5.253 min_dist_between_inflexion

DESCRIPTION LINKS AUTOMATON

Origin

Derived from `inflexion`

Constraint

`min_dist_between_inflexion(MINDIST, VARIABLES)`

Arguments

MINDIST	:	int
VARIABLES	:	collection(var−dvar)

Restrictions

MINDIST ≥ 0
MINDIST $\leq$ |VARIABLES|
required(VARIABLES, var)

Purpose

Given an integer value MINDIST and a sequence of variables VARIABLES enforce MINDIST to be greater than or equal to the smallest distance between two consecutive inflexions in the sequence VARIABLES, or to |VARIABLES| if no more than one inflexion exists.

An *inflexion* of a sequence of variables VARIABLES is a set of consecutive variables $V_i, V_{i+1}, \ldots, V_{j-1}, V_j$ $(i+1<j)$ such that one of the following conditions holds:

- $V_i < V_{i+1} \wedge V_{i+1} = \cdots = V_{j-1} \wedge V_{j-1} > V_j$,
- $V_i > V_{i+1} \wedge V_{i+1} = \cdots = V_{j-1} \wedge V_{j-1} < V_j$.

In this context, the index j is the *position* of the inflexion (i.e., the first instant when the inflexion is discovered when scanning the sequence of variables VARIABLES from left to right. The *distance between two consecutive inflexions* is the absolute value of the difference of their corresponding positions.

Example

$(2, \langle 2, 2, 3, 3, 2, 2, 1, 4, 4, 3\rangle)$

Figure 5.531 shows the three inflexions associated with the sequence 2, 2, 3, 3, 2, 2, 1, 4, 4, 3 and their respective positions 5, 8 and 10 in red. The `min_dist_between_inflexion` constraint holds since its first argument MINDIST $=2$ is greater than or equal to the smallest distance 2 between two consecutive inflexions of the sequence of variables VARIABLES.

Typical

MINDIST > 1
|VARIABLES| > 3
range(VARIABLES.var) > 1

Symmetries

- Items of VARIABLES can be reversed.
- One and the same constant can be added to the var attribute of all items of VARIABLES.

Counting

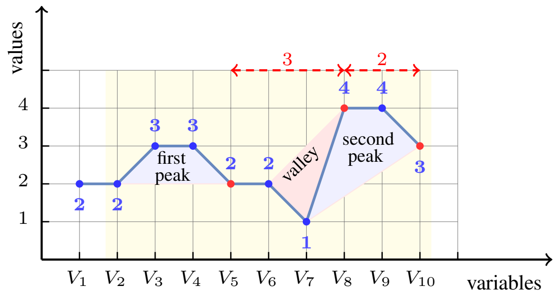
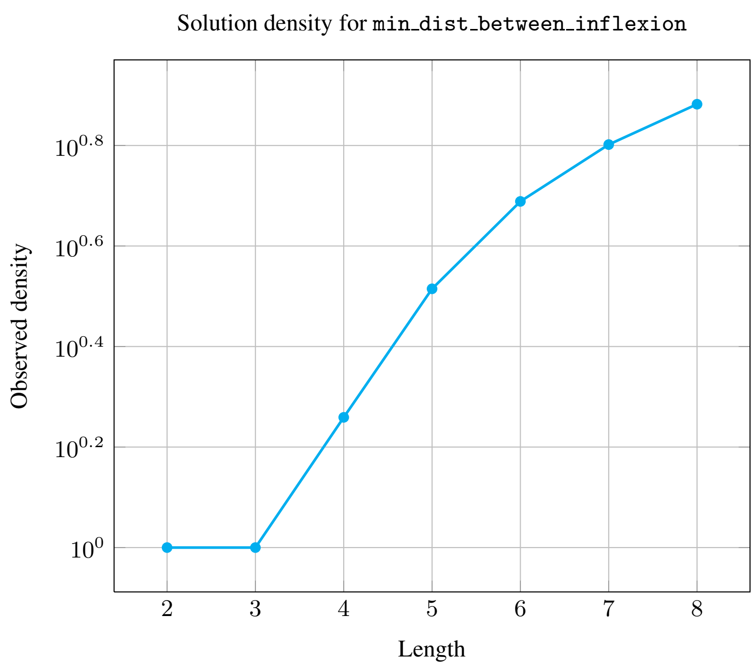
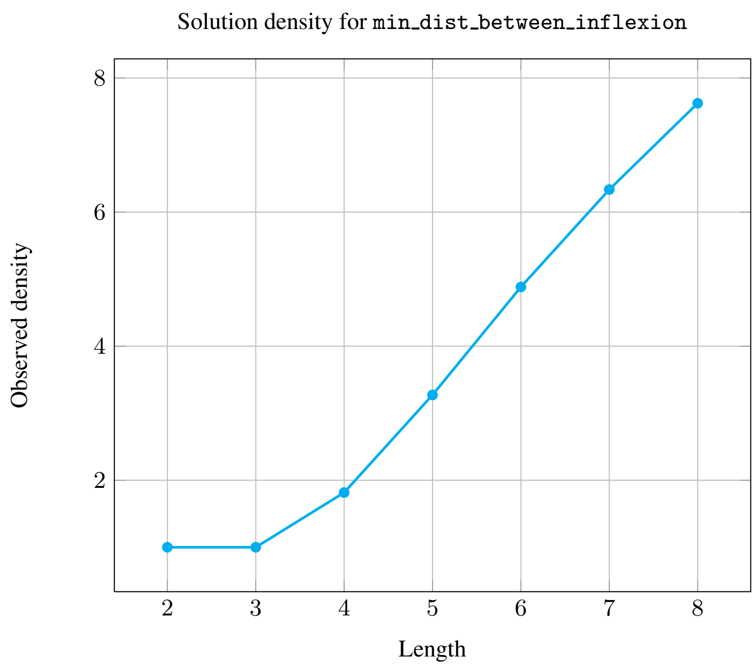


Figure 5.531: Illustration of the **Example** slot: a sequence of ten variables V_1, V_2, V_3, V_4, V_5, V_6, V_7, V_8, V_9, V_{10} respectively fixed to values 2, 2, 3, 3, 2, 2, 1, 4, 4, 3 and its three inflexions, two peaks and one valley; each red point denotes an instant where a new inflexion is discovered while scanning the sequence from left to right; as shown by the rightmost arrow, the minimum distance between two consecutive inflexions is equal to 2.

Length (n)	2	3	4	5	6	7	8
Solutions	9	64	1135	25444	574483	13287476	328156407

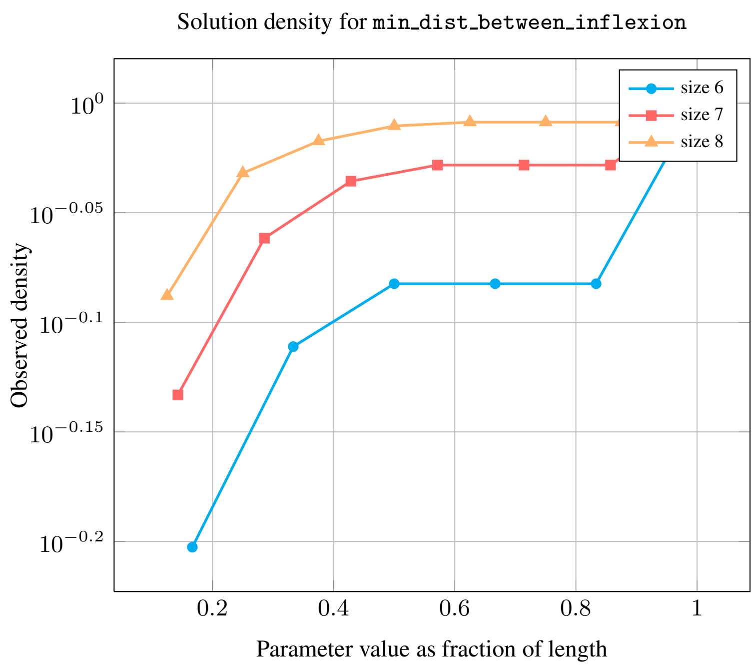
Number of solutions for `min_dist_between_inflexion`: domains $0..n$

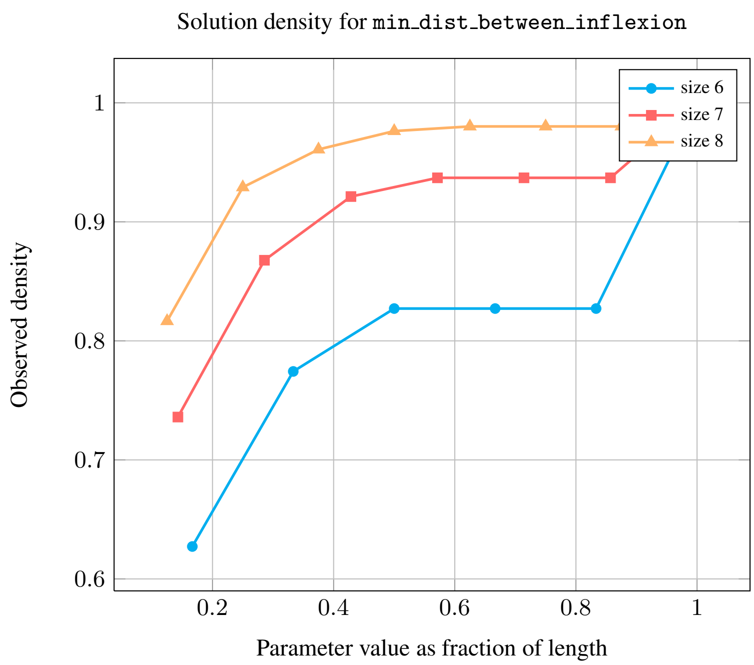




Length (n)		2	3	4	5	6	7	8
Total		9	64	1135	25444	574483	13287476	328156407
Parameter value	1	-	-	170	3598	73794	1543512	35152278
	2	9	-	170	4690	91098	1819764	39992562
	3	-	64	170	4690	97314	1932012	41360676
	4	-	-	625	4690	97314	1965012	42025560
	5	-	-	-	7776	97314	1965012	42192870
	6	-	-	-	-	117649	1965012	42192870
	7	-	-	-	-	-	2097152	42192870
	8	-	-	-	-	-	-	43046721

Solution count for `min_dist_between_inflexion`: domains $0..n$





See also **common keyword:** `inflexion`, `longest_decreasing_sequence`, `longest_increasing_sequence`, `peak`, `valley` *(sequence)*.

Keywords

characteristic of a constraint: automaton, automaton with counters, automaton with same input symbol.

combinatorial object: sequence.

constraint network structure: sliding cyclic(1) constraint network(3).

Automaton Figure 5.532 depicts the automaton associated with the `min_dist_between_inflexion` constraint.

STATES SEMANTICS

s	: stationary mode	$(=^*)$
$i0$	: increasing mode (no inflexion yet found)	$(< \{< \mid =\}^*)$
$d0$	: decreasing mode (no inflexion yet found)	$(> \{> \mid =\}^*)$
$i1$	: increasing mode (at least one inflexion already found)	$(< \{< \mid =\}^*)$
$d1$	: decreasing mode (at least one inflexion already found)	$(> \{> \mid =\}^*)$

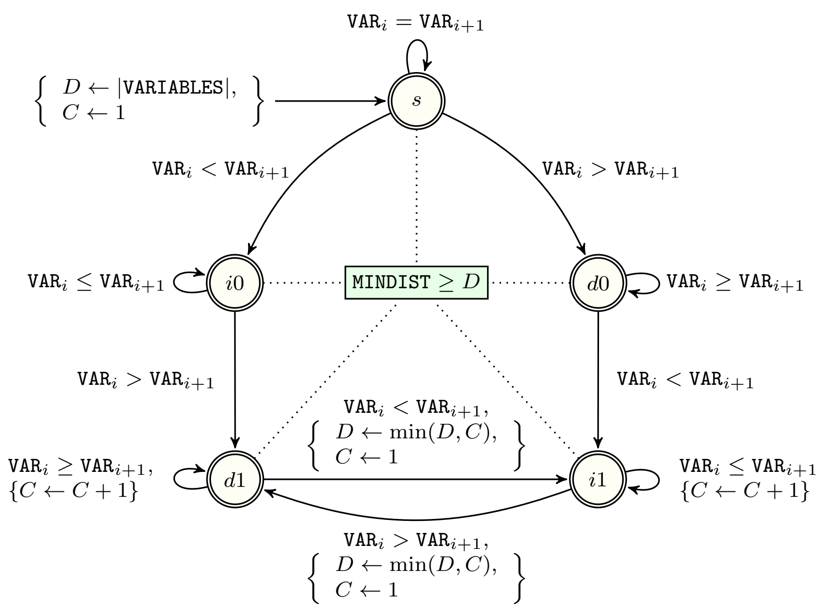


Figure 5.532: Automaton of the `min_dist_between_inflexion` constraint (state s means that we are in *stationary* mode, state $i0$ means that we are in *increasing* mode and that we did not yet found any inflexion, state $d0$ means that we are in *decreasing* mode and that we did not yet found any inflexion, state $i1$ means that we are in *increasing* mode and that we already found at least one inflexion, state $d1$ means that we are in *decreasing* mode and that we already found at least one inflexion, the minimum distance between two consecutive inflexions is updated each time we switch from $i1$ to $d1$ mode – or conversely from $d1$ to $i1$ mode – and the counter D is updated accordingly)

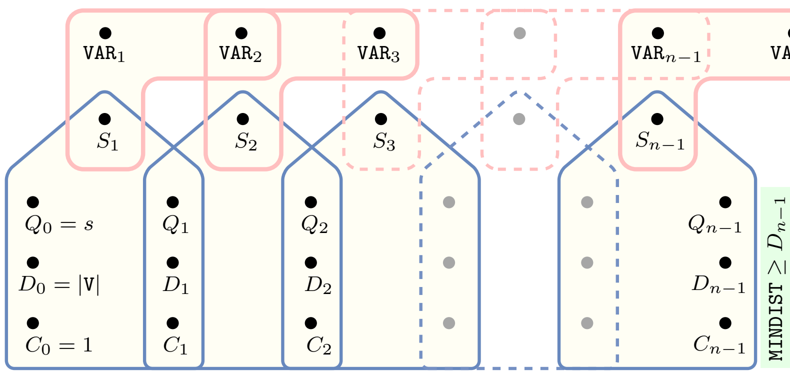


Figure 5.533: Hypergraph of the reformulation corresponding to the automaton of the min_dist_between_inflexion constraint where V is a shortcut for VARIABLES (since all states of the automaton are accepting there is no restriction on the last variable Q_{n-1})