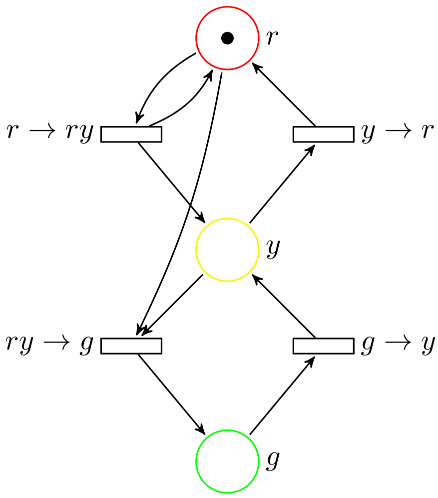
TD 9: Petri Nets

1 Modeling Using Petri Nets

Exercise 1 (Traffic Lights). Consider again the traffic lights example from the lecture notes:



1. How can you modify this Petri net so that it becomes 1-safe?
2. Extend your Petri net to model two traffic lights handling a street intersection.

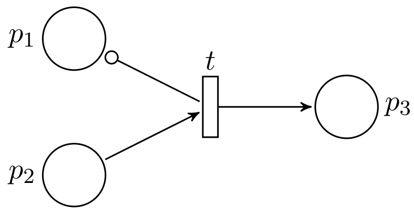
Exercise 2 (Producer/Consumer). A producer/consumer system gathers two types of processes:

producers who can make the actions *produce* (p) or *deliver* (d), and

consumers with the actions *receive* (r) and *consume* (c).

All the producers and consumers communicate through a single unordered channel.

1. Model a producer/consumer system with two producers and three consumers. How can you modify this system to enforce a maximal capacity of ten simultaneous items in the channel?
2. An *inhibitor arc* between a place p and a transition t makes t firable only if the current marking at p is zero. In the following example, there is such an inhibitor arc between p_1 and t. A marking $(0, 2, 1)$ allows to fire t to reach $(0, 1, 2)$, but $(1, 1, 1)$ does not allow to fire t.



Using inhibitor arcs, enforce a priority for the first producer and the first consumer on the channel: the other processes can use the channel only if it is not currently used by the first producer and the first consumer.

2 Unfoldings

Exercise 3 (Adequate Partial Orders)**.** A partial order $\prec$ between events is *adequate* if the three following conditions are verified:

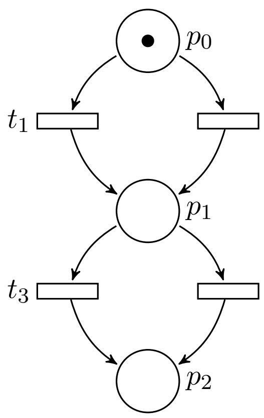
(a) $\prec$ is well-founded,

(b) $C_t \subsetneq C_{t'}$ implies $t \prec t'$, and

(c) $\prec$ is preserved by finite extensions: as in the lecture notes, if $t \prec t'$ and $M_t = M_{t'}$, and E and E' are two isomorphic extensions of C_t and $C_{t'}$ with $C_u = C_t \oplus E$ and $C_{u'} = C_{t'} \oplus E'$, then $u \prec u'$.

As you can guess, adequate partial orders result in complete unfoldings.

1. Show that $\prec_s$ defined by $t \prec_s t'$ iff $|C_t| < |C_{t'}|$ is adequate.
2. Construct the finite unfolding of the following Petri net using $\prec_s$; how does the size of this unfolding relate to the number of reachable markings?



3. Suppose we define an arbitrary total order $\ll$ on the transitions T of the Petri net, i.e. they are $t_1 \ll \cdots \ll t_n$. Given a set S of events and conditions of $\mathcal{Q}$, $\varphi(S)$ is

the sequence $t_1^{i_1} \cdots t_n^{i_n}$ in T^* where i_j is the number of events labeled by t_j in S. We also note $\ll$ for the lexicographic order on T^*.

Show that $\prec_e$ defined by $t \prec_e t'$ iff $|C_t| < |C_{t'}|$ or $|C_t| = |C_{t'}|$ and $\varphi(C_t) \ll \varphi(C_{t'})$ is adequate. Construct the finite unfolding for the previous Petri net using $\prec_e$.

4. There might still be examples where $\prec_e$ performs poorly. One solution would be to use a *total* adequate order. Give a 1-safe Petri net that shows that $\prec_e$ is not total.

3 Model Checking Petri Nets

Exercise 4 (Upper Bounds)**.** Let us fix a Petri net $\mathcal{N} = \langle P, T, F, W, m_0 \rangle$. We consider as usual propositional LTL, with a set of atomic propositions AP equal to P the set of places of the Petri net. We define proposition p to hold in a marking m in $\mathbb{N}^P$ if $m(p) > 0$.

The models of our LTL formulæ are *computations* $m_0 m_1 \cdots$ in $(\mathbb{N}^P)^\omega$ such that, for all $i \in \mathbb{N}$, $m_i \to_{\mathcal{N}} m_{i+1}$ is a transition step of the Petri net $\mathcal{N}$.

1. We want to prove that state-based LTL model checking can be performed in polynomial space for 1-safe Petri nets. For this, prove that one can construct an exponential-sized Büchi automaton $\mathcal{B}_{\mathcal{N}}$ from a 1-safe Petri net that recognizes all the infinite computations of $\mathcal{N}$ starting in m_0.

2. In the general case, state-based LTL model checking is undecidable. Prove it for Petri nets with at least two unbounded places, by a reduction from the halting problem for 2-counter Minsky machines.

3. We consider now a different set of atomic propositions, such that $\Sigma = 2^{\mathrm{AP}}$, and a labeled Petri net, with a labeling homomorphism $\lambda : T \to \Sigma$. The models of our LTL formulæ are infinite words $a_0 a_1 \cdots$ in Σ^ω such that $m_0 \xrightarrow{t_0}_{\mathcal{N}} m_1 \xrightarrow{t_1}_{\mathcal{N}} m_2 \cdots$ is an execution of $\mathcal{N}$ and $\lambda(t_i) = a_i$ for all i.

   Prove that action-based LTL model checking can be performed in polynomial space for labeled 1-safe Petri nets.

Exercise 5 (Lower Bounds for 1-Safe Petri Nets)**.** A *linear bounded automaton* (LBA) $\mathcal{M} = \langle Q, \Sigma \uplus \{\dashv, \vdash\}, \Gamma, \delta, q_0, \#, F \rangle$ is a Turing machine with a left endmarker $\dashv$ and a right endmarker $\vdash$,

- that cannot move left from $\dashv$ nor right from $\vdash$,
- that cannot print over $\dashv$ or $\vdash$, and
- that starts with input $\dashv x \vdash$ for some x in Σ^*.

A LBA is thus restricted to its initial tape contents. The membership problem for a LBA with input $\dashv x \vdash$ is PSPACE-hard.

1. Show how to simulate a LBA with input $\dashv x \vdash$ by a 1-safe Petri net of quadratic size.
2. Show that state-based LTL model checking is PSPACE-hard in the size of the Petri net for 1-safe Petri nets.
3. Show that action-based LTL model checking is PSPACE-hard in the size of the Petri net for labeled 1-safe Petri nets.

4 Coverability

The *coverability problem* for Petri nets is the following decision problem:

Instance: A Petri net $\mathcal{N} = \langle P, T, F, W, m_0 \rangle$ and a marking m_1 in $\mathbb{N}^P$.

Question: Does there exist m_2 in $\mathsf{Reach}_{\mathcal{N}}(m_0)$ such that $m_1 \leq m_2$?

For 1-safe Petri nets, coverability coincides with reachability, and is thus PSPACE-complete according to the previous exercises.

Exercise 6 (Inhibitor Arcs)**.** Prove that the coverability problem is undecidable for Petri nets having two inhibitor arcs.
(Hint: start by proving its undecidability for Petri nets with two places that are the sources of all the inhibitor arcs.)

Exercise 7 (Rackoff's Algorithm)**.** Let us fix a Petri net $\mathcal{N} = \langle P, T, F, W, m_0 \rangle$. We consider *generalized markings* in $\mathbb{Z}^P$. A *generalized computation* is a sequence $\mu_1 \cdots \mu_n$ in $(\mathbb{Z}^P)^*$ such that, for all $1 \leq i < n$, there is a transition t in T with $\mu_{i+1}(p) = \mu_i(p) - W(p,t) + W(t,p)$ for all $p \in P$ (i.e. we do not enforce enabling conditions). For a subset I of P, a generalized sequence is *I-admissible* if furthermore $\mu_i(p) \geq W(p,t)$ for all p in I at each step $1 \leq i < n$. For a value B in $\mathbb{N}$, it is *I–B-bounded* if furthermore $\mu_i(p) < B$ for all p in I at each step $1 \leq i \leq n$. A generalized sequence is an *I-covering* for m_1 if $\mu_1 = m_0$ and $\mu_n(p) \geq m_1(p)$ for all p in I.

Thus a computation is a P-admissible generalized computation, and a P-admissible P-covering for m_1 answers the coverability problem.

For a Petri net $\mathcal{N} = \langle P, T, F, W, m_0 \rangle$ and a marking m_1 in $\mathbb{N}^P$, let $\ell(\mathcal{N}, m_1)$ be the length of the shortest P-admissible P-covering for m_1 in $\mathcal{N}$ if one exists, and otherwise $\ell(\mathcal{N}, m_1) = 0$. For L, k in $\mathbb{N}$, define

$$M_L(k) = \sup\{\ell(\mathcal{N}, m_1) \mid |P| = k, \max\{W(p,t) \mid p \in P, t \in T\} + \max\{m_1(p) \mid p \in P\} \leq L\} .$$

1. Show that $M_L(0) \leq 1$.

2. We want to show that

$$M_L(k) \leq (L \cdot M_L(k-1))^k + M_L(k-1)$$

for all $k \geq 1$. To this end, we prove that, for every marking m_1 in $\mathbb{N}^P$ for a Petri net $\mathcal{N}$ with $|P| = k$,

$$\ell(\mathcal{N}, m_1) \leq (L \cdot M_L(k-1))^k + M_L(k-1)\ . \qquad (*)$$

Let

$$B = M_L(k-1) \cdot \max\{W(p,t) \mid p \in P, t \in T\} + \max\{m_1(p) \mid p \in P\}\ .$$

and suppose that there exists a P-admissible P-covering $w = \mu_1 \cdots \mu_n$ for m_1 in $\mathcal{N}$.

   (a) Show that, if w is P–B-bounded, then $(*)$ holds.

   (b) Assume the contrary: we can split w as $w_1 w_2$ such that w_1 is P–B-bounded and w_2 starts with a marking μ_j with a place p such that $\mu_j(p) \geq B$. Show that $(*)$ also holds.

3. Show that $M_L(|P|) \leq L^{(3 \cdot |P|)!}$ for $L = 2 + \max\{W(p,t) \mid p \in P, t \in T\} + \max\{m_1(p) \mid p \in P\}$.

4. Assuming that the size n of the instance $(\mathcal{N}, m_1)$ of the coverability problem is more than

$$\max\{\log L, |P|, \max\{\log W(t,p) \mid t \in T, p \in P\}\}\ ,$$

deduce that we can guess a P-admissible P-covering for m_1 of length at most $2^{2^{c \cdot n \log n}}$ for some constant c. Conclude that coverability can be solved in ExpSpace.

5 Vector Addition Systems

Exercise 8 (VASS)**.** An n-dimensional *vector addition system with states* (VASS) is a tuple $\mathcal{V} = \langle Q, \delta, q_0 \rangle$ where Q is a finite set of states, $q_0 \in Q$ the initial state, and $\delta \subseteq Q \times \mathbb{Z}^n \times Q$ the transition relation. A configuration of $\mathcal{V}$ is a pair (q, v) in $Q \times \mathbb{N}^n$. An execution of $\mathcal{V}$ is a sequence of configurations $(q_0, v_0)(q_1, v_1) \cdots (q_m, v_m)$ such that $v_0 = \bar{0}$, and for $0 < i \leq m$, $(q_{i-1}, v_i - v_{i-1}, q_i)$ is in δ.

1. Show that any VASS can be simulated by a Petri net.

2. Show that, conversely, any Petri net can be simulated by a VASS.

Exercise 9 (VAS)**.** An n-dimensional *vector addition system* (VAS) is a pair (v_0, W) where $v_0 \in \mathbb{N}^n$ is the initial vector and $W \subseteq \mathbb{Z}^n$ is the set of transition vectors. An execution of (v_0, W) is a sequence $v_0 v_1 \cdots v_m$ where $v_i \in \mathbb{N}$ for all $0 \leq i \leq m$ and $v_i - v_{i-1} \in W$ for all $0 < i \leq m$.

We want to show that any n-dimensional VASS $\mathcal{V}$ can be simulated by an $(n+3)$-dimensional VAS (v_0, W).

Hint: Let $k = |Q|$, and define the two functions $a(i) = i + 1$ and $b(i) = (k+1)(k-i)$. Encode a configuration (q_i, v) of $\mathcal{V}$ as the vector $(v(1), \ldots, v(n), a(i), b(i), 0)$. For every state q_i, $0 \leq i < k$, we add two transition vectors to W:

$$t_i = (0, \ldots, 0, -a(i), a(k-i) - b(i), b(k-i))$$
$$t'_i = (0, \ldots, 0, b(i), -a(k-i), a(i) - b(k-i))$$

For every transition $d = (q_i, w, q_j)$ of $\mathcal{V}$, we add one transition vector to W:

$$t_d = (w(1), \ldots, w(n), a(j) - b(i), b(j), -a(i))$$

1. Show that any execution of $\mathcal{V}$ can be simulated by (v_0, W) for a suitable v_0.
2. Conversely, show that this VAS (v_0, W) simulates $\mathcal{V}$ faithfully.