Reachability Graphs of Two-Transition Petri Nets

Evgeny Erofeev* and Harro Wimmel**

Parallel Systems, Department of Computing Science
Carl von Ossietzky Universität, D-26111 Oldenburg, Germany
{evgeny.erofeev,harro.wimmel}@informatik.uni-oldenburg.de

Abstract. The reachability graph of a Petri net is a labelled transition system that can have a very complex structure. A characterisation of reachability graphs of Petri nets with only two transitions gives insight into typical structures that also might occur with arbitrarily large nets. An important means for such a characterisation is region theory, which allows to draw conclusions from cycles occurring in labelled transition systems. Attention is drawn especially to generalised cycles, i.e. cycles in the underlying undirected graph. Our characterisation also gives rise to an algorithm for the over-approximation of a given prefix-closed language by a Petri net language.

1 Introduction

When looking at a Petri net [5] and its reachability graph we can take system analysis or system synthesis as a point of view on their relationship. In system analysis, we can e.g. model a system by a marked Petri net and construct its (unique) reachability graph to analyse its behaviour [6]. In system synthesis, a behavioural specification is typically given, and a system implementing it is sought. For example, one may try to find a Petri net whose reachability graph is isomorphic to a given labelled transition system [1].

Our ultimate aim is to characterise, graph-theoretically, exactly the labelled transition systems that are synthesisable into a place/transition Petri net. To our knowledge, such a characterisation is difficult and has not yet been achieved in general. In the special case of binary words, there have been approaches via pattern matching and via letter counting [2–4] which successfully lead to a characterisation of Petri net synthesisable words, i.e. where the reachability graph consists of a single, specific sequence of states without any branching.

In this paper, we try to lift this limitation a bit and find the common structure of reachability graphs of Petri nets with only two transitions (generating sets of binary words). Based on region theory [1], we show how these reachability graphs can be partitioned into only a few classes each having strong common properties.

* Supported by DFG through grant Be 1267/14-1 CAVER and Graduiertenkolleg GRK-1765 SCARE http://www.uni-oldenburg.de/en/scare/
** Supported by DFG through grant Be 1267/15-1 ARS and grant Be 1267/16-1 ASYST

The classes closely depend on the presence or absence of generalised cycles in the reachability graph (i.e. cycles in the underlying undirected graph).

In case a given behaviour cannot be synthesised, we might ask for a small over-approximation, i.e. a reachability graph including all the required behaviour and allowing as few as possible additional words. If the given behaviour is finite, we can extend our results to obtain an algorithm for computing the minimal over-approximation by computing a convex hull for a subset of $\mathbb{N}^2$.

In section 2 we briefly recapitulate some basic definitions about labelled transition systems, Petri nets, and regions. Section 3 describes shapes of reachability graphs containing generalized cycles with non-zero Parikh vectors, while section 4 handles those where all generalised cycles have Parikh vector zero, essentially allowing a transformation to some subset of $\mathbb{N}^2$. In section 5 we present the algorithm for over-approximating the behaviour of a finite, binary, prefix-closed language, before concluding the paper in section 6.

2 Basic Concepts

LTS. A *labelled transition system* (LTS) with initial state is a tuple $TS = (S, \to, T, s_0)$ with nodes S (a countable set of states), edge labels T (a finite set of letters), edges $\to \subseteq (S \times T \times S)$, and an initial state $s_0 \in S$. An edge $(s, t, s') \in \to$ may be written as $s \xrightarrow{t} s'$ or $s' \xleftarrow{t} s$. We use $s \overset{t}{\rightsquigarrow} s'$ as an abbreviation for $(s \xrightarrow{t} s' \lor s \xleftarrow{t} s')$ and call it a generalised edge or g-edge (in the underlying undirected graph). A path (g-path) $\sigma \in T^*$ from s to s', written as $s \xrightarrow{\sigma} s'$ $(s \overset{\sigma}{\rightsquigarrow} s')$, is given inductively by $s = s'$ for the empty word $\sigma = \varepsilon$ and by $\exists s'' \in S$: $s \xrightarrow{w} s'' \xrightarrow{t} s'$ $(s \overset{w}{\rightsquigarrow} s'' \overset{t}{\rightsquigarrow} s')$ for $\sigma = wt$ with $w \in T^*$ and $t \in T$. A path $s \xrightarrow{\sigma} s'$ (g-path $s \overset{\sigma}{\rightsquigarrow} s'$) is a cycle (g-cycle) if and only if $s = s'$. It is called elementary if $|\sigma|$ is the number of different states occurring on the path, i.e. every state appears only once. The Parikh vector $\wp(\sigma) : T \to \mathbb{Z}$ of a g-path $s \overset{\sigma}{\rightsquigarrow} s'$ is a mapping defined by $\wp(\varepsilon) = \mathbf{0}$ (with $\mathbf{0}(t) = 0$ for all $t \in T$) for the empty word $\sigma = \varepsilon$, by $\wp(wt)(t') = \wp(w)(t')$ for $\sigma = wt$ and $t \neq t'$, by $\wp(wt)(t) = \wp(w)(t) + 1$ in case the g-path is $s \overset{w}{\rightsquigarrow} s'' \xrightarrow{t} s'$, and $\wp(wt)(t) = \wp(w)(t) - 1$ if the g-path is $s \overset{w}{\rightsquigarrow} s'' \xleftarrow{t} s'$ for some $s'' \in S$.

Two labelled transition systems $TS_1 = (S_1, \to_1, T, s_{01})$ and $TS_2 = (S_2, \to_2, T, s_{02})$ are isomorphic if there is a bijection $\zeta \colon S_1 \to S_2$ with $\zeta(s_{01}) = s_{02}$ and $(s, t, s') \in \to_1 \Leftrightarrow (\zeta(s), t, \zeta(s')) \in \to_2$, for all $s, s' \in S_1$. We call TS_1 a *sub-LTS* of TS_2 if $S_1 \subseteq S_2$, $\to_1 \subseteq \to_2$, and $s_{01} = s_{02}$. A *binary LTS* (bLTS) is an LTS $(S, \to, T, s_0)$ with $|T| = 2$, in this paper usually with $T = \{a, b\}$. A *word over* T is a sequence $w \in T^*$, and it is *binary* if $|T| = 2$. A word $w = t_1 t_2 \ldots t_n$ of length $n \in \mathbb{N}$ uniquely corresponds to a finite transition system $TS(w) = (\{\wp(\epsilon), \wp(t_1), \ldots, \wp(t_1 \ldots t_n)\}, \{(\wp(t_1 \ldots t_{i-1}), t_i, \wp(t_1 \ldots t_i)) \mid 0 < i \leq n \land t_i \in T\}, T, \wp(\epsilon))$. A *language over* T is a set $L \subseteq T^*$. For a finite language L we can uniquely define a transition system $TS(L) = \bigcup_{w \in L} TS(w)$, where for words $w_1, w_2 \in T^*$ and $TS(w_1) = (S_1, \to_1, T, \wp(\epsilon))$, $TS(w_2) = (S_2, \to_2, T, \wp(\epsilon))$ we write $TS(w_1) \cup TS(w_2) = (S_1 \cup S_2, \to_1 \cup \to_2, T, \wp(\epsilon))$.

Since we are interested in behaviour expressed as an LTS, we will assume LTS to be totally reachable, i.e. for every state there is a path from the initial state to it. This also means that each two states are connected by a g-path. In case of a not totally reachable LTS we may drop all unreachable states without loss of behaviour.

Petri net. An *initially marked Petri net* is denoted as $N = (P, T, F, M_0)$ where P is a finite set of places, T is a finite set of transitions, F is the flow function $F\colon ((P \times T) \cup (T \times P)) \to \mathbb{N}$ specifying the arc weights, and M_0 is the initial marking (where a marking is a mapping $M\colon P \to \mathbb{N}$, indicating the number of tokens in each place). A side-place is a place p with $p^\bullet \cap {}^\bullet p \neq \emptyset$, where $p^\bullet = \{t \in T \mid F(p,t){>}0\}$ and ${}^\bullet p = \{t \in T \mid F(t,p){>}0\}$. N is pure or side-place free if it has no side-places. A transition $t \in T$ is enabled at a marking M, denoted by $M[t\rangle$, if $\forall p \in P\colon M(p) \geq F(p,t)$. The firing of t leads from M to M', denoted by $M[t\rangle M'$, if $M[t\rangle$ and $M'(p) = M(p) - F(p,t) + F(t,p)$. This can be extended, as usual, to $M[\sigma\rangle M'$ for sequences $\sigma \in T^*$, and $[M\rangle$ denotes the set of markings reachable from M. The reachability graph $RG(N)$ of a Petri net N is the labelled transition system with the set of vertices $[M_0\rangle$, initial state M_0, label set T, and set of edges $\{(M, t, M') \mid M, M' \in [M_0\rangle \wedge M[t\rangle M'\}$. If a labelled transition system TS is isomorphic to the reachability graph of a Petri net N, we say that N *PN-solves* (or simply *solves*) TS, and that TS is *synthesisable* to N. We say that N *solves* a word w if it solves $TS(w)$. We frequently identify the states of TS with the markings of N then, writing e.g. $s(p) \geq F(p,t)$.

Reachability graphs (and their underlying LTS) have some nice properties we can make use of: They are fully deterministic ($s_1 \xleftarrow{\sigma_1} s \xrightarrow{\sigma_2} s_2 \wedge \wp(\sigma_1) = \wp(\sigma_2) \Longrightarrow s_1 = s_2$ and $s_1 \xrightarrow{\sigma_1} s \xleftarrow{\sigma_2} s_2 \wedge \wp(\sigma_1) = \wp(\sigma_2) \Longrightarrow s_1 = s_2$), totally reachable ($\forall s \in S\ \exists \sigma \in T^*\colon s_0 \xrightarrow{\sigma} s$), and zero-path-cyclic ($s \overset{\sigma}{\rightsquigarrow} s' \wedge \wp(\sigma) = \mathbf{0} \Longrightarrow s = s'$).

Separation problem. Let a labelled transition system $TS = (S, \to, T, s_0)$ be given. If we want to synthesise a Petri net with isomorphic reachability graph, T must be used directly as the set of transitions, since we do not consider any transition labels. In case the LTS is finite, we have to solve $\frac{1}{2}\cdot|S|\cdot(|S|-1)$ state separation problems for the places and up to $|S|\cdot|T|$ event/state separation problems, as follows:

- A *state separation problem* consists of a set of states $\{s, s'\}$ with $s \neq s'$, and it can be solved by a place that distinguishes them, i.e. has a different number of tokens in the markings corresponding to the two states.
- An *event/state separation problem* consists of a pair $(s, t) \in S \times T$ with $\neg(s \xrightarrow{t})$. For every such problem, one needs a place p such that $M(p) < F(p,t)$ for the marking M corresponding to state s, where F refers to the arcs of the hoped-for net. On the other hand, for every edge $s' \xrightarrow{t} s''$ we must guarantee $M'(p) \geq F(p,t)$, M' being the marking corresponding to state s'.

If the LTS is infinite, also the number of separation problems (of each kind) becomes infinite.

Region. An *abstract region* r of an LTS $(S, \to, \{t_1, \ldots, t_n\}, s_0)$ is a tuple $r = (r_0, r_1, \ldots, r_n)$ where $r_0 \in \mathbb{N}^S$ and $r_i \in \mathbb{Z}$ for $1 \leq i \leq n$ with the following consistency property: $\forall s, s' \in S\ \forall i \in \{1, \ldots, n\}$: $(s \xrightarrow{t_i} s' \Longrightarrow r_0(s') = r_0(s) + r_i)$. The property implies that for every g-cycle $s \overset{\sigma}{\rightsquigarrow} s$ in the LTS holds $\sum_{i=1}^{n} \wp(\sigma)(t_i) \cdot r_i = 0$. It follows that the value $r_0(s)$ is fully defined by $r_0(s_0)$ and the r_i with $1 \leq i \leq n$.

An abstract region in a reachability graph gives rise to an equivalence class of places p where $M_0(p) = r_0(s_0)$ and $r_i = F(t_i, p) - F(p, t_i)$. If the region distinguishes the states s and s' of two reachable markings M and M' (by $r_0(s) \neq r_0(s')$), the firing rule of the Petri net ensures that $M(p) \neq M'(p)$.

Lemma 1 (Indistinguishable Regions). *Let $r = (r_0, r_1, \ldots, r_n)$ be an abstract region of some (fully reachable) LTS $(S, \rightarrow, \{t_1, \ldots, t_n\}, s_0)$ with $s, s' \in S$. If r does not solve the state separation problem $\{s, s'\}$ (by distinguishing s and s'), neither does $r' = (k \cdot r_0 + i, k \cdot r_1, \ldots k \cdot r_n)$ with $i, k \in \mathbb{Z}$.*

Proof. Note: If $k \cdot r_0(s) + i < 0$, r' is not a region and there is nothing to prove. Assume $r_0(s) = r_0(s')$. Then, $(k \cdot r_0(s') + i) - (k \cdot r_0(s) + i) = k \cdot (r_0(s') - r_0(s)) = 0$. □

Places constructed from abstract regions are sufficient to deal with all state separation problems but not event/state separation problems. For fully determined places (with exactly known values for $F(p, t_i)$ and $F(t_i, p)$) one would have to consider more refined regions. Instead, we argument directly about the loops at some place p in a Petri net, i.e. about the value $k_t := \min\{F(p, t), F(t, p)\}$ for each transition t. Together with a region for p, they fully determine the arc weights between p and its neighboring transitions. Places derived from the same region, distinguished by the loop values k_t only, can easily be unified.

Lemma 2 (Loop maximisation). *Let $N = (P, T, F, M_0)$ be a Petri net with $p_1, p_2 \in P$ and $k_t \in \mathbb{Z}$ for each $t \in T$ such that $M_0(p_2) = M_0(p_1)$, $\forall t \in T$: $F(p_2, t) = F(p_1, t) + k_t$ and $F(t, p_2) = F(t, p_1) + k_t$. Define a new Petri net N' by adding k_t to each of $F(p_1, t)$ and $F(t, p_1)$ if $k_t > 0$, for every $t \in T$, and then deleting p_2 including all adjacent arcs. N' has the same reachability graph as N.*

Proof. Note that $M(p_1) = M(p_2)$ holds for all reachable markings M in N. A transition t is not firable in N at M, if $M(p_1) < F(p_1, t)$ or $M(p_2) < F(p_2, t)$, or put differently, if $M(p_1) < \max\{F(p_1, t), F(p_2, t)\}$. This maximum is exactly the arc weight $F(p_1, t)$ as defined for N'. Since the token change on p_1 by firing a transition is the same in N and N' and enabledness of transitions also remains unchanged from N to N', both nets have the same reachability graph. □

So, of all places constructed from the same region, we need only one, i.e. the place with the maximal number k_t of loops, separately computed for each adjacent transition t in the Petri net.

3 Generalised Cycles with Non-Zero Parikh Vectors

We would like to characterise the reachability graphs of Petri nets with at most two transitions. As a first case we consider only reachability graphs that contain at least one generalised cycle with a non-zero Parikh vector.

Theorem 1 (Shapes with Non-Zero Parikh Cycles). *If the reachability graph of a Petri net $(P, \{a, b\}, F, M_0)$ contains a g-cycle with a non-zero Parikh vector, it has one of the seven general shapes shown in Figures 1, 3, and 5 or it consists of just the initial state without any edges.*

Proof. Let $(S, \rightarrow, \{a, b\}, s_0)$ be a synthesisable bLTS, i.e. the reachability graph of some Petri net $(P, \{a, b\}, F, M_0)$, with some g-cycle $s \overset{\sigma}{\rightsquigarrow} s$ such that $\wp(\sigma) \neq \mathbf{0}$.

From the definition of abstract regions we know that every g-cycle must have a weighted sum of zero in any region of the LTS. Let $r = (r_0, r_a, r_b)$ be any such region, then $\wp(\sigma)(a) \cdot r_a + \wp(\sigma)(b) \cdot r_b = 0$. Our knowledge about r_a and r_b depends directly on the Parikh vector of σ.

Case 1: $\wp(\sigma)(a) = 0$. Obviously then, $\wp(\sigma)(b) \neq 0$ and thus $r_b = 0$, but we know nothing about r_a. As a consequence, following a b-edge $s \overset{b}{\rightarrow} s'$ in the LTS cannot modify the region value, i.e. $r_0(s') = r_0(s) + r_b = r_0(s)$. Therefore, because no region can distinguish s and s', and neither can any place distinguish the corresponding markings in the Petri net, the markings must be identical. We conclude that $s = s'$ for every b-edge $s \overset{b}{\rightarrow} s'$, all b-edges must be loops in the reachability graph. Figure 1 depicts all possible reachability graphs, with a (class of) example Petri nets on the left. The shape of the reachability graph depends on whether the occurring regions have positive, zero, or negative r_a values:

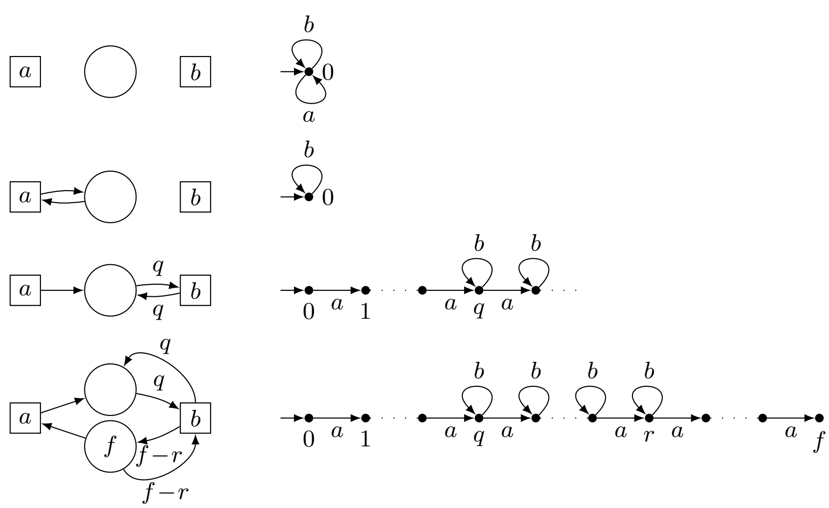
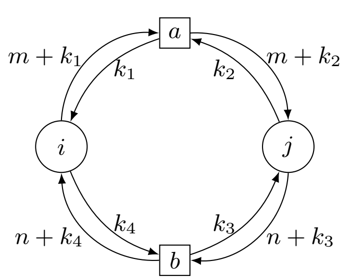


Fig. 1. Reachability graphs with loops (cycles of length 1). For an easy identification of states with markings in the Petri nets on the left side, consider states as consecutively named 0, 1, 2, ...

– If in all regions $r = (r_0, r_a, 0)$ holds $r_a = 0$, transition a cannot change the marking in the Petri net either. If a can fire at all, we obtain the reachability graph in the first row of Figure 1. If a cannot fire, we get the second row.
– If there is a region with $r_a > 0$ but none with $r_a < 0$, and assuming that a can fire at all, its firing will increase the number of tokens on any place connected to it. With increasing markings we obtain an infinite series of pairwise distinguishable states as in the third row of Figure 1. Since b occurs in the reachability graph, there is some earliest state q at which it forms a loop. As the markings rise from this point on, b is also activated at all later states.
– If there is a region with $r_a < 0$, there must be a place from which a removes tokens and eventually becomes deactivated. In the fourth line of Figure 1 a can fire exactly f times. The transition b can be deactivated at an earlier state r, if the place is a side-place of b (in the picture with arc weight $f - r$). If there is also a region with $r_a > 0$, b may not be activated until some state q just as in the previous subcase.

Case 2: $\wp(\sigma)(b) = 0$. This case is analogous to the previous one.

Case 3: $\wp(\sigma)(a) \cdot \wp(\sigma)(b) > 0$. Quite obviously then, $r_a \cdot r_b < 0$ and $r_b = -\frac{\wp(\sigma)(a)}{\wp(\sigma)(b)} \cdot r_a$, so r_a and r_b have a fixed ratio in every region. For every pair of such regions, we can use Lemma 1 to find a new region that is a common multiple with modified initial value r_0, i.e. any two such regions solve exactly the same state separation problems. Therefore, only two regions are of interest, one with $r_a < 0$ and $r_b > 0$ that can prevent an a, and one with $r_a > 0$ and $r_b < 0$ which can prevent b. By Lemma 2, the number of places constructed from each region can be reduced to one, the one with the maximal number of loops at each transition. Therefore, it is sufficient to have at most two regions, (r_0, r_a, r_b) and $(r'_0, -r_a, -r_b)$, and to construct one place from each region. A representative class of Petri nets with such two regions is shown in Figure 2.



$TS = (S, \to, \{a, b\}, 0)$ with

$S = \{-j, \ldots, +i\} \cap gcd(m, n) \cdot \mathbb{Z}$

$\forall x \in S:$

$x \xrightarrow{a} x + m \iff k_2 - j \leq x \wedge x + m \leq i - k_1$

$x \xrightarrow{b} x - n \iff k_3 - j \leq x - n \wedge x \leq i - k_4$

Fig. 2. A representative class of Petri nets for the case $\wp(\sigma)(a) \cdot \wp(\sigma)(b) > 0$. An LTS representing the reachability graph is shown on the right hand side. Some unreachable states are easy to exclude (by intersecting with $gcd(m, n) \cdot \mathbb{Z}$), others may occur if the initial state does not lie on a cycle

On the right hand side of Figure 2 we can see an LTS representing the reachability graph. For simplicity, we have named the initial state 0, which leads to states with negative numbers. By adding j, we can obtain the values of a possible region.

If we omit one of the two regions (places), the state space will become infinite, as either the boundary $-j$ or the boundary i will fall. We may think of this as replacing $-j$ by $-\infty$ or i by ∞. If we omit both places, the reachability graph collapses to the first row of Figure 1.

Figure 3 visualises an example of a reachability graph with a cycle. A Petri net according to Figure 2 has the parameters $m = 5$, $n = 3$, $i = 32$, $j = 0$, $k_1 = 0$, $k_2 = 0$, $k_3 = 9$, and $k_4 = 0$. The value $k_3 = 9$ ensures that the two initial a's cannot be undone. This also makes the states 1 to 4 and 6 to 8 unreachable. Incrementing i (by one) will add one a-edge at the highest state without such an edge to some new state and one b-edge at the new state. E.g., setting i to 33 will add an a-edge from 28 to 33 (new) and one b-edge from 33 to 30. Each increment will complete a new diamond in the graph (here with $28 \xrightarrow{b} 25 \xrightarrow{a} 30$). Incrementing j will analogously add a diamond at the other end. Adding loops via parameters k_1 to k_4 essentially cuts off such diamonds again, unless the cutting point is near the initial state. In this case, only one kind of edge is removed. The initial state will then not lie on a cycle anymore but form a path (using the other kind of edges) that leads to the cyclic part of the LTS. The parameters m and n determine the length of cycles in the LTS and the ratio of a and b on these cycles.

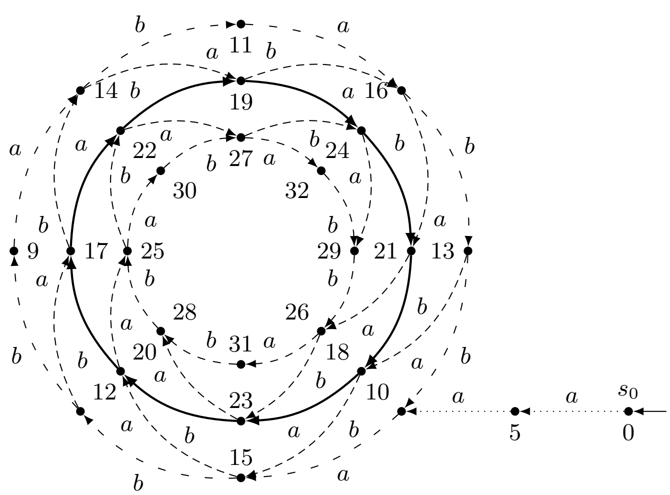
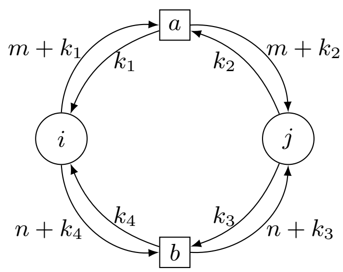


Fig. 3. Visualisation of a reachability graph with (standard) cycles

Case 4: $\wp(\sigma)(a) \cdot \wp(\sigma)(b) < 0$. With the same reasoning as in the previous case but $r_a \cdot r_b > 0$, we find that again two regions (r_0, r_a, r_b) and $(r'_0, -r_a, -r_b)$ and one place constructed from each region must be enough. This leads to the class of Petri nets shown in Figure 4.



$TS = (S, \to, \{a, b\}, 0)$ with

$S = \{0, \ldots, i\}$

$\forall x \in S$:

$x \xrightarrow{a} x + m \iff k_2 - j \leq x \wedge x + m \leq i - k_1$

$x \xrightarrow{b} x + n \iff k_3 - j \leq x \wedge x + n \leq i - k_4$

Fig. 4. A representative class of Petri nets for the case $\wp(\sigma)(a) \cdot \wp(\sigma)(b) < 0$. An LTS representing the reachability graph is shown on the right hand side. Some states may be unreachable

Unlike Case 3, there is a steady token flow from the left to the right place, no matter which transition is fired. Therefore, the right place determines when a transition may start firing and the left place when the firing must cease. Since the left place will at some point stop both transitions from firing, the reachability graph will normally be finite. Only if we remove the left place from the net, we can obtain an infinite behaviour (with states then named according to the number of tokens on the right place). These possibilities are shown in Figure 5.

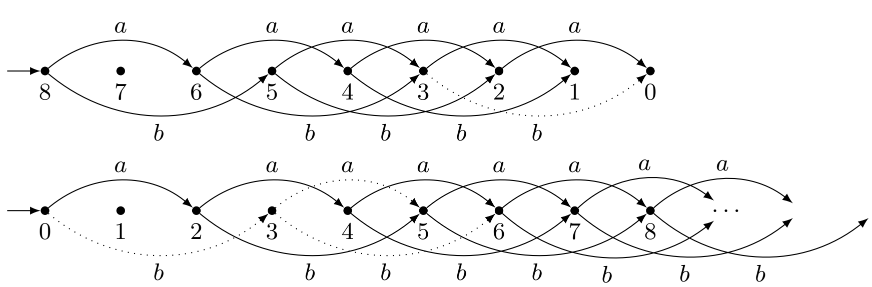


Fig. 5. Visualisation of reachability graphs containing g-cycles with non-zero Parikh vector, but no directed cycles

The upper reachability graph (with the dotted edge) corresponds e.g. to the Petri net with parameters $i = 8$, $j = 0$, $m = 2$, $n = 3$, and $k_1 = k_2 = k_3 = k_4 = 0$. Introducing loops at the left place will prevent the rightmost edges, i.e. setting

$k_4 = 1$ will eliminate the dotted b-edge, setting it to 2 will also prevent the b-edge ending at state 1, and so on.

The lower reachability graph corresponds to a Petri net without the left place and its adjacent edges. It has the parameters $j = 0$, $m = 2$, $n = 3$, and $k_2 = k_3 = 0$. Adding a loop at the right place prevents edges beginning at the initial state. Setting $k_3 = 1$ will remove the b-edge at 0 and make state 3 unreachable. Therefore, the edges from state 3 also become unusable (shown as dotted lines).

With cases 1 to 4, we have covered all possible g-cycles with non-zero Parikh vector that might occur in a reachability graph of a Petri net with transitions a and b. This concludes the proof of Theorem 1. ☐

4 Generalised Cycles with Zero Parikh Vectors only

Let us now assume, that our LTS does not have any g-cycle σ with $\wp(\sigma) \neq \mathbf{0}$. In this case, the transitions a and b are independent, and we may use a base transformation to project the LTS onto the plane $\mathbb{N}^2$, with the initial state mapped to $(0, 0)$. The transitions a and b become the base vectors, i.e. firing a increments the first component of a state, $(x, y) \xrightarrow{a} (x + 1, y)$, and firing b increments the second component, $(x, y) \xrightarrow{b} (x, y + 1)$, whenever the transitions are allowed.

Since all g-cycles σ have zero Parikh vectors, $\wp(\sigma) = \mathbf{0}$, the equation $\wp(\sigma)(a) \cdot r_a + \wp(\sigma)(b) \cdot r_b = 0$ does not restrict the values for r_a and r_b of a region in any way. If we distinguish regions (r_0, r_a, r_b) by the signs of r_a and r_b, there are essentially nine types of regions. Regions with non-negative values for r_a as well as r_b do not restrict the enabledness of transitions. With positive values for r_a and r_b, the five remaining types can be written as $(r_0, -r_a, +r_b)$, $(r_0, -r_a, 0)$, $(r_0, -r_a, -r_b)$, $(r_0, 0, -r_b)$, and $(r_0, +r_a, -r_b)$. An example LTS with one region of each of the five types is shown in Figure 6.

Let now $N = (P, \{a, b\}, F, M_0)$ be a Petri net and G the projection of the reachability graph onto $\mathbb{N}^2$. Furthermore, let R be the set of regions of G.

Lemma 3 (Inner States). *Let N be pure (i.e. $\forall p, t : F(p, t) \cdot F(t, p) = 0$) and $(x, y) \in G$ be some state. If $\forall (r_0, r_a, r_b) \in R$: $r_0((x+1, y)) \geq 0$, then $(x+1, y) \in G$ with $(x, y) \xrightarrow{a} (x+1, y)$. If $\forall (r_0, r_a, r_b) \in R$: $r_0((x, y+1)) \geq 0$, then $(x, y+1) \in G$ with $(x, y) \xrightarrow{b} (x, y + 1)$.*

Proof. Note first, that for any state $(x, y) \in \mathbb{N}^2$ and any region $(r_0, r_a, r_b) \in R$ holds $r_0((x + 1, y)) = r_0((x, y)) + r_a$ and $r_0((x, y + 1)) = r_0((x, y)) + r_b$ by the definition of a region, since we use a and b as base vectors in $\mathbb{N}^2$.

To prevent $(x, y) \xrightarrow{a}$ in G, there must be a region $(r_0, r_a, r_b) \in R$ with $r_a < 0$. For a place p constructed from such a region, $r_a = F(a, p) - F(p, a)$ must hold. Since N is pure, $F(p, a) > 0$ and $F(a, p) = 0$. Now, $r_0((x + 1, y)) - r_0((x, y)) = r_a = -F(p, a)$ and with $r_0((x+1, y)) \geq 0$ we conclude $r_0((x, y)) \geq F(p, a)$. Since by construction of p, $r_0((x, y))$ is the number of tokens on p at the state (x, y),

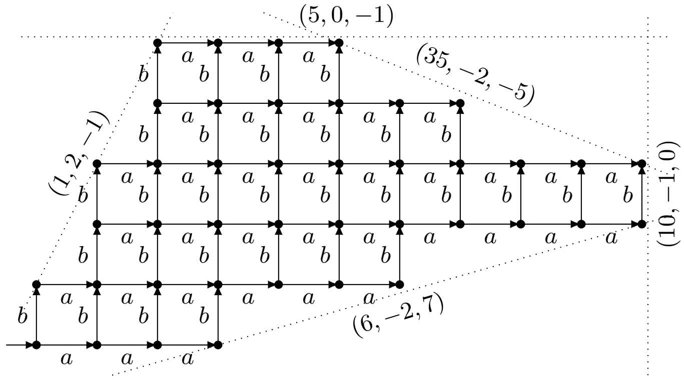


Fig. 6. An LTS without Parikh-non-zero g-cycles, limited by 5 regions (shown as dotted lines at which the value of a region (r_0, r_a, r_b) is zero). The initial state is at the lower left and can be viewed as the origin $(0, 0)$ of the plane $\mathbb{N}^2$, where a and b are the unit vectors of the two dimensions

p cannot prevent firing a at the corresponding marking. Since the region and p were arbitrary, $(x, y) \xrightarrow{a} (x + 1, y)$ holds in G, i.e. $(x + 1, y) \in G$.

An analogous reasoning holds for $(x, y + 1)$ and the transition b. □

Theorem 2 (Reachability graphs of pure nets without Parikh-non-zero g-cycles). *An LTS G in which all g-cycles have Parikh vector zero is the reachability graph of a pure Petri net* $N = (P, \{a, b\}, F, M_0)$ *if and only if it can be viewed as a weakly connected convex subset of* $\mathbb{N}^2$ *containing the initial state* $(0, 0)$ *such that for each two states* $(x, y), (x + 1, y) \in G$: $(x, y) \xrightarrow{a} (x + 1, y)$ *and for each two states* $(x, y), (x, y + 1) \in G$: $(x, y) \xrightarrow{b} (x, y + 1)$.

Proof. Note that convex subsets of $\mathbb{N}^2$ can be defined by cutting off parts of $\mathbb{N}^2$ using straight lines, and that these lines may not cut off the initial state, so our five types of regions give rise to exactly this kind of convex subset.

Using Lemma 3 proves that all necessary states and edges exist. In some extreme cases, e.g. with regions $(0, 1, -1)$, $(0, -1, 1)$, $(0, 1, 0)$, and $(0, 0, 1)$ we may obtain states that cannot be connected via g-paths, in this case the states (x, x) with $x \in \mathbb{N}$. Then, only the weakly connected component of the graph that contains the initial state forms the reachability graph.

It remains to show that a pure Petri net can be found such that its reachability graph does not identify any two of the states in $\mathbb{N}^2$. This can easily be done using the regions $(0, 1, 0)$ and $(0, 0, 1)$, i.e. by adding to each postset of a and b one new place with an empty initial marking, counting the number of a's or b's that have occurred. □

Figure 6 is a typical representative of such a reachability graph of a pure net (over transitions a and b). The regions cutting off parts of $\mathbb{N}^2$ may vary in number and direction, and they may not even make the graph finite, but the "inside area" (the convex subset of $\mathbb{N}^2$) will always be completely filled with states and edges. A pure Petri net synthesising the LTS from Figure 6 can be seen in Figure 7.

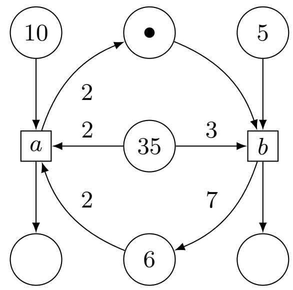


Fig. 7. A pure Petri net with the LTS from Figure 6 as its reachability graph. Each region forms one place. Though unnecessary in this case, we add regions $(0, 1, 0)$ and $(0, 0, 1)$ to ensure that no two states can have the same marking

To characterise the reachability graphs of non-pure nets in the same way, we need to consider the effect of adding a self-loop between a place (representing some region (r_0, r_a, r_b)) and the transition a or b. The dotted lines in Figure 6 show where the corresponding region has the value zero, or put differently, they are the border that edges cannot cross. We can also interpret them as two different lines at the same location, one preventing the a-edges, the other preventing the b-edges from crossing into the half-plane of negative region values. Let us call these lines the *a-line* and *b-line* of the region.

Lemma 4 (Shifting enabledness lines). *Let $N = (P, \{a, b\}, F, M_0)$ be a pure Petri net with a place $p \in P$ (with corresponding region (r_0, r_a, r_b)), and let N' be derived from N by adding $k \in \mathbb{N}$ to both $F(p, a)$ and $F(a, p)$. Let G and G' be the reachability graphs of N and N' projected to $\mathbb{N}^2$. Then, from G to G', the a-line is shifted by the fraction of $\frac{k}{r_a}$ of the unit-a-vector, unless $r_a = 0$, then it is shifted by the fraction of $\frac{k}{r_b}$ of the unit b-vector. If $r_a = r_b = 0$, either nothing happens (in case $r_0 \geq k$) or G' collapses and does not contain any a-edges (if $r_0 < k$).*

Proof. Note first, that G' is a subgraph of G since N' has a more restricted behaviour than N. If $r_a \neq 0$, any a-edge changes the number of tokens on p by r_a, therefore the $\frac{k}{r_a}$th fraction of an a-edge increases the number of tokens on p by k. This is exactly the additional amount of tokens needed in G' to allow an a-edge, thus the a-line moves from G to G' by the $\frac{k}{r_a}$th fraction of a unit-a-vector (possibly in the opposite direction if $r_a < 0$).

If $r_a = 0$, still a-edges are allowed only if the region value is at least k, since in N' we have $F(p, a) = k = F(a, p)$. A line representing the region value of k is exactly at a distance of $\frac{k}{r_b}$ times the unit-b-vector from the b-line, and becomes the new a-line.

If the region is $(r_0, 0, 0)$ neither transition can change the start value r_0. We either have $r_0 \geq k$, i.e. $r_0 \geq F(p, a) = F(a, p)$, so the number of tokens on p is sufficient to allow an a at all states. Or we have $r_0 < k$, in which case there are no a-edges in G' at all and G' represents some word from b^*. □

This lemma can obviously be formulated for the transition b as well. In Figure 8 we can see the effect of adding one self-loop with a for each of the five places representing one of the regions $(1, 2, -1)$, $(5, 0, -1)$, $(35, -2, -5)$, $(10, -1, 0)$, and $(6, -2, 7)$. The lemma also includes regions like $(0, 1, 0)$, that usually will not be shown in the LTS because they are positioned left of or below the origin. A shifted line from such a region can easily cut off the origin and thus collapse the reachability graph, as in the case of $(r_0, 0, 0)$ in the lemma.

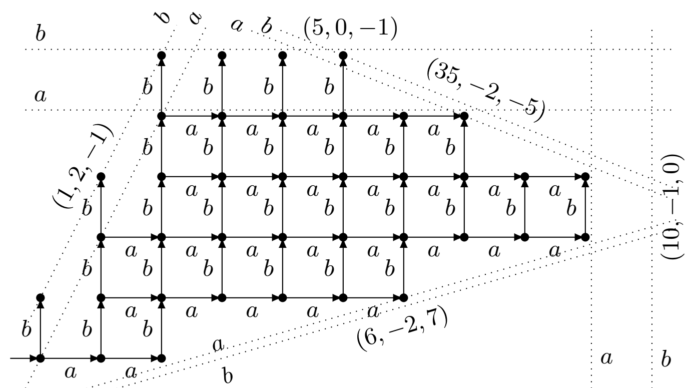


Fig. 8. The reachability graph of the net of Figure 7 if we add, for each of the five original regions, a self-loop between the corresponding place and a. The line where a region has value zero is split into two: one line that a-edges cannot permeate and one line that b-edges cannot cross (marked with letters a and b)

We can then characterise the reachability graphs of nets $(P, \{a, b\}, F, M_0)$ as follows.

Theorem 3 (Reachability graphs of nets without Parikh-non-zero g-cycles). *Let $C \subseteq \mathbb{N}^2$ be a convex area and let C_a, C_b be derived from C by shifting borders of C only. Let G be the graph including $(0, 0)$ such that $(x, y) \xrightarrow{a} (x + 1, y) \in G \iff (x, y) \in G \cap C_a, (x + 1, y) \in C_a$ and $(x, y) \xrightarrow{b} (x, y + 1) \in G \iff (x, y) \in G \cap C_b, (x, y + 1) \in C_b$. Then, G is the projection of a reachability graph of a Petri net $N = (P, \{a, b\}, F, M_0)$ to $\mathbb{N}^2$. If the reachability graph of a Petri net $(P, \{a, b\}, F, M_0)$ does not contain g-cycles σ with $\wp(\sigma) \neq \mathbf{0}$, C_a and C_b meeting the above conditions can always be found.*

5 Over-approximation of finite languages

Assume now that we have a finite language L over a binary alphabet $\{a, b\}$ given, and we aim to synthesise a Petri net which allows firing of all the words of this language. Due to the fact that for each Petri net, every prefix of a feasible transitions sequence is also feasible, we will assume L to be prefix-closed, i.e. for every $uv \in L$ with $u, v \in \{a, b\}^*$, it holds $u \in L$. For a Petri net N, let $\mathcal{L}(N)$ be the set of all transition sequences fireable in N. Since we consider only unlabelled Petri nets, it may be the case that there is no net N such that $\mathcal{L}(N) = L$. Nevertheless, there exists a net N such that $L \subseteq \mathcal{L}(N)$ (for instance, the net $N = (\{P_a, P_b\}, \{a, b\}, \{(P_a, a) \mapsto 1, (P_b, b) \mapsto 1\}, (l, l))$ with $l = \max_{w \in L} |w|$). Hence the challenging problem is to find a net N such that $L \subseteq \mathcal{L}(N)$ and the difference between $\mathcal{L}(N)$ and L is minimal. The characterisations established in Theorem 2 and 3 suggest a possible algorithm for such an over-approximation of finite languages.

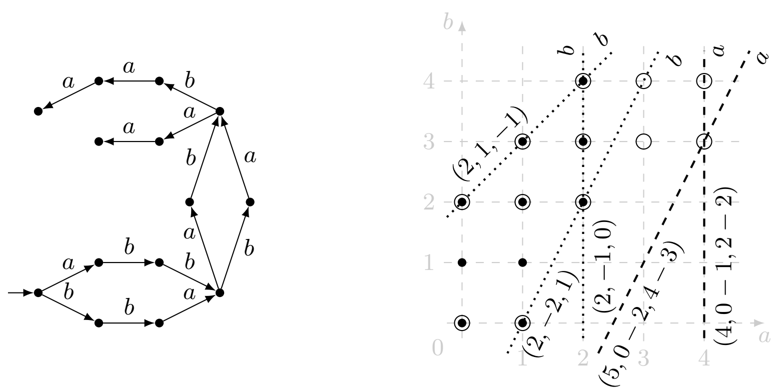


Fig. 9. The lts $TS(L)$ (l.h.s.) derived from language L. R.h.s.: solid dots denote the b-adjacent states, dotted lines represent the regions obtained from the convex hull of this set of states; circles denote a-adjacent states; dashed regions $(5, 0-2, 4-3)$ and $(4, 0-1, 2-2)$ were derived by transforming places corresponding to $(2, -2, 1)$ and $(2, -1, 0)$, respectively, into side-conditions.

Let $L = \overline{\{abbabaa,\ bbababaa\}}$, where over-line stands for prefix-closure, be an example language for which we seek to produce a Petri net whose language includes L. We can easily translate L into a labelled transition system $TS(L)$ without Parikh-non-zero g-cycles (l.h.s. in Figure 9). As we have established in Section 4, regions are represented as lines on $\mathbb{N}^2$, and these lines are essentially the borders that the projected arcs cannot cross. According to Lemma 4, being considered as a line, the same region can simultaneously impose for each transition its own border. In order to find the over-approximating Petri net, we first build regions for each letter, a and b, separately. E.g., for b in our example in Figure 9 we construct the regions $(2, 1, -1)$, $(2, -1, 0)$, and $(2, -2, 1)$, represented as dotted lines, which only take states into account that have an adjacent b-edge. These regions, together with the left and lower border of $\mathbb{N}^2$, form the convex hull for b, but some states only adjacent to a-edges may be outside, e.g. $(4, 3)$. Using the mechanism of Lemma 4 we can adjust the regions $(2, -1, 0)$ and $(2, -2, 1)$, obtaining new borders (drawn as dashed) for a-edges. Each region (border of the hull) is then translated into a place of a net. The same must be done for the states adjacent to a-edges, taking care of b-edges outside the convex hull. The sought net is obtained as a union of the places derived from the borders of a-adjacent and of b-adjacent states.

Algorithm 1 Over-approximation of a finite language

Input: finite (prefix-closed) language $L \in \{a, b\}^*$
Output: Petri net over-approximating L

compute sets $W_a = \{(x, y) \mid \wp(wa) = (x, y) \vee \wp(w) = (x, y),\ wa \in L\}$
$W_b = \{(x, y) \mid \wp(wb) = (x, y) \vee \wp(w) = (x, y),\ wb \in L\}$
find the convex hulls $H_a = ((x_i, y_i))_{0 \leq i \leq k_a} \subseteq W_a$ of W_a (enumerated clockwise)
$H_b = ((x_j, y_j))_{0 \leq j \leq k_b} \subseteq W_b$ of W_b (enumerated clockwise)
$(P_a, T, F_a, M_{0,a}) \leftarrow$ **partialSolution**(a, b, W_a, W_b, H_a)
$(P_b, T, F_b, M_{0,b}) \leftarrow$ **partialSolution**(b, a, W_b, W_a, H_b)
$N \leftarrow (P_a \cup P_b, \{a, b\}, F_a \cup F_b, M_{0,a} \cup M_{0,b})$
return N

procedure partialSolution(a, b, W_a, W_b, H)
{construct the net restricting the firings of one transition}
begin procedure
$m \leftarrow |H|$, $P \leftarrow \emptyset$ {find the size m of the hull, define the set of places P}
for $i = 0$ **to** $m - 1$ **do** {construct places from H taking into account W_b}
$r_a^i \leftarrow y_i - y_{i+1}$ {define a region as a line through two points}
$r_b^i \leftarrow x_{i+1} - x_i$ {orthogonal's direction accords with the ordering of H}
$r_0^i \leftarrow -r_a^i \cdot x_i - r_b^i \cdot y_i$
define place p_i
$M_0(p_i) \leftarrow r_0^i$
if $r_a^i \geq 0$ **then** $F(a, p_i) \leftarrow r_a^i$, $F(p_i, a) \leftarrow 0$ **else** $F(p_i, a) \leftarrow r_a^i$, $F(a, p_i) \leftarrow 0$
if $r_b^i \geq 0$ **then** $F(b, p_i) \leftarrow r_b^i$, $F(p_i, b) \leftarrow 0$ **else** $F(p_i, b) \leftarrow r_b^i$, $F(b, p_i) \leftarrow 0$
if $W_b \setminus W_a \neq \emptyset$
then
$(x', y') \leftarrow \arg\max_{\{(x,y) \in W_b \setminus W_a \mid r_0^i + x' \cdot r_a^i + y' \cdot r_b^i < 0\}} \{|r_0^i + x \cdot r_a^i + y \cdot r_b^i|\}$
{arg max returns the argument yielding the maximum of the set}
$k \leftarrow |x' \cdot r_a^i + y' \cdot r_b^i| - r_0^i$ {define the "moving factor"}
$M_0(p_i) \leftarrow M_0(p_i) + k$, $F(p_i, a) \leftarrow F(p_i, a) + k$, $F(a, p_i) \leftarrow F(a, p_i) + k$
{adjust the restrictions to include outer states}
add place p_i to P
endfor
return $(P, \{a, b\}, F, M_0)$
end procedure

Algorithm 1 describes this process formally. In order to construct a convex hull, one can use the `Quickhull` [7] algorithm which produces the hull in $O(n^2)$ in the worst case, where n is the size of the initial set (which is the set of states of the LTS in our case). For a language L we have $n \leq |L| \cdot l$ (l being the length of the longest word). The complexity of the **partialSolution** procedure is $O(n^2)$ in the worst case. Hence, the total complexity of the algorithm does not exceed $O(n^2)$. Applying the algorithm to the language L, we obtain the Petri net on the left hand side of Figure 10, its reachability graph being depicted on the right hand side.

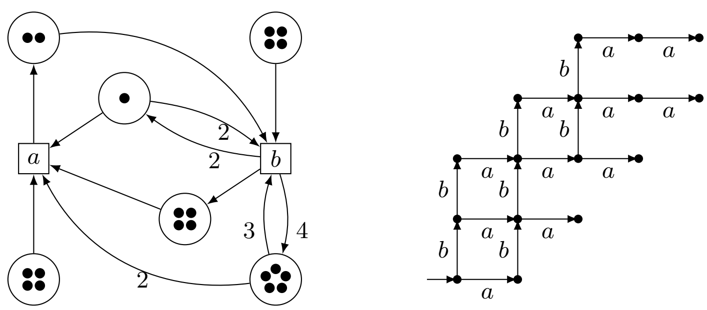


Fig. 10. The net (l.h.s.) obtained by the algorithm and its reachability graph (r.h.s.).

6 Conclusion

In this paper a graph-theoretical characterisation of the reachability graphs of Petri nets over the binary transition set is presented. The characterisation relies on the notion of generalised cycles. Based on this characterisation and the absence of Parikh-non-zero g-cycles, an algorithm for over-approximating a finite language by a Petri net language is suggested.

A natural continuation of this line of work is to use more than two transitions. Unluckily, just considering all pairs of transitions (others being "invisible") is insufficient, so even an extension to only three transitions is far from trivial.

References

1. É. Badouel, L. Bernardinello, P. Darondeau: Petri Net Synthesis. Springer-Verlag, ISBN 978-3-662-47966-7, 339 pages (2015).
2. Kamila Barylska, Eike Best, Evgeny Erofeev, Lukasz Mikulski, and Marcin Piatkowski: On binary words being Petri net solvable. In: ATAED'2015, Josep Carmona, Robin Bergenthum, Wil van der Aalst (eds), pp. 1-15, http://ceur-ws.org/Vol-1371.

3. Kamila Barylska, Eike Best, Evgeny Erofeev, Lukasz Mikulski, and Marcin Piatkowski: Conditions for Petri Net Solvable Binary Words. In ToPNoC XI (Transactions on Petri Nets and other Models of Concurrency), Jetty Kleijn, Jrg Desel (eds), Lecture Notes in Computer Science 9930, Springer, 2016, pp.137-159, DOI: 10.1007/978-3-662-53401-4_7.
4. E. Best, E. Erofeev, U. Schlachter, H. Wimmel: Characterising Petri Net Solvable Binary Words. In: Application and Theory of Petri Nets and Concurrency - 37th International Conference, (Petri Nets 2016), Fabrice Kordon, Daniel Moldt (eds.). Lecture Notes in Computer Science 9698, Springer, 2016, pp.39-58, DOI: 10.1007/978-3-319-39086-4_4.
5. T. Murata: Petri Nets: Properties, Analysis and Applications. Proc. of the IEEE, Vol. 77(4), 541-580 (1989).
6. W. Reisig: Understanding Petri Nets: Modeling Techniques, Analysis Methods, Case Studies. Springer-Verlag, ISBN ISBN 978-3-642-33278-4, 211 pages (2013).
7. W. Eddy: A new convex hull algorithm for planar sets. ACM Transactions on Mathematical Software, 1977.