5.338 same_interval

DESCRIPTION LINKS GRAPH

Origin Derived from `same`.

Constraint `same_interval(VARIABLES1, VARIABLES2, SIZE_INTERVAL)`

Arguments

```
VARIABLES1    : collection(var-dvar)
VARIABLES2    : collection(var-dvar)
SIZE_INTERVAL : int
```

Restrictions

```
|VARIABLES1| = |VARIABLES2|
required(VARIABLES1, var)
required(VARIABLES2, var)
SIZE_INTERVAL > 0
```

Purpose Let N_i (respectively M_i) denote the number of variables of the collection `VARIABLES1` (respectively `VARIABLES2`) that take a value in the interval $[\texttt{SIZE\_INTERVAL} \cdot i, \texttt{SIZE\_INTERVAL} \cdot i + \texttt{SIZE\_INTERVAL} - 1$. For all integer i we have $N_i = M_i$.

Example $(\langle 1, 7, 6, 0, 1, 7\rangle, \langle 8, 8, 8, 0, 1, 2\rangle, 3)$

In the example, the third argument `SIZE_INTERVAL` $= 3$ defines the following family of intervals $[3 \cdot k, 3 \cdot k + 2]$, where k is an integer. Consequently the values of the collection $\langle 1, 7, 6, 0, 1, 7\rangle$ are respectively located within intervals $[0, 2]$, $[6, 8]$, $[6, 8]$, $[0, 2]$, $[0, 2]$, $[6, 8]$. Therefore intervals $[0, 2]$ and $[6, 8]$ are respectively used 3 and 3 times. Similarly, the values of the collection $\langle 8, 8, 8, 0, 1, 2\rangle$ are respectively located within intervals $[6, 8]$, $[6, 8]$, $[6, 8]$, $[0, 2]$, $[0, 2]$, $[0, 2]$. As before intervals $[0, 2]$ and $[6, 8]$ are respectively used 3 and 3 times. Consequently the `same_interval` constraint holds. Figure 5.671 illustrates this correspondence.

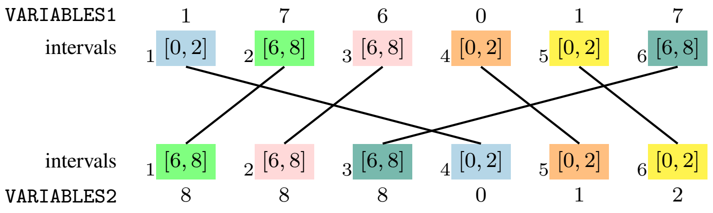

Figure 5.671: Illustration of the correspondence between the items of the `VARIABLES1` and of the `VARIABLES2` collections of the **Example** slot

Typical

$|\texttt{VARIABLES1}| > 1$
$\texttt{range}(\texttt{VARIABLES1.var}) > 1$
$\texttt{range}(\texttt{VARIABLES2.var}) > 1$
$\texttt{SIZE\_INTERVAL} > 1$
$\texttt{SIZE\_INTERVAL} < \texttt{range}(\texttt{VARIABLES1.var})$
$\texttt{SIZE\_INTERVAL} < \texttt{range}(\texttt{VARIABLES2.var})$

Symmetries

- Arguments are permutable w.r.t. permutation $(\texttt{VARIABLES1}, \texttt{VARIABLES2})$ $(\texttt{SIZE\_INTERVAL})$.
- Items of VARIABLES1 are permutable.
- Items of VARIABLES2 are permutable.
- An occurrence of a value of VARIABLES.var that belongs to the k-th interval, of size SIZE_INTERVAL, can be replaced by any other value of the same interval.

Arg. properties

Aggregate: VARIABLES1(union), VARIABLES2(union), SIZE_INTERVAL(id).

Algorithm

See algorithm of the same constraint.

Used in

k_same_interval.

See also

implies: used_by_interval.

soft variant: soft_same_interval_var *(variable-based violation measure)*.

specialisation: same *(*variable/constant *replaced by* variable*)*.

system of constraints: k_same_interval.

Keywords

characteristic of a constraint: sort based reformulation.

combinatorial object: permutation.

constraint arguments: constraint between two collections of variables.

modelling: interval.

Arc input(s)	`VARIABLES1` `VARIABLES2`
Arc generator	$PRODUCT$ $\mapsto$ `collection(variables1, variables2)`
Arc arity	2

Arc constraint(s)	`variables1.var/SIZE_INTERVAL =`
	`variables2.var/SIZE_INTERVAL`

Graph property(ies)	• for all connected components: **NSOURCE**=**NSINK** • **NSOURCE**= |`VARIABLES1`| • **NSINK**= |`VARIABLES2`|

Graph model Parts (A) and (B) of Figure 5.672 respectively show the initial and final graph associated with the **Example** slot. Since we use the **NSOURCE** and **NSINK** graph properties, the source and sink vertices of the final graph are stressed with a double circle. Since there is a constraint on each connected component of the final graph we also show the different connected components. Each of them corresponds to an equivalence class according to the arc constraint. The `same_interval` constraint holds since:

- Each connected component of the final graph has the same number of sources and of sinks.
- The number of sources of the final graph is equal to |`VARIABLES1`|.
- The number of sinks of the final graph is equal to |`VARIABLES2`|.

Signature Since the initial graph contains only sources and sinks, and since isolated vertices are eliminated from the final graph, we make the following observations:

- Sources of the initial graph cannot become sinks of the final graph,
- Sinks of the initial graph cannot become sources of the final graph.

From the previous observations and since we use the $PRODUCT$ arc generator on the collections `VARIABLES1` and `VARIABLES2`, we have that the maximum number of sources and sinks of the final graph is respectively equal to |`VARIABLES1`| and |`VARIABLES2`|. Therefore we can rewrite **NSOURCE** = |`VARIABLES1`| to **NSOURCE** $\geq$ |`VARIABLES1`| and simplify $\underline{\overline{\mathbf{NSOURCE}}}$ to $\overline{\mathbf{NSOURCE}}$. In a similar way, we can rewrite **NSINK** = |`VARIABLES2`| to **NSINK** $\geq$ |`VARIABLES2`| and simplify $\underline{\overline{\mathbf{NSINK}}}$ to $\overline{\mathbf{NSINK}}$.

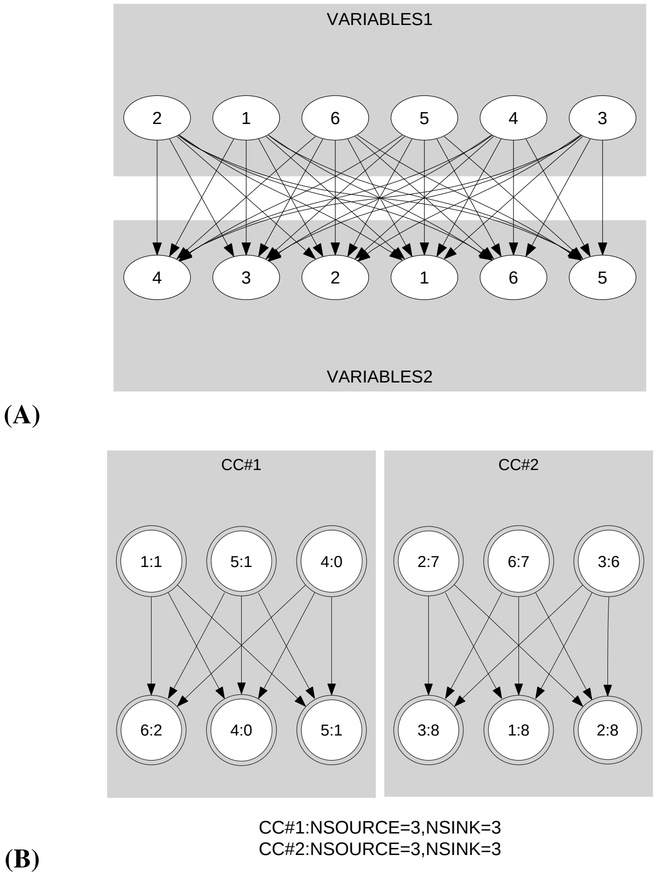

Figure 5.672: Initial and final graph of the `same_interval` constraint