5.117 element

DESCRIPTION LINKS GRAPH AUTOMATON

Origin [380]

Constraint element(INDEX, TABLE, VALUE)

Synonyms nth, element_var, array.

Arguments

INDEX : dvar
TABLE : collection(value−dvar)
VALUE : dvar

Restrictions

$\text{INDEX} \geq 1$
$\text{INDEX} \leq |\text{TABLE}|$
required(TABLE, value)

Purpose VALUE is equal to the INDEX^{th} item of TABLE.

Example $(3, \langle 6, 9, 2, 9\rangle, 2)$

The element constraint holds since its third argument VALUE = 2 is equal to the 3^{th} (INDEX = 3) item of the collection $\langle 6, 9, 2, 9\rangle$.

Typical

$|\text{TABLE}| > 1$
range(TABLE.value) > 1

Symmetry All occurrences of two distinct values in TABLE.value or VALUE can be swapped; all occurrences of a value in TABLE.value or VALUE can be renamed to any unused value.

Usage See elem.

Remark In the original element constraint of **CHIP** the index attribute was not explicitly present in the table of values. It was implicitly defined as the position of a value in the previous table.

The element constraint is called nth in **Choco** (http://choco.sourceforge.net/). It is also sometimes called element_var when the second argument corresponds to a table of variables.

The case constraint [90] is a generalisation of the element constraint, where the table is replaced by a directed acyclic graph describing the set of solutions.

Systems nth in **Choco**, element in **Gecode**, element in **JaCoP**, element in **SICStus**.

See also

common keyword: elem_from_to, element_greatereq, element_lesseq, element_matrix, element_product, element_sparse *(array constraint)*, elementn, elements_sparse, in_relation, stage_element, sum *(data constraint)*.

generalisation: cond_lex_cost *(*variable *replaced by* tuple *of* variables*)*.

implied by: elem.

implies: elem.

system of constraints: elements.

uses in its reformulation: elements_alldifferent, tree_range, tree_resource.

Keywords

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

constraint network structure: centered cyclic(2) constraint network(1).

constraint type: data constraint.

filtering: arc-consistency.

modelling: array constraint, table, functional dependency, variable indexing, variable subscript, disjunction, assignment to the same set of values, sequence dependent set-up.

modelling exercises: assignment to the same set of values, sequence dependent set-up, zebra puzzle.

puzzles: zebra puzzle.

Derived Collection

$$\texttt{col}\left(\begin{array}{l}\texttt{ITEM}-\texttt{collection}(\texttt{index}-\texttt{dvar},\texttt{value}-\texttt{dvar}),\\ [\texttt{item}(\texttt{index}-\texttt{INDEX},\texttt{value}-\texttt{VALUE})]\end{array}\right)$$

Arc input(s) ITEM TABLE

Arc generator $PRODUCT \mapsto$ `collection`(`item`, `table`)

Arc arity 2

Arc constraint(s)

- `item.index` = `table.key`
- `item.value` = `table.value`

Graph property(ies) $\mathbf{NARC} = 1$

Graph model

The original `element` constraint with three arguments. We use the derived collection `ITEM` for putting together the `INDEX` and `VALUE` parameters of the `element` constraint. Within the arc constraint we use the implicit attribute `key` that associates to each item of a collection its position within the collection.

Parts (A) and (B) of Figure 5.237 respectively show the initial and final graph associated with the **Example** slot. Since we use the **NARC** graph property, the unique arc of the final graph is stressed in bold.

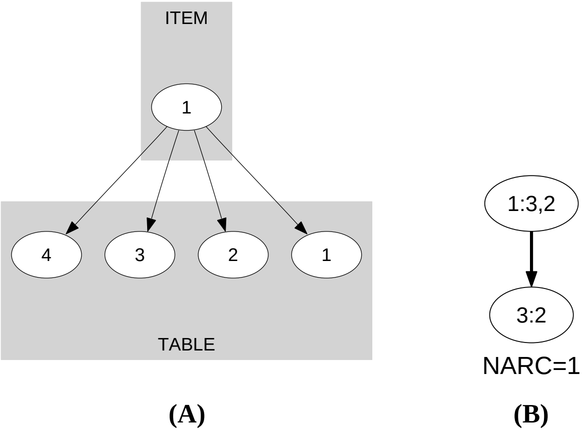

Figure 5.237: Initial and final graph of the `element` constraint

Signature

Because of the first condition of the arc constraint the final graph cannot have more than one arc. Therefore we can rewrite $\mathbf{NARC} = 1$ to $\mathbf{NARC} \geq 1$ and simplify $\overline{\mathbf{NARC}}$ to $\underline{\overline{\mathbf{NARC}}}$.

Automaton Figure 5.238 depicts the automaton associated with the `element` constraint. Let $\mathtt{VALUE}_i$ be the `value` attribute of the i^{th} item of the `TABLE` collection. To each triple $(\mathtt{INDEX}, \mathtt{VALUE}, \mathtt{VALUE}_i)$ corresponds a 0-1 signature variable $\mathtt{S}_i$ as well as the following signature constraint: $(\mathtt{INDEX} = i \wedge \mathtt{VALUE} = \mathtt{VALUE}_i) \Leftrightarrow \mathtt{S}_i$.

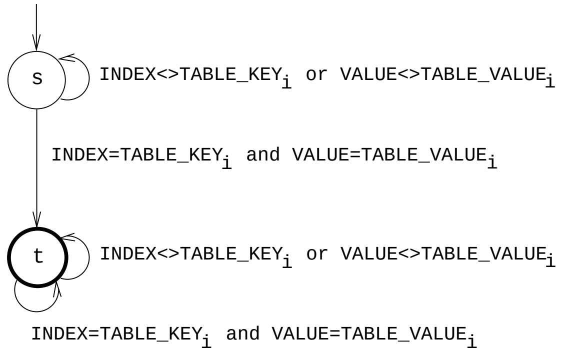
s
INDEX<>TABLE_KEY$_i$ or VALUE<>TABLE_VALUE$_i$
INDEX=TABLE_KEY$_i$ and VALUE=TABLE_VALUE$_i$
t
INDEX<>TABLE_KEY$_i$ or VALUE<>TABLE_VALUE$_i$
INDEX=TABLE_KEY$_i$ and VALUE=TABLE_VALUE$_i$

Figure 5.238: Automaton of the `element` constraint

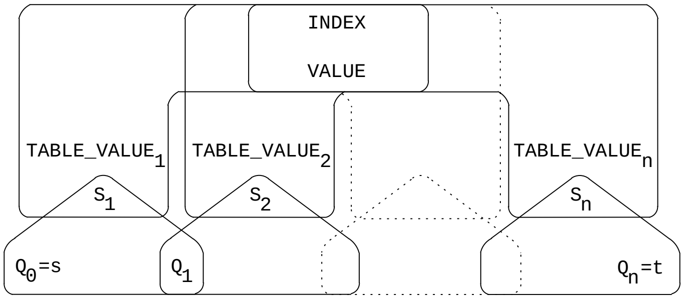

Figure 5.239: Hypergraph of the reformulation corresponding to the automaton of the `element` constraint