5.363 soft_same_modulo_var

DESCRIPTION LINKS GRAPH

Origin Derived from same_modulo

Constraint soft_same_modulo_var(C, VARIABLES1, VARIABLES2, M)

Synonym soft_same_modulo.

Arguments

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

Restrictions

$\mathtt{C} \geq 0$
$\mathtt{C} \leq |\mathtt{VARIABLES1}|$
$|\mathtt{VARIABLES1}| = |\mathtt{VARIABLES2}|$
required(VARIABLES1, var)
required(VARIABLES2, var)
$\mathtt{M} > 0$

Purpose

For each integer R in $[0, \mathtt{M}-1]$, let $N1_R$ (respectively $N2_R$) denote the number of variables of VARIABLES1 (respectively VARIABLES2) that have R as a rest when divided by M. C is the minimum number of values to change in the VARIABLES1 and VARIABLES2 collections so that for all R in $[0, \mathtt{M}-1]$ we have $N1_R = N2_R$.

Example

$(4, \langle 9, 9, 9, 9, 9, 1\rangle, \langle 9, 1, 1, 1, 1, 8\rangle, 3)$

In the example, the values of the collections $\langle 9, 9, 9, 9, 9, 1\rangle$ and $\langle 9, 1, 1, 1, 1, 8\rangle$ are respectively associated with the equivalence classes $9 \bmod 3 = 0$, $9 \bmod 3 = 0$, $9 \bmod 3 = 0$, $9 \bmod 3 = 0$, $9 \bmod 3 = 0$, $1 \bmod 3 = 1$ and $9 \bmod 3 = 0$, $1 \bmod 3 = 1$, $1 \bmod 3 = 1$, $1 \bmod 3 = 1$, $1 \bmod 3 = 1$, $8 \bmod 3 = 2$. Since there is a correspondence between two pairs of equivalence classes we must unset at least $6-2$ items (6 is the number of items of the VARIABLES1 and VARIABLES2 collections). Consequently, the soft_same_modulo_var constraint holds since its first argument C is set to $6-2$.

Typical

$\mathtt{C} > 0$
$|\mathtt{VARIABLES1}| > 1$
range(VARIABLES1.var) > 1
range(VARIABLES2.var) > 1
$\mathtt{M} > 1$
M <maxval(VARIABLES1.var)
M <maxval(VARIABLES2.var)

Symmetries

- Arguments are permutable w.r.t. permutation (`C`) (`VARIABLES1`, `VARIABLES2`) (`M`).
- Items of `VARIABLES1` are permutable.
- Items of `VARIABLES2` are permutable.
- An occurrence of a value u of `VARIABLES1.var` can be replaced by any other value v such that v is congruent to u modulo `M`.
- An occurrence of a value u of `VARIABLES2.var` can be replaced by any other value v such that v is congruent to u modulo `M`.

Usage

A soft `same_modulo` constraint.

Algorithm

See algorithm of the `soft_same_var` constraint.

See also

hard version: `same_modulo`.

implies: `soft_used_by_modulo_var`.

Keywords

characteristic of a constraint: modulo.

constraint arguments: constraint between two collections of variables.

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

Arc input(s)	VARIABLES1 VARIABLES2
Arc generator	*PRODUCT* ↦ collection(variables1, variables2)
Arc arity	2
Arc constraint(s)	variables1.var mod M = variables2.var mod M
Graph property(ies)	**NSINK_NSOURCE** $= \lvert\mathtt{VARIABLES1}\rvert - \mathtt{C}$

Graph model Parts (A) and (B) of Figure 5.705 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_same_modulo_var constraint holds since the cost 4 corresponds to the difference between the number of variables of VARIABLES1 and the sum over the different connected components of the minimum number of sources and sinks.

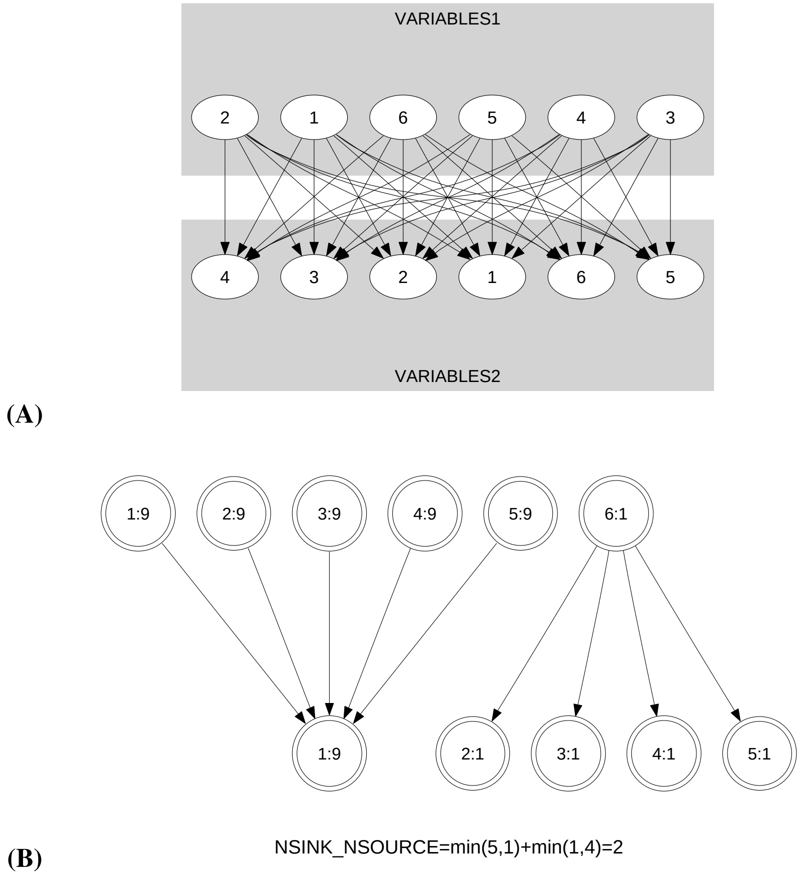

Figure 5.705: Initial and final graph of the soft_same_modulo_var constraint