5.44 balance_cycle

DESCRIPTION LINKS GRAPH

Origin derived from `balance` and `cycle`

Constraint `balance_cycle(BALANCE, NODES)`

Arguments

BALANCE	:	dvar
NODES	:	collection(index−int, succ−dvar)

Restrictions

$\texttt{BALANCE} \geq 0$
$\texttt{BALANCE} \leq \texttt{max}(0, |\texttt{NODES}| - 2)$
$\texttt{required}(\texttt{NODES}, [\texttt{index}, \texttt{succ}])$
$\texttt{NODES.index} \geq 1$
$\texttt{NODES.index} \leq |\texttt{NODES}|$
$\texttt{distinct}(\texttt{NODES}, \texttt{index})$
$\texttt{NODES.succ} \geq 1$
$\texttt{NODES.succ} \leq |\texttt{NODES}|$

Purpose Consider a digraph G described by the `NODES` collection. Partition G into a set of vertex disjoint circuits in such a way that each vertex of G belongs to a single circuit. `BALANCE` is equal to the difference between the number of vertices of the largest circuit and the number of vertices of the smallest circuit.

Example

$$\left(\begin{array}{l}1, \left\langle\begin{array}{ll}\texttt{index}-1 & \texttt{succ}-2,\\ \texttt{index}-2 & \texttt{succ}-1,\\ \texttt{index}-3 & \texttt{succ}-5,\\ \texttt{index}-4 & \texttt{succ}-3,\\ \texttt{index}-5 & \texttt{succ}-4\end{array}\right\rangle\end{array}\right)$$
$$\left(\begin{array}{l}0, \left\langle\begin{array}{ll}\texttt{index}-1 & \texttt{succ}-2,\\ \texttt{index}-2 & \texttt{succ}-3,\\ \texttt{index}-3 & \texttt{succ}-1,\\ \texttt{index}-4 & \texttt{succ}-5,\\ \texttt{index}-5 & \texttt{succ}-6,\\ \texttt{index}-6 & \texttt{succ}-4\end{array}\right\rangle\end{array}\right)$$
$$\left(\begin{array}{l}4, \left\langle\begin{array}{ll}\texttt{index}-1 & \texttt{succ}-2,\\ \texttt{index}-2 & \texttt{succ}-3,\\ \texttt{index}-3 & \texttt{succ}-4,\\ \texttt{index}-4 & \texttt{succ}-5,\\ \texttt{index}-5 & \texttt{succ}-1,\\ \texttt{index}-6 & \texttt{succ}-6\end{array}\right\rangle\end{array}\right)$$

In the first example we have the following two circuits: $1 \rightarrow 2 \rightarrow 1$ and $3 \rightarrow 5 \rightarrow 4 \rightarrow 3$. Since $\texttt{BALANCE} = 1$ is the difference between the number of vertices of the largest circuit (i.e., 3) and the number of vertices of the smallest circuit (i.e., 2) the corresponding `balance_cycle` constraint holds.

All solutions

Figure 5.118 gives all solutions to the following non ground instance of the `balance_cycle` constraint: $\texttt{BALANCE} \in [0,1]$, $\texttt{S}_1 \in [1,2]$, $\texttt{S}_2 \in [1,3]$, $\texttt{S}_3 \in [3,5]$, $\texttt{S}_4 \in [3,4]$, $\texttt{S}_5 \in [2,5]$, $\texttt{balance\_cycle}(\texttt{BALANCE}, \langle 1\ \texttt{S}_1, 2\ \texttt{S}_2, 3\ \texttt{S}_3, 4\ \texttt{S}_4, 5\ \texttt{S}_5\rangle)$.

① $(\mathbf{0}, \langle \mathbf{1}_1, \mathbf{2}_2, \mathbf{3}_3, \mathbf{4}_4, \mathbf{5}_5\rangle)$
② $(\mathbf{1}, \langle \mathbf{1}_1, \mathbf{2}_2, \mathbf{4}_3, \mathbf{3}_4, \mathbf{5}_5\rangle)$
③ $(\mathbf{1}, \langle \mathbf{1}_1, \mathbf{2}_2, \mathbf{5}_3, \mathbf{4}_4, \mathbf{3}_5\rangle)$
④ $(\mathbf{1}, \langle \mathbf{2}_1, \mathbf{1}_2, \mathbf{3}_3, \mathbf{4}_4, \mathbf{5}_5\rangle)$
⑤ $(\mathbf{1}, \langle \mathbf{2}_1, \mathbf{1}_2, \mathbf{4}_3, \mathbf{3}_4, \mathbf{5}_5\rangle)$
⑥ $(\mathbf{1}, \langle \mathbf{2}_1, \mathbf{1}_2, \mathbf{5}_3, \mathbf{3}_4, \mathbf{4}_5\rangle)$
⑦ $(\mathbf{1}, \langle \mathbf{2}_1, \mathbf{1}_2, \mathbf{5}_3, \mathbf{4}_4, \mathbf{3}_5\rangle)$

Figure 5.118: All solutions corresponding to the non ground example of the `balance_cycle` constraint of the **All solutions** slot; the `index` attribute is displayed as indices of the `succ` attribute, and all vertices of a same cycle are coloured by the same colour.

Typical

$|\texttt{NODES}| > 2$

Symmetry

Items of `NODES` are permutable.

Arg. properties

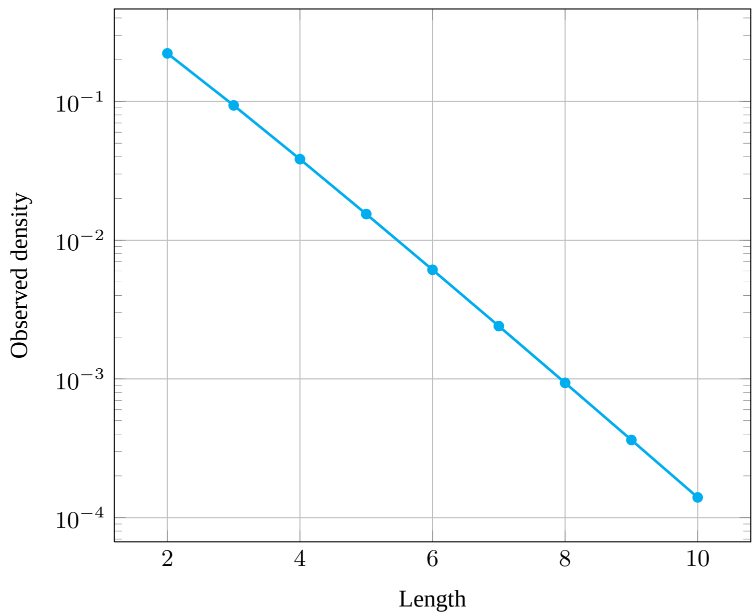
Functional dependency: `BALANCE` determined by `NODES`.

Counting

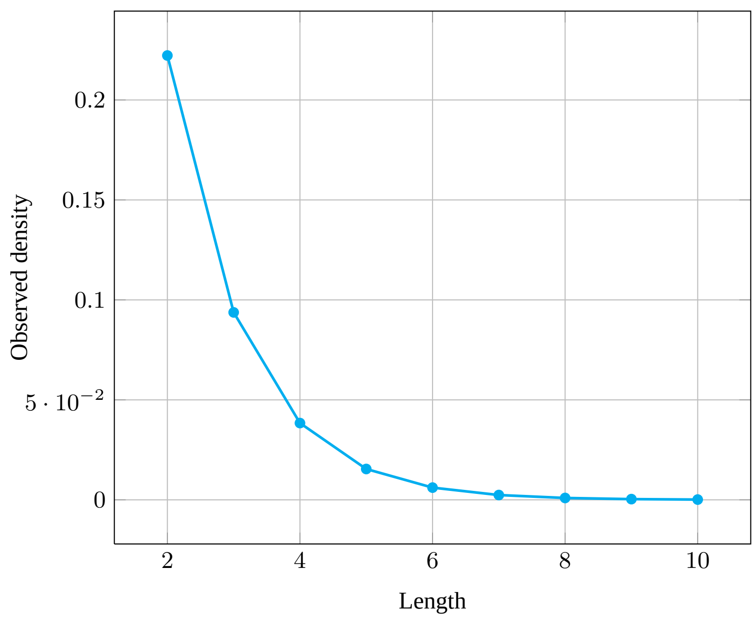
Length (n)	2	3	4	5	6	7	8	9	10
Solutions	2	6	24	120	720	5040	40320	362880	3628800

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

Solution density for `balance_cycle`



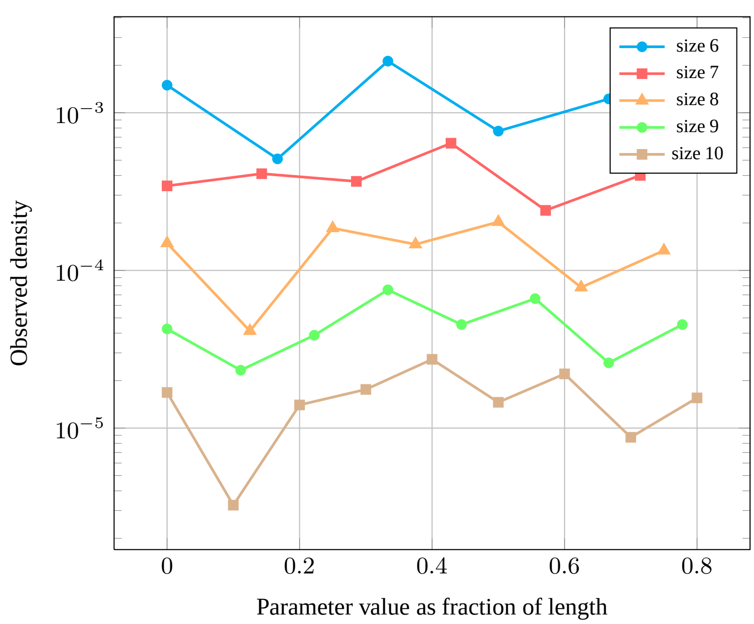
Solution density for `balance_cycle`

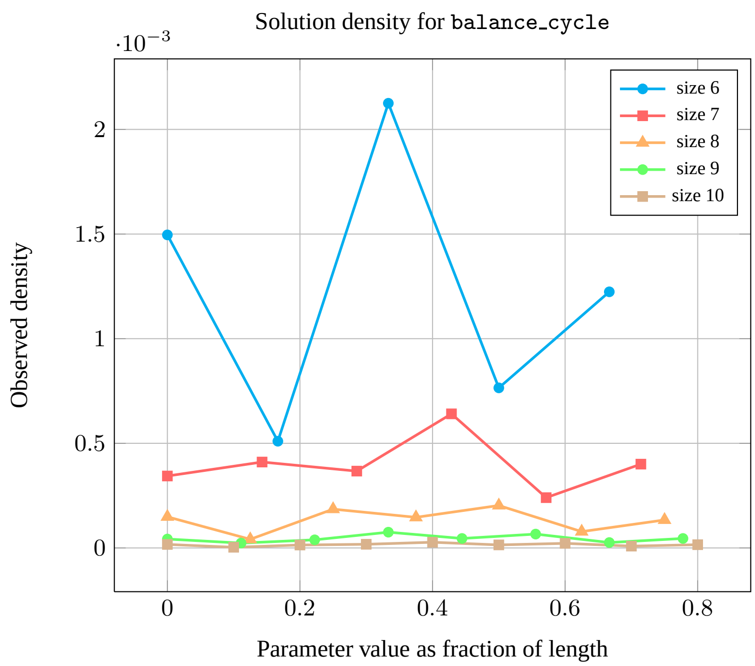


Length (n)		2	3	4	5	6	7	8	9	10
Total		2	6	24	120	720	5040	40320	362880	3628800
Parameter value	0	2	3	10	25	176	721	6406	42561	436402
	1	-	3	6	45	60	861	1778	23283	84150
	2	-	-	8	20	250	770	7980	38808	363680
	3	-	-	-	30	90	1344	6300	75348	456120
	4	-	-	-	-	144	504	8736	45360	708048
	5	-	-	-	-	-	840	3360	66240	378000
	6	-	-	-	-	-	-	5760	25920	572400
	7	-	-	-	-	-	-	-	45360	226800
	8	-	-	-	-	-	-	-	-	403200

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

Solution density for `balance_cycle`





See also

related: `balance` *(equivalence classes correspond to vertices in same cycle rather than variables assigned to the same value)*, `cycle` *(do not care how many cycles but how balanced the cycles are)*.

Keywords

combinatorial object: permutation.

constraint type: graph constraint, graph partitioning constraint.

filtering: DFS-bottleneck.

final graph structure: circuit, connected component, strongly connected component, one_succ.

modelling: cycle, functional dependency.

Cond. implications

- `balance_cycle`(BALANCE, NODES)
  with BALANCE > 0
  and BALANCE ≤ 2
  implies `all_differ_from_at_least_k_pos`(K : BALANCE, VECTORS : NODES).

- `balance_cycle`(BALANCE, NODES)
  implies `permutation`(VARIABLES : NODES).

Arc input(s)	NODES
Arc generator	*CLIQUE*$\mapsto$collection(nodes1, nodes2)
Arc arity	2
Arc constraint(s)	nodes1.succ = nodes2.index
Graph property(ies)	• **NTREE**= 0 • **RANGE_NCC**= BALANCE
Graph class	ONE_SUCC

Graph model From the restrictions and from the arc constraint, we deduce that we have a bijection from the successor variables to the values of interval $[1, |\text{NODES}|]$. With no explicit restrictions it would have been impossible to derive this property.

In order to express the binary constraint that links two vertices one has to make explicit the identifier of the vertices. This is why the balance_cycle constraint considers objects that have two attributes:

- One fixed attribute index that is the identifier of the vertex,
- One variable attribute succ that is the successor of the vertex.

The graph property **NTREE** = 0 is used in order to avoid having vertices that both do not belong to a circuit and have at least one successor located on a circuit. This concretely means that all vertices of the final graph should belong to a circuit.

Parts (A) and (B) of Figure 5.119 respectively show the initial and final graph associated with the first example of the **Example** slot. Since we use the **RANGE_NCC** graph property, we show the connected components of the final graph. The constraint holds since all the vertices belong to a circuit (i.e., **NTREE** = 0) and since BALANCE = **RANGE_NCC** = 1.

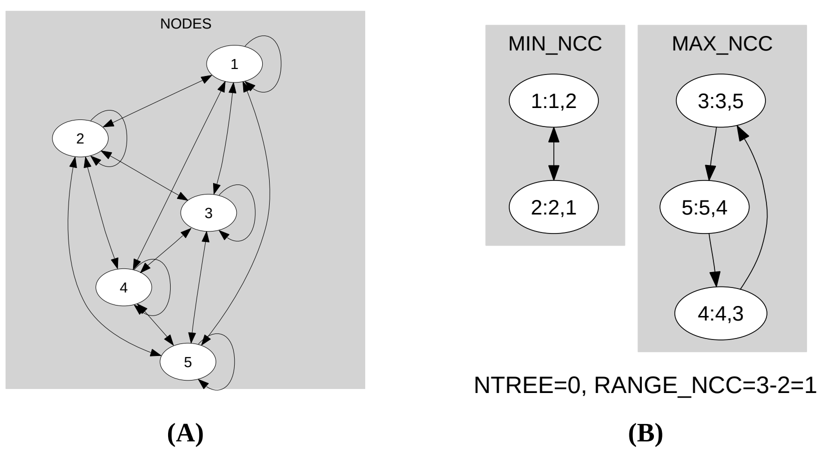


Figure 5.119: Initial and final graph of the balance_cycle constraint