

5.20 among_modulo

	DESCRIPTION	LINKS	GRAPH	AUTOMATON
Origin	Derived from among .			
Constraint	<code>among_modulo(NVAR, VARIABLES, REMAINDER, QUOTIENT)</code>			
Arguments	<pre>NVAR : dvar VARIABLES : collection(var-dvar) REMAINDER : int QUOTIENT : int</pre>			
Restrictions	<pre>NVAR ≥ 0 NVAR ≤ VARIABLES required(VARIABLES, var) REMAINDER ≥ 0 REMAINDER < QUOTIENT QUOTIENT > 0</pre>			
Purpose	<div style="border: 1px solid pink; padding: 5px;"> <p>NVAR is the number of variables of the collection VARIABLES taking a value that is congruent to REMAINDER modulo QUOTIENT.</p> </div>			
Example	<div style="border: 1px solid blue; padding: 5px; display: inline-block;"> $(3, \langle 4, 5, 8, 4, 1 \rangle, 0, 2)$ </div> <p>In this example $\text{REMAINDER} = 0$ and $\text{QUOTIENT} = 2$ specifies that we count the number of even values taken by the different variables. As a consequence the <code>among_modulo</code> constraint holds since exactly 3 values of the collection $\langle 4, 5, 8, 4, 1 \rangle$ are even.</p>			
Typical	<pre>NVAR > 0 NVAR < VARIABLES VARIABLES > 1 QUOTIENT > 1 QUOTIENT < maxval(VARIABLES.var)</pre>			
Symmetries	<ul style="list-style-type: none"> • Items of VARIABLES are permutable. • An occurrence of a value u of VARIABLES.var such that $u \bmod \text{QUOTIENT} = \text{REMAINDER}$ (resp. $u \bmod \text{QUOTIENT} \neq \text{REMAINDER}$) can be replaced by any other value v such that $v \bmod \text{QUOTIENT} = \text{REMAINDER}$ (resp. $u \bmod \text{QUOTIENT} \neq \text{REMAINDER}$). 			
Remark	By giving explicitly all values v that satisfy the equality $v \bmod \text{QUOTIENT} = \text{REMAINDER}$, the <code>among_modulo</code> constraint can be modelled with the among constraint. However the <code>among_modulo</code> constraint provides a more compact form.			
See also	generalisation: among (list of values v such that $v \bmod \text{QUOTIENT} = \text{REMAINDER}$ replaced by list of values).			

Keywords

characteristic of a constraint: modulo, automaton, automaton with counters.

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

constraint type: value constraint, counting constraint.

filtering: arc-consistency.

Arc input(s)	VARIABLES
Arc generator	<i>SELF</i> \mapsto collection(variables)
Arc arity	1
Arc constraint(s)	variables.var mod QUOTIENT = REMAINDER
Graph property(ies)	NARC = NVAR

Graph model

The arc constraint corresponds to a unary constraint. For this reason 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 5.36 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.

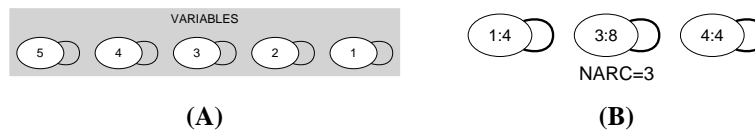


Figure 5.36: Initial and final graph of the `among_modulo` constraint

Automaton

Figure 5.37 depicts the automaton associated with the `among_modulo` constraint. To each variable VAR_i of the collection `VARIABLES` corresponds a 0-1 signature variable S_i . The following signature constraint links VAR_i and S_i : $VAR_i \bmod QUOTIENT = REMAINDER \Leftrightarrow S_i$.

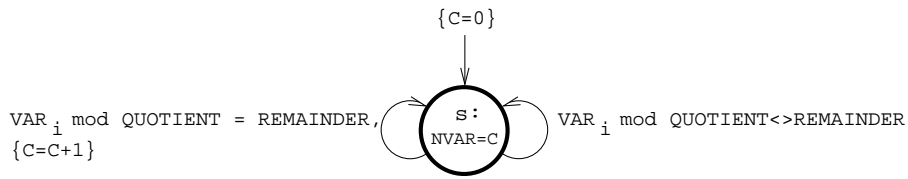


Figure 5.37: Automaton of the `among_modulo` constraint

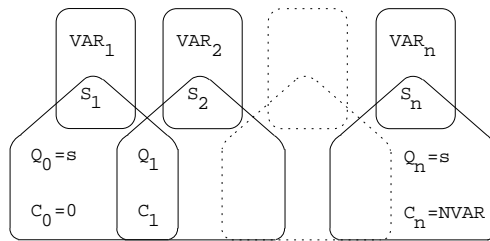


Figure 5.38: Hypergraph of the reformulation corresponding to the automaton of the `among_modulo` constraint