5.398 symmetric_cardinality

DESCRIPTION LINKS GRAPH

Origin Derived from global_cardinality by W. Kocjan.

Constraint symmetric_cardinality(VARS, VALS)

Arguments

VARS : collection(idvar−int, var−svar, l−int, u−int)
VALS : collection(idval−int, val−svar, l−int, u−int)

Restrictions

required(VARS, [idvar, var, l, u])
$|\mathtt{VARS}| \geq 1$
$\mathtt{VARS.idvar} \geq 1$
$\mathtt{VARS.idvar} \leq |\mathtt{VARS}|$
distinct(VARS, idvar)
$\mathtt{VARS.l} \geq 0$
$\mathtt{VARS.l} \leq \mathtt{VARS.u}$
$\mathtt{VARS.u} \leq |\mathtt{VALS}|$
required(VALS, [idval, val, l, u])
$|\mathtt{VALS}| \geq 1$
$\mathtt{VALS.idval} \geq 1$
$\mathtt{VALS.idval} \leq |\mathtt{VALS}|$
distinct(VALS, idval)
$\mathtt{VALS.l} \geq 0$
$\mathtt{VALS.l} \leq \mathtt{VALS.u}$
$\mathtt{VALS.u} \leq |\mathtt{VARS}|$

Purpose Put in relation two sets: for each element of one set gives the corresponding elements of the other set to which it is associated. In addition, it constraints the number of elements associated with each element to be in a given interval.

Example

$$\left(\begin{array}{l}\left\langle\begin{array}{llll}\mathtt{idvar}-1 & \mathtt{var}-\{3\} & \mathtt{l}-0 & \mathtt{u}-1,\\ \mathtt{idvar}-2 & \mathtt{var}-\{1\} & \mathtt{l}-1 & \mathtt{u}-2,\\ \mathtt{idvar}-3 & \mathtt{var}-\{1,2\} & \mathtt{l}-1 & \mathtt{u}-2,\\ \mathtt{idvar}-4 & \mathtt{var}-\{1,3\} & \mathtt{l}-2 & \mathtt{u}-3\end{array}\right\rangle,\\ \left\langle\begin{array}{llll}\mathtt{idval}-1 & \mathtt{val}-\{2,3,4\} & \mathtt{l}-3 & \mathtt{u}-4,\\ \mathtt{idval}-2 & \mathtt{val}-\{3\} & \mathtt{l}-1 & \mathtt{u}-1,\\ \mathtt{idval}-3 & \mathtt{val}-\{1,4\} & \mathtt{l}-1 & \mathtt{u}-2,\\ \mathtt{idval}-4 & \mathtt{val}-\emptyset & \mathtt{l}-0 & \mathtt{u}-1\end{array}\right\rangle\end{array}\right)$$

The symmetric_cardinality constraint holds since:

- $3 \in \mathtt{VARS}[1].\mathtt{var} \Leftrightarrow 1 \in \mathtt{VALS}[3].\mathtt{val}$,
- $1 \in \mathtt{VARS}[2].\mathtt{var} \Leftrightarrow 2 \in \mathtt{VALS}[1].\mathtt{val}$,
- $1 \in \mathtt{VARS}[3].\mathtt{var} \Leftrightarrow 3 \in \mathtt{VALS}[1].\mathtt{val}$,
- $2 \in \mathtt{VARS}[3].\mathtt{var} \Leftrightarrow 3 \in \mathtt{VALS}[2].\mathtt{val}$,

- $1 \in$ VARS$[4]$.var $\Leftrightarrow 4 \in$ VALS$[1]$.val,
- $3 \in$ VARS$[4]$.var $\Leftrightarrow 4 \in$ VALS$[3]$.val,
- The number of elements of VARS$[1]$.var $= \{3\}$ belongs to interval $[0, 1]$,
- The number of elements of VARS$[2]$.var $= \{1\}$ belongs to interval $[1, 2]$,
- The number of elements of VARS$[3]$.var $= \{1, 2\}$ belongs to interval $[1, 2]$,
- The number of elements of VARS$[4]$.var $= \{1, 3\}$ belongs to interval $[2, 3]$,
- The number of elements of VALS$[1]$.val $= \{2, 3, 4\}$ belongs to interval $[3, 4]$,
- The number of elements of VALS$[2]$.val $= \{3\}$ belongs to interval $[1, 1]$,
- The number of elements of VALS$[3]$.val $= \{1, 4\}$ belongs to interval $[1, 2]$,
- The number of elements of VALS$[4]$.val $= \emptyset$ belongs to interval $[0, 1]$.

Typical

$|$VARS$| > 1$
$|$VALS$| > 1$

Symmetrics

- Items of VARS are permutable.
- Items of VALS are permutable.

Usage

The most simple example of applying `symmetric_gcc` is a variant of personnel assignment problem, where one person can be assigned to perform between n and m $(n \leq m)$ jobs, and every job requires between p and q $(p \leq q)$ persons. In addition every job requires different kind of skills. The previous problem can be modelled as follows:

- For each person we create an item of the VARS collection,
- For each job we create an item of the VALS collection,
- There is an arc between a person and the particular job if this person is qualified to perform it.

Remark

The `symmetric_gcc` constraint generalises the `global_cardinality` constraint by allowing a variable to take more than one value.

Algorithm

A first flow-based arc-consistency algorithm for the `symmetric_cardinality` constraint is described in [241]. A second arc-consistency filtering algorithm exploiting matching theory [148] is described in [129, 130].

See also

common keyword: `link_set_to_booleans` (*constraint involving set variables*).

generalisation: `symmetric_gcc` (`fixed interval` *replaced by* `variable`).

root concept: `global_cardinality`.

used in graph description: `in_set`.

Keywords

application area: assignment.

combinatorial object: relation.

constraint arguments: constraint involving set variables.

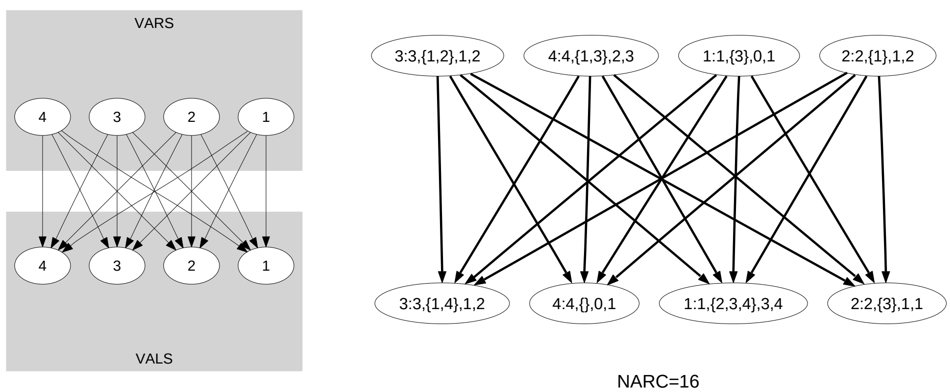
constraint type: decomposition, timetabling constraint.

filtering: flow, bipartite matching.

Arc input(s)	`VARS` `VALS`
Arc generator	$PRODUCT \mapsto$ `collection`(`vars`, `vals`)
Arc arity	2
Arc constraint(s)	• `in_set`(`vars.idvar`, `vals.val`) $\Leftrightarrow$ `in_set`(`vals.idval`, `vars.var`) • `vars.l` $\leq$ `card_set`(`vars.var`) • `vars.u` $\geq$ `card_set`(`vars.var`) • `vals.l` $\leq$ `card_set`(`vals.val`) • `vals.u` $\geq$ `card_set`(`vals.val`)
Graph property(ies)	$\mathbf{NARC} = |$`VARS`$| * |$`VALS`$|$

Graph model The graph model used for the `symmetric_cardinality` is similar to the one used in the `domain_constraint` or in the `link_set_to_booleans` constraints: we use an equivalence in the arc constraint and ask all arc constraints to hold.

Parts (A) and (B) of Figure 5.756 respectively show the initial and final graph associated with the **Example** slot. Since we use the $\mathbf{NARC}$ graph property, all the arcs of the final graph are stressed in bold.



(A) **(B)**

Figure 5.756: Initial and final graph of the `symmetric_cardinality` constraint

Signature Since we use the $PRODUCT$ arc generator on the collections `VARS` and `VALS`, the number of arcs of the initial graph is equal to $|$`VARS`$| \cdot |$`VALS`$|$. Therefore the maximum number of arcs of the final graph is also equal to $|$`VARS`$| \cdot |$`VALS`$|$ and we can rewrite $\mathbf{NARC} = |$`VARS`$| \cdot |$`VALS`$|$ to $\mathbf{NARC} \geq |$`VARS`$| \cdot |$`VALS`$|$. So we can simplify $\underline{\overline{\mathbf{NARC}}}$ to $\overline{\mathbf{NARC}}$.