5.9 all_equal_valley_min

DESCRIPTION LINKS AUTOMATON

Origin Derived from valley and all_equal.

Constraint all_equal_valley_min(VARIABLES)

Argument VARIABLES : collection(var−dvar)

Restrictions
|VARIABLES| > 0
required(VARIABLES, var)

Purpose A variable V_k $(1 < k < m)$ of the sequence of variables 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 $V_k < V_{k+1}$.
Enforce all the valleys of the sequence VARIABLES to be assigned the same value, i.e. to be located at the same altitude corresponding to the minimum value of the sequence VARIABLES.

Example $(\langle 2, 5, 5, 4, 2, 2, 6, 2, 7\rangle)$

The all_equal_valley_min constraint holds since the two valleys, in bold, of the sequence 2 5 5 4 2 **2** 6 **2** 7 are located at the same altitude 2 that is also the minimum value of the sequence 2 5 5 4 2 2 6 2 7. Figure 5.18 depicts the solution associated with the example.

Note that the all_equal_valley_min constraint does not enforce that the sequence VARIABLES contains at least one valley.

Typical
|VARIABLES| ≥ 5
range(VARIABLES.var) > 1
valley(VARIABLES.var) ≥ 2

Symmetries
- Items of VARIABLES can be reversed.
- 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	605	6707	81648	1065542	14829903

Number of solutions for all_equal_valley_min: domains $0..n$

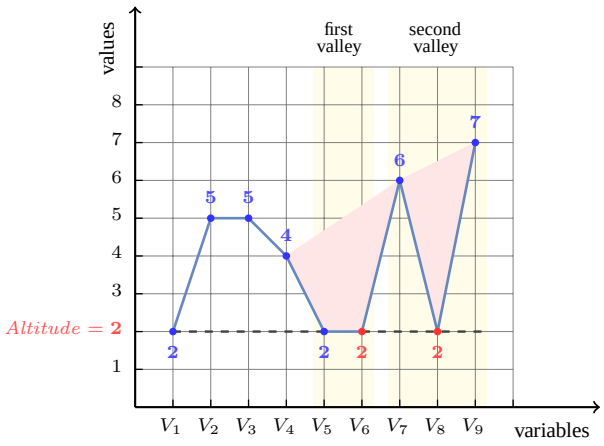
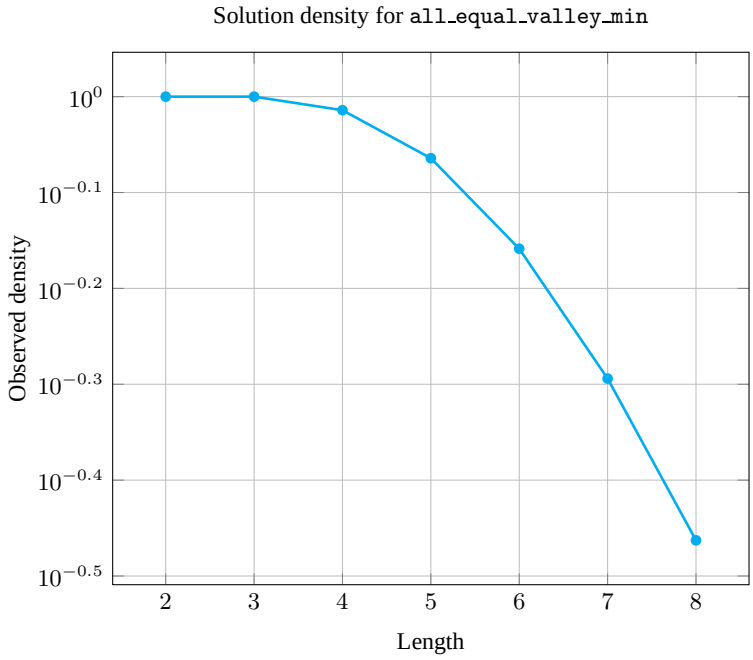
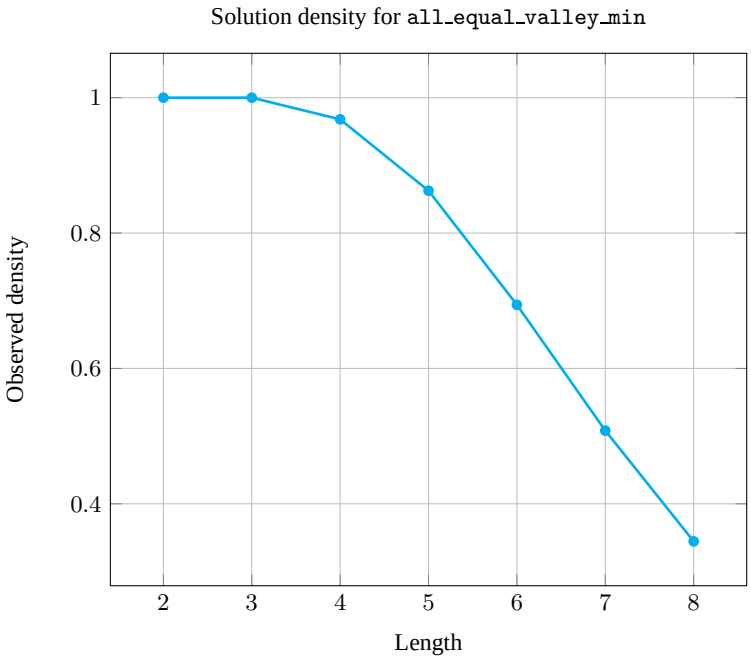


Figure 5.18: Illustration of the **Example** slot: a sequence of nine variables V_1, V_2, V_3, V_4, V_5, V_6, V_7, V_8, V_9 respectively fixed to values 2, 5, 5, 4, 2, 2, 6, 2, 7 and its corresponding two valleys, in red, both located at altitude 2 that also corresponds to the minimum value of the sequence





See also **implied by:** no_valley.

implies: all_equal_valley.

related: all_equal_peak_max, 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

- all_equal_valley_min(VARIABLES) with valley(VARIABLES.var) > 1 **implies** some_equal(VARIABLES).
- all_equal_valley_min(VARIABLES) with valley(VARIABLES.var) > 0 **implies** not_all_equal(VARIABLES).

Automaton

Figure 5.19 depicts the automaton associated with the `all_equal_valley_min` 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: $(\texttt{VAR}_i < \texttt{VAR}_{i+1} \Leftrightarrow S_i = 0) \wedge (\texttt{VAR}_i = \texttt{VAR}_{i+1} \Leftrightarrow S_i = 1) \wedge (\texttt{VAR}_i > \texttt{VAR}_{i+1} \Leftrightarrow S_i = 2)$.

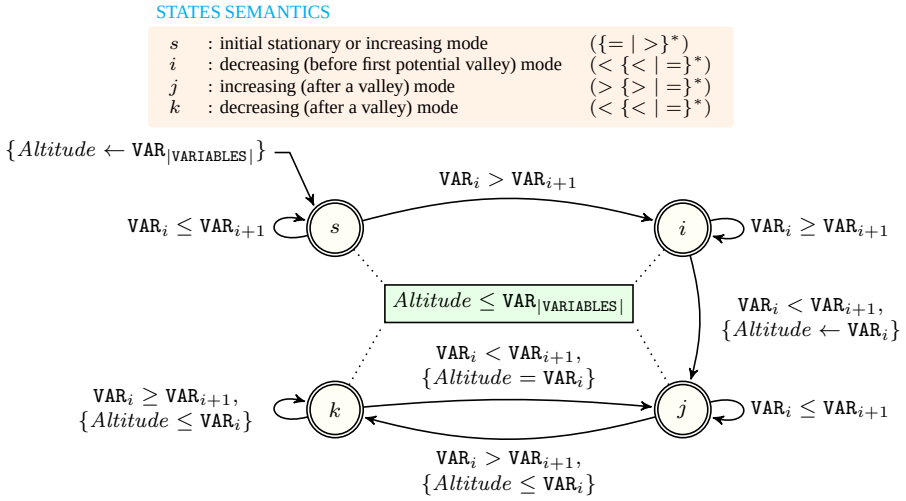


Figure 5.19: Automaton for the `all_equal_valley_min` constraint; note the conditional transition from state k to state j testing that the counter $Altitude$ is equal to $\texttt{VAR}_i$ for enforcing that all valleys are located at the same altitude; the conditional transitions from j to k and from k to k and the final check $Altitude \leq \texttt{VAR}_{|\texttt{VARIABLES}|}$ enforce the minimum value of the sequence `VARIABLES` to not be located below the altitude of the eventual valleys.

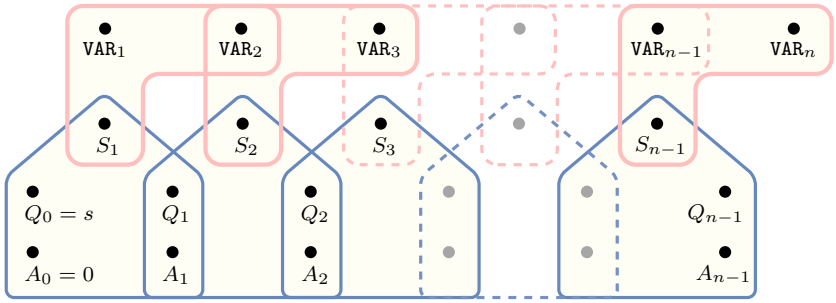


Figure 5.20: Hypergraph of the reformulation corresponding to the automaton of the `all_equal_valley_min` constraint where A stands for the value of the counter $Altitude$ (since all states of the automaton are accepting there is no restriction on the last variable Q_{n-1})