

5.6 all_equal_peak

	DESCRIPTION	LINKS	AUTOMATON
Origin	Derived from peak and all_equal .		
Constraint	<code>all_equal_peak(VARIABLES)</code>		
Argument	VARIABLES : <code>collection(var-dvar)</code>		
Restrictions	$ VARIABLES > 0$ <code>required(VARIABLES, var)</code>		
Purpose	<p>A variable V_k ($1 < k < m$) of the sequence of variables $VARIABLES = V_1, \dots, V_m$ is a <i>peak</i> if and only if there exists an i ($1 < i \leq k$) such that $V_{i-1} < V_i$ and $V_i = V_{i+1} = \dots = V_k$ and $V_k > V_{k+1}$.</p> <p>Enforce all the peaks of the sequence $VARIABLES$ to be assigned the same value, i.e. to be located at the same altitude.</p>		
Example	<div style="border: 1px solid black; padding: 5px; display: inline-block;"> $((1, 5, 5, 4, 3, 5, 2, 7))$ </div>		

The `all_equal_peak` constraint holds since the two peaks, in bold, of the sequence 1 5 5 4 3 5 2 7 are located at the same altitude 5. Figure 5.7 depicts the solution associated with the example.

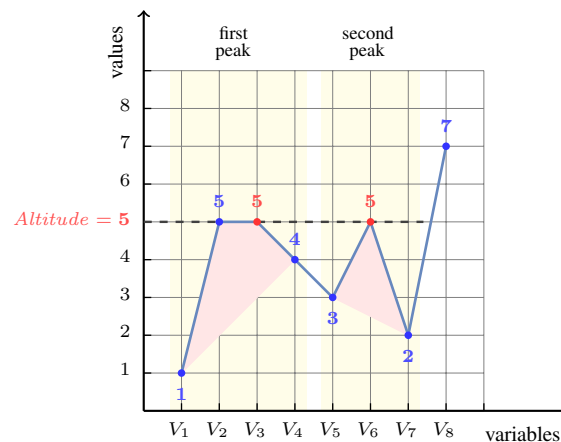


Figure 5.7: Illustration of the **Example** slot: a sequence of eight variables $V_1, V_2, V_3, V_4, V_5, V_6, V_7, V_8$ respectively fixed to values 1, 5, 5, 4, 3, 5, 2, 7 and its corresponding two peaks, in red, both located at altitude 5

Note that the `all_equal_peak` constraint does not enforce that the maximum value of the sequence $VARIABLES$ corresponds to the altitude of its peaks since, as shown by the

example, the sequence can end up with an increasing subsequence that go beyond the altitude of its peaks. It also does not enforce that the sequence VARIABLES contains at least one peak.

All solutions

Figure 5.8 gives all solutions to the following non ground instance of the `all_equal_peak` constraint: $V_1 \in \{0, 5\}$, $V_2 \in [2, 3]$, $V_3 = 2$, $V_4 \in [3, 4]$, $V_5 = 1$, `all_equal_peak`($\langle V_1, V_2, V_3, V_4, V_5 \rangle$).

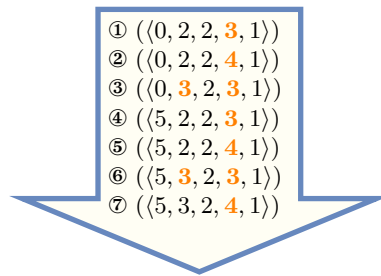


Figure 5.8: All solutions corresponding to the non ground example of the `all_equal_peak` constraint of the **All solutions** slot where each peak is coloured in orange

Typical

```
|VARIABLES| ≥ 5
range(VARIABLES.var) > 1
peak(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.

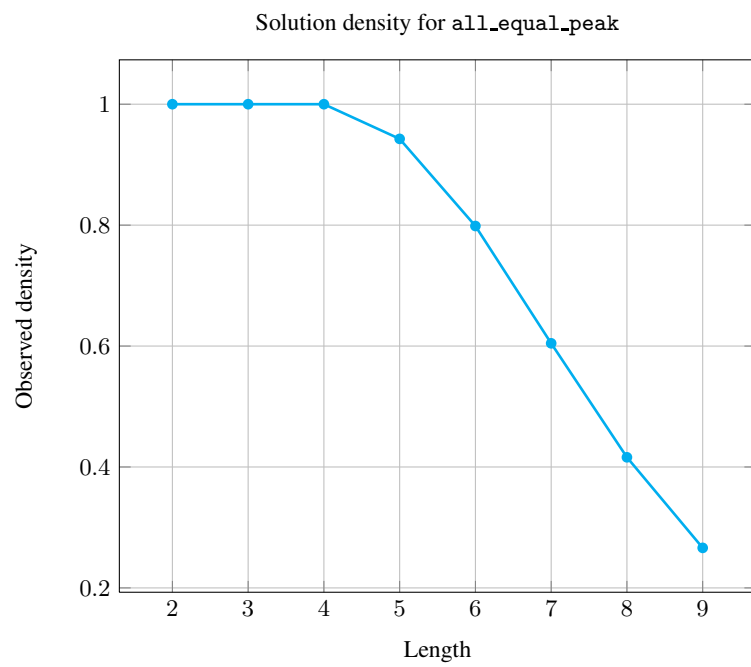
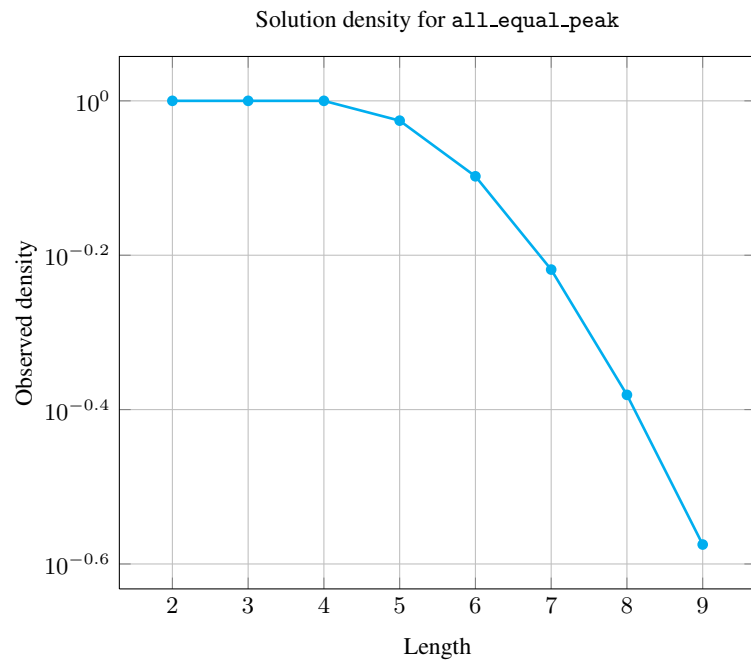
Arg. properties

- [Prefix-contractible](#) wrt. VARIABLES.
- [Suffix-contractible](#) wrt. VARIABLES.

Counting

Length (n)	2	3	4	5	6	7	8	9
Solutions	9	64	625	7330	93947	1267790	17908059	266201992

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



See also

[implied by: `all_equal_peak_max`.](#)

[implies: `decreasing_peak`, `increasing_peak`.](#)

related: `all_equal_valley`, `peak`.

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_peak(VARIABLES)`
with `peak(VARIABLES.var) > 1`
implies `some_equal(VARIABLES)`.
- `all_equal_peak(VARIABLES)`
with `peak(VARIABLES.var) > 0`
implies `not_all_equal(VARIABLES)`.

Automaton

Figure 5.9 depicts the automaton associated with the `all_equal_peak` constraint. To each pair of consecutive variables ($\text{VAR}_i, \text{VAR}_{i+1}$) of the collection `VARIABLES` corresponds a signature variable S_i . The following signature constraint links $\text{VAR}_i, \text{VAR}_{i+1}$ and S_i : $(\text{VAR}_i < \text{VAR}_{i+1} \Leftrightarrow S_i = 0) \wedge (\text{VAR}_i = \text{VAR}_{i+1} \Leftrightarrow S_i = 1) \wedge (\text{VAR}_i > \text{VAR}_{i+1} \Leftrightarrow S_i = 2)$.

STATES SEMANTICS

s	: initial stationary or decreasing mode	$(\{= >\}^*)$
i	: increasing (before first potential peak) mode	$(\{< < =\}^*)$
j	: decreasing (after a peak) mode	$(\{> > =\}^*)$
k	: increasing (after a peak) mode	$(\{< < =\}^*)$

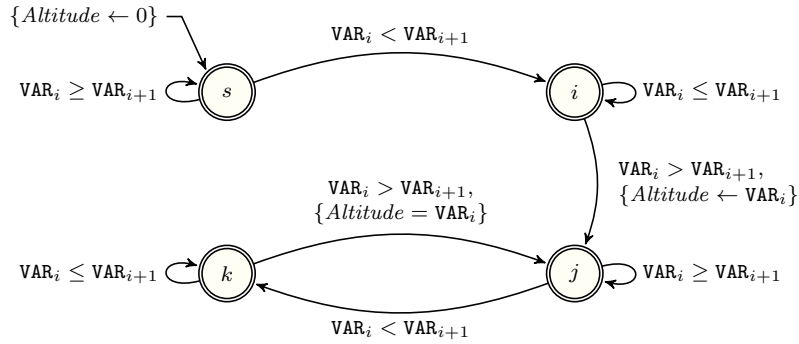


Figure 5.9: Automaton for the `all_equal_peak` constraint (note the conditional transition from state k to state j testing that the counter *Altitude* is equal to VAR_i for enforcing that all peaks are located at the same altitude)

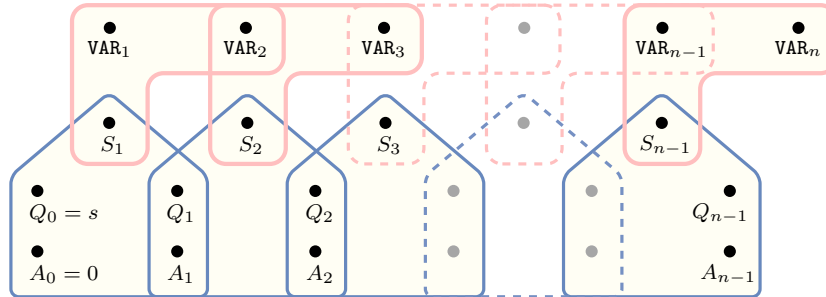


Figure 5.10: Hypergraph of the reformulation corresponding to the automaton of the `all_equal_peak` constraint where A_i stands for the value of the counter *Altitude* (since all states of the automaton are accepting there is no restriction on the last variable Q_{n-1})

20130107

483