5.48 balance_path

DESCRIPTION LINKS GRAPH

Origin

derived from `balance` and `path`

Constraint

`balance_path(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 paths in such a way that each vertex of G belongs to a single path. `BALANCE` is equal to the difference between the number of vertices of the largest path and the number of vertices of the smallest path.

Example

$$\left(3, \left\langle \begin{array}{ll} \texttt{index}-1 & \texttt{succ}-1, \\ \texttt{index}-2 & \texttt{succ}-3, \\ \texttt{index}-3 & \texttt{succ}-5, \\ \texttt{index}-4 & \texttt{succ}-4, \\ \texttt{index}-5 & \texttt{succ}-1, \\ \texttt{index}-6 & \texttt{succ}-6, \\ \texttt{index}-7 & \texttt{succ}-7, \\ \texttt{index}-8 & \texttt{succ}-6 \end{array} \right\rangle \right)$$
$$\left(0, \left\langle \begin{array}{ll} \texttt{index}-1 & \texttt{succ}-2, \\ \texttt{index}-2 & \texttt{succ}-3, \\ \texttt{index}-3 & \texttt{succ}-4, \\ \texttt{index}-4 & \texttt{succ}-4, \\ \texttt{index}-5 & \texttt{succ}-6, \\ \texttt{index}-6 & \texttt{succ}-7, \\ \texttt{index}-7 & \texttt{succ}-8, \\ \texttt{index}-8 & \texttt{succ}-8 \end{array} \right\rangle \right)$$
$$\left(6, \left\langle \begin{array}{ll} \texttt{index}-1 & \texttt{succ}-2, \\ \texttt{index}-2 & \texttt{succ}-3, \\ \texttt{index}-3 & \texttt{succ}-4, \\ \texttt{index}-4 & \texttt{succ}-5, \\ \texttt{index}-5 & \texttt{succ}-6, \\ \texttt{index}-6 & \texttt{succ}-7, \\ \texttt{index}-7 & \texttt{succ}-7, \\ \texttt{index}-8 & \texttt{succ}-8 \end{array} \right\rangle \right)$$

In the first example we have the following four paths: $2 \rightarrow 3 \rightarrow 5 \rightarrow 1$, $8 \rightarrow 6$, 4, and 7. Since $\texttt{BALANCE} = 3$ is the difference between the number of vertices of the largest path (i.e., 4) and the number of vertices of the smallest path (i.e., 1) the corresponding `balance_path` constraint holds.

All solutions

Figure 5.125 gives all solutions to the following non ground instance of the `balance_path` constraint: $\texttt{BALANCE} = 0$, $\texttt{S}_1 \in [1,2]$, $\texttt{S}_2 \in [1,3]$, $\texttt{S}_3 \in [3,5]$, $\texttt{S}_4 \in [3,4]$, $\texttt{S}_5 \in [2,5]$, $\texttt{S}_6 \in [5,6]$, $\texttt{balance\_path}(\texttt{BALANCE}, \langle 1\ \texttt{S}_1, 2\ \texttt{S}_2, 3\ \texttt{S}_3, 4\ \texttt{S}_4, 5\ \texttt{S}_5, 6\ \texttt{S}_6\rangle)$.

① $(\mathbf{0}, \langle \mathbf{1}, \mathbf{1}, \mathbf{3}, \mathbf{3}, \mathbf{5}, \mathbf{5}\rangle)$
② $(\mathbf{0}, \langle \mathbf{1}, \mathbf{1}, \mathbf{4}, \mathbf{4}, \mathbf{5}, \mathbf{5}\rangle)$
③ $(\mathbf{0}, \langle \mathbf{1}, \mathbf{2}, \mathbf{3}, \mathbf{4}, \mathbf{5}, \mathbf{6}\rangle)$
④ $(\mathbf{0}, \langle \mathbf{2}, \mathbf{2}, \mathbf{3}, \mathbf{3}, \mathbf{5}, \mathbf{5}\rangle)$
⑤ $(\mathbf{0}, \langle \mathbf{2}, \mathbf{2}, \mathbf{4}, \mathbf{4}, \mathbf{5}, \mathbf{5}\rangle)$
⑥ $(\mathbf{0}, \langle \mathbf{2}, \mathbf{3}, \mathbf{3}, \mathbf{4}, \mathbf{4}, \mathbf{5}\rangle)$

Figure 5.125: All solutions corresponding to the non ground example of the `balance_path` constraint of the **All solutions** slot; the `index` attribute is displayed as indices of the `succ` attribute and all vertices of a same path are coloured by the same colour.

Typical

$|\texttt{NODES}| > 2$

Symmetry

Items of `NODES` are permutable.

Arg. properties

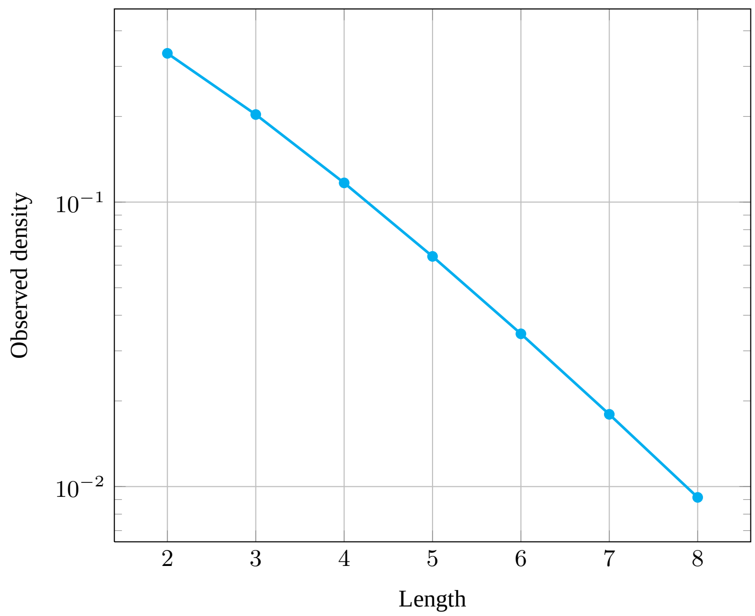
Functional dependency: `BALANCE` determined by `NODES`.

Counting

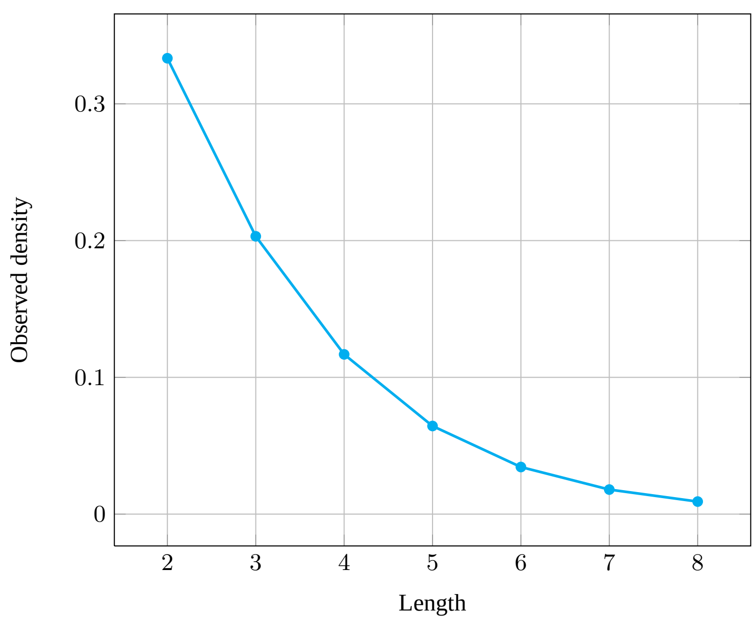
Length (n)	2	3	4	5	6	7	8
Solutions	3	13	73	501	4051	37633	394353

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

Solution density for `balance_path`



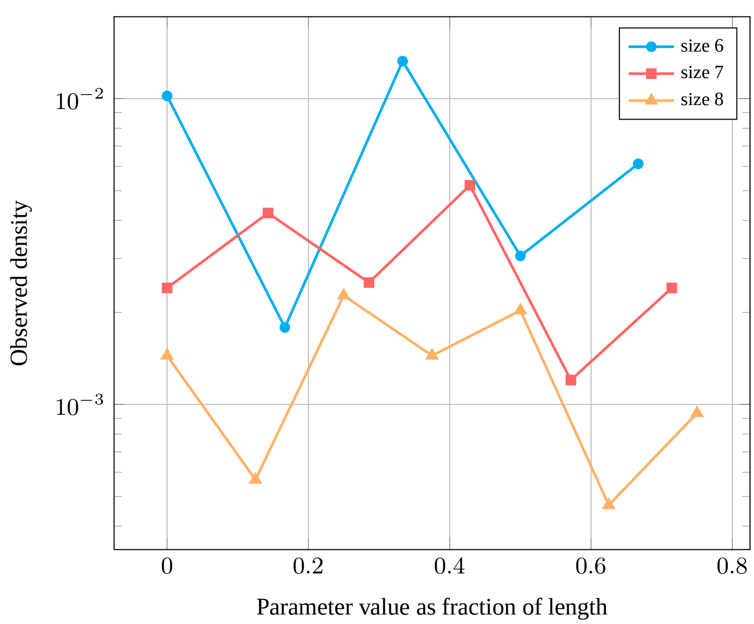
Solution density for `balance_path`

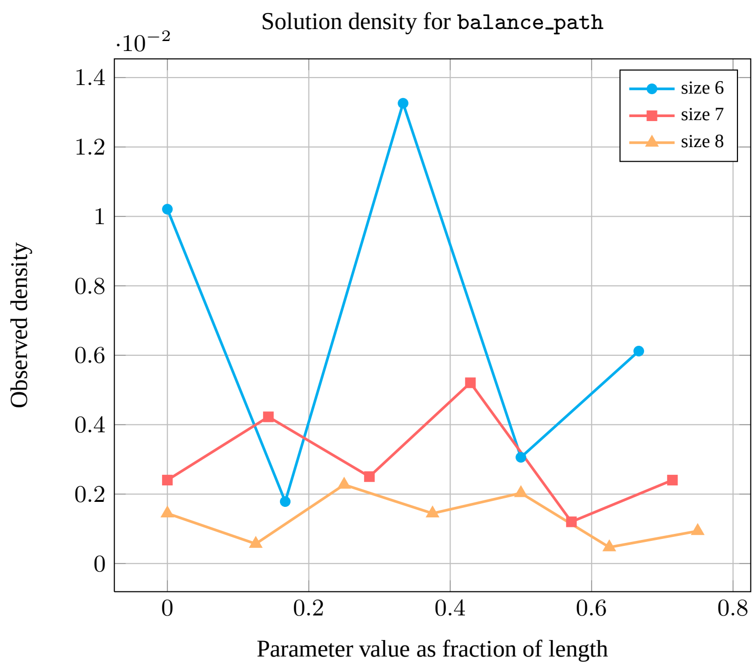


Length (n)		2	3	4	5	6	7	8
Total		3	13	73	501	4051	37633	394353
Parameter value	0	3	7	37	121	1201	5041	62161
	1	-	6	12	200	210	8862	24416
	2	-	-	24	60	1560	5250	97776
	3	-	-	-	120	360	10920	62160
	4	-	-	-	-	720	2520	87360
	5	-	-	-	-	-	5040	20160
	6	-	-	-	-	-	-	40320

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

Solution density for `balance_path`





See also

implies: `balance_tree`.

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

Keywords

combinatorial object: path.

constraint type: graph constraint, graph partitioning constraint.

filtering: DFS-bottleneck.

final graph structure: connected component, tree, one_succ.

modelling: functional dependency.

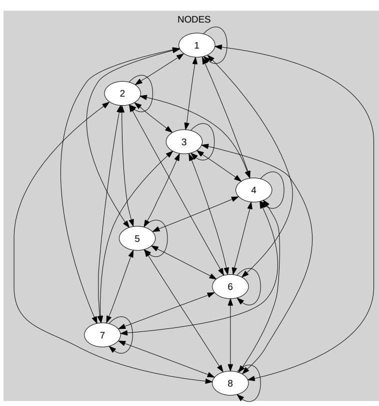
Arc input(s)	NODES
Arc generator	*CLIQUE*↦collection(nodes1, nodes2)
Arc arity	2
Arc constraint(s)	nodes1.succ = nodes2.index
Graph property(ies)	• **MAX_NSCC**≤ 1 • **MAX_ID**≤ 1 • **RANGE_NCC**= BALANCE
Graph class	ONE_SUCC

Graph model 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_path constraint considers objects that have two attributes:

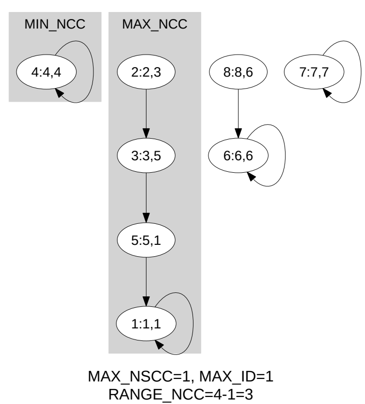
- One fixed attribute index that is the identifier of the vertex,
- One variable attribute succ that is the successor of the vertex.

We use the graph property **MAX_NSCC**≤ 1 in order to specify the fact that the size of the largest strongly connected component should not exceed one. In fact each root of a tree is a strongly connected component with a single vertex. The graph property **MAX_ID**≤ 1 constraints the maximum in-degree of the final graph to not exceed 1. **MAX_ID** does not consider loops: This is why we do not have any problem with the final node of each path.

Parts (A) and (B) of Figure 5.126 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 path and since BALANCE = **RANGE_NCC** = 3.



(A)



(B)

Figure 5.126: Initial and final graph of the balance_path constraint