Global Constraint Catalog

PDF PL PNG XML

5.417. valley

DESCRIPTION LINKS AUTOMATON

Origin

Derived from inflexion.

Constraint

valley(N, VARIABLES)

Arguments

N	dvar
VARIABLES	collection(var − dvar)

Restrictions

$N \geq 0$
$2 * N \leq \max(|VARIABLES| - 1, 0)$
required(VARIABLES, var)

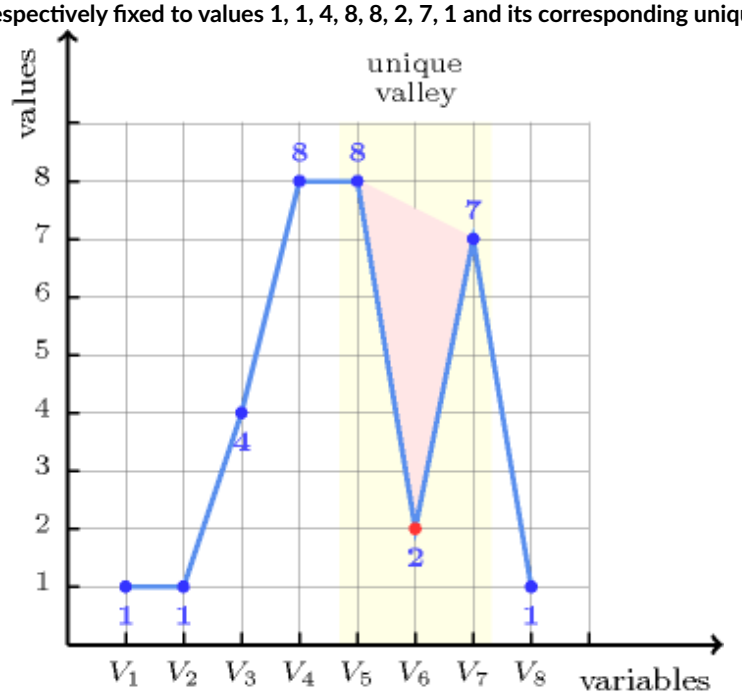
Purpose

A variable V_v $(1 < v < m)$ of the sequence of variables $VARIABLES = V_1, \cdots, V_m$ is a *valley* if and only if there exists an i (with $1 < i \leq v$) such that $V_{i-1} > V_i$ and $V_i = V_{i+1} = \cdots = V_v$ and $V_v < V_{v+1}$. N is the total number of valleys of the sequence of variables VARIABLES.

Example

$(1, \langle 1, 1, 4, 8, 8, 2, 7, 1 \rangle)$
$(0, \langle 1, 1, 4, 5, 8, 8, 4, 1 \rangle)$
$(4, \langle 1, 0, 4, 0, 8, 2, 4, 1, 2 \rangle)$
The first valley constraint holds since the sequence 11488271 contains one valley that corresponds to the variable that is assigned to value 2.

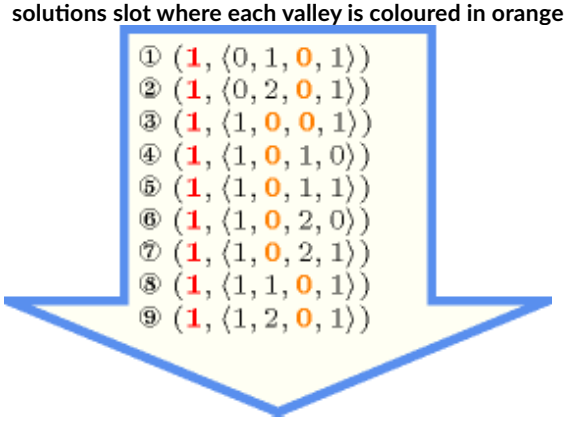
Figure 5.417.1. Illustration of the first example of the Example slot: a sequence of eight variables V_1, V_2, V_3, V_4, V_5, V_6, V_7, V_8 respectively fixed to values 1, 1, 4, 8, 8, 2, 7, 1 and its corresponding unique valley ($\mathtt{N} = 1$)



All solutions

Figure 5.417.2 gives all solutions to the following non ground instance of the `valley` constraint: $\mathtt{N} \in [1, 2]$, $V_1 \in [0, 1]$, $V_2 \in [0, 2]$, $V_3 \in [0, 2]$, $V_4 \in [0, 1]$, $\mathtt{valley}(\mathtt{N}, \langle V_1, V_2, V_3, V_4 \rangle)$.

Figure 5.417.2. All solutions corresponding to the non ground example of the `valley` constraint of the All solutions slot where each valley is coloured in orange



Typical

$|\mathtt{VARIABLES}| > 2$
$\mathtt{range}(\mathtt{VARIABLES}.\mathtt{var}) > 1$

Symmetries

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

Arg. properties

- Functional dependency: `N` determined by `VARIABLES`.
- Contractible wrt. `VARIABLES` when $\texttt{N} = 0$.

Usage

Useful for constraining the number of *valleys* of a sequence of domain variables.

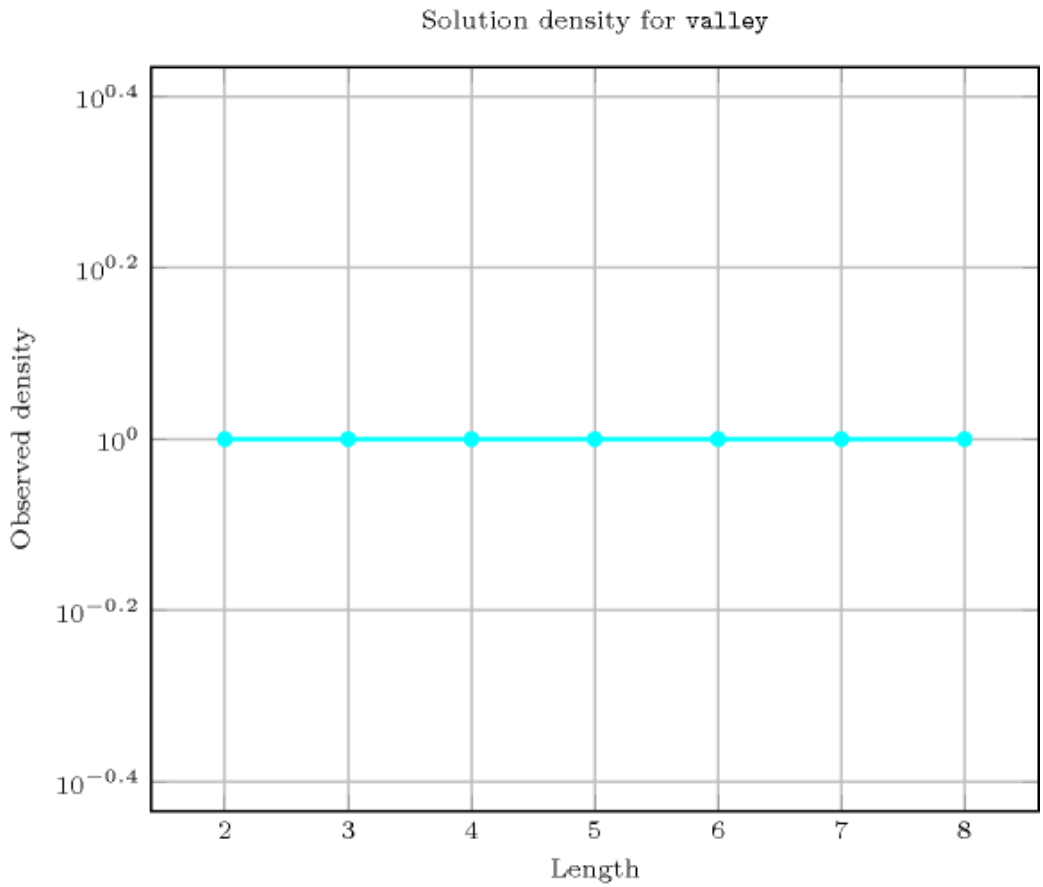
Remark

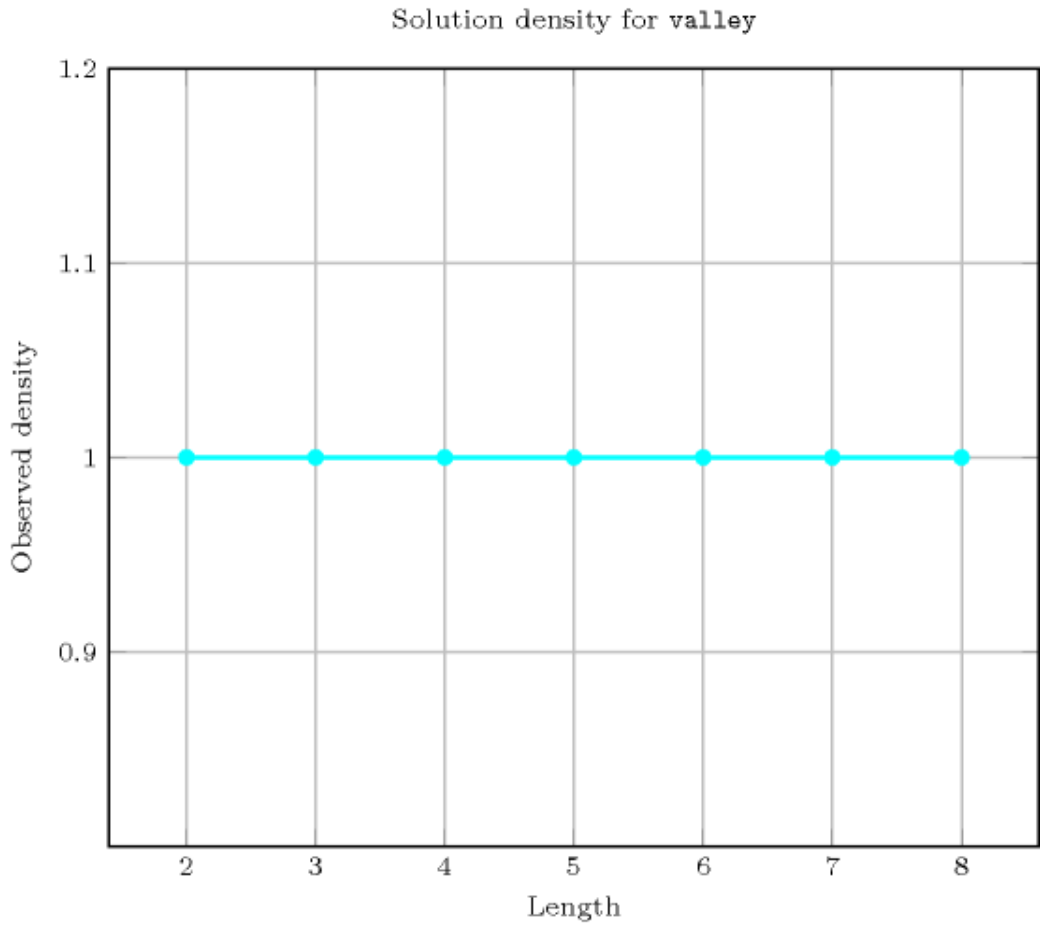
Since the arity of the arc constraint is not fixed, the `valley` constraint cannot be currently described with the graph-based representation. However, this would not hold anymore if we were introducing a slot that specifies how to merge adjacent vertices of the final graph.

Counting

Length (n)	2	3	4	5	6	7	8
Solutions	9	64	625	7776	117649	2097152	43046721

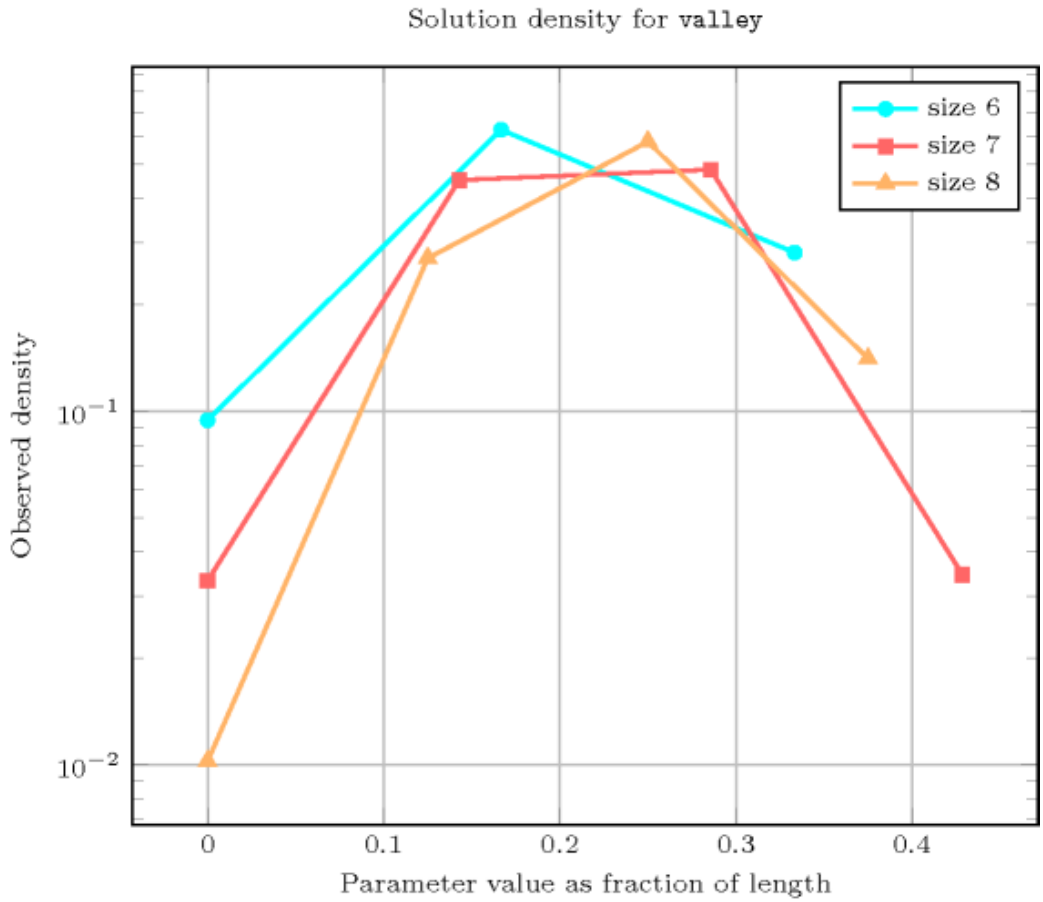
Number of solutions for `valley`: domains $0..n$

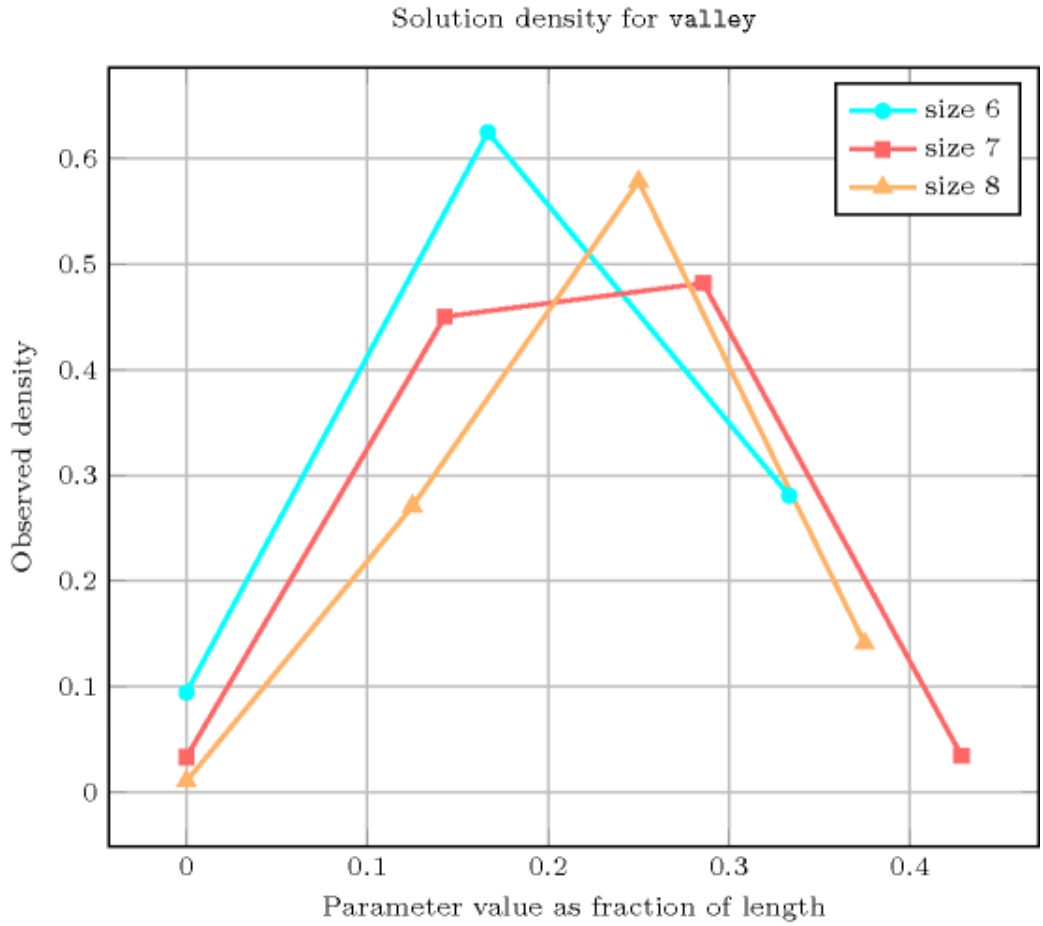




Length (n)	2	3	4	5	6	7	8	
Total	9	64	625	7776	117649	2097152	43046721	
Parameter value	0	9	50	295	1792	11088	69498	439791
	1	-	14	330	5313	73528	944430	11654622
	2	-	-	-	671	33033	1010922	24895038
	3	-	-	-	-	-	72302	6057270

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


Solution density for valley
size 6
size 7
size 8
Observed density
10^{-1}
10^{-2}
0
0.1
0.2
0.3
0.4
Parameter value as fraction of length


Solution density for valley
size 6
size 7
size 8
Observed density
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.1
0.2
0.3
0.4
Parameter value as fraction of length

See also

common keyword: deepest_valley, inflexion, min_dist_between_inflexion, min_width_valley *(sequence)*.

comparison swapped: peak.

generalisation: big_valley *(a tolerance parameter is added for counting only big valleys)*.

related: all_equal_valley, all_equal_valley_min, decreasing_valley, increasing_valley, no_peak.

specialisation: no_valley *(the variable counting the number of valleys is set to 0 and removed)*.

Keywords

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

combinatorial object: sequence.

constraint arguments: reverse of a constraint, pure functional dependency.

constraint network structure: sliding cyclic(1) constraint network(2).

filtering: glue matrix.

modelling: functional dependency.

Cond. implications

- valley(N, VARIABLES)

  with $\mathtt{N} > 0$

  implies atleast_nvalue(NVAL, VARIABLES)

  when $\mathtt{NVAL} = 2$.

- valley(N, VARIABLES)

  implies inflexion(N, VARIABLES)

  when $\mathtt{N} = \mathtt{peak}(\mathtt{VARIABLES.var}) + \mathtt{valley}(\mathtt{VARIABLES.var})$.

Automaton

Figure 5.417.3 depicts the automaton associated with the valley constraint. To each pair of consecutive variables $(\mathtt{VAR}_i, \mathtt{VAR}_{i+1})$ of the collection VARIABLES corresponds a signature variable S_i. The following signature constraint links $\mathtt{VAR}_i$, $\mathtt{VAR}_{i+1}$ and S_i: $(\mathtt{VAR}_i < \mathtt{VAR}_{i+1} \Leftrightarrow S_i = 0) \wedge (\mathtt{VAR}_i = \mathtt{VAR}_{i+1} \Leftrightarrow S_i = 1) \wedge (\mathtt{VAR}_i > \mathtt{VAR}_{i+1} \Leftrightarrow S_i = 2)$.

Figure 5.417.3. Automaton of the valley constraint

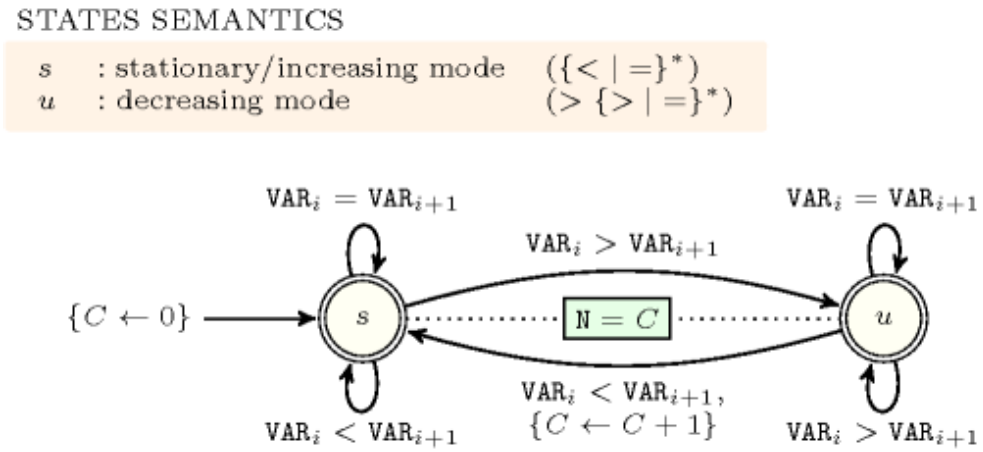


Figure 5.417.4. Hypergraph of the reformulation corresponding to the automaton of the `valley` constraint (since all states of the automaton are accepting there is no restriction on the last variable Q_{n-1})

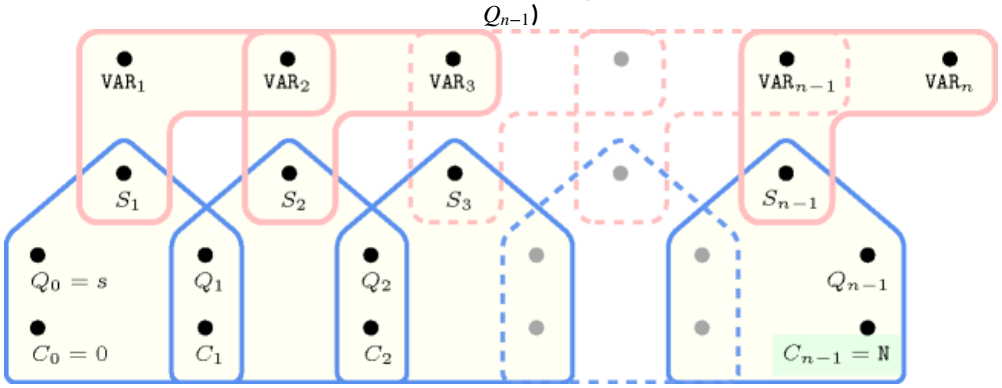


Figure 5.417.5. Glue matrix of the `valley` constraint

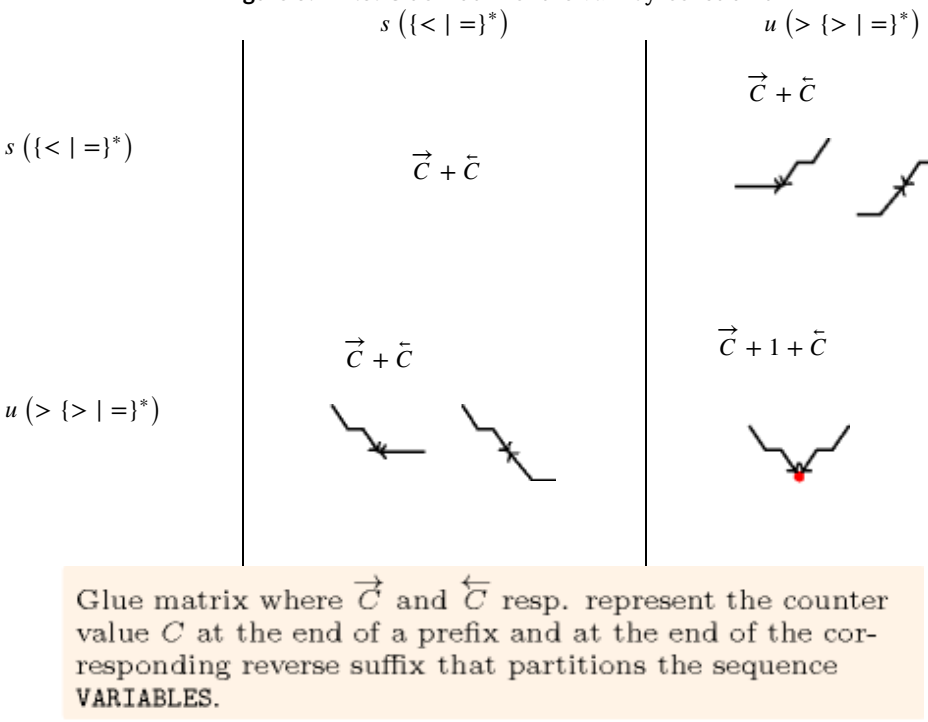
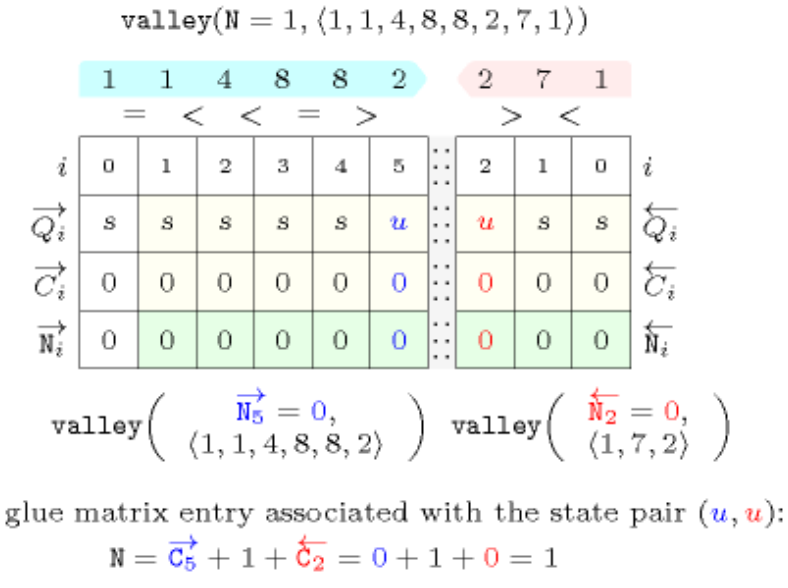


Figure 5.417.6. Illustrating the use of the state pair (u, u) of the glue matrix for linking $\mathbb{N}$ with the counters variables obtained after reading the prefix $1, 1, 4, 8, 8, 2$ and corresponding suffix $2, 7, 1$ of the sequence $1, 1, 4, 8, 8, 2, 7, 1$; note that the suffix $2, 7, 1$ (in pink) is proceed in reverse order; the left (resp. right) table shows the initialisation (for $i = 0$) and the evolution (for $i > 0$) of the state of the automaton and its counter C upon reading the prefix $1, 1, 4, 8, 8, 2$ (resp. the reverse suffix $1, 7, 2$).



W3C: XHTML - last update: 2014-6-10. SD.