5.369 soft_used_by_var

DESCRIPTION **LINKS** **GRAPH**

Origin

Derived from `used_by`

Constraint

`soft_used_by_var(C, VARIABLES1, VARIABLES2)`

Synonym

`soft_used_by`.

Arguments

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

Restrictions

$C \geq 0$
$C \leq |VARIABLES2|$
$|VARIABLES1| \geq |VARIABLES2|$
required(VARIABLES1, var)
required(VARIABLES2, var)

Purpose

C is the minimum number of values to change in the VARIABLES1 and VARIABLES2 collections so that all the values of the variables of collection VARIABLES2 are used by the variables of collection VARIABLES1.

Example

$(2, \langle 9, 1, 1, 8, 8 \rangle, \langle 9, 9, 9, 1 \rangle)$

As illustrated by Figure 5.712, there is a correspondence between two pairs of values of the collections $\langle 9, 1, 1, 8, 8 \rangle$ and $\langle 9, 9, 9, 1 \rangle$. Consequently, we must unset at least $4 - 2$ items (4 is the number of items of the VARIABLES2 collection). The `soft_used_by_var` constraint holds since its first argument C is set to $4 - 2$.

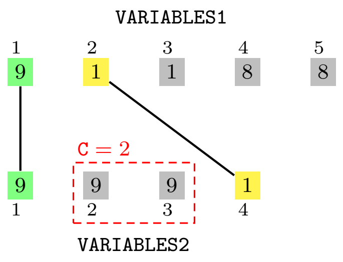

Figure 5.712: Illustration of the partial correspondence between the items of the VARIABLES2 $= \langle 9, 9, 9, 1 \rangle$ and of the VARIABLES1 $= \langle 9, 1, 1, 8, 8 \rangle$ collections of the **Example** slot, i.e., C $= 2$ items of the VARIABLES2 or of the VARIABLES1 collections need to be changed in order to cover all elements of VARIABLES2

Typical

$\texttt{C} > 0$
$|\texttt{VARIABLES1}| > 1$
$|\texttt{VARIABLES2}| > 1$
$\texttt{range}(\texttt{VARIABLES1.var}) > 1$
$\texttt{range}(\texttt{VARIABLES2.var}) > 1$

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.

Usage A soft `used_by` constraint.

Algorithm A filtering algorithm achieving arc-consistency is described in [129, 130].

See also

hard version: `used_by`.

implied by: `soft_same_var`.

Keywords

constraint arguments: constraint between two collections of variables.

constraint type: soft constraint, relaxation, variable-based violation measure.

filtering: bipartite matching.

Arc input(s)	`VARIABLES1 VARIABLES2`
Arc generator	$PRODUCT \mapsto$ `collection(variables1, variables2)`
Arc arity	2
Arc constraint(s)	`variables1.var` $=$ `variables2.var`
Graph property(ies)	**NSINK_NSOURCE** $= \vert$`VARIABLES2`$\vert -$ `C`

Graph model Parts (A) and (B) of Figure 5.713 respectively show the initial and final graph associated with the **Example** slot. Since we use the **NSINK_NSOURCE** graph property, the source and sink vertices of the final graph are stressed with a double circle. The `soft_used_by_var` constraint holds since the cost 2 corresponds to the difference between the number of variables of `VARIABLES2` and the sum over the different connected components of the minimum number of sources and sinks.

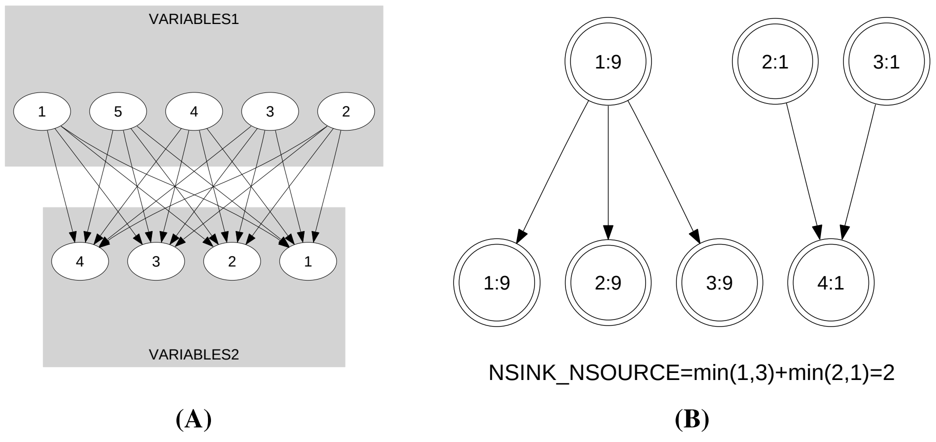

Figure 5.713: Initial and final graph of the `soft_used_by_var` constraint