5.327 precedence

DESCRIPTION LINKS GRAPH

Origin Scheduling

Constraint precedence(TASKS)

Argument TASKS : collection(origin−dvar, duration−dvar)

Restrictions required(TASKS, [origin, duration])
TASKS.duration ≥ 0

Purpose All consecutive pairs of tasks of the collection TASKS should be ordered (i.e., the end of the first task of a pair should be less than or equal to the start of the second task of the same pair).

Example

$$\left(\left\langle \begin{array}{ll} \texttt{origin} - 1 & \texttt{duration} - 3, \\ \texttt{origin} - 4 & \texttt{duration} - 0, \\ \texttt{origin} - 5 & \texttt{duration} - 2, \\ \texttt{origin} - 8 & \texttt{duration} - 1 \end{array} \right\rangle \right)$$

Since the tasks are ordered (i.e., $1 + 3 \leq 4$, $4 + 0 \leq 5$, $5 + 2 \leq 8$) the precedence constraint holds.

Typical |TASKS| > 2
TASKS.duration ≥ 1

Symmetries

- TASKS.duration can be decreased to any value ≥ 0.
- One and the same constant can be added to the origin attribute of all items of TASKS.

Arg. properties

Contractible wrt. TASKS.

See also

common keyword: increasing *(order constraint)*.

implies: disjunctive.

implies (items to collection): lex_chain_lesseq.

Keywords

constraint type: decomposition, order constraint.

filtering: arc-consistency.

Arc input(s)	TASKS
Arc generator	$PATH \mapsto$ collection(tasks1, tasks2)
Arc arity	2
Arc constraint(s)	tasks1.origin + tasks1.duration $\leq$ tasks2.origin
Graph property(ies)	**NARC**$= \lvert$TASKS$\rvert - 1$

Graph model Since we are only interested by the constraints linking two consecutive items of the collection TASKS we use *PATH* to generate the arcs of the initial graph.

Parts (A) and (B) of Figure 5.675 respectively show the initial and final graph of the first example of the **Example** slot. Since we use the **NARC** graph property, the arcs of the final graph are stressed in bold.

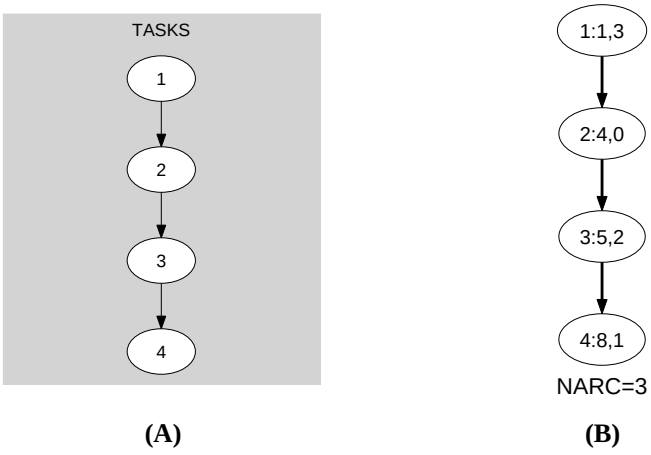

Figure 5.675: Initial and final graph of the precedence constraint