5.416 uses

DESCRIPTION LINKS GRAPH

Origin [63]

Constraint uses(VARIABLES1, VARIABLES2)

Arguments

VARIABLES1 : collection(var−dvar)
VARIABLES2 : collection(var−dvar)

Restrictions

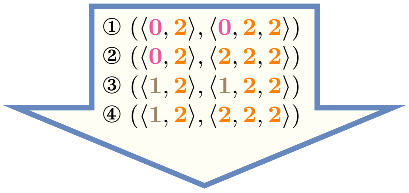
$\texttt{min}(1, |\texttt{VARIABLES1}|) \geq \texttt{min}(1, |\texttt{VARIABLES2}|)$
required(VARIABLES1, var)
required(VARIABLES2, var)

Purpose The set of values assigned to the variables of the collection of variables VARIABLES2 is included within the set of values assigned to the variables of the collection of variables VARIABLES1.

Example $(\langle 3, 3, 4, 6\rangle , \langle 3, 4, 4, 4, 4\rangle)$

The uses constraint holds since the set of values $\{3, 4\}$ assigned to the items of collection $\langle 3, 4, 4, 4, 4\rangle$ is included within the set of values $\{3, 4, 6\}$ occurring within $\langle 3, 3, 4, 6\rangle$.

All solutions Figure 5.788 gives all solutions to the following non ground instance of the uses constraint: $U_1 \in [0, 1]$, $U_2 \in [1, 2]$, $V_1 \in [0, 2]$, $V_2 \in [2, 4]$, $V_3 \in [2, 4]$, $\texttt{uses}(\langle U_1, U_2\rangle, \langle V_1, V_2, V_3\rangle)$.



① $(\langle 0, 2\rangle, \langle 0, 2, 2\rangle)$
② $(\langle 0, 2\rangle, \langle 2, 2, 2\rangle)$
③ $(\langle 1, 2\rangle, \langle 1, 2, 2\rangle)$
④ $(\langle 1, 2\rangle, \langle 2, 2, 2\rangle)$

Figure 5.788: All solutions corresponding to the non ground example of the uses constraint of the **All solutions** slot where identical values are coloured in the same way in both collections

Typical

$|\texttt{VARIABLES1}| > 1$
$\texttt{range}(\texttt{VARIABLES1.var}) > 1$
$|\texttt{VARIABLES2}| > 1$
$\texttt{range}(\texttt{VARIABLES2.var}) > 1$
$|\texttt{VARIABLES1}| \leq |\texttt{VARIABLES2}|$

Symmetries

- Items of VARIABLES1 are permutable.
- Items of VARIABLES2 are permutable.
- All occurrences of two distinct values in VARIABLES1.var or VARIABLES2.var can be swapped; all occurrences of a value in VARIABLES1.var or VARIABLES2.var can be renamed to any unused value.

Arg. properties

- Contractible wrt. VARIABLES2.
- Extensible wrt. VARIABLES1.
- Aggregate: VARIABLES1(union), VARIABLES2(union).

Remark

It was shown in [63] that, finding out whether a uses constraint has a solution or not is NP-hard. This was achieved by reduction from 3-SAT.

See also

generalisation: common.

implied by: used_by.

related: roots.

Keywords

complexity: 3-SAT.

constraint arguments: constraint between two collections of variables.

final graph structure: acyclic, bipartite, no loop.

modelling: inclusion.

Arc input(s)	`VARIABLES1 VARIABLES2`
Arc generator	$PRODUCT$ ↦ `collection`(`variables1`, `variables2`)
Arc arity	2
Arc constraint(s)	`variables1.var` = `variables2.var`
Graph property(ies)	**NSINK**= |`VARIABLES2`|
Graph class	• `ACYCLIC` • `BIPARTITE` • `NO_LOOP`

Graph model Parts (A) and (B) of Figure 5.789 respectively show the initial and final graph associated with the **Example** slot. Since we use the **NSINK** graph property, the sink vertices of the final graph are stressed with a double circle. Note that all the vertices corresponding to the variables that take values 9 or 2 were removed from the final graph since there is no arc for which the associated equality constraint holds.

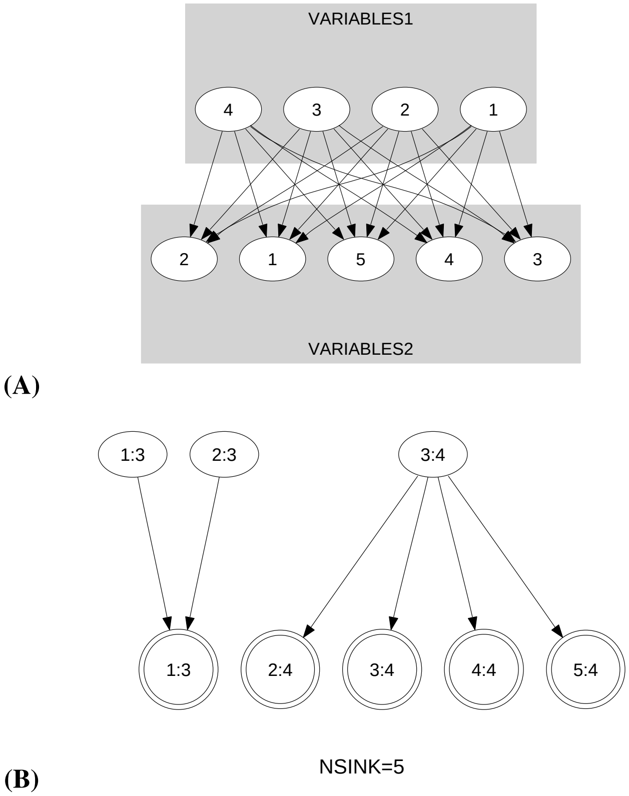


Figure 5.789: Initial and final graph of the `uses` constraint