5.19 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 $(\langle 5,9,1,5\rangle, \langle 2,1,6,9,6,2\rangle)$

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.

All solutions Figure 5.45 gives all solutions to the following non ground instance of the alldifferent_on_intersection constraint: $U_1 \in [2,3]$, $U_2 \in [1,2]$, $V_1 \in [2,3]$, $V_2 \in [2,2]$, $V_3 \in [0,1]$, alldifferent_on_intersection$(\langle U_1, U_2\rangle, \langle V_1, V_2, V_3\rangle)$.

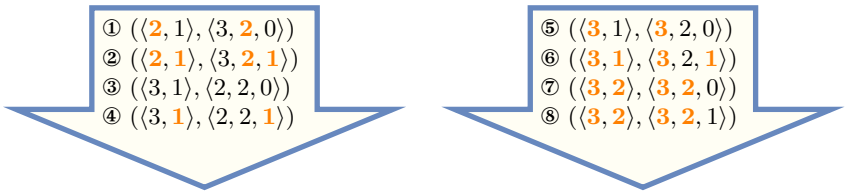

Figure 5.45: All solutions corresponding to the non ground example of the alldifferent_on_intersection constraint of the **All solutions** slot, where values that occur in both collections are coloured in orange

Typical

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

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.

Arg. properties

- Contractible wrt. `VARIABLES1`.
- Contractible wrt. `VARIABLES2`.

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	$PRODUCT \mapsto$ 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 5.46 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 components 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.

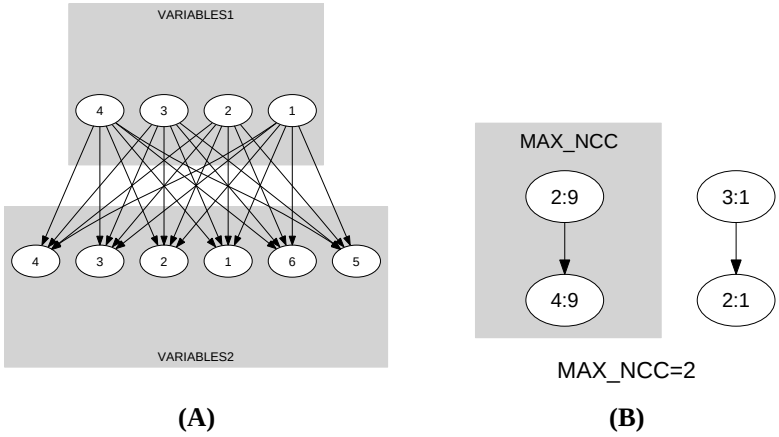

Figure 5.46: Initial and final graph of the alldifferent_on_intersection constraint

Automaton

Figure 5.47 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.

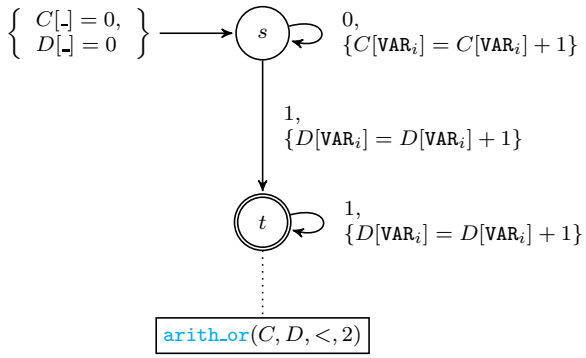

Figure 5.47: Automaton of the `alldifferent_on_intersection` constraint