

## 5.28 among\_seq

	DESCRIPTION	LINKS	GRAPH
Origin	[41]		
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 $\geq$ 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	(1, 2, 4, $\langle$ 9, 2, 4, 5, 5, 7, 2 $\rangle$ , $\langle$ 0, 2, 4, 6, 8 $\rangle$ )		
	The among_seq constraint holds since the different sequences of 4 consecutive variables contains respectively 2, 2, 1 and 1 even numbers.		
All solutions	Figure 5.76 gives all solutions to the following non ground instance of the among_seq constraint: $V_1 \in [1, 2]$ , $V_2 \in [8, 9]$ , $V_3 \in [5, 6]$ , $V_4 \in [2, 3]$ , among_seq(0, 1, 2, $\langle$ V <sub>1</sub> , V <sub>2</sub> , V <sub>3</sub> , V <sub>4</sub> $\rangle$ , $\langle$ 0, 2, 4, 6, 8 $\rangle$ ).		
Typical	LOW < SEQ UP > 0 SEQ > 1 SEQ <  VARIABLES   VARIABLES  > 1  VALUES  > 0  VARIABLES  >  VALUES  LOW > 0 $\vee$ UP < SEQ		

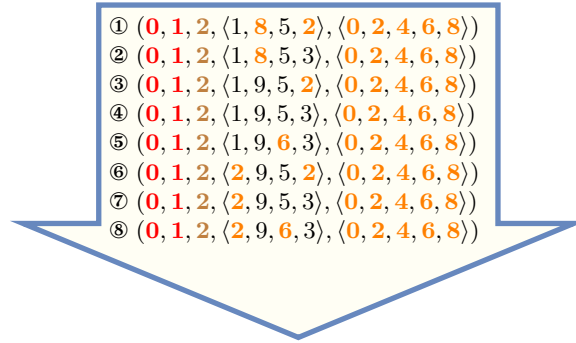


Figure 5.76: All solutions corresponding to the non ground example of the `among_seq` constraint of the **All solutions** slot, where each sequence of two consecutive variables (`SEQ = 2`) does not contain more than one occurrence (`LOW = 0`, `UP = 1`) of values 0, 2, 4, 6, 8

#### Symmetries

- Items of `VARIABLES` can be [reversed](#).
- Items of `VALUES` are [permutable](#).
- `LOW` can be [decreased](#) to any value  $\geq 0$ .
- `UP` can be [increased](#) to any value  $\leq \text{SEQ}$ .
- 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`).

#### Arg. properties

- [Contractible](#) wrt. `VARIABLES` when `UP = 0`.
- [Contractible](#) wrt. `VARIABLES` when `SEQ = 1`.
- [Prefix-contractible](#) wrt. `VARIABLES`.
- [Suffix-contractible](#) wrt. `VARIABLES`.

#### Usage

The `among_seq` constraint occurs in many timetabling problems. As a typical example taken from [426], 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 [30] have proposed a first incomplete filtering algorithm for the `among_seq` constraint. Later on, W.-J. van Hoes *et al.* proposed two filtering algorithms [426] 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 [87] that provides a complete filtering algorithm. One year later, Maher *et al.* use a reformulation in term of a [linear program](#) [273] 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](#).

<b>Arc input(s)</b>	VARIABLES
<b>Arc generator</b>	<i>PATH</i> $\mapsto$ collection
<b>Arc arity</b>	SEQ
<b>Arc constraint(s)</b>	among_low_up(LOW, UP, collection, VALUES)
<b>Graph property(ies)</b>	<b>NARC</b> = $ VARIABLES  - SEQ + 1$
<b>Graph model</b>	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 <i>PATH</i> . The constraint associated with an arc corresponds to the among_low_up constraint defined at another entry of this catalogue.
<b>Signature</b>	Since we use the <i>PATH</i> arc generator with an arity of SEQ on the items of the VARIABLES collection, the expression $ VARIABLES  - SEQ + 1$ corresponds to the maximum number of arcs of the final graph. Therefore we can rewrite the graph property <b>NARC</b> = $ VARIABLES  - SEQ + 1$ to <b>NARC</b> $\geq  VARIABLES  - SEQ + 1$ and simplify <u><b>NARC</b></u> to <b>NARC</b> .