5.182 in_same_partition

Origin   Used for defining several entries of this catalog.

Constraint   in_same_partition(VAR1, VAR2, PARTITIONS)

Type   VALUES : collection(val−int)

Arguments
```
VAR1       : dvar
VAR2       : dvar
PARTITIONS : collection(p − VALUES)
```

Restrictions
$|\text{VALUES}| \geq 1$
required(VALUES, val)
distinct(VALUES, val)
required(PARTITIONS, p)
$|\text{PARTITIONS}| \geq 2$

Purpose   Enforce VAR1 and VAR2 to be respectively assigned to values v_1 and v_2 that both belong to a same partition of the collection PARTITIONS.

Example $(6, 2, \langle \text{p} − \langle 1, 3 \rangle, \text{p} − \langle 4 \rangle, \text{p} − \langle 2, 6 \rangle \rangle)$

The in_same_partition constraint holds since its first and second arguments VAR1 = 6 and VAR2 = 2 both belong to the third partition $\langle 2, 6 \rangle$ of its third argument PARTITIONS.

Typical   VAR1 $\neq$ VAR2

Symmetries
- Arguments are permutable w.r.t. permutation (VAR1, VAR2) (PARTITIONS).
- Items of PARTITIONS are permutable.
- Items of PARTITIONS.p are permutable.

Arg. properties   Extensible wrt. PARTITIONS.

Used in   alldifferent_partition,   balance_partition,   change_partition,
common_partition,   nclass,   same_partition,   soft_same_partition_var,
soft_used_by_partition_var, used_by_partition.

See also   **common keyword:** alldifferent_partition *(partition)*, in *(value constraint)*.

used in graph description: in.

Keywords

Derived Collection

$$\text{col} \left(\begin{array}{l} \texttt{VARIABLES} - \texttt{collection}(\texttt{var} - \texttt{dvar}), \\ [\texttt{item}(\texttt{var} - \texttt{VAR1}), \texttt{item}(\texttt{var} - \texttt{VAR2})] \end{array} \right)$$

Arc input(s)	VARIABLES PARTITIONS
Arc generator	$PRODUCT \mapsto$ collection(variables, partitions)
Arc arity	2
Arc constraint(s)	in(variables.var, partitions.p)
Graph property(ies)	• **NSOURCE**$= 2$ • **NSINK**$= 1$

Graph model

VAR1 and VAR2 are put together in the derived collection VARIABLES. Since both VAR1 and VAR2 should take their value in one of the partition depicted by the PARTITIONS collection, the final graph should have two sources corresponding respectively to VAR1 and VAR2. Since two, possibly distinct, values should be assigned to VAR1 and VAR2 and since these values belong to the same partition p the final graph should only have one sink. This sink corresponds in fact to partition p.

Parts (A) and (B) of Figure 5.424 respectively show the initial and final graph associated with the **Example** slot. Since we both use the **NSOURCE** and **NSINK** graph properties, the source and sink vertices of the final graph are shown with a double circle.
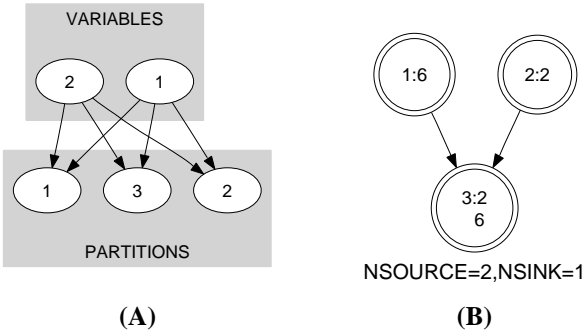

Figure 5.424: Initial and final graph of the in_same_partition constraint

Signature

Note that the sinks of the initial graph cannot become sources of the final graph since isolated vertices are eliminated from the final graph. Since the final graph contains two sources it also includes one arc between a source and a sink. Therefore the minimum number of sinks of the final graph is equal to one. So we can rewrite **NSINK** $= 1$ to **NSINK** ≥ 1 and simplify $\overline{\text{NSINK}}$ to **NSINK**.

Automaton   Figure 5.425 depicts the automaton associated with the in_same_partition constraint. Let $\texttt{VALUES}_i$ be the p attribute of the ith item of the PARTITIONS collection. To each triple $(\texttt{VAR1}, \texttt{VAR2}, \texttt{VALUES}_i)$ corresponds a 0-1 signature variable S_i as well as the following signature constraint: $((\texttt{VAR1} \in \texttt{VALUES}_i) \land (\texttt{VAR2} \in \texttt{VALUES}_i)) \Leftrightarrow S_i$.
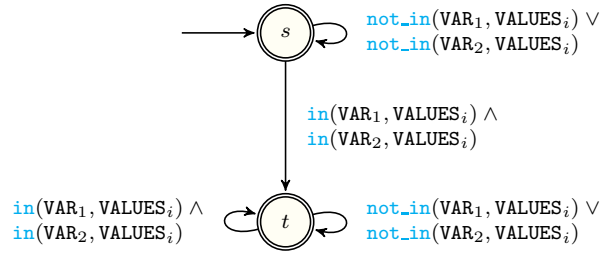

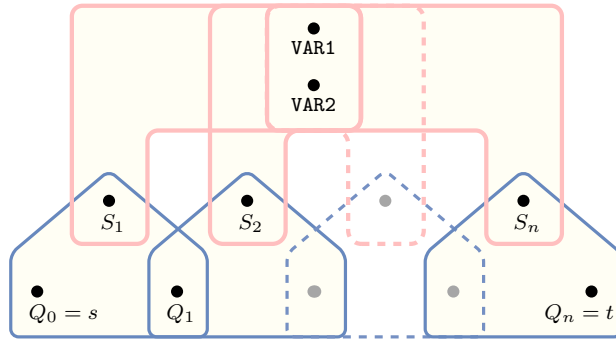Figure 5.425: Automaton of the in_same_partition constraint

Figure 5.426: Hypergraph of the reformulation corresponding to the automaton of the in_same_partition constraint