

5.84 cumulative_product

	DESCRIPTION	LINKS	GRAPH
Origin	Derived from <code>cumulative</code> .		
Constraint	<code>cumulative_product(TASKS, LIMIT)</code>		
Arguments	$\begin{array}{l} \text{TASKS} : \text{collection} \left(\begin{array}{l} \text{origin-dvar,} \\ \text{duration-dvar,} \\ \text{end-dvar,} \\ \text{height-dvar} \end{array} \right) \\ \text{LIMIT} : \text{int} \end{array}$		
Restrictions	<pre>require_at_least(2, TASKS, [origin, duration, end]) required(TASKS, height) TASKS.duration ≥ 0 TASKS.origin ≤ TASKS.end TASKS.height ≥ 1 LIMIT ≥ 0</pre>		
Purpose	<p>Consider a set \mathcal{T} of tasks described by the <code>TASKS</code> collection. The <code>cumulative_product</code> constraint enforces that at each point in time, the product of the heights of the set of tasks that overlap that point, does not exceed a given limit. 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 <code>origin + duration = end</code>.</p>		
Example	$\left(\left\langle \begin{array}{l} \text{origin} - 1 \quad \text{duration} - 3 \quad \text{end} - 4 \quad \text{height} - 1, \\ \text{origin} - 2 \quad \text{duration} - 9 \quad \text{end} - 11 \quad \text{height} - 2, \\ \text{origin} - 3 \quad \text{duration} - 10 \quad \text{end} - 13 \quad \text{height} - 1, \\ \text{origin} - 6 \quad \text{duration} - 6 \quad \text{end} - 12 \quad \text{height} - 1, \\ \text{origin} - 7 \quad \text{duration} - 2 \quad \text{end} - 9 \quad \text{height} - 3 \end{array} \right\rangle, 6 \right)$		

Figure 5.171 shows the solution associated with the example. To each task of the `cumulative_product` constraint corresponds a set of rectangles coloured with the same colour: the sum of the lengths of the rectangles corresponds to the duration of the task, while the height of the rectangles (i.e., all the rectangles associated with a task have the same height) corresponds to the height of the task. The profile corresponding to the product of the heights of the tasks that overlap a given point is depicted by a thick red line. The `cumulative_product` constraint holds since at each point in time the product of the heights of the tasks that overlap that point is not strictly greater than the upper limit 6 enforced by the last argument of the `cumulative_product` constraint.

Typical

```

|TASKS| > 1
range(TASKS.origin) > 1
range(TASKS.duration) > 1
range(TASKS.end) > 1
range(TASKS.height) > 1
TASKS.duration > 0
LIMIT <prod(TASKS.height)

```

Symmetries

- Items of TASKS are **permutable**.
- TASKS.height can be **decreased** to any value ≥ 0 .
- One and the same constant can be **added** to the origin and end attributes of all items of TASKS.
- LIMIT can be **increased**.

Reformulation

The cumulative_product constraint can be expressed in term of a set of reified constraints and of $|\text{TASKS}|$ constraints of the form $h_1 \cdot h_2 \cdot \dots \cdot h_{|\text{TASKS}|} \leq l$:

1. For each pair of tasks $\text{TASKS}[i], \text{TASKS}[j]$ ($i, j \in [1, |\text{TASKS}|]$) of the TASKS collection we create a variable H_{ij} which is set to the height of task $\text{TASKS}[j]$ if task $\text{TASKS}[j]$ overlaps the origin attribute of task $\text{TASKS}[i]$, and to 1 otherwise:
 - If $i = j$:
 - $H_{ij} = \text{TASKS}[i].\text{height}$.
 - If $i \neq j$:
 - $H_{ij} = \text{TASKS}[j].\text{height} \vee H_{ij} = 1$.
 - $((\text{TASKS}[j].\text{origin} \leq \text{TASKS}[i].\text{origin} \wedge \text{TASKS}[j].\text{end} > \text{TASKS}[i].\text{origin}) \wedge (H_{ij} = \text{TASKS}[j].\text{height})) \vee ((\text{TASKS}[j].\text{origin} > \text{TASKS}[i].\text{origin} \vee \text{TASKS}[j].\text{end} \leq \text{TASKS}[i].\text{origin}) \wedge (H_{ij} = 1))$
2. For each task $\text{TASKS}[i]$ ($i \in [1, |\text{TASKS}|]$) we impose a constraint of the form $H_{i1} \cdot H_{i2} \cdot \dots \cdot H_{i|\text{TASKS}|} \leq \text{LIMIT}$.

See also

common keyword: cumulative (*resource constraint*).

used in graph description: product_ctr.

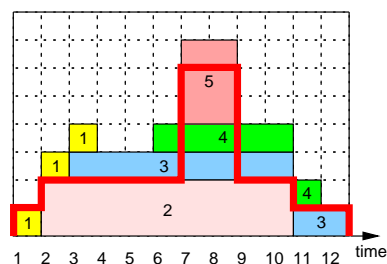


Figure 5.171: Solution of the cumulative_product constraint

Keywords

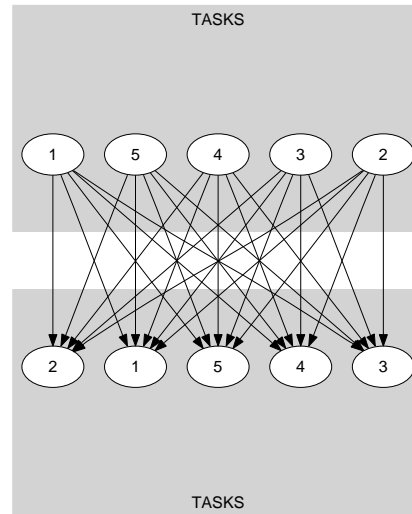
characteristic of a constraint: product.

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

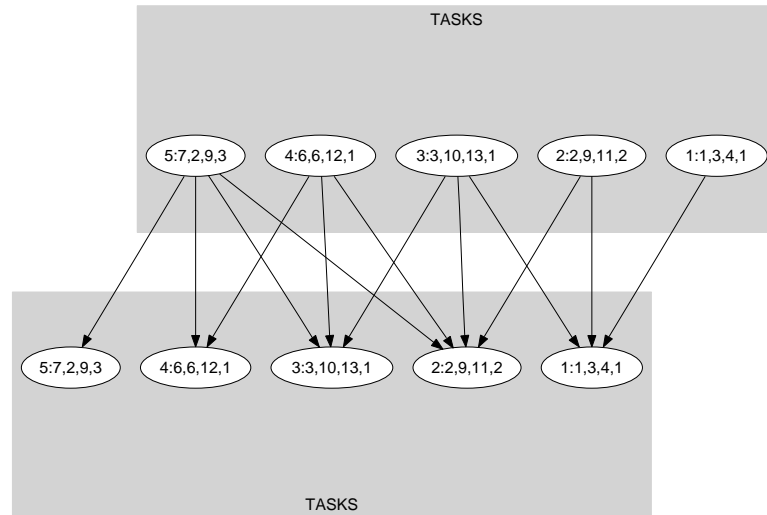
filtering: compulsory part.

modelling: zero-duration task.

Arc input(s)	TASKS
Arc generator	$SELF \mapsto \text{collection}(\text{tasks})$
Arc arity	1
Arc constraint(s)	$\text{tasks.origin} + \text{tasks.duration} = \text{tasks.end}$
Graph property(ies)	$\overline{NARC} = \text{TASKS} $
Arc input(s)	TASKS TASKS
Arc generator	$PRODUCT \mapsto \text{collection}(\text{tasks1}, \text{tasks2})$
Arc arity	2
Arc constraint(s)	<ul style="list-style-type: none"> • $\text{tasks1.duration} > 0$ • $\text{tasks2.origin} \leq \text{tasks1.origin}$ • $\text{tasks1.origin} < \text{tasks2.end}$
Graph class	<ul style="list-style-type: none"> • ACYCLIC • BIPARTITE • NO_LOOP
Sets	$SUCC \mapsto \left[\begin{array}{l} \text{source}, \\ \text{variables} - \text{col} \left(\begin{array}{l} \text{VARIABLES} - \text{collection}(\text{var} - \text{dvar}), \\ [\text{item}(\text{var} - \text{ITEMS.height})] \end{array} \right) \end{array} \right]$
Constraint(s) on sets	$\text{product_ctr}(\text{variables}, \leq, \text{LIMIT})$
Graph model	<p>Parts (A) and (B) of Figure 5.172 respectively show the initial and final graph associated with the second graph constraint of the Example slot. On the one hand, each source vertex of the final graph can be interpreted as a time point. On the other hand the successors of a source vertex correspond to those tasks that overlap that time point. The <code>cumulative_product</code> constraint holds since for each successor set S of the final graph the product of the heights of the tasks in S does not exceed the limit $\text{LIMIT} = 6$.</p>
Signature	<p>Since TASKS is the maximum number of vertices of the final graph of the first graph constraint we can rewrite $\overline{NARC} = \text{TASKS}$ to $\overline{NARC} \geq \text{TASKS}$. This leads to simplify \overline{NARC} to \overline{NARC}.</p>



(A)



(B)

Figure 5.172: Initial and final graph of the `cumulative_product` constraint

20030820

735