5.103 cycle

DESCRIPTION LINKS GRAPH

Origin	[41]
Constraint	`cycle(NCYCLE, NODES)`

Arguments	`NCYCLE : dvar`
	`NODES : collection(index-int, succ-dvar)`

Restrictions	`NCYCLE` ≥ 1
	`NCYCLE` $\leq$ |`NODES`|
	`required(NODES, [index, succ])`
	`NODES.index` ≥ 1
	`NODES.index` $\leq$ |`NODES`|
	`distinct(NODES, index)`
	`NODES.succ` ≥ 1
	`NODES.succ` $\leq$ |`NODES`|

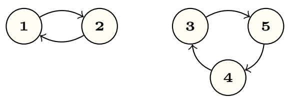
Purpose	Consider a digraph G described by the `NODES` collection. `NCYCLE` is equal to the number of circuits for covering G in such a way that each vertex of G belongs to a single circuit. `NCYCLE` can also be interpreted as the number of cycles of the permutation associated with the successor variables of the `NODES` collection.

Example

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

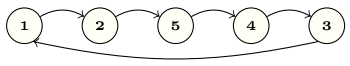
- In the first example we have the following 2 (`NCYCLE` $= 2$) cycles: $1 \mapsto 2 \mapsto 1$ and $3 \mapsto 5 \mapsto 4 \mapsto 3$. Consequently, the corresponding `cycle` constraint holds.

`cycle`(**2**, ⟨1 2, 2 1, 3 5, 4 3, 5 4⟩)



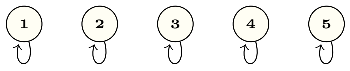
- In the second example we have 1 (NCYCLE $=1$) cycle: $1 \mapsto 2 \mapsto 5 \mapsto 4 \mapsto 3 \mapsto 1$.

cycle(**1**, $\langle 1\,2,\ 2\,5,\ 3\,1,\ 4\,3,\ 5\,4\rangle$)



- In the third example we have the following 5 (NCYCLE $=5$) cycles: $1 \mapsto 1$, $2 \mapsto 2$, $3 \mapsto 3$, $4 \mapsto 4$ and $5 \mapsto 5$.

cycle(**5**, $\langle 1\,1,\ 2\,2,\ 3\,3,\ 4\,4,\ 5\,5\rangle$)



All solutions Figure 5.228 gives all solutions to the following non ground instance of the `cycle` constraint: $N \in [1,2]$, $V_1 \in [2,4]$, $V_2 \in [2,3]$, $V_3 \in [1,6]$, $V_4 \in [2,5]$, $V_5 \in [2,3]$, $V_6 \in [1,6]$, `cycle`$(N, \langle 1\,V_1,\ 2\,V_2,\ 3\,V_3,\ 4\,V_4,\ 5\,V_5,\ 6\,V_6\rangle)$.

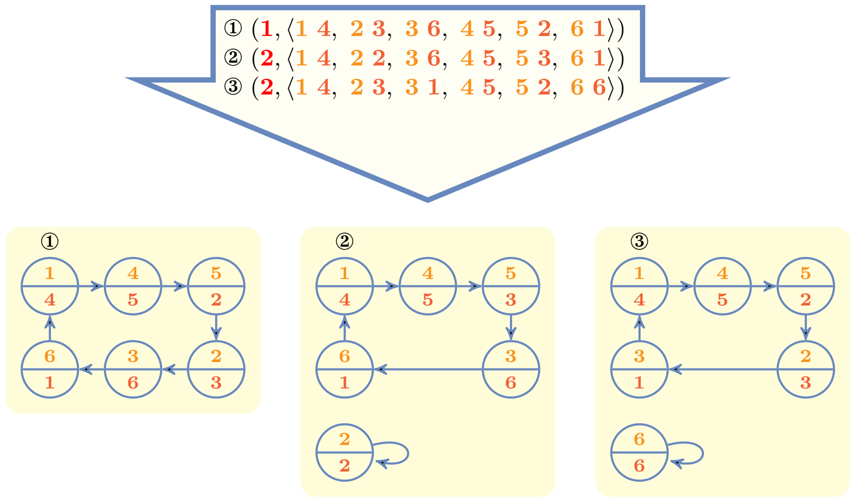


Figure 5.228: All solutions corresponding to the non ground example of the `cycle` constraint of the **All solutions** slot

Typical

NCYCLE $<$ |NODES|
|NODES| > 2

Symmetries

- Items of NODES are permutable.
- Attributes of NODES are permutable w.r.t. permutation (`index`, `succ`) (*permutation applied to all items*).

Arg. properties

Functional dependency: NCYCLE determined by NODES.

Usage The PhD thesis of Éric Bourreau [84] mentions the following applications of extensions of the `cycle` constraint:

- The balanced Euler knight problem where one tries to cover a rectangular chessboard of size $N \cdot M$ by C knights that all have to visit between $2 \cdot \lfloor\lfloor(N \cdot M)/C\rfloor/2\rfloor$ and $2 \cdot \lceil\lceil(N \cdot M)/C\rceil/2\rceil$ distinct locations. For some values of N, M and C there does not exist any solution to the previous problem. This is for instance the case when $N = M = C = 6$. Figure 5.229 depicts the graph associated with the 6×6 chessboard as well as examples of balanced solutions with respectively 1, 2, 3, 4 and 5 knights.
- Some pick-up delivery problems where a fleet of vehicles has to transport a set of orders. Each order is characterised by its initial location, its final destination and its weight. In addition one also has to take into account the capacity of the different vehicles.

Remark In the original `cycle` constraint of **CHIP** the `index` attribute was not explicitly present. It was implicitly defined as the position of a variable in a list.

In an early version of the **CHIP** there was a constraint named `circuit` that, from a declarative point of view, was equivalent to $\texttt{cycle}(1, \texttt{NODES})$. In ALICE [256] the `circuit` constraint was also present.

Given a complete digraph of n vertices as well as an unrestricted number of circuits `NCYCLE`, the total number of solutions to the corresponding `cycle` constraint corresponds to the sequence A000142 of the On-Line Encyclopaedia of Integer Sequences [392]. Given a complete digraph of n vertices as well as a fixed number of circuits `NCYCLE` between 1 and n, the total number of solutions to the corresponding `cycle` constraint corresponds to the so called *Stirling number of first kind*.

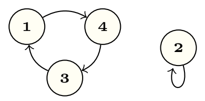
Algorithm Since all `succ` variables have to take distinct values one can reuse the algorithms associated with the `alldifferent` constraint. A second necessary condition is to have no more than $\overline{\texttt{NCYCLE}}$ strongly connected components. Pruning for enforcing this condition, as soon as we have $\overline{\texttt{NCYCLE}}$ strongly connected components, can be done by forcing all strong bridges to belong to the final solution, since otherwise we would have more than $\overline{\texttt{NCYCLE}}$ strongly connected components. Since all the vertices of a circuit belong to the same strongly connected component an arc going from one strongly connected component to another strongly connected component has to be removed.

Reformulation

Let n and $s_1, s_2, \ldots, s_n$ respectively denotes the number of vertices (i.e., $|\texttt{NODES}|$) and the successor variables associated with vertices $1, 2, \ldots, n$. The `cycle` constraint can be reformulated as a conjunction of one `alldifferent` constraint, $n \cdot (n-1)$ `element` constraints, n `minimum` constraints, and one `nvalue` constraint.

- First, we state an `alldifferent`$(\langle s_1, s_2, \ldots, s_n\rangle)$ constraint for enforcing distinct values to be assigned to the successor variables.
- Second, the key idea is to extract for each vertex i (with $i \in [1, n]$) all the vertices that belong to the same cycle. This is done by stating a conjunction of $n-1$ `element` constraints of the form:
  `element`$(i, \langle s_1, s_2, \ldots, s_n\rangle, s_{i,1})$,
  `element`$(s_{i,1}, \langle s_1, s_2, \ldots, s_n\rangle, s_{i,2})$,
  ....
  `element`$(s_{i,n-2}, \langle s_1, s_2, \ldots, s_n\rangle, s_{i,n-1})$.
  Then, using a `minimum`$(m_i, \langle i, s_{i,1}, s_{i,2}, \ldots, s_{i,n-1}\rangle)$ constraint, we get a unique representative for the cycle containing vertex i.
- Third, using a `nvalue`(NCYCLE, $\langle m_1, m_2, \ldots, m_n\rangle$) constraint, we get the number of distinct cycles.

Illustration of the reformulation of
`cycle`(**2**, $\langle 1\ 4,\ 2\ 2,\ 3\ 1,\ 4\ 3\rangle$)



`alldifferent`$(\langle 4, 2, 1, 3\rangle)$

`element`$(\mathbf{1}, \langle 4, 2, 1, 3\rangle, \boxed{4})$
`element`$(4, \langle 4, 2, 1, 3\rangle, \boxed{3})$
`element`$(3, \langle 4, 2, 1, 3\rangle, \boxed{1})$
(*representative of the cycle containing vertex* **1**) min $= \boxed{1}$

`element`$(\mathbf{2}, \langle 4, 2, 1, 3\rangle, \boxed{2})$
`element`$(2, \langle 4, 2, 1, 3\rangle, \boxed{2})$
`element`$(2, \langle 4, 2, 1, 3\rangle, \boxed{2})$
(*representative of the cycle containing vertex* **2**) min $= \boxed{2}$

`element`$(\mathbf{3}, \langle 4, 2, 1, 3\rangle, \boxed{1})$
`element`$(1, \langle 4, 2, 1, 3\rangle, \boxed{4})$
`element`$(4, \langle 4, 2, 1, 3\rangle, \boxed{3})$
(*representative of the cycle containing vertex* **3**) min $= \boxed{1}$

`element`$(\mathbf{4}, \langle 4, 2, 1, 3\rangle, \boxed{3})$
`element`$(3, \langle 4, 2, 1, 3\rangle, \boxed{1})$
`element`$(1, \langle 4, 2, 1, 3\rangle, \boxed{4})$
(*representative of the cycle containing vertex* **4**) min $= \boxed{1}$

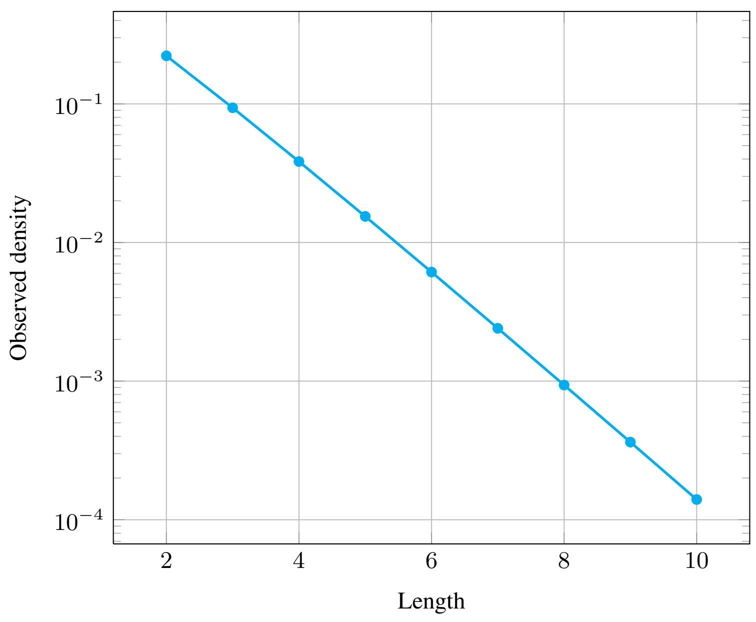
`nvalue`=**2**

Counting

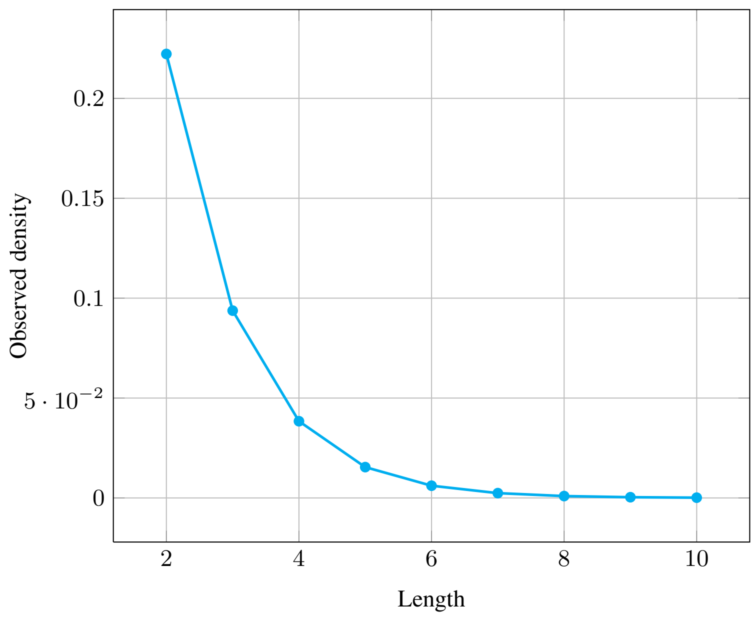
Length (n)	2	3	4	5	6	7	8	9	10
Solutions	2	6	24	120	720	5040	40320	362880	3628800

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

Solution density for `cycle`

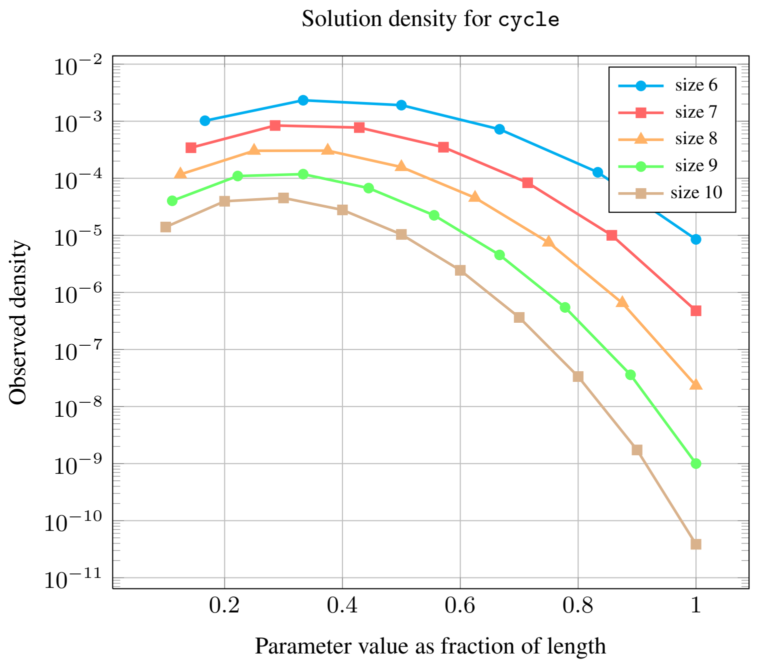


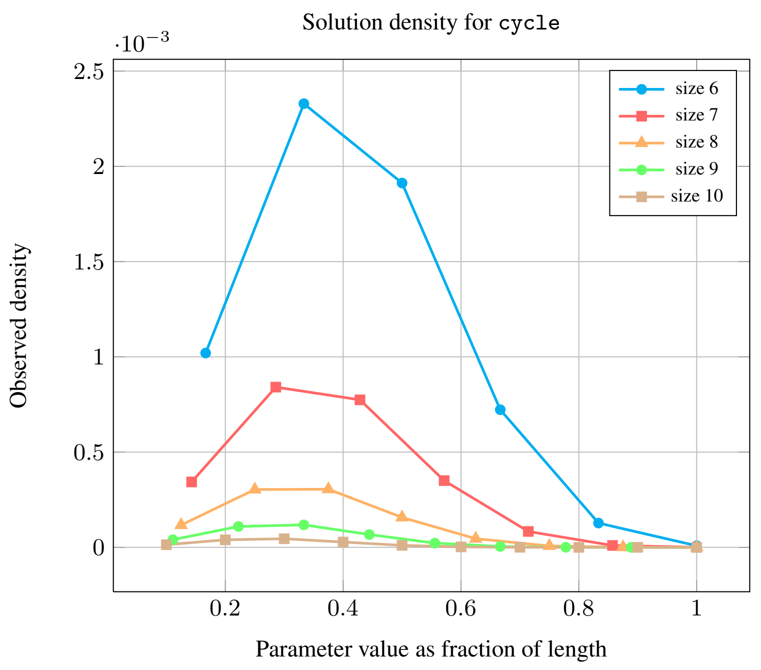
Solution density for `cycle`



Length (n)		2	3	4	5	6	7	8	9	10
Total		2	6	24	120	720	5040	40320	362880	3628800
Parameter value	1	1	2	6	24	120	720	5040	40320	362880
	2	1	3	11	50	274	1764	13068	109584	1026576
	3	-	1	6	35	225	1624	13132	118124	1172700
	4	-	-	1	10	85	735	6769	67284	723680
	5	-	-	-	1	15	175	1960	22449	269325
	6	-	-	-	-	1	21	322	4536	63273
	7	-	-	-	-	-	1	28	546	9450
	8	-	-	-	-	-	-	1	36	870
	9	-	-	-	-	-	-	-	1	45
	10	-	-	-	-	-	-	-	-	1

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





See also **common keyword:** `alldifferent` *(permutation)*,
`circuit_cluster` *(graph constraint, one_succ)*,
`cycle_card_on_path` *(permutation,graph partitioning constraint)*,
`cycle_or_accessibility` *(graph constraint)*,
`cycle_resource` *(graph partitioning constraint)*,
`derangement` *(permutation)*,
`graph_crossing` *(graph constraint,graph partitioning constraint)*,
`inverse` *(permutation)*,
`map` *(graph partitioning constraint)*,
`symmetric_alldifferent` *(permutation)*,
`tour` *(graph constraint)*,
`tree` *(graph partitioning constraint)*.

implies: `alldifferent`.

implies (items to collection): `atleast_nvector`.

related: `balance_cycle` *(counting number of cycles versus controlling how balanced the cycles are)*.

specialisation: `circuit` *(*`NCYCLE` *set to* 1*)*.

used in reformulation: `alldifferent`, `element`, `minimum`, `nvalue`.

Keywords **characteristic of a constraint:** core.

combinatorial object: permutation.

constraint arguments: business rules.

constraint type: graph constraint, graph partitioning constraint.

filtering: strong bridge, DFS-bottleneck.

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

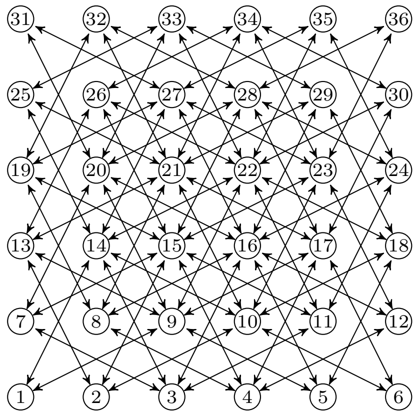
modelling: cycle, functional dependency.

problems: pick-up delivery.

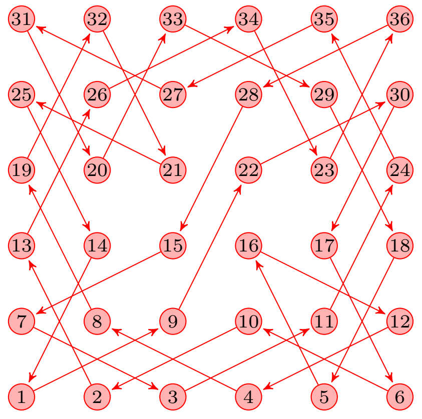
puzzles: Euler knight.

Cond. implications

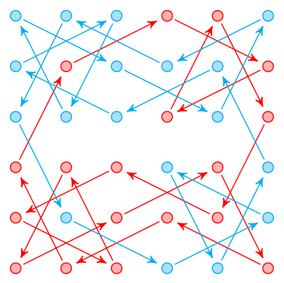
- `cycle`(NCYCLE, NODES)
  with NCYCLE $= 1$
  implies `balance_cycle`(BALANCE, NODES)
  when BALANCE $= 0$.
- `cycle`(NCYCLE, NODES)
  implies `permutation`(VARIABLES : NODES).



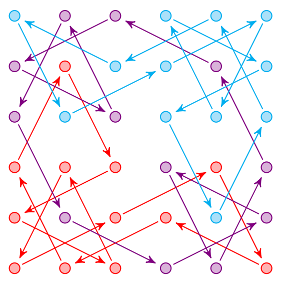
Graph of potential moves
of a 6 × 6 chessboard



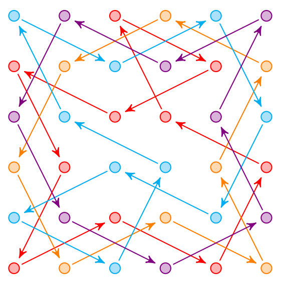
1 knight
(**36** moves)



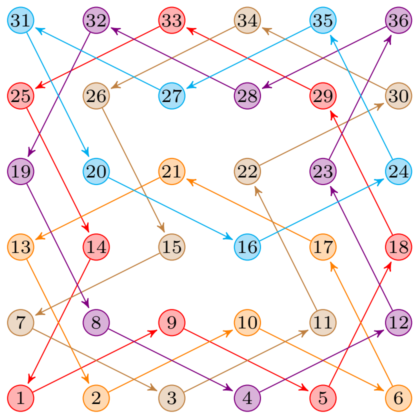
2 knights
(**18** and **18** moves)



3 knights
(**12**, **12** and **12** moves)



4 knights
(**8**, **8**, **10** and **10** moves)



5 knights
(**6**, **6**, **8**, **8** and **8** moves)

`NODES` (`i` for `index`, `s` for `succ`)

`i` − **1**	`s` − 9,	`i` − **19**	`s` − 8,
`i` − **2**	`s` − 10,	`i` − **20**	`s` − 16,
`i` − **3**	`s` − 11,	`i` − **21**	`s` − 13,
`i` − **4**	`s` − 12,	`i` − **22**	`s` − 30,
`i` − **5**	`s` − 18,	`i` − **23**	`s` − 36,
`i` − **6**	`s` − 17,	`i` − **24**	`s` − 35,
`i` − **7**	`s` − 3,	`i` − **25**	`s` − 14,
`i` − **8**	`s` − 4,	`i` − **26**	`s` − 15,
`i` − **9**	`s` − 5,	`i` − **27**	`s` − 31,
`i` − **10**	`s` − 6,	`i` − **28**	`s` − 32,
`i` − **11**	`s` − 22,	`i` − **29**	`s` − 33,
`i` − **12**	`s` − 23,	`i` − **30**	`s` − 34,
`i` − **13**	`s` − 2,	`i` − **31**	`s` − 20,
`i` − **14**	`s` − 1,	`i` − **32**	`s` − 19,
`i` − **15**	`s` − 7,	`i` − **33**	`s` − 25,
`i` − **16**	`s` − 24,	`i` − **34**	`s` − 26,
`i` − **17**	`s` − 21,	`i` − **35**	`s` − 27,
`i` − **18**	`s` − 29,	`i` − **36**	`s` − 28

Figure 5.229: Graph of potential moves of a 6 × 6 chessboard, corresponding balanced knight's tours with 1 up to 5 knights, and collection of nodes passed to the `cycle` constraint corresponding to the solution with 5 knights; note that their is no balanced knight's tour on a 6 × 6 chessboard where each knight exactly performs 6 moves.

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**$= 0$ • **NCC**$=$ `NCYCLE`
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 $[1, |\texttt{NODES}|]$. 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 `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 5.230 respectively show the initial and final graph associated with the first example of the **Example** slot. Since we use the **NCC** graph property, we show the two connected components of the final graph. The constraint holds since all the vertices belong to a circuit (i.e., **NTREE** $= 0$) and since `NCYCLE` $=$ **NCC** $= 2$.

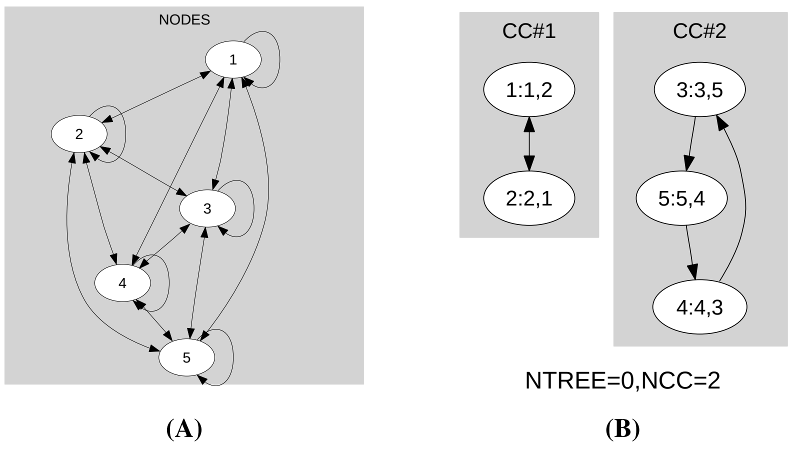


Figure 5.230: Initial and final graph of the `cycle` constraint

Quiz

EXERCISE 1 (checking whether a ground instance holds or not)[a]

A. Does the constraint $\texttt{cycle}(1, \langle 1\ 2, 2\ 1, 3\ 2\rangle)$ hold?

B. Does the constraint $\texttt{cycle}(2, \langle 1\ 3, 2\ 2, 3\ 1\rangle)$ hold?

C. Does the constraint $\texttt{cycle}(3, \langle 1\ 1, 2\ 2, 3\ 3\rangle)$ hold?

D. Does the constraint $\texttt{cycle}(2, \langle 1\ 5, 2\ 4, 3\ 3, 4\ 2, 5\ 1\rangle)$ hold?

[a]Hint: go back to the definition of `cycle`.

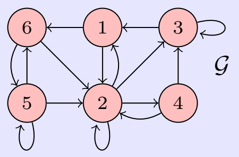
EXERCISE 2 (finding all solutions)[a]

Give all the solutions to the constraint:

$$\begin{cases} N \in \{2, 4\}, \\ V_1 \in \{1, 3, 4, 5\}, \quad V_2 \in \{3, 4\}, \quad V_3 \in \{2, 3, 5, 6\}, \\ V_4 \in \{1, 4, 6\}, \quad V_5 \in \{2, 6\}, \quad V_6 \in \{3, 4, 6\}, \\ \texttt{cycle}(N, \langle 1\ V_1,\ 2\ V_2,\ 3\ V_3,\ 4\ V_4,\ 5\ V_5,\ 6\ V_6\rangle). \end{cases}$$

[a]Hint: follow the order induced by the functional dependency between the arguments of `cycle`, start with variables that have the smallest domain.

EXERCISE 3 (identifying infeasible values)[a]



A. Describe the following digraph $\mathcal{G}$ in terms of successor variables and their corresponding domains. Give the implicit assumption behind this description.

B. Model with a single `cycle` constraint the problem of finding a Hamiltonian cycle[b] in the graph $\mathcal{G}$.

C. Identify variable-value pairs that do not belong to any solution to the `cycle` constraint stated in the previous question.

[a]Hint: make a link between the successor variables and the arcs of the graph, identify the basic constraint on the successor variables, make a what-if reasoning wrt. the arcs and the strongly connected components.

[b]Given a digraph $\mathcal{G}$ with p vertices, a *Hamiltonian cycle* of $\mathcal{G}$ is a succession of arcs $v_1 \mapsto v_2, v_2 \mapsto v_3, \cdots, v_{p-1} \mapsto v_p, v_p \mapsto v_1$ of $\mathcal{G}$ such that the vertices $v_1, v_2, \cdots, v_p$ are all distinct.

EXERCISE 4 (variable-based degree of violation)[a]

A. Compute the variable-based degree of violation[b] of the following constraints:

(a) $\texttt{cycle}(4, \langle 1\ 2, 2\ 3, 3\ 1, 4\ 4\rangle)$,

(b) $\texttt{cycle}(1, \langle 1\ 3, 2\ 4, 3\ 3, 4\ 4\rangle)$,

(c) $\texttt{cycle}(6, \langle 1\ 2, 2\ 2, 3\ 4, 4\ 4, 5\ 6, 6\ 5\rangle)$.

B. Give a formula for evaluating the variable-based degree of violation of any ground instance of the `cycle` constraint.

[a]Hint: focus first on the basic constraint on the successor variables, then on the first argument of `cycle`.

[b]Given a constraint for which all variables are fixed, the *variable-based degree of violation* is the minimum number of variables to assign differently in order to satisfy the constraint.

EXERCISE 5 (De Bruijn sequence)[a]

Given an alphabet $A = \{0, 1, \ldots, n-1\}$ and an integer $m > 0$ the corresponding *De Bruijn digraph* $\mathcal{G}_m^n = (V, E)$ of order m is defined as follows:

- The set of vertices V consist of every potential word of length m over the alphabet A.
- The set E contains all arcs $w_1 \mapsto w_2$ where w_1 and w_2 are words of length m over the alphabet A such that the last $m-1$ letters of w_1 coincide with the first $m-1$ first letters of w_2.

Given an alphabet $A = \{0, 1, \ldots, n-1\}$ and an integer $m > 0$ a *De Bruijn sequence* s_m^n of order m is a word over the alphabet A such that every word of length m over the alphabet A occurs[b] exactly once in s.

A. Given an alphabet $A = \{0, 1, \ldots, n-1\}$ define a De Bruijn sequence of order m wrt the *De Bruijn digraph* of order m defined on the same alphabet A. Illustrate this link on the De Bruijn sequence $0\ 1\ 0\ 1\ 1\ 1\ 0\ 0$ when $n = 2$, $m = 3$ and $A = \{0, 1\}$.

B. Based on the previous correspondence give a compact model for De Bruijn sequences of order m that uses a single `cycle` constraint.

[a]Hint: define the vertices of the De Bruijn digraph $\mathcal{G}_3^2$, define the arcs of $\mathcal{G}_3^2$, search a pattern on $\mathcal{G}_3^2$ corresponding to a De Bruijn sequence.

[b]A word $w = w_0 w_1 \cdots w_{m-1}$ *occurs* in a sequence $s = s_0 s_1 \cdots s_{p-1}$ $(p \geq m)$ if there exists a position i $(0 \leq i < p)$ such that $w_0 = s_i, w_1 = s_{(i+1) \bmod p}, \cdots, w_{m-1} = s_{(i+m-1) \bmod p}$.

SOLUTION TO EXERCISE 1

A. *No, since the successor attributes* 2, 1, 2 *are not all different.*

B. *Yes, since we have two cycles namely* $1 \mapsto 3 \mapsto 1$ *and* $2 \mapsto 2$.

C. *Yes, since we have three cycles namely* $1 \mapsto 1$, $2 \mapsto 2$ *and* $3 \mapsto 3$.

D. *No, since we have three cycles namely* $1 \mapsto 5 \mapsto 1$, $2 \mapsto 4 \mapsto 2$ *and* $3 \mapsto 3$*, rather than two cycles as stated by the first argument of the* `cycle` *constraint.*

SOLUTION TO EXERCISE 2
(*variables of a same cycle are coloured with the same colour*)

the five solutions

$N, \langle 1\ V_1, 2\ V_2, 3\ V_3, 4\ V_4, 5\ V_5, 6\ V_6 \rangle$

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

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

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

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

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

SOLUTION TO EXERCISE 3

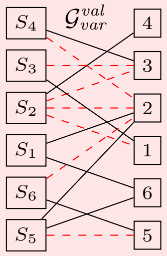
A. *To each vertex v of $\mathcal{G}$ we associate a* successor *variable S_v whose initial domain is set to the labels of the successors of v. Thus we have:*

$$\begin{cases} S_1 \in \{2,6\}, & S_2 \in \{1,2,3,4\}, & S_3 \in \{1,3\}, \\ S_4 \in \{2,3\}, & S_5 \in \{2,5,6\}, & S_6 \in \{2,5\}. \end{cases}$$

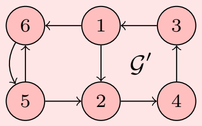
The implicit hypothesis is that, in solutions to the modelled problem, each vertex of the corresponding induced subgraph of $\mathcal{G}$ has exactly one successor.

B. *Since we were asked to have a single cycle we set the first argument of* `cycle` *to 1 and obtain* `cycle`$(1, \langle 1\ S_1, 2\ S_2, 3\ S_3, 4\ S_4, 5\ S_5, 6\ S_6\rangle)$.

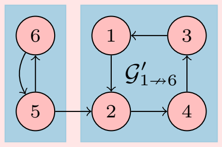
C. *Since there is a single cycle, $S_i \neq i$ (with $i \in [1,6]$). A necessary condition for the* `cycle` *constraint is that all its successor variables are assigned distinct values, i.e. each vertex has exactly one predecessor in a ground solution. Consequently, infeasible variable-value pairs for* `alldifferent` *are also infeasible for* `cycle`. *Any edge that does not belong to a matching of cardinality 6 in the corresponding variable-value graph $\mathcal{G}_{var}^{val}$ given on the right can not be part of a solution. As a result $\mathcal{G}'$ is shown below on the right.*



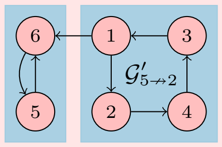
We now deal with the fact that we should have a single cycle. A necessary condition is that the graph $\mathcal{G}'$ consists of a single strongly connected component. We identify the arcs $u \mapsto v$ of $\mathcal{G}'$ such that, if they were removed, the number of strongly connected components of $\mathcal{G}'$ would be greater than one. For such arcs $u \mapsto v$ we remove all arcs $w \mapsto v$ (with $w \neq u$).



- *If we remove $1 \mapsto 6$ from $\mathcal{G}'$ we obtain $\mathcal{G}'_{1 \not\mapsto 6}$, which has the two strongly connected components depicted by the two blue rectangles. Consequently* the arc $5 \mapsto 6$ *is forbidden.*



- *If we remove $5 \mapsto 2$ from $\mathcal{G}'$ we obtain $\mathcal{G}'_{5 \not\mapsto 2}$, which has the two strongly connected components depicted by the two blue rectangles. Consequently the arc $1 \mapsto 2$ is forbidden.*



As a consequence we have a unique solution $S_1 = 6$, $S_2 = 4$, $S_3 = 1$, $S_4 = 3$, $S_5 = 2$, $S_6 = 5$ corresponding to the Hamiltonian cycle $1 \mapsto 6 \mapsto 5 \mapsto 2 \mapsto 4 \mapsto 3 \mapsto 1$.

SOLUTION TO EXERCISE 4

A. (a) *The variable-based degree of violation is equal to 1 since the* `alldifferent` *constraint holds and since we just have to correct the number of cycles (we have the two cycles $1 \mapsto 2 \mapsto 3 \mapsto 1$ and $4 \mapsto 4$ rather than one cycle). Therefore we only need to set the first argument of the* `cycle` *constraint to 2.*

$$\texttt{cycle}(\overset{2}{4}, \langle 1\ 2, 2\ 3, 3\ 1, 4\ 4\rangle)$$

(b) *Since we have two occurrences of 3 and two occurrences of 4 in the successor variables the variable-based degree of violation is at least equal to 2. Since, as shown below, it is possible the building of a single cycle $1 \mapsto 3 \mapsto 2 \mapsto 4 \mapsto 1$ by just changing the assignment of two variables, the variable-based degree of violation is equal to 2.*

$$\texttt{cycle}(1, \langle 1\ 3, 2\ 4, 3\ \overset{2}{3}, 4\ \overset{1}{4}\rangle)$$

(c) *Since we have two occurrences of 2 and two occurrences of 4 in the successor variables the variable-based degree of violation is at least equal to 2. Since just changing the values of two successor variables does not allow the building of 6 cycles the variable-based degree of violation is at least equal to 3. It is equal to 3 as shown by the following assignment that corresponds to the three cycles $1 \mapsto 2 \mapsto 1$, $3 \mapsto 4 \mapsto 3$, $5 \mapsto 6 \mapsto 5$.*

$$\texttt{cycle}(\overset{3}{6}, \langle 1\ 2, 2\ \overset{1}{2}, 3\ 4, 4\ \overset{3}{4}, 5\ 6, 6\ 5\rangle)$$

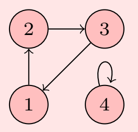
B. *Within the graph associated with the* `cycle` *constraint let $ncycle$, $nmap$ and $nsource$ respectively denote the number of connected components corresponding to a single cycle, the number of connected components with at least one source, and the number of sources.*

Given `N` *the first argument of the* `cycle` *constraint the variable-based degree of violation is equal to $nsource + \delta$ where δ is equal to 0 if* `N` $\in [ncycle + (nmap > 0), ncycle + nmap]$ *and 1 otherwise. The idea is that we have to change at least $nsource$ successor variables to fulfil the* `alldifferent` *constraint, and possibly the first argument* `N` *if we can not reach* `N` *cycles by just changing $nsource$ successor variables. The figures below illustrate the formula for the three examples of the previous question:*

(a) *We have* $0 + 4 \notin [2 + (0 > 0), 2 + 0] = 1$,

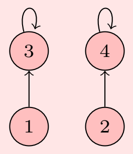
(b) *We have* $2 + 1 \notin [0 + (2 > 0), 0 + 2] = 2$,

(c) *We have* $2 + 6 \notin [1 + (2 > 0), 1 + 2] = 3$.



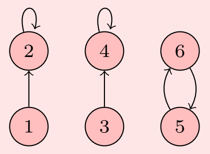
$ncycle = 2$
$nmap = 0$
$source = 0$

(a)



$ncycle = 0$
$nmap = 2$
$source = 2$

(b)

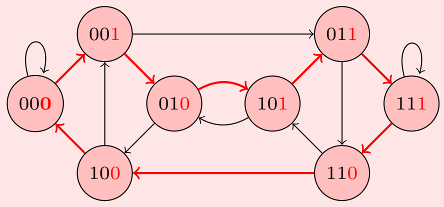


$ncycle = 1$
$nmap = 2$
$source = 2$

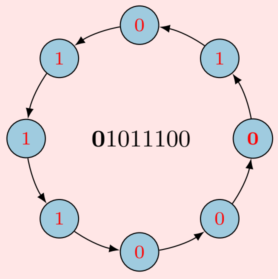
(c)

SOLUTION TO EXERCISE 5

A. *A De Bruijn sequence of order m over an alphabet $A = \{0, 1, \ldots, n-1\}$ can be seen as a Hamiltonian cycle*[a] *on the De Bruijn digraph of order m defined over the same alphabet A, where the sequence of letters corresponds to the sequence of last letters of the words associated with the successive vertices of the cycle. Visiting once each vertex of the digraph allows the corresponding cyclic sequence to contain exactly once each word of length m of the alphabet A.*



De Bruijn digraph of order 3 over $A = \{0, 1\}$



A De Bruijn sequence of order 3 over $A = \{0, 1\}$

B. *Each vertex of the De Bruijn graph associated with a word w is labelled by the decimal number plus one*[b] *corresponding to w. Then to each vertex of the De Bruijn graph corresponds a successor variable whose initial domain is set to the labels of the successors of v. Finally a `cycle` constraint with one cycle is posted.*

$$\left\{\begin{array}{llll} S_1 \in \{1,2\}, & S_2 \in \{3,4\}, & S_3 \in \{5,6\}, & S_4 \in \{7,8\}, \\ S_5 \in \{1,2\}, & S_6 \in \{3,4\}, & S_7 \in \{5,6\}, & S_8 \in \{7,8\}, \\ \multicolumn{4}{l}{\texttt{cycle}(1, \langle 1\ S_1, 2\ S_2, 3\ S_3, 4\ S_4, 5\ S_5, 6\ S_6, 7\ S_7, 8\ S_8 \rangle).} \end{array}\right.$$

A solution corresponds to the sequence $(S_1 - 1) \bmod n$, $(S_2 - 1) \bmod n$, $\cdots$, $(S_8 - 1) \bmod n$.

[a] Given a digraph $\mathcal{G}$ with p vertices, a *Hamiltonian cycle* of $\mathcal{G}$ is a succession of arcs $v_1 \mapsto v_2, v_2 \mapsto v_3, \cdots, v_{p-1} \mapsto v_p, v_p \mapsto v_1$ of $\mathcal{G}$ such that the vertices $v_1, v_2, \cdots, v_p$ are all distinct.

[b] $+1$ since, within the `cycle` constraint, vertices are labelled from 1 up to the total number of vertices.