5.18 alldifferent_modulo

DESCRIPTION LINKS GRAPH AUTOMATON

Origin

Derived from `alldifferent`.

Constraint

`alldifferent_modulo(VARIABLES, M)`

Synonyms

`alldiff_modulo`, `alldistinct_modulo`.

Arguments

`VARIABLES : collection(var−dvar)`
`M : int`

Restrictions

`required(VARIABLES, var)`
$\mathtt{M} > 0$
$\mathtt{M} \geq |\mathtt{VARIABLES}|$

Purpose

Enforce all variables of the collection `VARIABLES` to have a distinct rest when divided by `M`.

Example

$(\langle 25, 1, 14, 3\rangle, 5)$

The equivalence classes associated with values 25, 1, 14 and 3 are respectively equal to $25 \bmod 5 = 0$, $1 \bmod 5 = 1$, $14 \bmod 5 = 4$ and $3 \bmod 5 = 3$. Since they are distinct the `alldifferent_modulo` constraint holds.

Typical

$|\mathtt{VARIABLES}| > 2$
$\mathtt{M} > 1$

Symmetries

- Items of `VARIABLES` are permutable.
- A value u of `VARIABLES.var` can be renamed to any value v such that v is congruent to u modulo `M`.
- Two distinct values u and v of `VARIABLES.var` such that $u \bmod \mathtt{M} \neq v \bmod \mathtt{M}$ can be swapped.

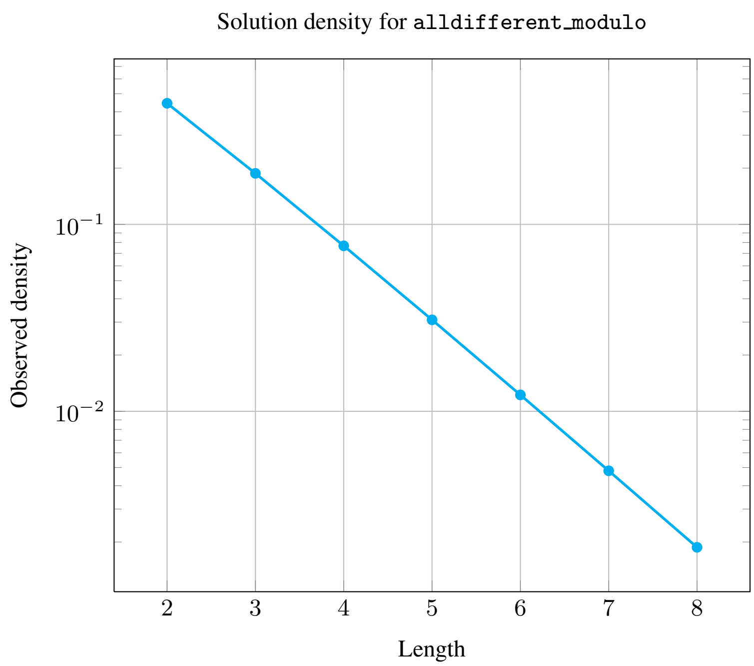
Arg. properties

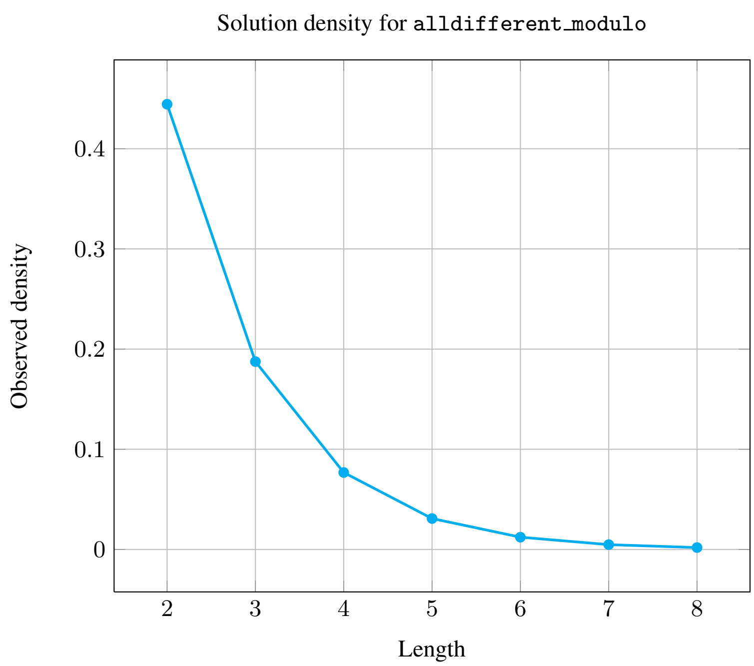
Contractible wrt. `VARIABLES`.

Counting

Length (n)	2	3	4	5	6	7	8
Solutions	4	12	48	240	1440	10080	80640

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

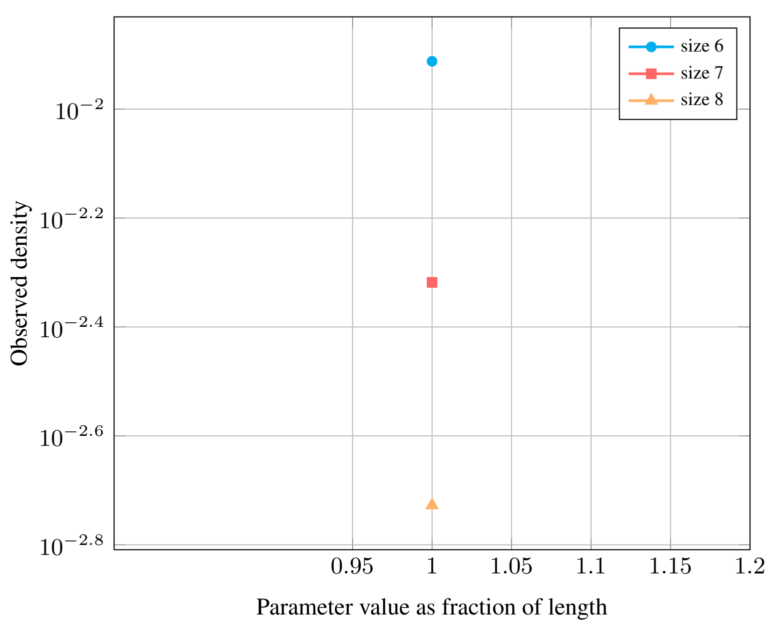

Solution density for alldifferent_modulo
Observed density
10^{-1}
10^{-2}
2 3 4 5 6 7 8
Length

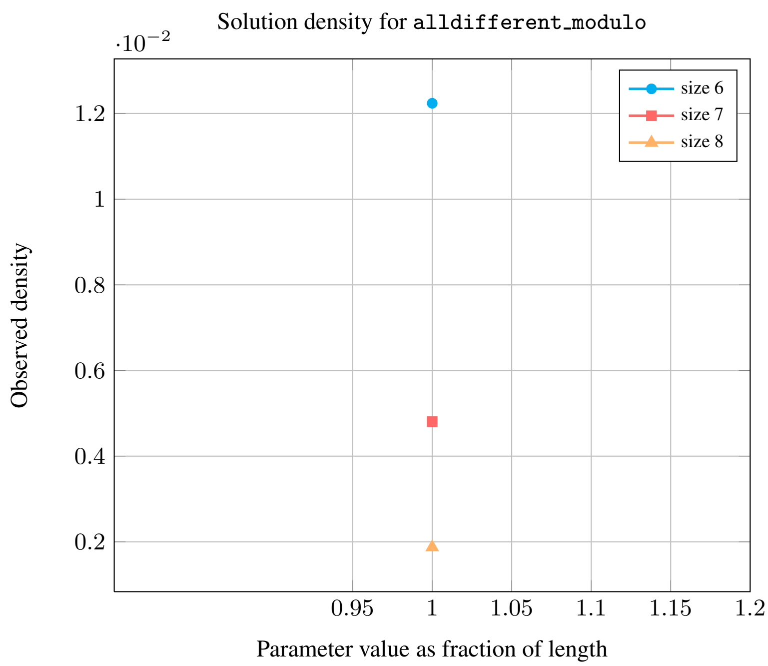

Solution density for alldifferent_modulo
Observed density
0.4
0.3
0.2
0.1
0
2 3 4 5 6 7 8
Length

Length (n)		2	3	4	5	6	7	8
Total		4	12	48	240	1440	10080	80640
Parameter value	2	4	-	-	-	-	-	-
	3	-	12	-	-	-	-	-
	4	-	-	48	-	-	-	-
	5	-	-	-	240	-	-	-
	6	-	-	-	-	1440	-	-
	7	-	-	-	-	-	10080	-
	8	-	-	-	-	-	-	80640

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

Solution density for `alldifferent_modulo`





See also

implies: `soft_alldifferent_var`.

specialisation: `alldifferent` *(*`variable` mod `constant` *replaced by* `variable`*)*.

Keywords

characteristic of a constraint: modulo, all different, sort based reformulation, automaton, automaton with array of counters.

constraint type: value constraint.

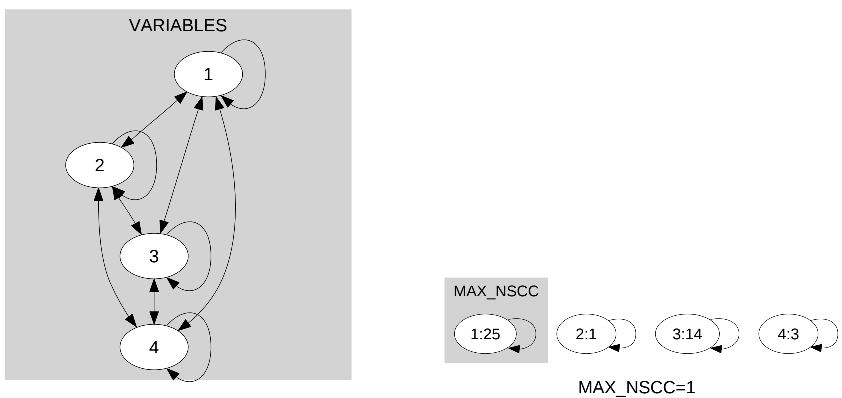
filtering: arc-consistency.

final graph structure: one_succ.

Arc input(s)	VARIABLES
Arc generator	$CLIQUE \mapsto$ collection(variables1, variables2)
Arc arity	2
Arc constraint(s)	variables1.var mod M = variables2.var mod M
Graph property(ies)	**MAX_NSCC** ≤ 1
Graph class	ONE_SUCC

Graph model Exploit the same model used for the alldifferent constraint. We replace the binary *equality* constraint by another equivalence relation depicted by the arc constraint. We generate a *clique* with a binary *equality modulo* M constraint between each pair of vertices (including a vertex and itself) and state that the size of the largest strongly connected component should not exceed 1.

Parts (A) and (B) of Figure 5.41 respectively show the initial and final graph associated with the **Example** slot. Since we use the **MAX_NSCC** graph property we show one of the largest strongly connected components of the final graph.



(A) **(B)**

Figure 5.41: Initial and final graph of the alldifferent_modulo constraint

Automaton Figure 5.42 depicts the automaton associated with the `alldifferent_modulo` constraint. To each item of the collection `VARIABLES` corresponds a signature variable $\mathtt{S}_i$ that is equal to 1. The automaton counts for each equivalence class the number of used values and finally imposes that each equivalence class is used at most one time.

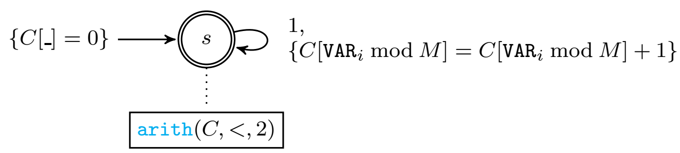


Figure 5.42: Automaton of the `alldifferent_modulo` constraint