvert\mathtt{RESOURCE}\rvert + \lvert\mathtt{TASK}\rvert$

For all items of RESOURCE:

Arc input(s)	RESOURCE_TASK
Arc generator	$CLIQUE \mapsto \mathtt{collection}(\mathtt{resource\_task1}, \mathtt{resource\_task2})$
Arc arity	2
Arc constraint(s)	• resource_task1.succ = resource_task2.index • resource_task1.name = resource_task2.name • resource_task1.name = RESOURCE.id
Graph property(ies)	**NVERTEX** $= \mathtt{RESOURCE.nb\_task} + 1$

Graph model For the second graph constraint, part (A) of Figure 5.769 shows the initial graphs associated with resources 1, 2 and 3 of the **Example** slot. For the second graph constraint, part (B) of Figure 5.769 shows the corresponding final graphs associated with resources 1, 2 and 3. Since we use the **NVERTEX** graph property, the vertices of the final graphs are stressed in bold. To each resource corresponds a tree of respectively 4, 0 and 1 task-vertices.

Signature Since the initial graph of the first graph constraint contains $\lvert\mathtt{RESOURCE}\rvert + \lvert\mathtt{TASK}\rvert$ vertices, the corresponding final graph cannot have more than $\lvert\mathtt{RESOURCE}\rvert + \lvert\mathtt{TASK}\rvert$ vertices. Therefore we can rewrite the graph property $\mathbf{NVERTEX} = \lvert\mathtt{RESOURCE}\rvert + \lvert\mathtt{TASK}\rvert$ to $\mathbf{NVERTEX} \geq \lvert\mathtt{RESOURCE}\rvert + \lvert\mathtt{TASK}\rvert$ and simplify $\underline{\mathbf{NVERTEX}}$ to $\mathbf{NVERTEX}$.

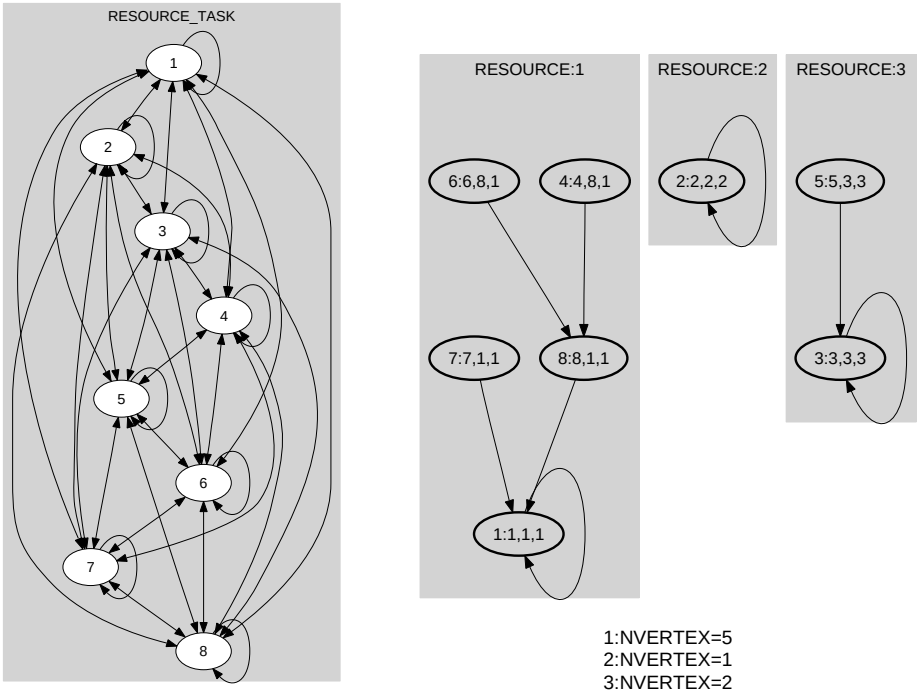

Figure 5.769: Initial and final graph of the `tree_resource` constraint