5.127 disjunctive_or_same_end

DESCRIPTION LINKS GRAPH

Origin Scheduling.

Constraint `disjunctive_or_same_end(TASKS)`

Synonyms `same_end_or_disjunctive`, `non_overlap_or_same_end`, `same_end_or_non_overlap`.

Argument $\texttt{TASKS} : \texttt{collection}(\texttt{origin}-\texttt{dvar}, \texttt{duration}-\texttt{dvar})$

Restrictions $\texttt{required}(\texttt{TASKS}, [\texttt{origin}, \texttt{duration}])$
$\texttt{TASKS.duration} \geq 0$

Purpose All pairs of tasks of the collection TASKS that have a duration strictly greater than 0 should either not overlap either have the same end, i.e. $\forall i \in [1, |\texttt{TASKS}|], \forall j \in [i+1, |\texttt{TASKS}|] : \texttt{TASKS}[i].\texttt{duration} = 0 \vee \texttt{TASKS}[j].\texttt{duration} = 0 \vee \texttt{TASKS}[i].\texttt{origin} + \texttt{TASKS}[i].\texttt{duration} \leq \texttt{TASKS}[j].\texttt{origin} \vee \texttt{TASKS}[j].\texttt{origin} + \texttt{TASKS}[j].\texttt{duration} \leq \texttt{TASKS}[i].\texttt{origin} \vee \texttt{TASKS}[i].\texttt{origin} + \texttt{TASKS}[i].\texttt{duration} = \texttt{TASKS}[j].\texttt{origin} + \texttt{TASKS}[j].\texttt{duration}$.

Example

$$\left(\left\langle\begin{array}{ll}\texttt{origin}-4 & \texttt{duration}-3,\\ \texttt{origin}-7 & \texttt{duration}-2,\\ \texttt{origin}-5 & \texttt{duration}-2\end{array}\right\rangle\right)$$

Since the ends of the first and third tasks coincide, and since the second task does neither overlap the first task nor the third task, the `disjunctive_or_same_end` constraint holds.

Typical $|\texttt{TASKS}| > 2$
$\texttt{TASKS.duration} \geq 1$

Symmetries

- Items of TASKS are permutable.
- 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:** `disjunctive`, `disjunctive_or_same_start` (*scheduling constraint*).
implied by: `disjunctive`.

Keywords **constraint type:** scheduling constraint, resource constraint, decomposition.
modelling: disjunction, zero-duration task.

Arc input(s)	`TASKS`
Arc generator	$\mathit{CLIQUE}(<) \mapsto \texttt{collection}(\texttt{tasks1}, \texttt{tasks2})$
Arc arity	2
Arc constraint(s)	$\bigvee \left(\begin{array}{l} \texttt{tasks1.duration} = 0, \\ \texttt{tasks2.duration} = 0, \\ \texttt{tasks1.origin} + \texttt{tasks1.duration} \leq \texttt{tasks2.origin}, \\ \texttt{tasks2.origin} + \texttt{tasks2.duration} \leq \texttt{tasks1.origin}, \\ \quad \texttt{tasks1.origin} + \texttt{tasks1.duration} = \\ \quad \texttt{tasks2.origin} + \texttt{tasks2.duration} \end{array} \right)$
Graph property(ies)	$\mathbf{NARC} = \lvert\texttt{TASKS}\rvert * (\lvert\texttt{TASKS}\rvert - 1)/2$

Graph model We generate a *clique* with a non-overlapping constraint or a same end constraint between each pair of distinct tasks and state that the number of arcs of the final graph should be equal to the number of arcs of the initial graph.

Parts (A) and (B) of Figure 5.284 respectively show the initial and final graph associated with the **Example** slot. The `disjunctive_or_same_end` constraint holds since all the arcs of the initial graph belong to the final graph.

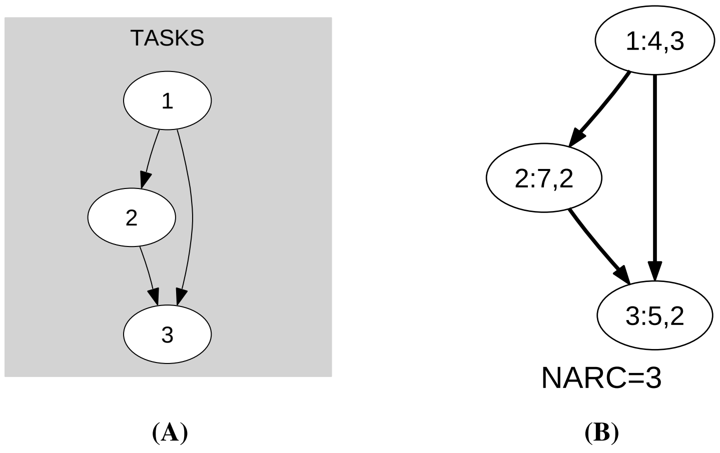

Figure 5.284: Initial and final graph of the `disjunctive_or_same_end` constraint