5.12 alldifferent_on_intersection

DESCRIPTION LINKS GRAPH AUTOMATON

Origin

Derived from `common` and `alldifferent`.

Constraint

`alldifferent_on_intersection`(`VARIABLES1`, `VARIABLES2`)

Synonyms

`alldiff_on_intersection`, `alldistinct_on_intersection`.

Arguments

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

Restrictions

`required`(`VARIABLES1`, `var`)
`required`(`VARIABLES2`, `var`)

Purpose

The values that both occur in the `VARIABLES1` and `VARIABLES2` collections have only one occurrence.

Example

$$\left(\begin{array}{l}\langle 5,9,1,5\rangle,\\ \left\langle\begin{array}{l}\texttt{var}-2,\\ \texttt{var}-1,\\ \texttt{var}-6,\\ \texttt{var}-9,\\ \texttt{var}-6,\\ \texttt{var}-2\end{array}\right\rangle\end{array}\right)$$

The `alldifferent_on_intersection` constraint holds since the values 9 and 1 that both occur in $\langle 5,9,1,5\rangle$ as well as in $\langle 2,1,6,9,6,2\rangle$ have exactly one occurrence in each collection.

Typical

$|\texttt{VARIABLES1}| > 1$
$|\texttt{VARIABLES2}| > 1$

Symmetries

- Arguments are permutable w.r.t. permutation (`VARIABLES1`, `VARIABLES2`).
- 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.

See also

common keyword: `common`, `nvalue_on_intersection` (*constraint on the intersection*).

implied by: `disjoint`.

implies: `same_intersection`.

root concept: `alldifferent`.

Keywords

characteristic of a constraint: all different, automaton, automaton with array of counters.

constraint arguments: constraint between two collections of variables.

constraint type: constraint on the intersection, value constraint.

final graph structure: connected component, acyclic, bipartite, no loop.

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

Graph model Parts (A) and (B) of Figure 5.16 respectively show the initial and final graph associated with the **Example** slot. Since we use the **MAX_NCC** graph property we show one of the largest connected component of the final graph. The `alldifferent_on_intersection` constraint holds since each connected component has at most two vertices. Note that all the vertices corresponding to the variables that take values 5, 2 or 6 were removed from the final graph since there is no arc for which the associated equality constraint holds.

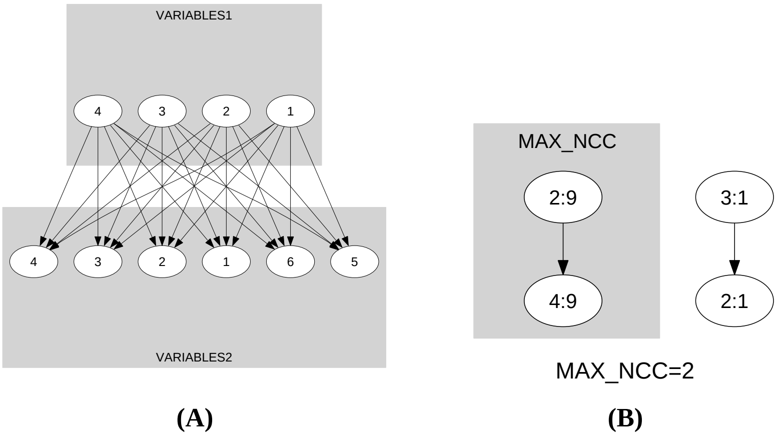

Figure 5.16: Initial and final graph of the `alldifferent_on_intersection` constraint

Automaton Figure 5.17 depicts the automaton associated with the `alldifferent_on_intersection` constraint. To each variable $\texttt{VAR1}_i$ of the collection `VARIABLES1` corresponds a signature variable $\texttt{S}_i$ that is equal to 0. To each variable $\texttt{VAR2}_i$ of the collection `VARIABLES2` corresponds a signature variable $\texttt{S}_{i+|\texttt{VARIABLES1}|}$ that is equal to 1. The automaton first counts the number of occurrences of each value assigned to the variables of the `VARIABLES1` collection. It then counts the number of occurrences of each value assigned to the variables of the `VARIABLES2` collection. Finally, the automaton imposes that each value is not taken by two variables of both collections.

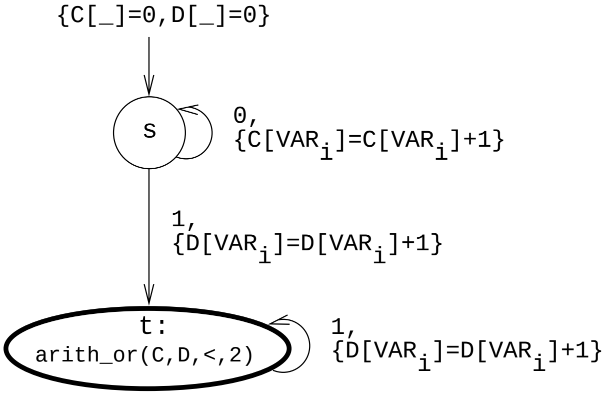

Figure 5.17: Automaton of the `alldifferent_on_intersection` constraint