5.136 domain_constraint

DESCRIPTION LINKS GRAPH AUTOMATON

Origin [339]

Constraint domain_constraint(VAR, VALUES)

Synonym domain.

Arguments

VAR : dvar
VALUES : collection(var01−dvar, value−int)

Restrictions

required(VALUES, [var01, value])
VALUES.var01 ≥ 0
VALUES.var01 ≤ 1
distinct(VALUES, value)

Purpose Make the link between a domain variable VAR and those 0-1 variables that are associated with each potential value of VAR: The 0-1 variable associated with the value that is taken by variable VAR is equal to 1, while the remaining 0-1 variables are all equal to 0.

Example

$$\left(5, \left\langle \begin{array}{ll} \texttt{var01}-0 & \texttt{value}-9, \\ \texttt{var01}-1 & \texttt{value}-5, \\ \texttt{var01}-0 & \texttt{value}-2, \\ \texttt{var01}-0 & \texttt{value}-7 \end{array} \right\rangle \right)$$

The domain_constraint holds since VAR $=$ 5 is set to the value corresponding to the 0-1 variable set to 1, while the other 0-1 variables are all set to 0.

Typical $|\texttt{VALUES}| > 1$

Symmetry Items of VALUES are permutable.

Usage This constraint is used in order to make the link between a formulation using finite domain constraints and a formulation exploiting 0-1 variables.

Reformulation The domain_constraint(VAR,

$$\begin{array}{l}\langle \texttt{var01}-B_1\ \texttt{value}-v_1, \\ \ \texttt{var01}-B_2\ \texttt{value}-v_2, \\ \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ \ \texttt{var01}-B_{|\texttt{VALUES}|}\ \texttt{value}-v_{|\texttt{VALUES}|}\rangle)\end{array}$$

constraint can be expressed in term of the following reified constraint $(\texttt{VAR} = v_1 \wedge B_1 = 1) \vee (\texttt{VAR} = v_2 \wedge B_2 = 1) \vee \cdots \vee (\texttt{VAR} = v_{|\texttt{VALUES}|} \wedge B_{|\texttt{VALUES}|} = 1)$.

Systems domainChanneling in **Choco**, channel in **Gecode**, in in **SICStus**, in_set in **SICStus**.

See also

common keyword: `link_set_to_booleans` *(channelling constraint)*.

related: `roots`.

Keywords

characteristic of a constraint: automaton, automaton without counters, reified automaton constraint, derived collection.

constraint network structure: centered cyclic(1) constraint network(1).

constraint type: decomposition.

filtering: linear programming, arc-consistency.

modelling: channelling constraint, domain channel, Boolean channel.

Derived Collection

$$\texttt{col}\left(\begin{array}{l}\texttt{VALUE}-\texttt{collection}(\texttt{var01}-\texttt{int},\texttt{value}-\texttt{dvar}),\\ [\texttt{item}(\texttt{var01}-1,\texttt{value}-\texttt{VAR})]\end{array}\right)$$

Arc input(s)	VALUE VALUES
Arc generator	$PRODUCT \mapsto \texttt{collection}(\texttt{value},\texttt{values})$
Arc arity	2
Arc constraint(s)	$\texttt{value.value} = \texttt{values.value} \Leftrightarrow \texttt{values.var01} = 1$
Graph property(ies)	$\textbf{NARC} = \lvert\texttt{VALUES}\rvert$

Graph model The `domain_constraint` constraint is modelled with the following bipartite graph:

- The first class of vertices corresponds to a single vertex containing the domain variable.
- The second class of vertices contains one vertex for each item of the collection VALUES.

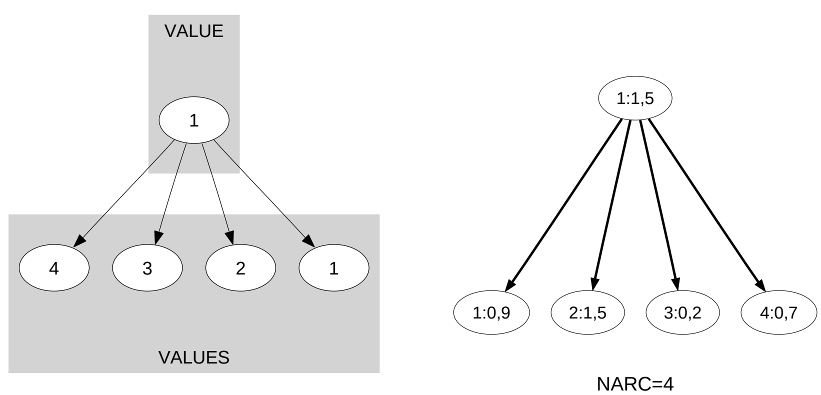
$PRODUCT$ is used in order to generate the arcs of the graph. In our context it takes a collection with a single item $\langle \texttt{var01}-1\ \texttt{value}-\texttt{VAR}\rangle$ and the collection VALUES.

The arc constraint between the variable VAR and one potential value v expresses the following:

- If the 0-1 variable associated with v is equal to 1, VAR is equal to v.
- Otherwise, if the 0-1 variable associated with v is equal to 0, VAR is not equal to v.

Since all arc constraints should hold the final graph contains exactly |VALUES| arcs.

Parts (A) and (B) of Figure 5.291 respectively show the initial and final graph associated with the **Example** slot. Since we use the **NARC** graph property, the arcs of the final graph are stressed in bold.



(A) **(B)**

Figure 5.291: Initial and final graph of the `domain_constraint` constraint

Signature

Since the number of arcs of the initial graph is equal to VALUES the maximum number of arcs of the final graph is also equal to VALUES. Therefore we can rewrite the graph property $\mathbf{NARC} = |\mathtt{VALUES}|$ to $\mathbf{NARC} \geq |\mathtt{VALUES}|$. This leads to simplify $\underline{\overline{\mathbf{NARC}}}$ to $\overline{\mathbf{NARC}}$.

Automaton Figure 5.292 depicts the automaton associated with the domain_constraint constraint. Let $\texttt{VAR01}_i$ and $\texttt{VALUE}_i$ respectively be the var01 and the value attributes of the i^{th} item of the VALUES collection. To each triple $(\texttt{VAR}, \texttt{VAR01}_i, \texttt{VALUE}_i)$ corresponds a 0-1 signature variable S_i as well as the following signature constraint: $((\texttt{VAR} = \texttt{VALUE}_i) \Leftrightarrow \texttt{VAR01}_i) \Leftrightarrow S_i$.

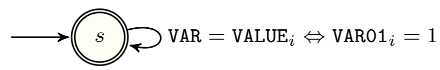


Figure 5.292: Automaton of the domain_constraint constraint

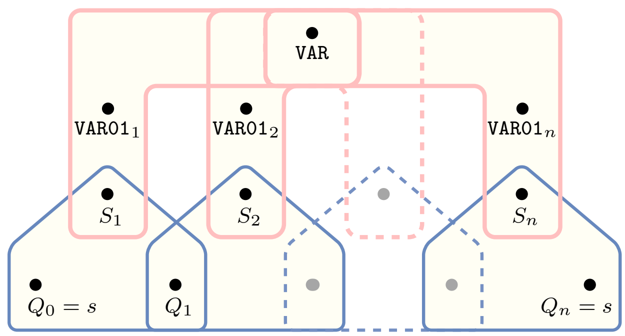


Figure 5.293: Hypergraph of the reformulation corresponding to the automaton of the domain_constraint constraint: since all states variables $Q_0, Q_1, \ldots, Q_n$ are fixed to the unique state s of the automaton, the transitions constraints involve only a single variable and the constraint network is Berge-acyclic