5.122 element_sparse

DESCRIPTION LINKS GRAPH AUTOMATON

Origin CHIP

Constraint element_sparse(ITEM, TABLE, DEFAULT)

Usual name element

Arguments

ITEM	:	collection(index−dvar, value−dvar)
TABLE	:	collection(index−int, value−int)
DEFAULT	:	int

Restrictions

required(ITEM, [index, value])
ITEM.index ≥ 1
|ITEM| = 1
required(TABLE, [index, value])
TABLE.index ≥ 1
distinct(TABLE, index)

Purpose ITEM[1].value is equal to one of the entries of the table TABLE or to the default value DEFAULT if the entry ITEM[1].index does not exist in TABLE.

Example

$$\left(\begin{array}{l}\langle \texttt{index}-2\ \texttt{value}-5\rangle,\\ \left\langle\begin{array}{ll}\texttt{index}-1 & \texttt{value}-6,\\ \texttt{index}-2 & \texttt{value}-5,\\ \texttt{index}-4 & \texttt{value}-2,\\ \texttt{index}-8 & \texttt{value}-9\end{array}\right\rangle, 5\end{array}\right)$$

The element_sparse constraint holds since its first argument ITEM corresponds to the second item of the TABLE collection.

Typical

|TABLE| > 1
range(TABLE.value) > 1

Symmetries

- Items of TABLE are permutable.
- All occurrences of two distinct values in ITEM.value, TABLE.value or DEFAULT can be swapped; all occurrences of a value in ITEM.value, TABLE.value or DEFAULT can be renamed to any unused value.

Usage A sometimes more compact form of the element constraint: we are not obliged to specify explicitly the table entries that correspond to the specified default value. This can sometimes reduce drastically memory utilisation.

Remark The original constraint of **CHIP** had an additional parameter SIZE giving the maximum value of ITEM.index.

Reformulation Let I and V respectively denote ITEM[1].index and ITEM[1].value. The element_sparse(ITEM, TABLE, DEFAULT) constraint can be expressed in term of a reified constraint of the form:
((I = TABLE[1].index ∧ V = TABLE[1].value) ∨
(I = TABLE[2].index ∧ V = TABLE[2].value) ∨
...
(I = TABLE[|TABLE|].index ∧ V = TABLE[TABLE|].value)) ∨
((I ≠ TABLE[1].index) ∧
(I ≠ TABLE[2].index) ∧
...
(I ≠ TABLE[|TABLE|].index) ∧
(V = DEFAULT)).

See also **common keyword:** elem, element *(array constraint)*, elements_sparse *(sparse table)*.

implies: elements_sparse.

system of constraints: elements_sparse.

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

constraint arguments: binary constraint.

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

constraint type: data constraint.

filtering: arc-consistency.

modelling: array constraint, table, sparse table, sparse functional dependency, variable indexing.

Derived Collections

col(DEF−collection(index−int, value−int), [item(index − 0, value − DEFAULT)])

col(TABLE_DEF−collection(index−dvar, value−dvar), [item(index − TABLE.index, value − TABLE.value), item(index − DEF.index, value − DEF.value)])

Arc input(s) ITEM TABLE_DEF

Arc generator PRODUCT ↦ collection(item, table_def)

Arc arity 2

Arc constraint(s)
- item.value = table_def.value
- item.index = table_def.index ∨ table_def.index = 0

Graph property(ies) NARC ≥ 1

Graph model The final graph has between one and two arc constraints: it has two arcs when the default value DEFAULT occurs also in the table TABLE; otherwise it has only one arc.

Parts (A) and (B) of Figure 5.250 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 outline with thick lines.

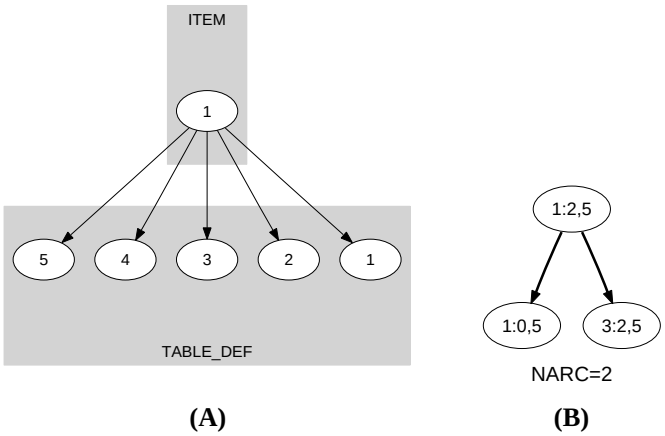

Figure 5.250: Initial and final graph of the element_sparse constraint

Automaton

Figure 5.251 depicts the automaton associated with the `element_sparse` constraint. Let `INDEX` and `VALUE` respectively be the `index` and the `value` attributes of the unique item of the `ITEM` collection. Let $\texttt{INDEX}_i$ and $\texttt{VALUE}_i$ respectively be the `index` and the `value` attributes of the i^{th} item of the `TABLE` collection. To each quintuple $(\texttt{INDEX}, \texttt{VALUE}, \texttt{DEFAULT}, \texttt{INDEX}_i, \texttt{VALUE}_i)$ corresponds a signature variable $\texttt{S}_i$ as well as the following signature constraint:

$$\begin{cases} (\texttt{INDEX} \neq \texttt{INDEX}_i \ \wedge \texttt{VALUE} \neq \texttt{DEFAULT}) & \Leftrightarrow \quad \texttt{S}_i = 0 \ \wedge \\ (\texttt{INDEX} = \texttt{INDEX}_i \ \wedge \texttt{VALUE} = \texttt{VALUE}_i \ ) & \Leftrightarrow \quad \texttt{S}_i = 1 \ \wedge \ . \\ (\texttt{INDEX} \neq \texttt{INDEX}_i \ \wedge \texttt{VALUE} = \texttt{DEFAULT}) & \Leftrightarrow \quad \texttt{S}_i = 2 \end{cases}$$

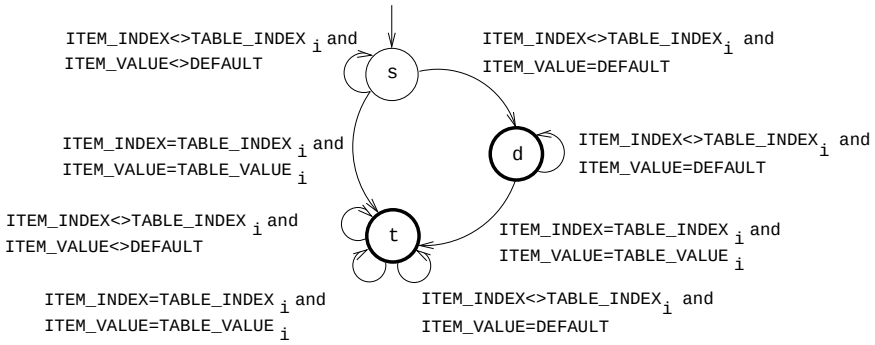

Figure 5.251: Automaton of the `element_sparse` constraint

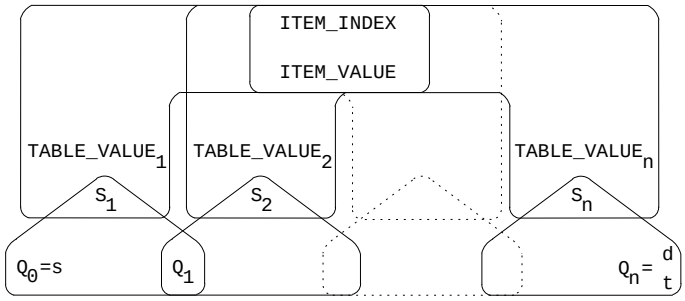

Figure 5.252: Hypergraph of the reformulation corresponding to the automaton of the `element_sparse` constraint