5.191 increasing_valley

DESCRIPTION LINKS AUTOMATON

Origin

Derived from `valley` and `increasing`.

Constraint

`increasing_valley(VARIABLES)`

Argument

`VARIABLES : collection(var−dvar)`

Restrictions

$|\texttt{VARIABLES}| > 0$
`required(VARIABLES, var)`

Purpose

A variable V_k $(1 < k < m)$ of the sequence of variables $\texttt{VARIABLES} = V_1, \ldots, V_m$ is a *valley* if and only if there exists an i $(1 < i \leq k)$ such that $V_{i-1} > V_i$ and $V_i = V_{i+1} = \cdots = V_k$ and $V_k < V_{k+1}$.
When considering all the valleys of the sequence `VARIABLES` from left to right enforce all valleys to be increasing, i.e. the altitude of each valley is greater than or equal to the altitude of its preceding valley when it exists.

Example

$(\langle 3, 5, 1, 4, 3, 5, 3, 3, 7, 2\rangle)$

The `increasing_valley` constraint holds since the sequence 3 5 **1** 4 **3** 5 3 **3** 7 2 contains three valleys, in bold, that are increasing.

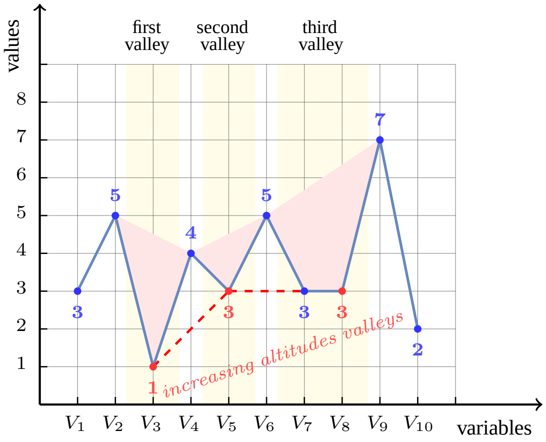


Figure 5.441: Illustration of the **Example** slot: a sequence of ten variables V_1, V_2, V_3, V_4, V_5, V_6, V_7, V_8, V_9, V_{10} respectively fixed to values 3, 5, 1, 4, 3, 5, 3, 3, 7, 2 and its corresponding three valleys, in red, respectively located at altitudes 1, 3 and 3

Typical

$|\texttt{VARIABLES}| \geq 7$
$\texttt{range}(\texttt{VARIABLES.var}) > 1$
$\texttt{valley}(\texttt{VARIABLES.var}) \geq 3$

Symmetry

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

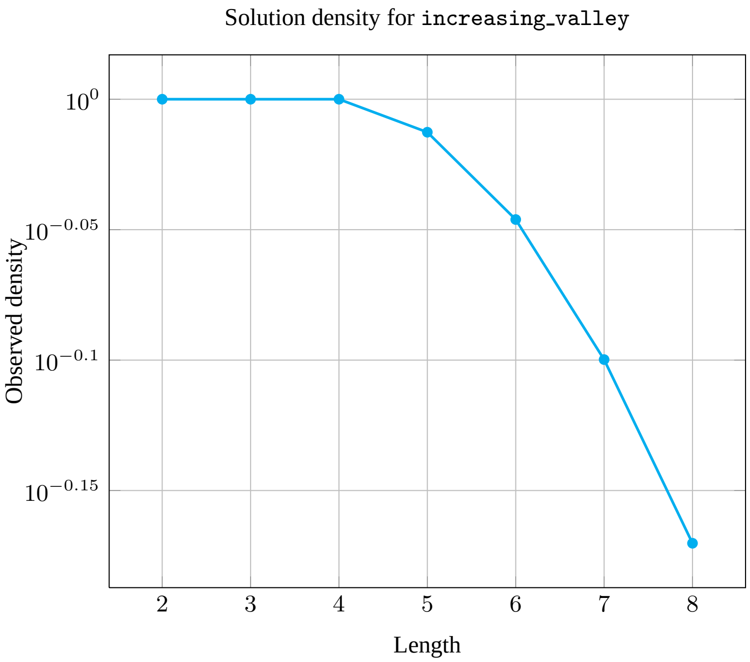
Arg. properties

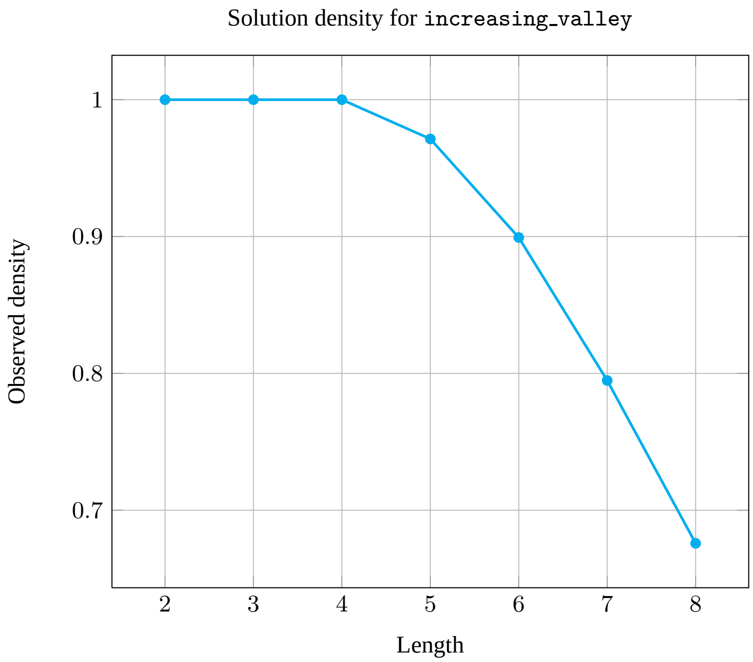
- Prefix-contractible wrt. `VARIABLES`.
- Suffix-contractible wrt. `VARIABLES`.

Counting

Length (n)	2	3	4	5	6	7	8
Solutions	9	64	625	7553	105798	1666878	29090469

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





See also **implied by:** `all_equal_valley`.

related: `decreasing_valley`, `valley`.

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(2).

Cond. implications `increasing_valley`(`VARIABLES`)
with `valley`(`VARIABLES.var`) > 0
implies `not_all_equal`(`VARIABLES`).

Automaton Figure 5.442 depicts the automaton associated with the `increasing_valley` constraint. To each pair of consecutive variables $(\texttt{VAR}_i, \texttt{VAR}_{i+1})$ of the collection `VARIABLES` corresponds a signature variable S_i. The following signature constraint links $\texttt{VAR}_i$, $\texttt{VAR}_{i+1}$ and S_i: $(\texttt{VAR}_i < \texttt{VAR}_{i+1} \Leftrightarrow S_i = 0) \wedge (\texttt{VAR}_i = \texttt{VAR}_{i+1} \Leftrightarrow S_i = 1) \wedge (\texttt{VAR}_i > \texttt{VAR}_{i+1} \Leftrightarrow S_i = 2)$.

STATES SEMANTICS

s	: initial stationary or increasing mode	$(\{= \mid <\}^*)$
u	: decreasing (before first potential valley) mode	$(> \{> \mid =\}^*)$
v	: increasing (after a valley) mode	$(< \{< \mid =\}^*)$
w	: decreasing (after a valley) mode	$(> \{> \mid =\}^*)$

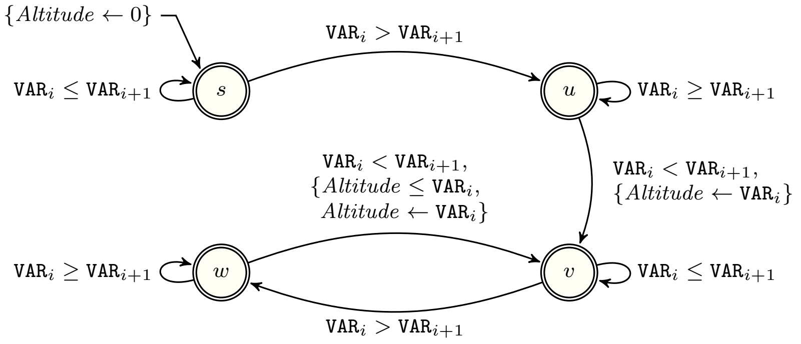

$\{Altitude \leftarrow 0\}$
$\texttt{VAR}_i > \texttt{VAR}_{i+1}$
$\texttt{VAR}_i \leq \texttt{VAR}_{i+1}$
s
u
$\texttt{VAR}_i \geq \texttt{VAR}_{i+1}$
$\texttt{VAR}_i < \texttt{VAR}_{i+1}$, $\{Altitude \leq \texttt{VAR}_i$, $Altitude \leftarrow \texttt{VAR}_i\}$
$\texttt{VAR}_i < \texttt{VAR}_{i+1}$, $\{Altitude \leftarrow \texttt{VAR}_i\}$
$\texttt{VAR}_i \geq \texttt{VAR}_{i+1}$
w
v
$\texttt{VAR}_i \leq \texttt{VAR}_{i+1}$
$\texttt{VAR}_i > \texttt{VAR}_{i+1}$

Figure 5.442: Automaton for the `increasing_valley` constraint (note the conditional transition from state w to state v testing that the counter $Altitude$ is less than or equal to $\texttt{VAR}_i$ for enforcing that all valleys from left to right are in increasing altitude)

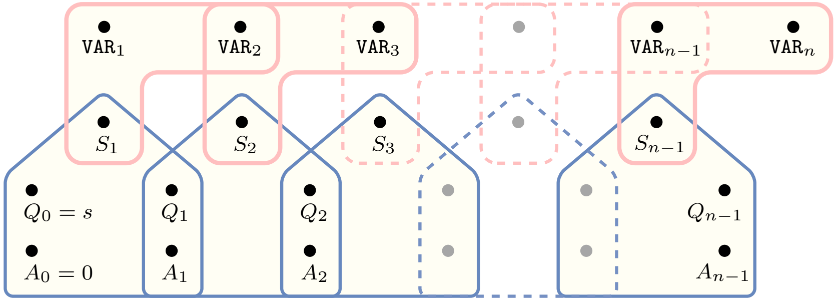


Figure 5.443: Hypergraph of the reformulation corresponding to the automaton of the `increasing_valley` constraint where A_i stands for the value of the counter $Altitude$ (since all states of the automaton are accepting there is no restriction on the last variable Q_{n-1})