5.74 coloured_cumulatives

DESCRIPTION LINKS GRAPH

Origin Derived from cumulatives and nvalues.

Constraint coloured_cumulatives(TASKS, MACHINES)

Synonym colored_cumulatives.

Arguments

TASKS	:	collection	(machine−dvar, origin−dvar, duration−dvar, end−dvar, colour−dvar)
MACHINES	:	collection	(id−int, capacity−int)

Restrictions

required(TASKS, [machine, colour])
require_at_least(2, TASKS, [origin, duration, end])
TASKS.duration ≥ 0
TASKS.origin ≤ TASKS.end
required(MACHINES, [id, capacity])
distinct(MACHINES, id)
MACHINES.capacity ≥ 0

Purpose Consider a set $\mathcal{T}$ of tasks described by the TASKS collection. The coloured_cumulatives constraint forces for each machine m of the MACHINES collection the following condition: at each point in time p, the numbers of distinct colours of the set of tasks that both overlap that point p and are assigned to machine m does not exceed the capacity of machine m. A task overlaps a point i if and only if (1) its origin is less than or equal to i, and (2) its end is strictly greater than i. It also imposes for each task of $\mathcal{T}$ the constraint origin + duration = end.

Example

$$\left(\begin{array}{l} \left\langle \begin{array}{lllll} \texttt{machine}-1 & \texttt{origin}-6 & \texttt{duration}-6 & \texttt{end}-12 & \texttt{colour}-2, \\ \texttt{machine}-1 & \texttt{origin}-2 & \texttt{duration}-9 & \texttt{end}-11 & \texttt{colour}-3, \\ \texttt{machine}-2 & \texttt{origin}-7 & \texttt{duration}-3 & \texttt{end}-10 & \texttt{colour}-3, \\ \texttt{machine}-1 & \texttt{origin}-1 & \texttt{duration}-2 & \texttt{end}-3 & \texttt{colour}-1, \\ \texttt{machine}-2 & \texttt{origin}-4 & \texttt{duration}-5 & \texttt{end}-9 & \texttt{colour}-3, \\ \texttt{machine}-1 & \texttt{origin}-3 & \texttt{duration}-10 & \texttt{end}-13 & \texttt{colour}-2 \end{array} \right\rangle, \\ \langle \texttt{id}-1\ \texttt{capacity}-2, \texttt{id}-2\ \texttt{capacity}-1 \rangle \end{array}\right)$$

Figure 5.178 shows the solution associated with the example. Each rectangle of the figure corresponds to a task of the coloured_cumulatives constraint. Tasks that have their colour attribute set to 1 and 2 are respectively coloured in blue and pink. The coloured_cumulatives constraint holds since for machine 1 we have at most two distinct colours in parallel (which is the maximum capacity for machine 1), while for machine 2 we have no more than a single colour in parallel (which is actually the maximum capacity for machine 2).

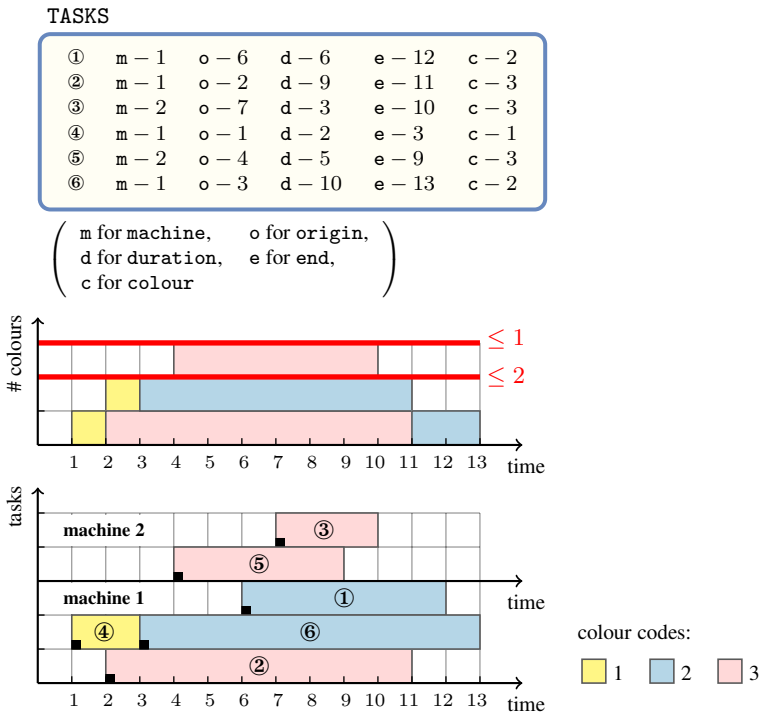


Figure 5.178: The coloured cumulative solution to the **Example** slot with at most two distinct colours in parallel on machine 1 and at most one distinct colour in parallel on machine 2

Typical

$|\texttt{TASKS}| > 1$
$\texttt{range}(\texttt{TASKS.machine}) > 1$
$\texttt{range}(\texttt{TASKS.origin}) > 1$
$\texttt{range}(\texttt{TASKS.duration}) > 1$
$\texttt{range}(\texttt{TASKS.end}) > 1$
$\texttt{range}(\texttt{TASKS.colour}) > 1$
$\texttt{TASKS.duration} > 0$
$|\texttt{MACHINES}| > 1$
$\texttt{MACHINES.capacity} > 0$
$\texttt{MACHINES.capacity} < \texttt{nval}(\texttt{TASKS.colour})$
$|\texttt{TASKS}| > |\texttt{MACHINES}|$

Symmetries

- Items of TASKS are permutable.
- Items of MACHINES are permutable.
- MACHINES.capacity can be increased.
- All occurrences of two distinct values in TASKS.machine or MACHINES.id can be swapped; all occurrences of a value in TASKS.machine or MACHINES.id can be renamed to any unused value.

Arg. properties

Contractible wrt. TASKS.

Usage

Useful for scheduling problems where several machines are available and where you have to assign each task to a specific machine. In addition each machine can only proceed in parallel a maximum number of tasks of distinct types.

Reformulation

The coloured_cumulatives constraint can be expressed in term of a set of reified constraints and of |TASKS| nvalue constraints:

1. For each pair of tasks $\texttt{TASKS}[i], \texttt{TASKS}[j]$ $(i, j \in [1, |\texttt{TASKS}|])$ of the TASKS collection we create a variable C_{ij} which is set to the colour of task $\texttt{TASKS}[j]$ if both tasks are assigned to the same machine and if task $\texttt{TASKS}[j]$ overlaps the origin attribute of task $\texttt{TASKS}[i]$, and to the colour of task $\texttt{TASKS}[i]$ otherwise:
   - If $i = j$:
     - $C_{ij} = \texttt{TASKS}[i].\texttt{colour}$.
   - If $i \neq j$:
     - $C_{ij} = \texttt{TASKS}[i].\texttt{colour} \vee C_{ij} = \texttt{TASKS}[j].\texttt{colour}$.
     - $((\texttt{TASKS}[j].\texttt{machine} = \texttt{TASKS}[i].\texttt{machine} \wedge \texttt{TASKS}[j].\texttt{origin} \leq \texttt{TASKS}[i].\texttt{origin} \wedge \texttt{TASKS}[j].\texttt{end} > \texttt{TASKS}[i].\texttt{origin}) \wedge (C_{ij} = \texttt{TASKS}[j].\texttt{colour})) \vee ((\texttt{TASKS}[j].\texttt{machine} \neq \texttt{TASKS}[i].\texttt{machine} \vee \texttt{TASKS}[j].\texttt{origin} > \texttt{TASKS}[i].\texttt{origin} \vee \texttt{TASKS}[j].\texttt{end} \leq \texttt{TASKS}[i].\texttt{origin}) \wedge (C_{ij} = \texttt{TASKS}[i].\texttt{colour}))$
2. For each task $\texttt{TASKS}[i]$ $(i \in [1, |\texttt{TASKS}|])$ we create a variable N_i which gives the number of distinct colours associated with the tasks that both are assigned to the same machine as task $\texttt{TASKS}[i]$ and overlap the origin of task $\texttt{TASKS}[i]$ ($\texttt{TASKS}[i]$ overlaps its own origin) and we impose N_i to not exceed the maximum number of distinct colours LIMIT allowed at each instant:
   - $N_i \geq 1 \wedge N_i \leq \texttt{LIMIT}$.
   - $\texttt{nvalue}(N_i, \langle C_{i1}, C_{i2}, \ldots, C_{i|\texttt{TASKS}|} \rangle)$.

See also

assignment dimension removed: coloured_cumulative *(*machine *attribute removed)*, cumulative *(*machine *attribute removed and number of distinct* colours *replaced by sum of* task heights*)*.

common keyword: cumulative, cumulatives *(resource constraint)*.

related: nvalue.

used in graph description: nvalues.

Keywords

characteristic of a constraint: coloured.

constraint type: scheduling constraint, resource constraint, temporal constraint.

filtering: compulsory part.

modelling: number of distinct values, assignment dimension, zero-duration task.

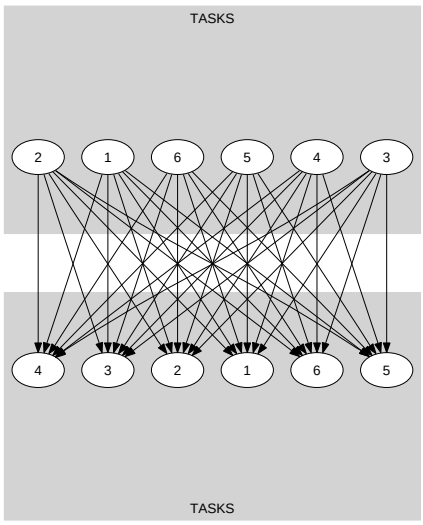
Arc input(s)	TASKS
Arc generator	$SELF \mapsto$ collection(tasks)
Arc arity	1
Arc constraint(s)	tasks.origin + tasks.duration = tasks.end
Graph property(ies)	**NARC** = |TASKS|

For all items of MACHINES:

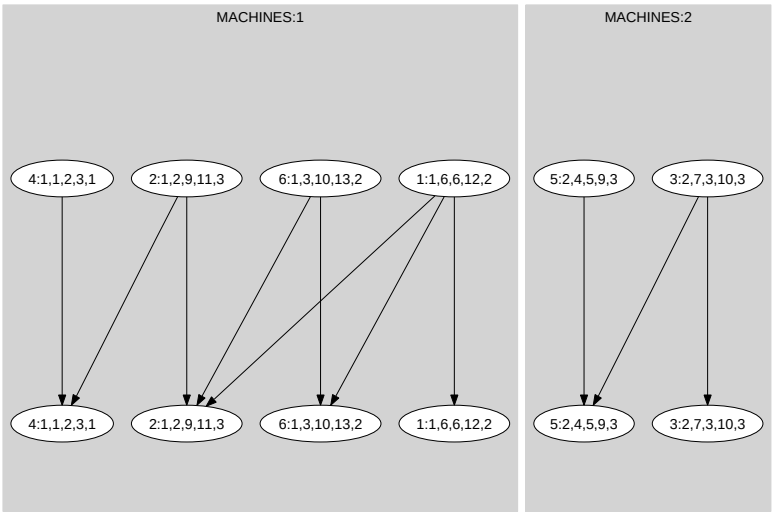
Arc input(s)	TASKS TASKS
Arc generator	$PRODUCT \mapsto$ collection(tasks1, tasks2)
Arc arity	2
Arc constraint(s)	• tasks1.machine = MACHINES.id • tasks1.machine = tasks2.machine • tasks1.duration > 0 • tasks2.origin ≤ tasks1.origin • tasks1.origin < tasks2.end
Graph class	• ACYCLIC • BIPARTITE • NO_LOOP
Sets	$\text{SUCC} \mapsto \left[\begin{array}{l}\text{source},\\ \text{variables} - \text{col}\left(\begin{array}{l}\text{VARIABLES}-\text{collection}(\text{var}-\text{dvar}),\\ [\text{item}(\text{var} - \text{TASKS.colour})]\end{array}\right)\end{array}\right]$
Constraint(s) on sets	nvalues(variables, ≤, MACHINES.capacity)

Graph model Parts (A) and (B) of Figure 5.179 respectively shows the initial and final graph associated with machines 1 and 2 involved in the **Example** slot. On the one hand, each source vertex of the final graph can be interpreted as a time point p on a specific machine m. On the other hand the successors of a source vertex correspond to those tasks that both overlap that time point p and are assigned to machine m. The coloured_cumulatives constraint holds since for each successor set $\mathcal{S}$ of the final graph the number of distinct colours in $\mathcal{S}$ does not exceed the capacity of the machine corresponding to the time point associated with $\mathcal{S}$.

Signature Since TASKS is the maximum number of vertices of the final graph of the first graph constraint we can rewrite **NARC** = |TASKS| to **NARC** ≥ |TASKS|. This leads to simplify **NARC** to **NARC**.



(A)



(B)

Figure 5.179: Initial and final graph of the `coloured_cumulatives` constraint