Pattern Matching for Superpositional Graphs and Separable Permutations

Mati Tombak, Neeme Loorits, Ahti Peder, Leo Võhandu

University of Tartu/Tallinn University of Technology

July 4, 2013

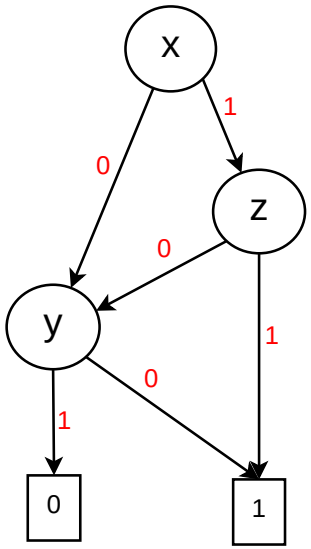
Binary Decision Diagram (BDD)

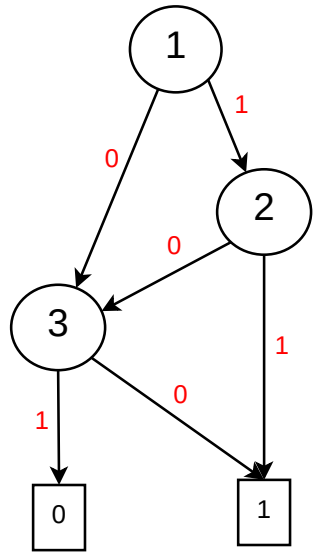
BDD for a Boolean function is constructed by recursive application of Shannon Expansion:

$$f(x_1,\ldots,x_n) = x_1 \,\&\, f_{x_1}(x_2,\ldots,x_n) \vee \overline{x_1} \,\&\, f_{\overline{x_1}}(x_2,\ldots,x_n)$$

C. Lee, “Representation of switching circuits by binary decision diagrams,” Bell. Syst. Tech. Journal, 38, 1959, pp. 985–999.

R. Bryant, “Graph-based algorithms for Boolean function manipulation,” IEEE Transaction on Computers, Vol. C-35 No 8., 1986, pp. 677–691.





BDD for a Boolean function $(x \& z) \vee \overline{y}$ and the corresponding binary graph.

Binary graph

Definition

A *binary graph* is an oriented acyclic connected graph with root and two terminals (sinks), 0 and 1. Every internal node v has two successors: $high(v)$ and $low(v)$.

An edge $a \to b$ is 0-*edge* (1-*edge*) if $low(a) = b$ ($high(a) = b$).

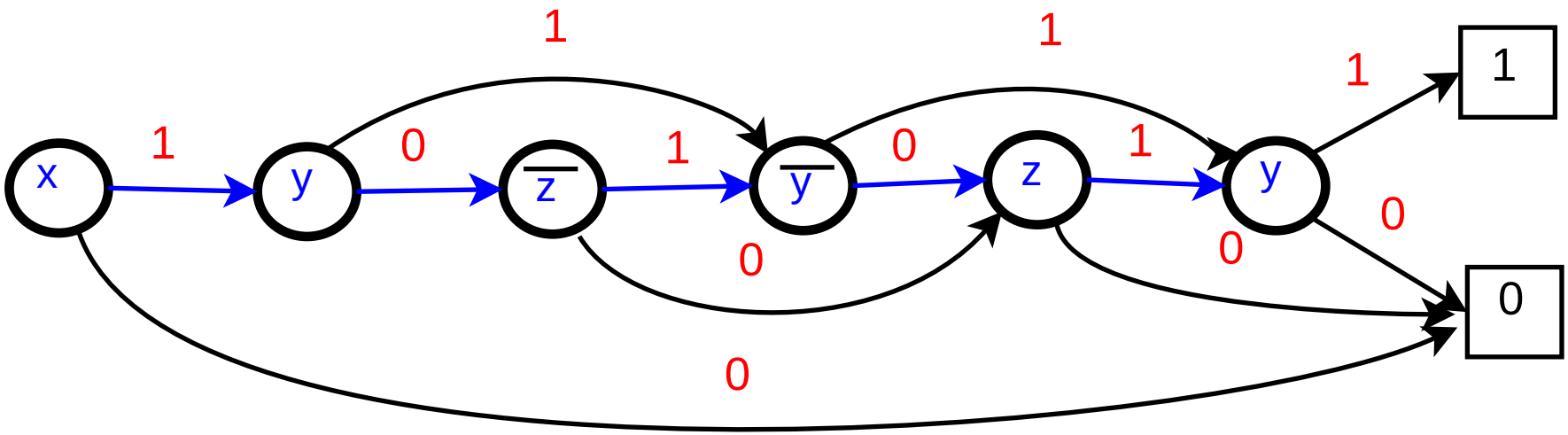
Binary graphs are skeletons of binary decision diagrams (BDD): a BDD is a binary graph, in which internal nodes are labeled by propositional variables.

Structurally Synthesized Binary Decision Diagram (SSBDD)

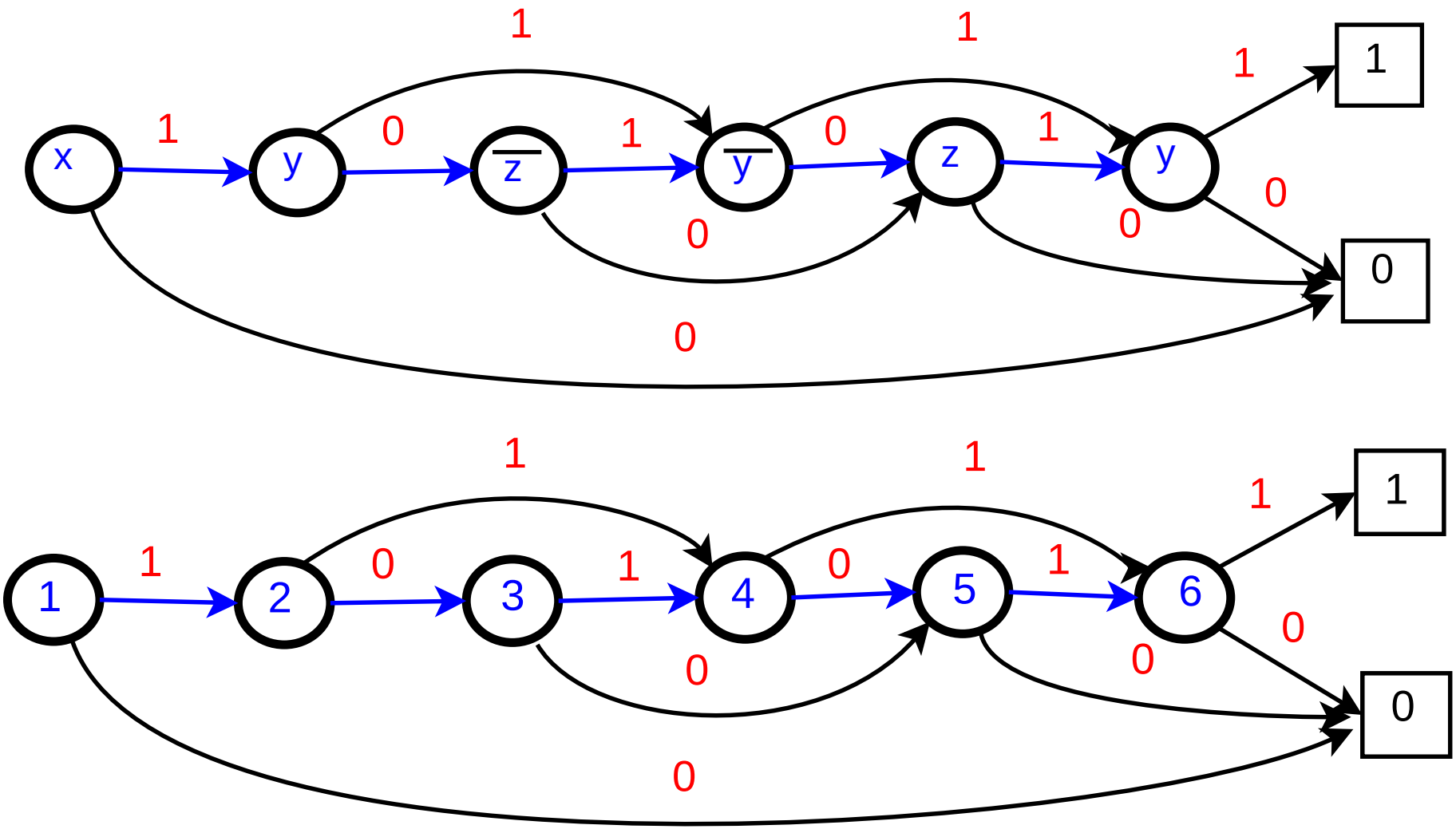
Suppose we have a Boolean function $f(x_1, \ldots, x_n)$, given by CNF and an assignment $(\alpha_1, \ldots, \alpha_n) \in \{0,1\}^n$.
We have to calculate the value of $f(\alpha_1, \ldots, \alpha_n)$.
We can do it by the definition, but if the performance is important, then better idea would be to do it from left to right.

$$(x \vee y \vee \overline{z}) \;\&\; (\overline{y} \vee z \vee y) \;\&\; (x \vee z \vee y)$$

$x \;\&\; (y \vee \overline{z}) \;\&\; (\overline{y} \vee z) \;\&\; y$



$x \;\&\; (y \vee \overline{z}) \;\&\; (\overline{y} \vee z) \;\&\; y$



The idea can be easily generalized to formula with arbitrary depth.

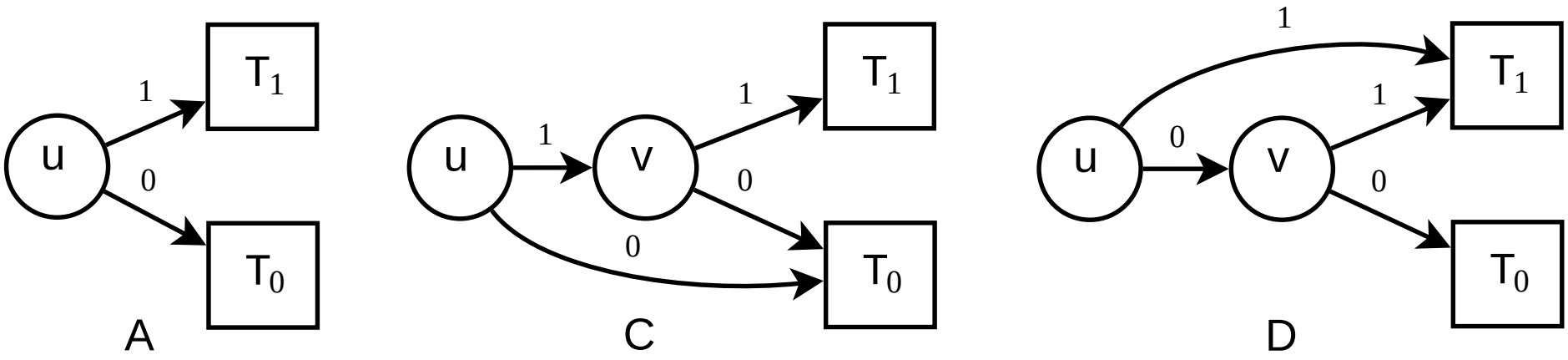
Superposition

Definition

A *superposition* of a binary graph E into a binary graph G instead of an internal node v, denoted by $G_{v \leftarrow E}$, is a graph, which we get by

- deleting v from G,
- redirecting all edges, pointing to v, to the root of E,
- redirecting all edges of E pointing to terminal 1, to the node $high(v)$,
- redirecting all edges, pointing to the terminal 0, to the node $low(v)$.

SuperPositional Graph (SPG)



Elementary graphs A, C ja D.

Definition

1° $A \in SPG$;

2° if $G \in SPG$ and $v \in V(G)$, then $G_{v \leftarrow C} \in SPG$ and $G_{v \leftarrow D} \in SPG$.

Constructors $\mathbf{C}, \mathbf{D}$

Elementary graphs C and D can be considered as constructors of superpositional graphs (we use bold $\mathbf{C}, \mathbf{D}$ to emphasize their role as constructors): if E and F are SPG with different sets of nodes, then

$$\mathbf{C}(E,F) = (\mathbf{C}[u \leftarrow E])[v \leftarrow F]$$

$$\mathbf{D}(E,F) = (\mathbf{D}[u \leftarrow E])[v \leftarrow F]$$

are SPGs. It is easy to see that constructors of superpositional graphs **C** and **D** are associative, so it is legal to use "long" constructors $\mathbf{C}(E_1, \ldots, E_n)$ and $\mathbf{D}(E_1, \ldots, E_n)$.

Can SPG be described without superposition?

We reproduce here the necessary and sufficient conditions for a binary graph to be a superpositional graph from the paper:

Ahti Peder and Mati Tombak, Superpositional graphs. Acta et Commentationes Universitatis Tartuensis de Mathematica*, 13, (2009), 51-64.*

Property 1: *Traceability*

Definition

A binary graph is *traceable*, if there exists a directed path through all intermediate nodes (Hamiltonian path).

It is easy to see, that if a Hamiltonian path exists in a binary graph, then it is unique. Therefore, it determines a canonical enumeration of nodes.

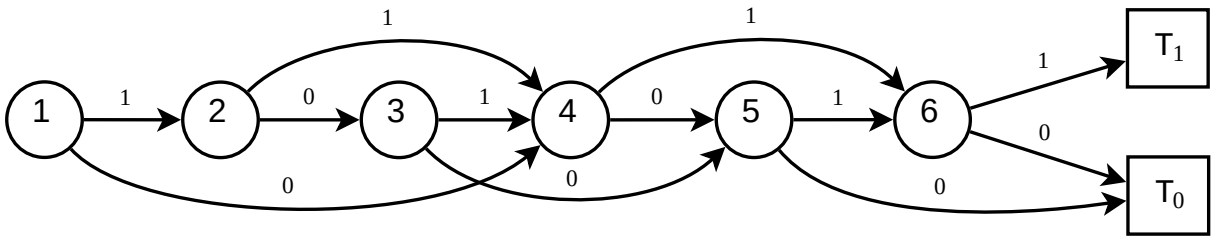
Theorem

Every SPG is traceable.

Property 2. *Homogeneity*

Definition

A binary graph is homogenous if only one type of edges (i.e. either 1-edges only or 0-edges only) enters into every node.



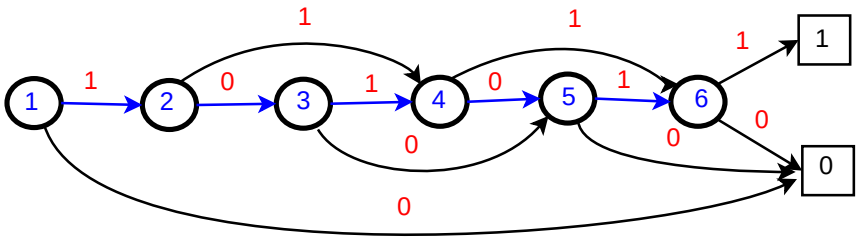
Theorem

Every SPG is homogenous.

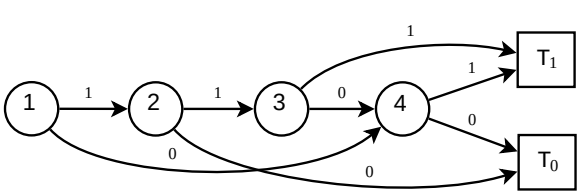
Property 3. *Strong planarity*

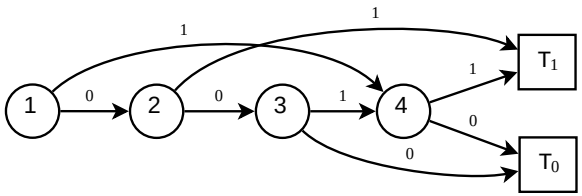
Definition

We say that a binary traceable graph is *strongly planar* if it has no crossing 0-edges and no crossing 1-edges in stretched drawing.



Strongly planar binary graph.





Binary graphs, which are not strongly planar.

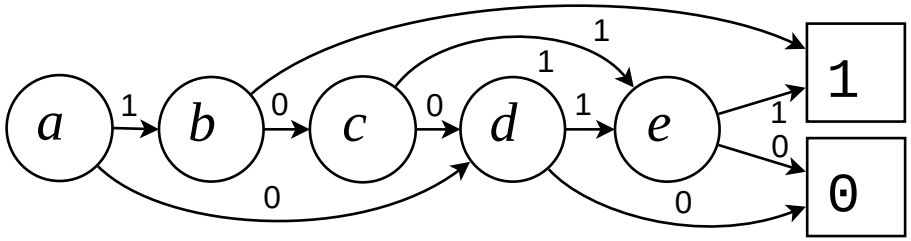
Theorem

Every SPG is strongly planar.

Property 4. *Cofinality*

Definition

We say that a binary traceable graph is 1-*cofinal (*0-*cofinal)* if all 1-edges (0-edges), starting between the endpoints of some 0-edge (1-edge) and crossing it, are entering into the same node. We say that a binary traceable graph is *cofinal* if it is 1-cofinal and 0-cofinal.



A binary graph which is not 1-cofinal.

Theorem

Every SPG is cofinal.

Necessary and sufficient conditions

Theorem

A binary graph is a superpositional graph if and only if it is a homogenous strongly planar cofinal traceable graph.

The proof of *only if* part of the theorem is a direct consequence of previous theorems. The proof of *if* part is more tricky and uses *Decomposition Lemma*.

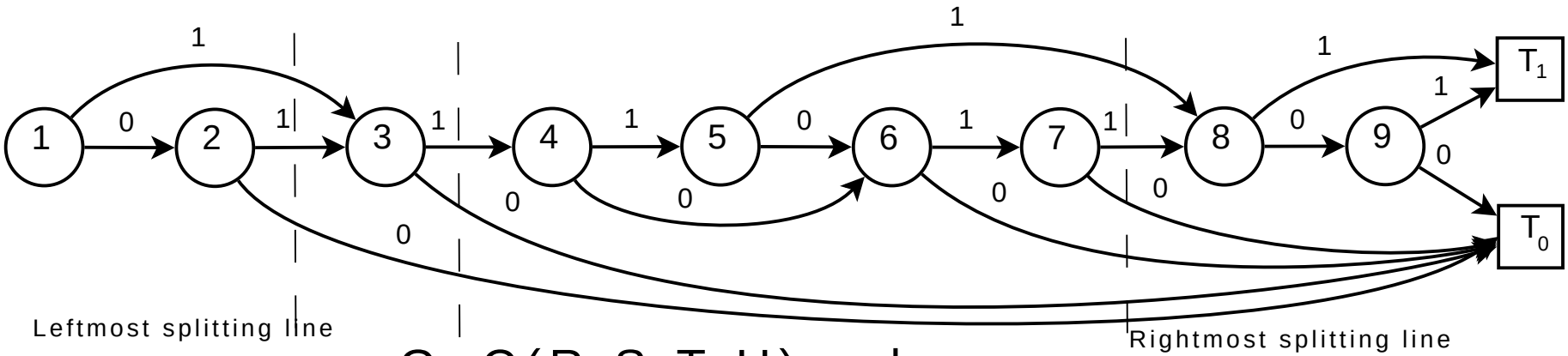
Decomposition Lemma

Lemma (Decomposition Lemma)

If G is an traceable homogenous strongly planar cofinal binary graph with nodes $1,\ldots,n$ $(n>1)$ in canonical order, m is a least node such that $m \xrightarrow{1} T_1$ and l is a least node such that $l \xrightarrow{0} T_0$, then:

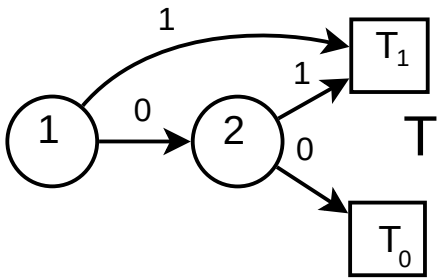
1. *If $l<m$ then G can be uniquely represented as $\boldsymbol{C}(G_1,\ldots,G_k)$ $(k>1)$ for some superpositional graphs $G_1,\ldots,G_k$. (we say that G is of type C).*
2. *If $m<l$ then G can be uniquely represented as $\boldsymbol{D}(G_1,\ldots,G_k)$ $(k>1)$ for some superpositional graphs $G_1,\ldots,G_k$. (we say that G is of type D).*

Decomposition of G into **C**(R,S,T,U)

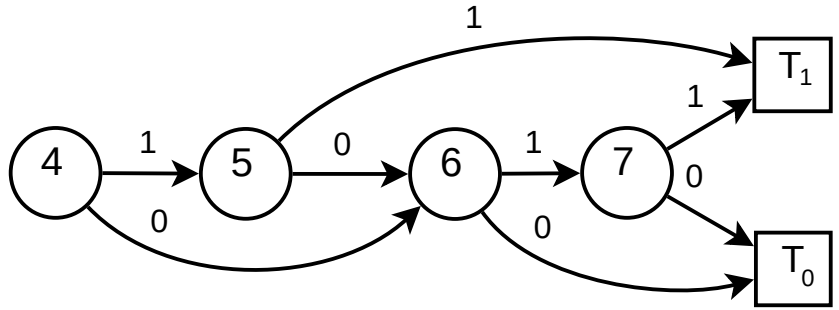


$G = C(R, S, T, U)$, where

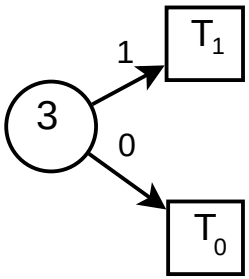
$R = G[1:2]$



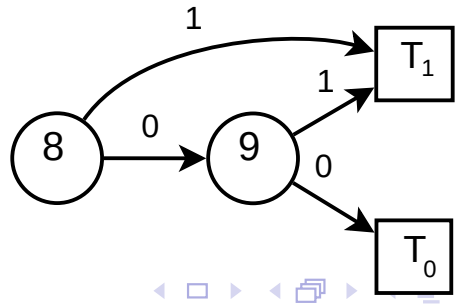
$T = G[4:7]$



$S = G[3,3]$



$U = G[8,9]$



Pattern Matching for SPG

Definition

The *pattern matching problem for superpositional graphs* is the following: Let T (text) and P (pattern) be superpositional graphs with internal nodes $1,\ldots,n$ and $1,\ldots,k$ ($k \leq n$). We say, that P matches into T if there exists a sequence of integers $i_1,\ldots,i_k$ such that:

1. For every arrow $l \xrightarrow{1} T1$ in P there exists a 1-path $i_l \rightsquigarrow T1$ in T, which consists of nodes from the set $\{i_l, i_l+1,\ldots,i_{l+1}-1\}$.
2. For every arrow $l \xrightarrow{0} T0$ in P there exists a 0-path $i_l \rightsquigarrow T0$ in T, which consists of nodes from the set $\{i_l, i_l+1\ldots,i_{l+1}-1\}$.
3. For every arrow $l \xrightarrow{1} m$ $(m \leq k)$ in P there exists a 1-path $i_l \rightsquigarrow i_m$ or there are indexes $r,s:\ r < i_m < s$ such that there exists a 1-path $i_l \rightsquigarrow r$ and $r-1 \xrightarrow{0} s$ in T.
4. For every arrow $l \xrightarrow{0} m$ $(m \leq k)$ in P there exists a 0-path $i_l \rightsquigarrow i_m$ or there are indexes $r,s:\ r < i_m < s$ such that there exists a 0-path $i_l \rightsquigarrow r$ and $r-1 \xrightarrow{1} s$ in T.

Notations

We denote by $G[k : l]$ a subgraph of G, induced by nodes $k, k+1, \ldots, l, T1, T0$ in which every edge $i \xrightarrow{1} m$ $(i \xrightarrow{0} m)$ for $m > l$ is redirected to $T1$ $(T0)$.
Let T be an SPG with nodes $T_1, \ldots, T_n$ (the text) and
P be an SPG with nodes $P_1, \ldots, P_k$ (the pattern).

Notations

If A and B are sets of sequences of integers, then $A \cup B$ denotes a union and $A \times B$ a Cartesian product of A and B. Note, that $A \times \emptyset = \emptyset \times A = \emptyset$.
$\{r\ r+1\ \ldots\ s\}$ is a set which consists of a single sequence $r\ r+1\ \ldots\ s$
$\{r, r+1, \ldots, s\}$ consists of $s-r+1$ sequences, each of length 1.
Variables X, Y, Z, V, W in Algorithm are local variables of type *set of integer sequences*.
Function *equivalent* checks if his arguments are equivalent up to the labels of internal nodes.
Function $split(G)$ returns a leftmost splitting point of G.
Our algorithm for pattern matching makes use of Decomposition Lemma and calculates matches recursively for a pair of SPG, induced by segments of $T_1, \ldots, T_n$ and $P_1, \ldots, P_k$.

An algorithm

match$(T[r:s], P[u:v])$
//returns a set of integer sequences, which are matches of SPG $P[u:v]$ into SPG $T[r:s]$.
Basis of recursion:
if $u = v$ **then return** $\{r, r+1, \ldots, s$ **fi**
//the pattern has length 1 – every position of the text is a match.
if $(v-u) > (s-r)$ **then return** $\emptyset$ **fi**
//the pattern is bigger than text.
if $(v-u) = (s-r)$ **then**
if *equivalent*$(T[r:s], P[u:v])$
then return $\{r\ r+1 \ldots s\}$
else return $\emptyset$
fi
fi

The recursive part I

// $(v-u) < (s-r)$
$dt = split(T[r:s])$;
$X := match(T[r:dt], P[u:v])$; //all matches of $P[u:v]$ in the left part.
$Y := match(T[dt+1:s], P[u:v])$; //all matches of $P[u:v]$ in the right part.
$Z := X \cup Y$

The recursive part II

if $type(T[r:s]) = type(P[u:v])$

for every splitting point dp in $P[u:v]$.

do

$V := match(T[r:dt], P[u:dp]);$

$W := match(T[dt+1:s], P[dp+1:v]);$

$Z := Z \cup (V \times W);$

fi:

od

return Z

end

Performance

An obvious modification of the Algorithm will calculate the number of matches.

By the Decomposition Lemma, every cut between the nodes of T and P performs the role of a splitting point exactly once in the full decomposition of the SPG.
It means, that the number of recursive calls of **match**$(T[r:s], P[u:v])$ would be at most nk if there were not multiple calls with the same pair of arguments.
We have an example with multiple calls.
To avoid multiple calls we have to store the number of matches for every combination of text and pattern. There are $n-1$ splitting points in the text and $p-1$ splitting points in the pattern, so we need a two-dimensional array $COUNT[1:n-1, 1:k-1]$. We assume, that we have prepared global array $COUNT[1:n-1, 1:k-1]$, filled in with constants -1.
So we achieve the time complexity $O(nk)$.

Mapping separable permutations into SPG

Definition

A *separable* n-permutation is a permutation, avoiding patterns 2413 and 3142, i.e. the class of permutations $S_n(2413, 3142)$.

Theorem

There is a bijection between a set of separable n-permutations and a set of superpositional graphs with n internal nodes.

A proof of the theorem can be found in:

L.Vohandu, A. Peder, M. Tombak. Permutations and bijections. Frontiers in AI and Applications, v. 237, 419-437. IOS Press (2012)

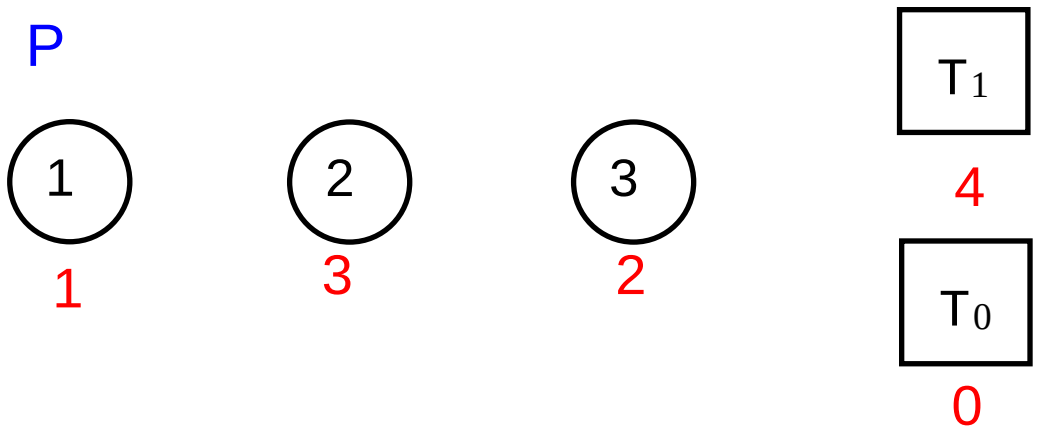
We need here the algorithm, implementing the mapping from separable permutations to SPG. Let G_p denote a superpositional graph, corresponding to a permutation p and p_G be a permutation, corresponding to a superpositional graph G.

sepperm2SPG

(separable permutation $p = p_1 \dots p_n$)
//returns a superpositional graph G_p.
begin
Augment the permutation to indices T_0 and T_1
taking $p(T_0) = 0$, $p(T_1) = n+1$.
Start with $n+2$ isolated nodes $1, \dots, n, T_1, T_0$;
for $i := 1$ **step** 1 **until** n
do set $i \xrightarrow{0} j$, where $j \in \{i+1, \dots, n, T_0\}$
is a least index for which $p(j) < p(i)$.
elseset $i \xrightarrow{1} j$, where $j \in \{i+2, \dots, n, T_1\}$
is a least index for which $p(j) > p(i)$.
od
end

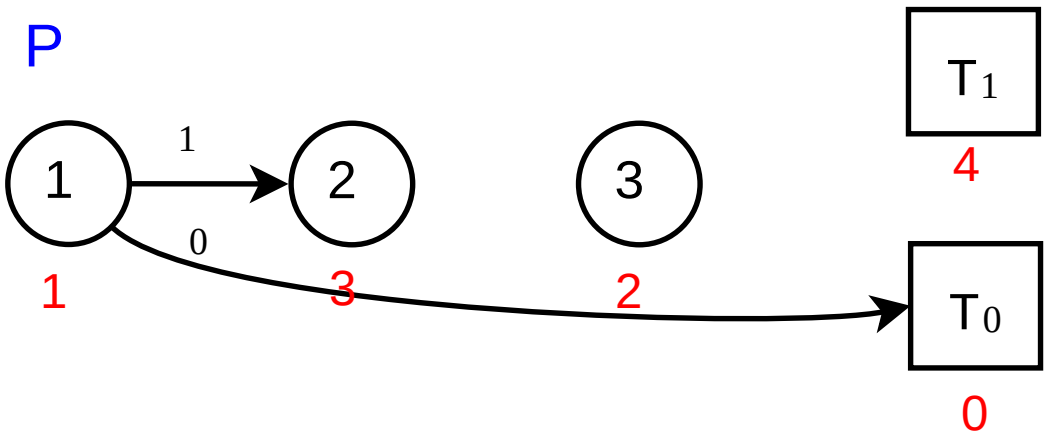
An example

$p = 132$



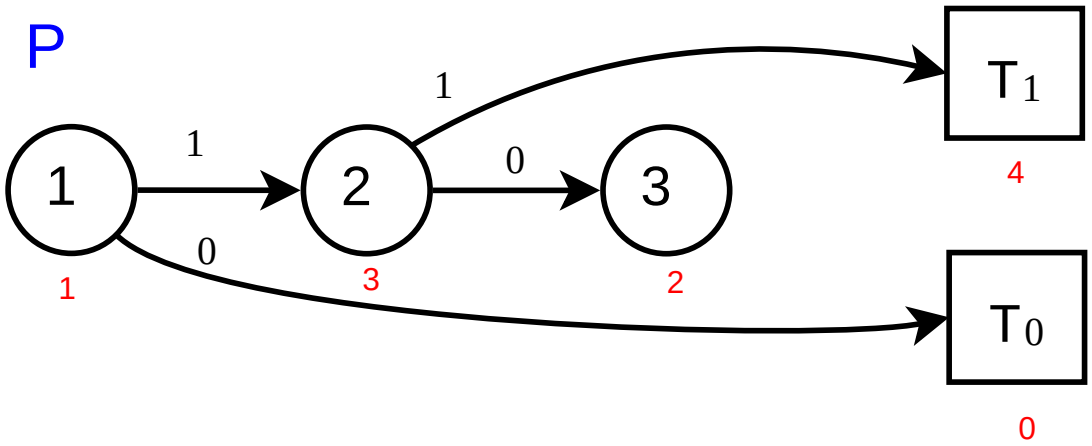
An example

$p = 132$



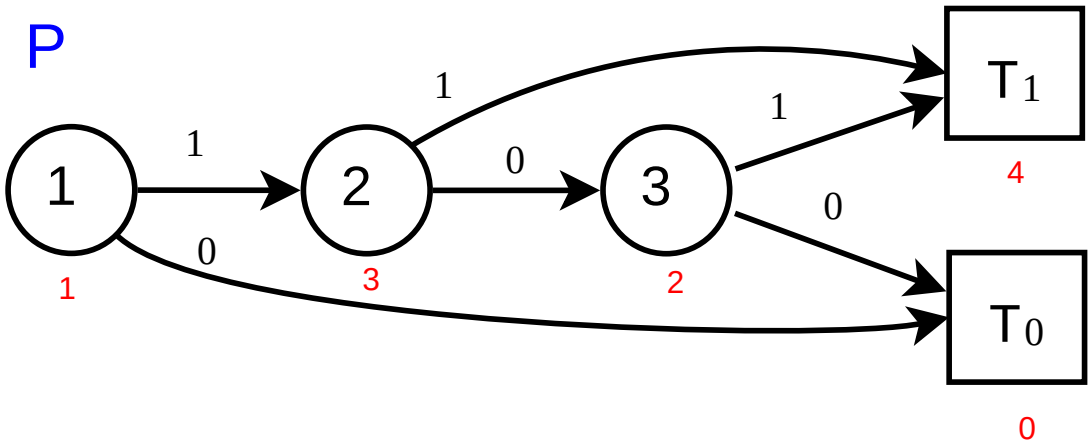
An example

$p = 132$



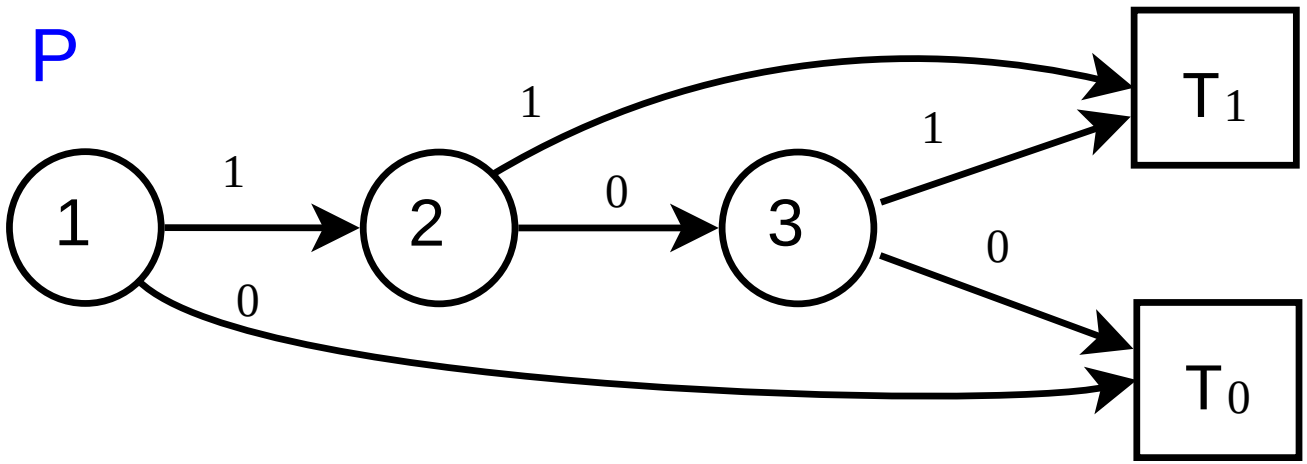
An example

$p = 132$



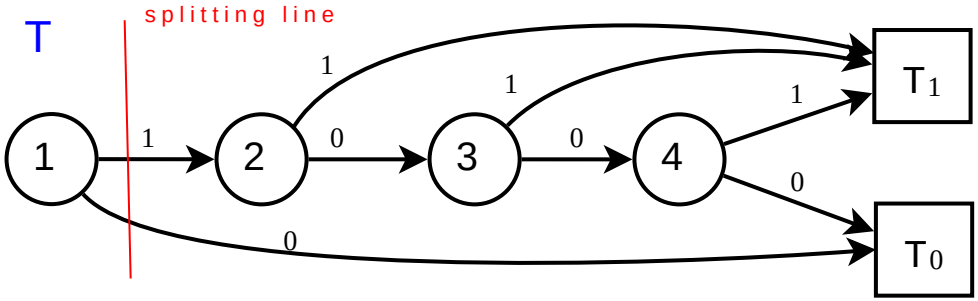
An example

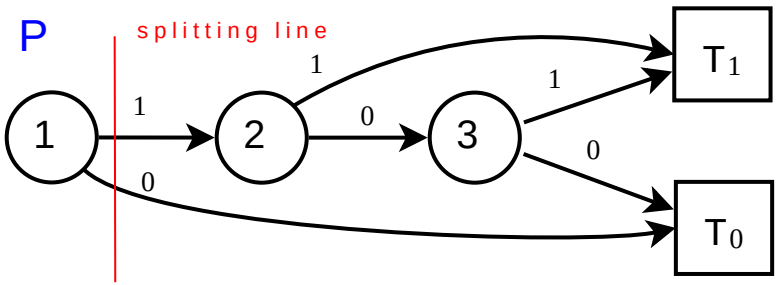
$p = 132$



Pattern matching. Example.

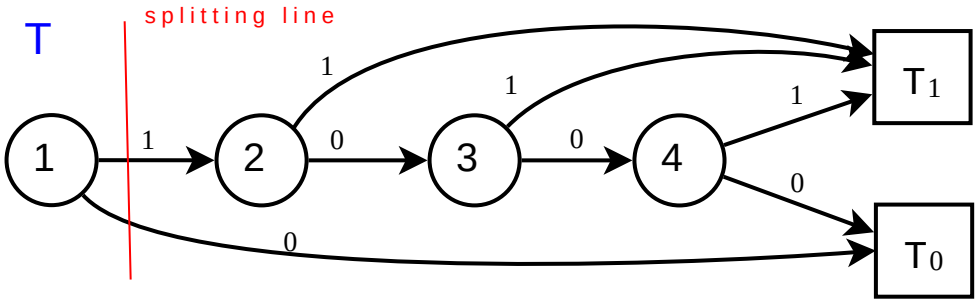
Let $t = 1432$, $p = 132$. Corresponding SPG-s are:

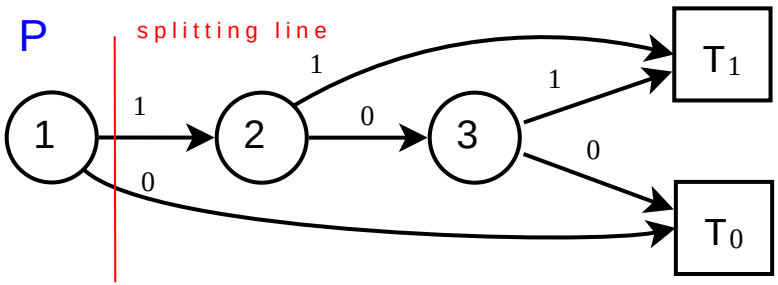




Let us apply algorithm *match* to these text and pattern!

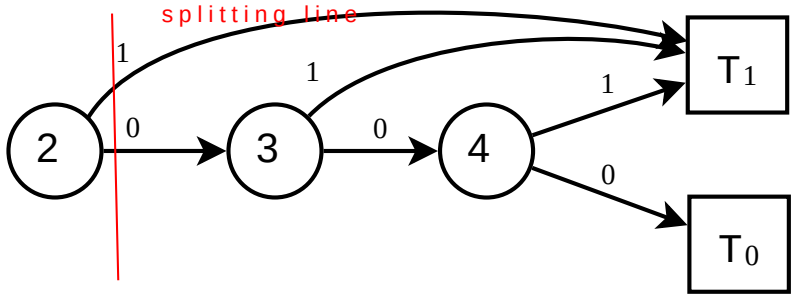
Pattern matching. Example.



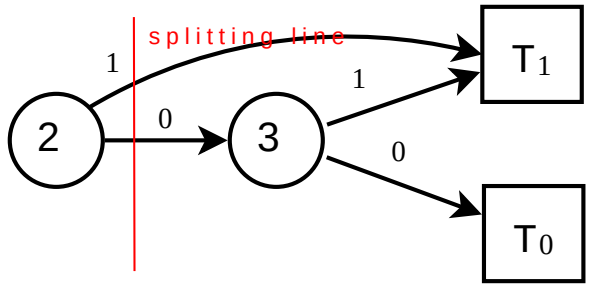


$$match(T[1:4], P[1:3]) =$$
$$= match(T[1:1], P[1:3]) \cup match(T[2:4], P[1:3]) \cup$$
$$\cup (match(T[1:1], P[1:1]) \times match(T[2:4], P[2:3])) =$$
$$= \emptyset \cup \emptyset \cup (\{1\} \times match(T[2:4], P[2:3])) =$$

T [2 : 4]



P [2 : 3]



$$\{1\} \times match(T[2:4], P[2:3]) =$$
$$\{1\} \times (match(T[2:2], P[2:3]) \cup match(T[3:4], P[2:3]) \cup$$
$$\cup (match(T[2:2], P[2:2]) \times match(T[3:4], P[3:3]))) =$$
$$\{1\} \times (\emptyset \cup \{34\} \cup (\{2\} \times \{3,4\})) =$$
$$\{1\} \times (\{34\} \cup (\{2\} \times \{3,4\})) =$$
$$\{134, 123, 124\}$$

Let me remind you, that $t = 1432$, $p = 132$.

Some theorems

Theorem

Algorithm maich$(T[1:n], P[1:k])$ *computes correctly all matches of pattern* P *in text* T*.*

Theorem

Let $t = t_1 \dots t_n$; $p = p_1 \dots p_k$ *be two separable permutations and* G_t, G_p *their superpositional graphs. A sequence of indexes* $i_1, \dots, i_k$ *is a match of* p *in* t *iff it is a match of* G_p *in* G_t*.*

Computer experimens for $n \leq 9$

Class of graphs	Sequence	Class of permutations	OEIS No.
1. Traceable	1, 2, 8, 48, 384,...	Double downgraded p.	A000165
2. Homogenous	1, 2, 6, 24, 120,...	Permutation	A000142
3. Strongly planar	1, 2, 6, 22, 92,...	Baxter permutation	A001181
4. Cofinal	1, 2, 6, 22, 90,...	Separable permutation	A006318

Computer experimens for $n \leq 9$

Class of graphs	Sequence	Class of permutations	OEIS No.
1. Traceable	1, 2, 8, 48, 384,...	Double downgraded p.	A000165
2. Homogenous	1, 2, 6, 24, 120,...	Permutation	A000142
3. Strongly planar	1, 2, 6, 22, 92,...	Baxter permutation	A001181
4. Cofinal	1, 2, 6, 22, 90,...	Separable permutation	A006318

1. Easy to see $t(n) = 2^{(n-1)} \cdot (n-1)!$.

Computer experimens for $n \leq 9$

Class of graphs	Sequence	Class of permutations	OEIS No.
1. Traceable	1, 2, 8, 48, 384,...	Double downgraded p.	A000165
2. Homogenous	1, 2, 6, 24, 120,...	Permutation	A000142
3. Strongly planar	1, 2, 6, 22, 92,...	Baxter permutation	A001181
4. Cofinal	1, 2, 6, 22, 90,...	Separable permutation	A006318

1. Easy to see $t(n) = 2^{(n-1)} \cdot (n-1)!$.
2. Bijection in [Vohandu, Peder, Tombak].

Computer experimens for $n \leq 9$

Class of graphs	Sequence	Class of permutations	OEIS No.
1. Traceable	1, 2, 8, 48, 384,...	Double downgraded p.	A000165
2. Homogenous	1, 2, 6, 24, 120,...	Permutation	A000142
3. Strongly planar	1, 2, 6, 22, 92,...	Baxter permutation	A001181
4. Cofinal	1, 2, 6, 22, 90,...	Separable permutation	A006318

1. Easy to see $t(n) = 2^{(n-1)} \cdot (n-1)!$.
2. Bijection in [Vohandu, Peder, Tombak].
3. Computer experiments until $n = 9$.

Computer experimens for $n \leq 9$

Class of graphs	Sequence	Class of permutations	OEIS No.
1. Traceable	1, 2, 8, 48, 384,...	Double downgraded p.	A000165
2. Homogenous	1, 2, 6, 24, 120,...	Permutation	A000142
3. Strongly planar	1, 2, 6, 22, 92,...	Baxter permutation	A001181
4. Cofinal	1, 2, 6, 22, 90,...	Separable permutation	A006318

1. Easy to see $t(n) = 2^{(n-1)} \cdot (n-1)!$.
2. Bijection in [Vohandu, Peder, Tombak].
3. Computer experiments until $n = 9$.
4. Bijection in [Vohandu, Peder, Tombak].

Computer experimens for $n \leq 9$

Class of graphs	Sequence	Class of permutations	OEIS No.
1. Traceable	1, 2, 8, 48, 384,...	Double downgraded p.	A000165
2. Homogenous	1, 2, 6, 24, 120,...	Permutation	A000142
3. Strongly planar	1, 2, 6, 22, 92,...	Baxter permutation	A001181
4. Cofinal	1, 2, 6, 22, 90,...	Separable permutation	A006318

1. Easy to see $t(n) = 2^{(n-1)} \cdot (n-1)!$.
2. Bijection in [Vohandu, Peder, Tombak].
3. Computer experiments until $n = 9$.
4. Bijection in [Vohandu, Peder, Tombak].

Problem

Find a bijection between permutations and homogenous binary graphs, which agrees with bijection [Vohandu, Peder, Tombak] and gives a bijection between Baxter permutations and strongly planar binary graphs, if limited to these classes.