5.39 atmost

DESCRIPTION LINKS GRAPH AUTOMATON

Origin CHIP

Constraint atmost(N, VARIABLES, VALUE)

Synonym count.

Arguments

N	:	int
VARIABLES	:	collection(var−dvar)
VALUE	:	int

Restrictions

N ≥ 0
required(VARIABLES, var)

Purpose At most N variables of the VARIABLES collection are assigned value VALUE.

Example $(1, \langle 4, 2, 4, 5\rangle, 2)$

The atmost constraint holds since at most 1 value of the collection $\langle 4, 2, 4, 5\rangle$ is equal to value 2.

All solutions Figure 5.91 gives all solutions to the following non ground instance of the atmost constraint: $V_1 \in [1, 2]$, $V_2 \in [2, 3]$, $V_3 \in [5, 6]$, $V_4 \in [2, 3]$, atmost($\mathbf{1}, \langle V_1, V_2, V_3, V_4\rangle, \mathbf{2}$).

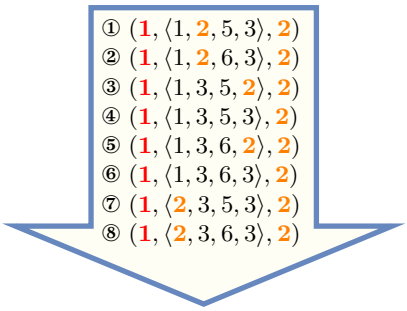


Figure 5.91: All solutions corresponding to the non ground example of the atmost constraint of the **All solutions** slot

Typical

N > 0
N < |VARIABLES|
|VARIABLES| > 1
atleast(1, VARIABLES, VALUE)

Symmetries

- Items of `VARIABLES` are permutable.
- `N` can be increased.
- An occurrence of a value of `VARIABLES.var` can be replaced by any other value that is different from `VALUE`.

Arg. properties

Contractible wrt. `VARIABLES`.

Systems

occurenceMax in **Choco**, count in **Gecode**, atmost in **Gecode**, count in **JaCoP**, at_most in **MiniZinc**, count in **SICStus**.

See also

common keyword: `among` *(value constraint)*.

comparison swapped: `atleast`.

generalisation: `cumulative` (`variable` *replaced by* `task`).

implied by: `exactly` ($\leq$`N` *replaced by* $=$`N`).

related: `roots`.

soft variant: `open_atmost` *(open constraint)*.

Keywords

characteristic of a constraint: automaton, automaton with counters.

constraint network structure: alpha-acyclic constraint network(2).

constraint type: value constraint.

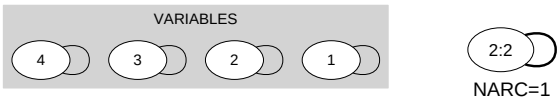
filtering: arc-consistency.

modelling: at most.

Arc input(s)	VARIABLES
Arc generator	*SELF* $\mapsto$ collection(variables)
Arc arity	1
Arc constraint(s)	variables.var = VALUE
Graph property(ies)	NARC $\leq$ N

Graph model Since each arc constraint involves only one vertex (VALUE is fixed), we employ the *SELF* arc generator in order to produce a graph with a single loop on each vertex.

Parts (A) and (B) of Figure 5.92 respectively show the initial and final graph associated with the **Example** slot. Since we use the NARC graph property, the loops of the final graph are stressed in bold.



(A) **(B)**

Figure 5.92: Initial and final graph of the atmost constraint

Automaton

Figure 5.93 depicts the automaton associated with the `atmost` constraint. To each variable $\texttt{VAR}_i$ of the collection `VARIABLES` corresponds a 0-1 signature variable S_i. The following signature constraint links $\texttt{VAR}_i$ and S_i: $\texttt{VAR}_i = \texttt{VALUE} \Leftrightarrow S_i$. The automaton counts the number of variables of the `VARIABLES` collection that are assigned value `VALUE` and finally checks that this number is less than or equal to `N`.

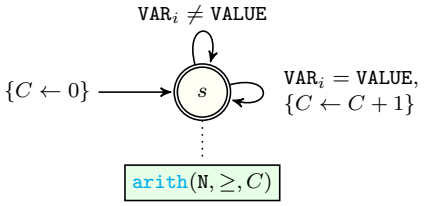


Figure 5.93: Automaton of the `atmost` constraint

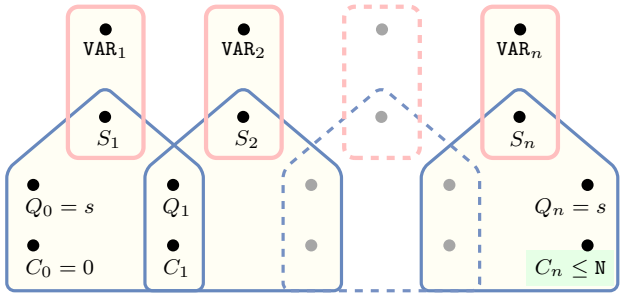


Figure 5.94: Hypergraph of the reformulation corresponding to the automaton (with one counter) of the `atmost` constraint: since all states variables $Q_0, Q_1, \ldots, Q_n$ are fixed to the unique state s of the automaton, the transitions constraints share only the counter variable C and the constraint network is Berge-acyclic