5.21 among_seq

DESCRIPTION LINKS GRAPH

Origin [37]

Constraint among_seq(LOW, UP, SEQ, VARIABLES, VALUES)

Synonym sequence.

Arguments

LOW	:	int
UP	:	int
SEQ	:	int
VARIABLES	:	collection(var−dvar)
VALUES	:	collection(val−int)

Restrictions

LOW ≥ 0
LOW $\leq$ |VARIABLES|
UP $\geq$ LOW
SEQ > 0
SEQ $\geq$ LOW
SEQ $\leq$ |VARIABLES|
required(VARIABLES, var)
required(VALUES, val)
distinct(VALUES, val)

Purpose Constrains all sequences of SEQ consecutive variables of the collection VARIABLES to take at least LOW values in VALUES and at most UP values in VALUES.

Example

$$\left(\begin{array}{c} \\ 1, 2, 4, \left\langle \begin{array}{l} \mathtt{var}-9, \\ \mathtt{var}-2, \\ \mathtt{var}-4, \\ \mathtt{var}-5, \\ \mathtt{var}-5, \\ \mathtt{var}-7, \\ \mathtt{var}-2 \end{array} \right\rangle, \\ \langle 0, 2, 4, 6, 8 \rangle \end{array}\right)$$

The among_seq constraint holds since the different sequences of 4 consecutive variables contains respectively 2, 2, 1 and 1 even numbers.

Typical

LOW $<$ SEQ
UP > 0
SEQ > 1
SEQ $<$ |VARIABLES|
|VARIABLES| > 1
|VALUES| > 0
|VARIABLES| $>$ |VALUES|

Symmetries

- Items of VARIABLES can be reversed.
- Items of VALUES are permutable.
- LOW can be decreased to any value ≥ 0.
- UP can be increased to any value $\leq$ SEQ.
- An occurrence of a value of VARIABLES.var that belongs to VALUES.val (resp. does not belong to VALUES.val) can be replaced by any other value in VALUES.val (resp. not in VALUES.val).

Usage

The among_seq constraint occurs in many timetabling problems. As a typical example taken from [387], consider for instance a nurse-rostering problem where each nurse can work at most 2 night shifts during every period of 7 consecutive days.

Algorithm

Beldiceanu and Carlsson [29] have proposed a first incomplete filtering algorithm for the among_seq constraint. Later on, W.-J. van Hoeve *et al.* proposed two filtering algorithms [387] establishing arc-consistency as well as an incomplete filtering algorithm based on dynamic programming concepts. In 2007 Brand *et al.* came up with a reformulation [78] that provides a complete filtering algorithm. One year later, Maher *et al.* use a reformulation in term of a linear program [244] where (1) each coefficient is an integer in $\{-1, 0, 1\}$, (2) each column has a block of consecutive 1's or -1's. From this reformulation they derive a flow model that leads to an algorithm that achieves a complete filtering in $O(n^2)$ along a branch of the search tree.

Systems

sequence in **Gecode**, sequence in **JaCoP**.

See also

generalisation: sliding_distribution *(single set of values replaced by individual values)*.

part of system of constraints: among_low_up.

root concept: among.

used in graph description: among_low_up.

Keywords

characteristic of a constraint: hypergraph.

combinatorial object: sequence.

constraint type: system of constraints, decomposition, sliding sequence constraint.

filtering: arc-consistency, linear programming, flow.

Arc input(s)	VARIABLES
Arc generator	$PATH \mapsto$ collection
Arc arity	SEQ
Arc constraint(s)	among_low_up(LOW, UP, collection, VALUES)
Graph property(ies)	$\mathbf{NARC} = \lvert\text{VARIABLES}\rvert - \text{SEQ} + 1$

Graph model A constraint on sliding sequences of consecutive variables. Each vertex of the graph corresponds to a variable. Since they link SEQ variables, the arcs of the graph correspond to hyperarcs. In order to link SEQ consecutive variables we use the arc generator $PATH$. The constraint associated with an arc corresponds to the among_low_up constraint defined at another entry of this catalogue.

Signature Since we use the $PATH$ arc generator with an arity of SEQ on the items of the VARIABLES collection, the expression $\lvert\text{VARIABLES}\rvert - \text{SEQ} + 1$ corresponds to the maximum number of arcs of the final graph. Therefore we can rewrite the graph property $\mathbf{NARC} = \lvert\text{VARIABLES}\rvert - \text{SEQ} + 1$ to $\mathbf{NARC} \geq \lvert\text{VARIABLES}\rvert - \text{SEQ} + 1$ and simplify $\overline{\mathbf{NARC}}$ to $\overline{\mathbf{NARC}}$.