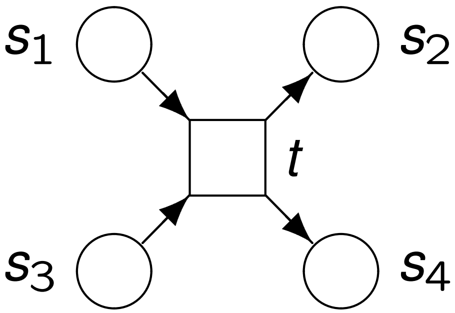
Petri nets

Petri nets

Petri nets are a basic model of parallel and distributed systems (named after Carl Adam Petri). The basic idea is to describe state changes in a system with transitions.



Petri nets contain places ○ and transitions □ that may be connected by directed arcs.

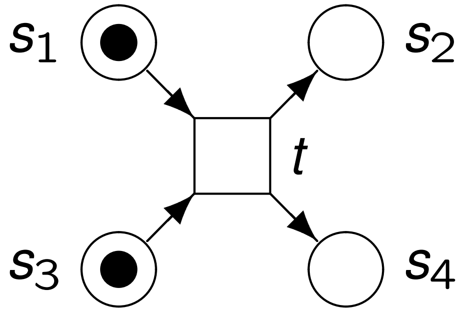
Places symbolise states, conditions, or resources that need to be met/be available before an action can be carried out.

Transitions symbolise actions.

Behaviour of Petri nets

Places may contain tokens that may move to other places by executing ("firing") actions.

A token on a place means that the corresponding condition is fulfilled or that a resource is available:



In the example, transition t may "fire" if there are tokens on places s_1 and s_3. Firing t will remove those tokens and place new tokens on s_2 and s_4.

Why Petri Nets?

low-level model for concurrent systems

expressly models concurrency, conflict, causality, . . .

finite-state or infinite-state models

Content:

Semantics of Petri nets

Modelling with Petri nets

Analysis methods: finite/infinite-state case, structural analysis

Remark: Many variants of Petri nets exist in the literature; we regard a special simple case also called P/T nets.

Petri Net

A Petri net is a tuple $N = \langle P, T, F, W, m_0 \rangle$, where

- P is a finite set of places,
- T is a finite set of transitions,
- the places P and transitions T are disjoint ($P \cap T = \emptyset$),
- $F \subseteq (P \times T) \cup (T \times P)$ is the flow relation,
- $W\colon ((P \times T) \cup (T \times P)) \to \mathbb{N}$ is the arc weight mapping (where $W(f) = 0$ for all $f \notin F$, and $W(f) > 0$ for all $f \in F$), and
- $m_0\colon P \to \mathbb{N}$ is the initial marking representing the initial distribution of tokens.

Semantics

Let $N = \langle P, T, F, W, m_0 \rangle$ be a Petri net. We associate with it the transition system $\mathcal{M} = \langle S, \Sigma, \Delta, I, AP, \ell \rangle$, where:

$$S = \{\, m \mid m \colon P \to \mathbb{N} \,\},\;\; I = \{m_0\}$$

$$\Sigma = T$$

$$\Delta = \{\, (m, t, m') \mid \forall p \in P : m(p) \geq W(p,t) \wedge m'(p) = m(p) - W(p,t) + W(t,p) \,\}$$

$$AP = P,\;\; \ell(m) = \{\, p \in P \mid m(p) > 0 \,\}$$

When $(m, t, m') \in \Delta$, we say that t is enabled in m and that its firing produces the successor marking m'; we also write $m \xrightarrow{t} m'$.

Semantics (remark)

The semantics given on the previous slide is also called interleaving semantics (one transition fires at a time).

Alternatively, one could define a step semantics, which better expresses the concurrent behaviours.

In step semantics, one allows a *multiset* of transitions to fire simultaneously; i.e. a multiset A is enabled in marking m if m contains enough tokens to fire all transitions in A.

However, for our purposes the interleaving semantics is sufficient.

Petri nets: Remarks

If $\langle p, t\rangle \in F$ for a transition t and a place p, then p is an input place of t,

If $\langle t, p\rangle \in F$ for a transition t and a place p, then p is an output place of t,

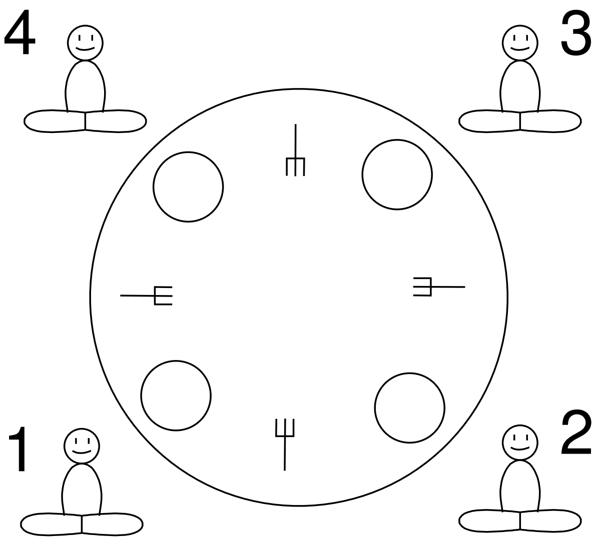
Let $a \in P \cup T$. The set ${}^\bullet a = \{a' \mid \langle a', a\rangle \in F\}$ is called the pre-set of a, and the set $a^\bullet = \{a' \mid \langle a, a'\rangle \in F\}$ is its post-set.

When drawing a Petri net, we usually omit arc weights of 1. Also, we may either denote tokens on a place either by black circles, or by a number.

Example: Dining philosophers

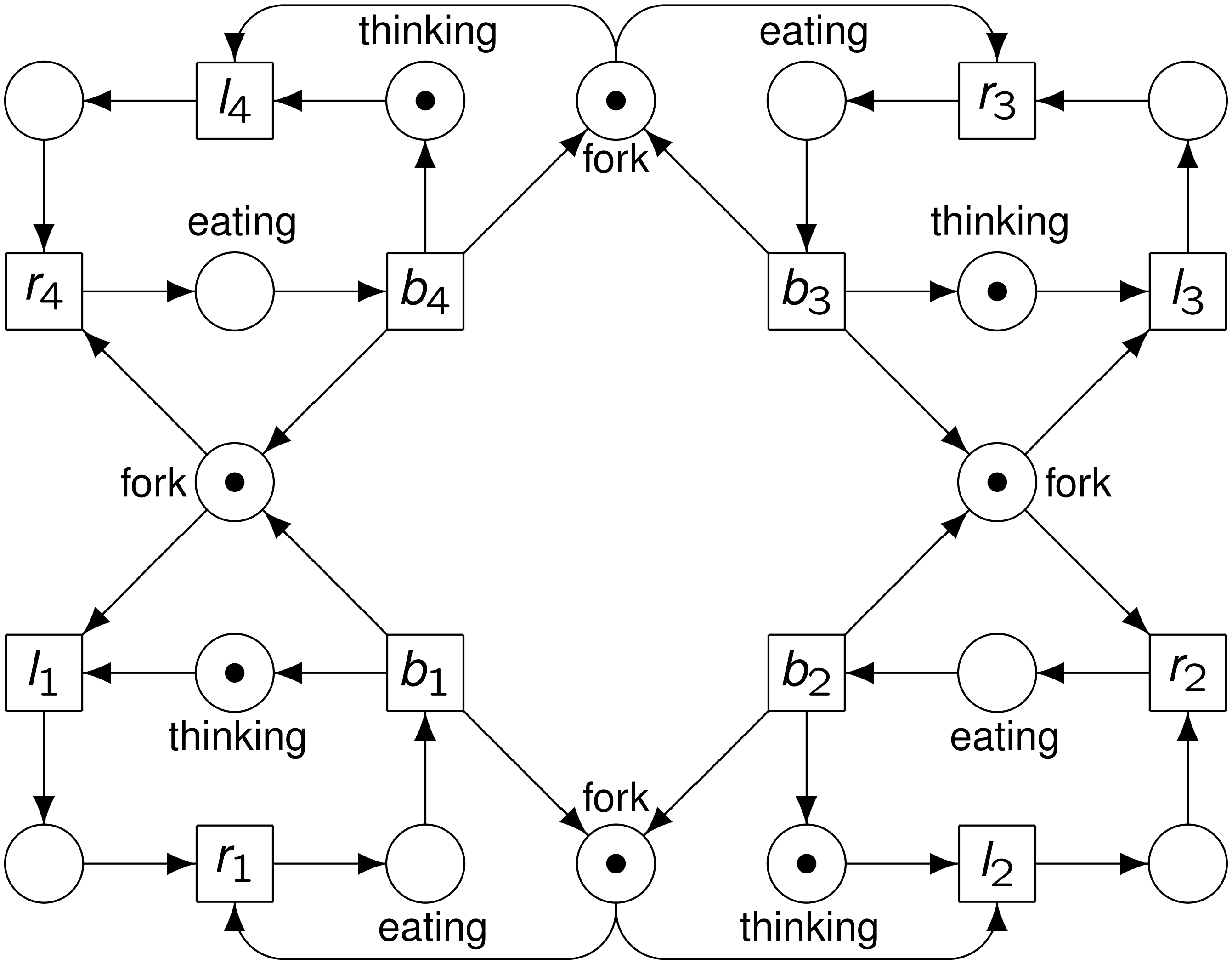
There are philosophers sitting around a round table.

There are forks on the table, one between each pair of philosophers.



The philosophers want to eat spaghetti from a large bowl in the center of the table.

Dining philosophers: Petri net



Synchronization by rendez-vous

Assume that we have a number of components with local actions and actions $!m$ (send message *m*) and $?m$ (receive message *m*).

Transition into Petri net:

Places = union of local states

Transitions:

– for local actions (p, a, p') build a Petri transition t labelled with a and $^\bullet t = \{p\}$, $t^\bullet = \{p'\}$;

– for pairs of actions $(p, !m, p')$ and $(q, ?m, q')$ build a Petri transition t labelled with m and $^\bullet t = \{p, q\}$, $t^\bullet = \{p', q'\}$.

Similar translations possible for other models discussed in the course (asynchronous product, TS with variables, . . .)

Notation for markings

Often we will fix an order on the places (e.g., matching the place numbering), and write, e.g., $m_0 = \langle 2, 5, 0 \rangle$ instead.

When no place contains more than one token, markings are in fact sets, in which case we often use set notation and write instead $m_0 = \{p_5, p_7, p_8\}$.

Alternatively, we could denote a marking as a multiset, e.g. $m_0 = \{p_1, p_1, p_2, p_2, p_2, p_2, p_2\}$.

Reachable markings

Let m be a marking of a Petri net $N = \langle P, T, F, W, m_0 \rangle$.

The set of markings reachable from m (the reachability set of m, written $\mathit{reach}(m)$), is the smallest set of markings such that:

1. $m \in \mathit{reach}(m)$, and
2. if $m' \xrightarrow{t} m''$ for some $t \in T$, $m' \in \mathit{reach}(m)$, then $m'' \in \mathit{reach}(m)$.

The set of reachable markings $\mathit{reach}(N)$ of a net $N = \langle P, T, F, W, m_0 \rangle$ is defined to be $\mathit{reach}(m_0)$.

Reachability Graph

Let $N = \langle P, T, F, W, m_0 \rangle$ be a Petri net with associated transition system $\mathcal{M} = \langle S, \Sigma, \Delta, I, AP, \ell \rangle$.

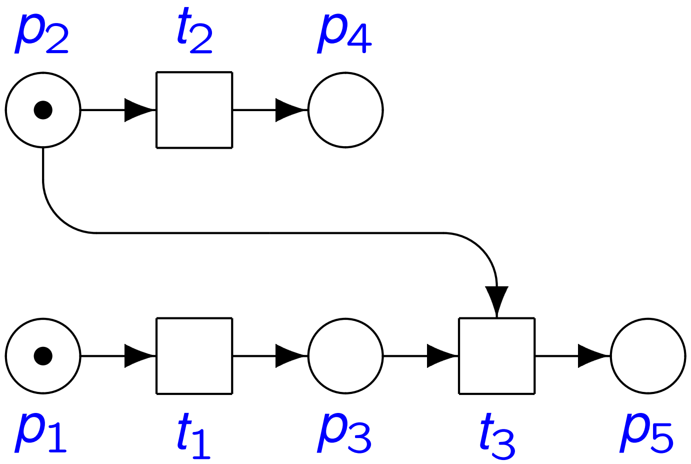
The reachability graph of N is the rooted, directed graph $G = \langle S', \Delta', m_0 \rangle$, where S' and Δ' are the restrictions of S and Δ to $\mathit{reach}(N)$.

The reachability graph can be constructed in iterative fashion, starting with the initial marking and then adding, step for step, all reachable markings.

k-safeness

Definition: Let N be a net. If no reachable marking of N can contain more than k tokens in any place (where $k \geq 0$ is some constant), then N is said to be k-safe.

Example: The following net is 1-safe.



Other example: the nets resulting from translating synchronous rendez-vous

k-safeness and Termination

A k-safe net has at most $(k+1)^{|P|}$ reachable markings; for 1-safe nets, the limit is $2^{|P|}$.

In this case, there are finitely many reachable markings, and the construction of the reachability graph terminates.

On the other hand, if a net is not k-safe for any k, then there are infinitely many markings, and the construction will not terminate.

Reachability problem for 1-safe nets

Let N be a Petri net and m be a marking. The *reachability problem* for N, m is to determine whether $m \in \mathit{reach}(N)$.

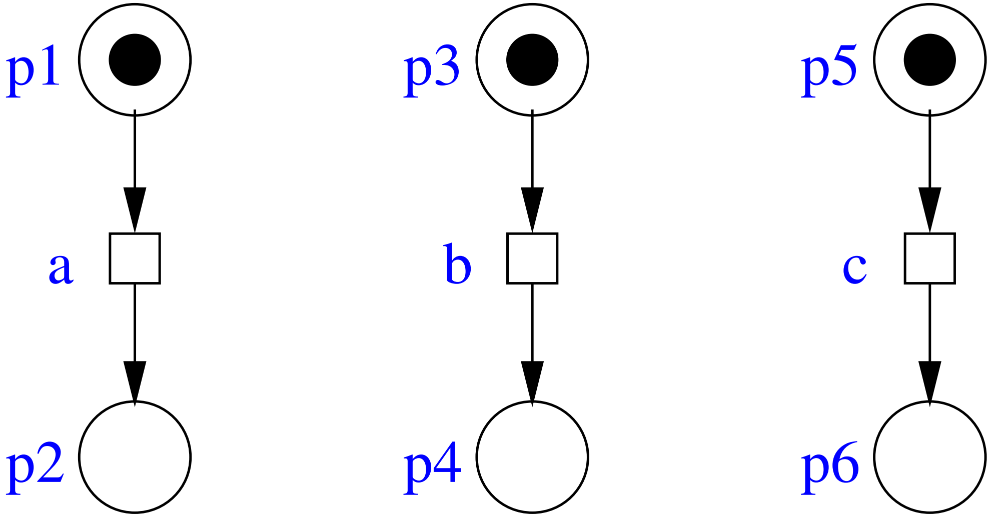
Theorem: The reachability problem for 1-safe Petri nets is PSPACE-complete.

Proof: (sketch)
upper bound: non-deterministically simulate net for at most $2^{|P|}$ steps;
hardness by reduction from QBF.

Corollary: Given a 1-safe net N and a place p, it is PSPACE-complete to determine whether $\mathit{reach}(N)$ contains a marking m such that $m(p) = 1$.

Algorithms for the reachability problem

The most straightforward way to solve the reachability problem on Petri nets is to construct the reachability graph. However, this can be very inefficient:



If there are n such components, then the reachability graph has size 2^n.

In fact, the reachability graph does not take advantage of the concurrent nature of a Petri net.

Later, we shall study a method more adapted to concurrent systems. It constructs a concise representation of the reachable markings called unfolding.

Given the unfolding of N, it is NP-complete to determine whether a given marking is reachable in N. One can thus take advantage of the advances made in SAT-checking.

First, we discuss a method to determine reachability information for non-safe nets.

Unbounded nets: Coverability graphs

Use of reachability graphs

If the net is not k-safe for any k, then it has infinitely many reachable markings, and one cannot effectively compute the reachability graph.

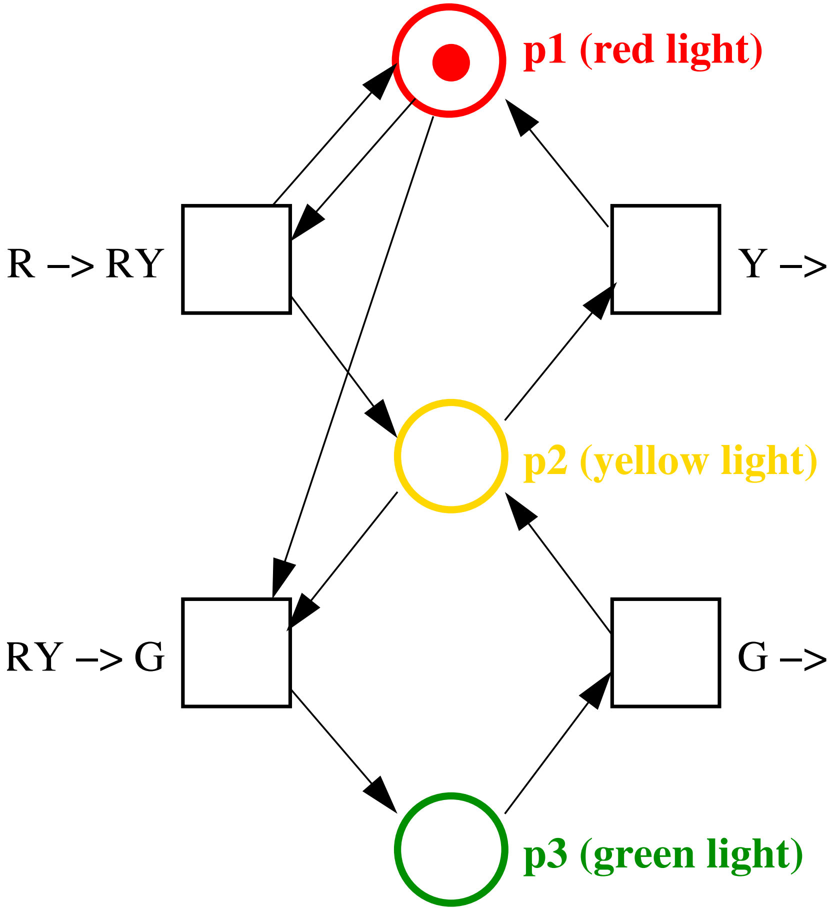
Nevertheless, the following problem is decidable: Given a (non-safe) net $\mathcal{P}$ and a marking m, is m reachable in $\mathcal{P}$?

This result is due to Mayr and Kosaraju (1981/82). However, the complexity of the problem is in general non-elementary, and no efficient methods are known.

Most of the time, though, one is interested in checking whether m is *part of* a reachable marking (one says that m is *coverable* in this case). This problem is somewhat easier to solve in practice, and we shall discuss it.

Example

Consider the following (slightly inept) attempt at modelling a traffic light:



Coverability Graphs

The reachability graph of the preceding net is infinite.

We will show the construction of a so-called coverability graph for it.

The coverability graph has the following properties:

- It can be used to find out whether the reachability graph is infinte.
- It is always finite, and its construction always terminates.
- Even for unbounded nets, it still gathers some information about reachable markings.

Computing with ω

First we introduce a new symbol ω to represent “arbitrarily many” tokens.

We extend the arithmetic on natural numbers with ω as follows. For all $n \in \mathbb{N}$:
$n + \omega = \omega + n = \omega$,
$\omega + \omega = \omega$,
$\omega - n = \omega$,
$0 \cdot \omega = 0$, $\omega \cdot \omega = \omega$,
$n \geq 1 \Rightarrow n \cdot \omega = \omega \cdot n = \omega$,
$n \leq \omega$, and $\omega \leq \omega$.

Note: $\omega - \omega$ remains undefined, but we will not need it.

ω-Markings

We extend the notion of markings to ω-markings. In an ω-marking, each place p will either have $n \in \mathbb{N}$ tokens, or ω tokens (arbitrarily many).

Note: This is a technical definition that we will need for constructing the coverability graph! The nets that we use only have *finite* markings.

An ω-marking such as $(1, \omega, 0)$ can also be interpreted as the set of (non-ω)-markings that have one token on the first place, no token on the third place, and any number of tokens on the second place.

Firing Rule with ω-markings

The firing condition and firing rule (reproduced below) neatly extend to ω-markings with the extended arithmetic rules:

> Firing condition:
>
> Transition $t \in T$ is M-enabled, written $M \xrightarrow{t}$, iff $\forall p \in {}^\bullet t : M(p) \geq W(p,t)$.
>
> Firing rule:
> An M-enabled transition t may fire, producing the successor marking M', where
>
> $$\forall p \in P : M'(p) = M(p) - W(p,t) + W(t,p).$$

If a transition has a place with ω tokens in its preset, that place is considered to have sufficiently many tokens for the transition to fire, regardless of the arc weight.

If a place contains an ω-marking, then firing any transition connected with an arc to that place will not change its marking.

Definition of Covering

An ω-marking M' covers an ω-marking M, denoted $M \leq M'$, iff

$$\forall p \in P\colon\ M(p) \leq M'(p).$$

An ω-marking M' strictly covers an ω-marking M, denoted $M < M'$, iff

$$M \leq M' \quad \text{and} \quad M' \neq M.$$

Coverability and Transition Sequences (1/2)

Observation: Let M and M' be two markings such that $M \leq M'$.
Then for all transitions t, the following holds:

$$\text{If } M \xrightarrow{t} \text{ then } M' \xrightarrow{t}.$$

In other words, if M' has at least as many tokens as M has (on each place), then M' enables at least the same transitions as M does.

This observation can be extended to *sequences* of transitions:
Define $M \xrightarrow{t_1 t_2 \ldots t_n} M'$ to denote:

$$\exists M_1, M_2, \ldots, M_n : M \xrightarrow{t_1} M_1 \xrightarrow{t_2} M_2 \cdots \xrightarrow{t_n} M_n = M'.$$

Now, if $M \xrightarrow{t_1 t_2 \ldots t_n}$ and $M \leq M'$, then $M' \xrightarrow{t_1 t_2 \ldots t_n}$.

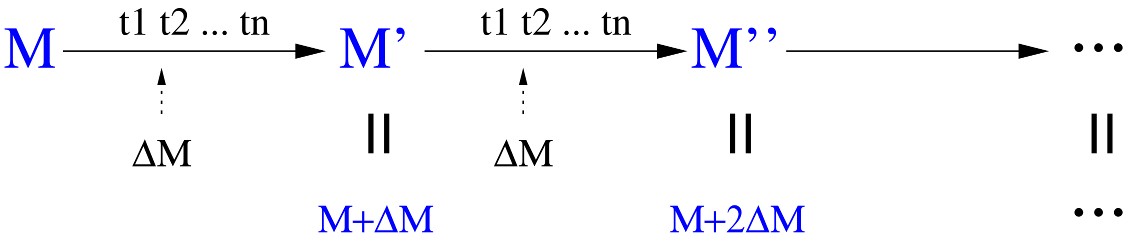
Coverability and Transition Sequences (2/2)

Let M, M' be markings such that $M < M'$, and assume that there is a sequence of transitions such that $M \xrightarrow{t_1 t_2 \ldots t_n} M'$ holds.

Thus, there is a marking M'' with $M' \xrightarrow{t_1 t_2 \ldots t_n} M''$.

Let $\Delta M := M' - M$ (place-wise difference). Because $M < M'$, the values of ΔM are non-negative and at least one value is non-zero.

Clearly, $M'' = M' + \Delta M = M + 2\Delta M$.



By firing the transition sequence $t_1 t_2 \ldots t_n$ repeatedly we can “pump” an arbitrary number of tokens to all the places having a non-zero marking in $\triangle M$.

The basic idea for constructing the coverability graph is now to replace the marking M' with a marking where all the places with non-zero tokens in $\triangle M$ are replaced by ω.

Coverability Graph Algorithm (1/2)

```
COVERABILITY-GRAPH($\langle P, T, F, W, M_0 \rangle$)
  $\langle V, E, v_0 \rangle := \langle \{M_0\}, \emptyset, M_0 \rangle$;
  $\mathit{Work}$ : set $:= \{M_0\}$;
  while $\mathit{Work} \neq \emptyset$
  do select $M$ from $\mathit{Work}$;
      $\mathit{Work} := \mathit{Work} \setminus \{M\}$;
      for $t \in$ enabled($M$)
      do $M' :=$ fire($M, t$);
          $M' :=$ AddOmegas($M, M', V$);
          if $M' \notin V$
            then $V := V \cup \{M'\}$
                 $\mathit{Work} := \mathit{Work} \cup \{M'\}$;
          $E := E \cup \{\langle M, t, M' \rangle\}$;
  return $\langle V, E, v_0 \rangle$;
```

The subroutine AddOmegas(M, M', V) will check if the sequences leading to M' can be repeated, strictly increasing the number of tokens on some places, and replace their values with ω.

Coverability Graph Algorithm (2/2)

The following notation us used in the AddOmegas subroutine:

- $M'' \rightarrow^* M$ iff the coverability graph currently contains a path (including the empty path!) leading from M'' to M.

```
AddOmegas($M, M', V$)
1  repeat $saved := M'$;
2          for all $M'' \in V$ s.t. $M'' \rightarrow^* M$
3          do if $M'' < M'$
4                then $M' := M' + ((M' - M'') \cdot \omega)$;
5  until $saved = M'$;
6  return $M'$;
```

In other words, repeated check all the predecessor markings of the new marking M' to see if they are strictly covered by M'. Line 5 causes all places whose number of tokens in M' is strictly larger than in the "parent" M'' to contain ω.

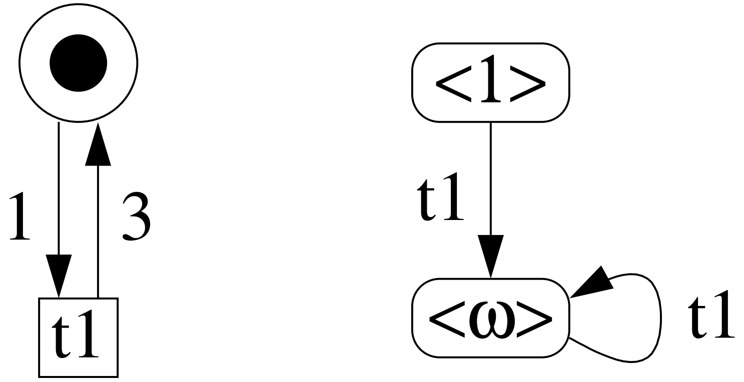
Properties of the coverability graph (1)

Let $N = \langle P, T, F, W, M_0 \rangle$ be a net.

The coverability graph has the following fundamental property:

> If a marking M of N is reachable, then M is covered by some vertex of the coverability graph of N.

Note that the reverse implication *does not* hold: A marking that is covered by some vertex of the coverability graph is not necessarily reachable, as shown by the following example:



Properties of the coverability graph (2)

The coverability graph could thus be said to compute an overapproximation of the reachable markings.

The construction of the coverability graph always terminates.
If N is bounded, then the coverabilibility graph is identical to the reachability graph.

Coverability graphs are not unique,
i.e. for a given net there may be more than one coverability graph, depending on the order of the worklist and the order in which firing transitions are considered.

Petri nets: Structural analysis

Structural Analysis: Motivation

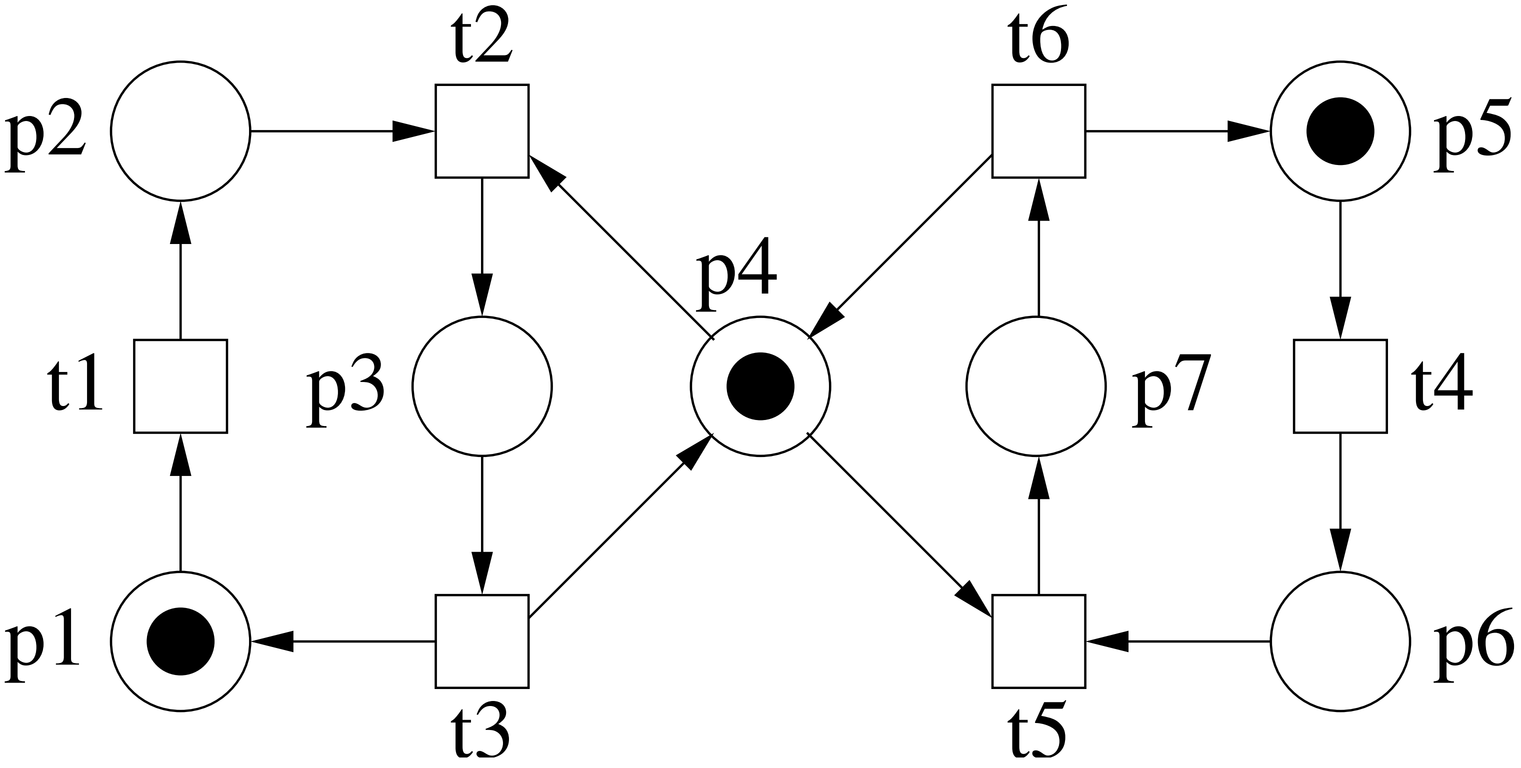
We shall consider another class of techniques that can extract information about the behaviour of the system by analyzing it locally (i.e., without first constructing an object that represents the entire behaviour of the net).

This class of techniques is called structural analysis. Some its components are:

Place invariants

Traps

Example 1



Incidence Matrix

Let $N = \langle P, T, F, W, M_0 \rangle$ be a P/T net. The corresponding incidence matrix $C\colon P \times T \to \mathbb{Z}$ is the matrix whose rows correspond to places and whose columns correspond to transitions. Column $t \in T$ denotes how the firing of t affects the marking of the net: $C(t, p) = W(t, p) - W(p, t)$.

The incidence matrix of Example 1:

$$\begin{array}{cccccc|c} t_1 & t_2 & t_3 & t_4 & t_5 & t_6 & \\ -1 & 0 & 1 & 0 & 0 & 0 & p_1 \\ 1 & -1 & 0 & 0 & 0 & 0 & p_2 \\ 0 & 1 & -1 & 0 & 0 & 0 & p_3 \\ 0 & -1 & 1 & 0 & -1 & 1 & p_4 \\ 0 & 0 & 0 & -1 & 0 & 1 & p_5 \\ 0 & 0 & 0 & 1 & -1 & 0 & p_6 \\ 0 & 0 & 0 & 0 & 1 & -1 & p_7 \end{array}$$

Markings as vectors

Let us now write markings as column vectors. E.g., the initial marking in Example 1 is $M_0 = (1\ 0\ 0\ 1\ 1\ 0\ 0)^T$.

Likewise, we can write firing counts as column vectors with one entry for each transition. E.g., if each of the transitions t_1, t_2, and t_4 fires once, we can express this with $u = (1\ 1\ 0\ 1\ 0\ 0)^T$.

Then, the result of firing these transitions can be computed as $M_0 + C \cdot u$.

$$\begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} -1 & 0 & 1 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & -1 & 1 \\ 0 & 0 & 0 & -1 & 0 & 1 \\ 0 & 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

Let N be a P/T net with indicence matrix C, and let M, M' be two markings of N. The following implication holds:

> If $M' \in reach(M)$, then there exists a vector u such that $M' = M + C \cdot u$ such that all entries in u are natural numbers.

Let N be a P/T net with indicence matrix C, and let M, M' be two markings of N. The following implication holds:

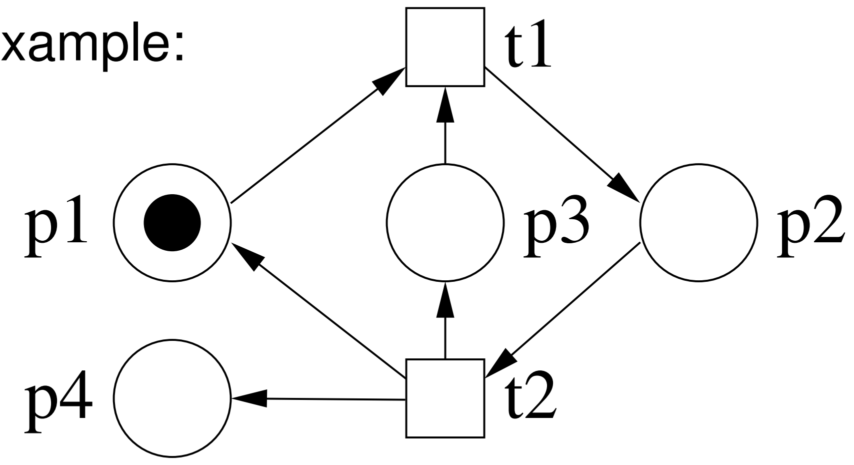
> If $M' \in reach(M)$, then there exists a vector u such that $M' = M + C \cdot u$ such that all entries in u are natural numbers.

Notice that the reverse implication does not hold in general!

E.g., bi-directional arcs (an arc from a place to a transition and back) cancel each other out in the matrix. For instance, if Example 1 contained a bi-directional arc between p_1 and t_3, the matrix would remain the same, but the marking $\{p_3, p_6\}$ (obtained on the previous slide) would be unreachable!

Example 2

A more complicated example:



Even though we have

$$\begin{pmatrix}1\\0\\0\\0\end{pmatrix} + \begin{pmatrix}-1 & 1\\1 & -1\\-1 & 1\\0 & 1\end{pmatrix} \cdot \begin{pmatrix}1\\1\end{pmatrix} = \begin{pmatrix}1\\0\\0\\1\end{pmatrix},$$

none of the sequences corresponding to $(1\ 1)^T$, i.e. $t_1 t_2$ or $t_2 t_1$, can happen.

Proving unreachability using the incidence matrix

To summarize: The markings obtained by computing with the incidence matrix are an over-approximation of the actual reachable markings

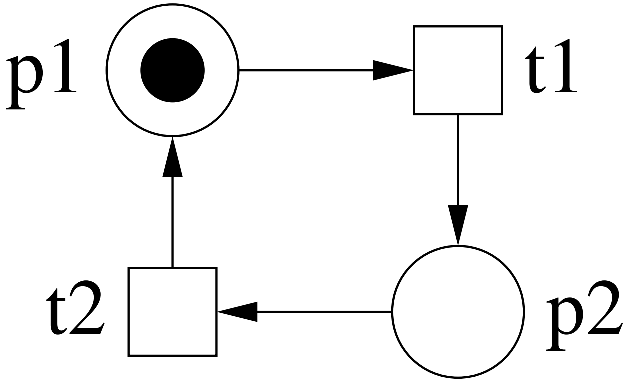
However, we *can* sometimes use the matrix equations to show that a marking M is *unreachable*. (Compare coverability graphs. . .)

I.e., a corollary of the previous implication is that if $M' = M + Cu$ has no natural solution for u, then $M' \notin reach(M)$.

Note: When we are talking about natural (integral) solutions of equations, we mean those whose components are natural (integral) numbers.

Example 3

Consider the following net and the marking $M = (1\ 1)^T$.



$$\begin{pmatrix}1\\0\end{pmatrix} + \begin{pmatrix}-1 & 1\\1 & -1\end{pmatrix} \cdot \begin{pmatrix}u_1\\u_2\end{pmatrix} = \begin{pmatrix}1\\1\end{pmatrix}$$

has no solution, and therefore M is not reachable.

Place invariants

Let N be a net and C its incidence matrix. A natural solution of the equation $C^T x = \mathbf{0}$ such that $x \neq \mathbf{0}$ is called a place invariant (or: P-invariant) of N.

Notice that a P-invariant is a vector with one entry for each place.

For instance, in Example 1, $x_1 = (1\ 1\ 1\ 0\ 0\ 0\ 0)^T$, $x_2 = (0\ 0\ 1\ 1\ 0\ 0\ 1)^T$, and $x_3 = (0\ 0\ 0\ 0\ 1\ 1\ 1)^T$ are all P-invariants.

A P-invariant indicates that the number of tokens in all reachable markings satisfies some linear invariant (see next slide).

Properties of P-invariants

Let M be marking reachable with a transition sequence whose firing count is expressed by u, i.e. $M = M_0 + Cu$. Let x be a P-invariant. Then, the following holds:

$$M^T x = (M_0 + Cu)^T x = M_0^T x + (Cu)^T x = M_0^T x + u^T C^T x = M_0^T x$$

For instance, invariant x_2 means that all reachable markings M satisfy (switching to the functional notation for markings):

$$M(p_3) + M(p_4) + M(p_7) = M_0(p_3) + M_0(p_4) + M_0(p_7) = 1 \qquad (1)$$

As a special case, a P-invariant in which all entries are either 0 or 1 indicates a set of places in which the number of tokens remains unchanged in all reachable markings.

Note that linear combinations of P-invariants (i.e. multiplying an invariant by a constant or component-wise addition of two invariants) will again yield a P-invariant.

We can use P-invariants to prove mutual exclusion properties.

Example: According to equation 1, in every reachable marking of Example 1 exactly one of the places p_3, p_4, and p_7 is marked. In particular, p_3 and p_7 cannot be marked concurrently!

Traps

Let $\langle P, T, F, W, M_0 \rangle$ be a P/T net. A trap is a set of places $S \subseteq P$ such that $S^\bullet \subseteq {}^\bullet S$.

In other words, each transition which removes tokens from a trap must also put at least one token back to the trap.

A trap S is called marked in marking M iff for at least one place $s \in S$ it holds that $M(s) \geq 1$.

Note: If a trap S is marked initially (i.e. in M_0), then it is also marked in all reachable markings.

In Example 4 (see next slide), $S_1 = \{nc_1, nc_2\}$ is a trap.

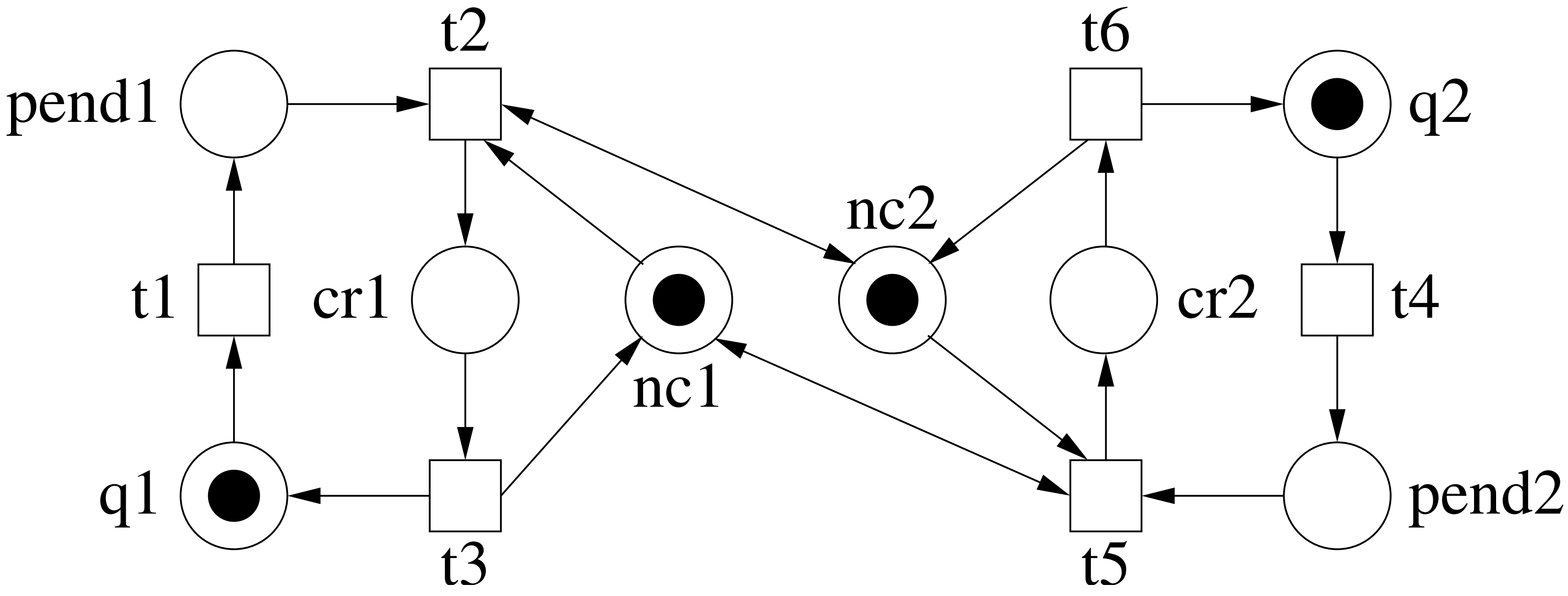
The only transitions that remove tokens from this set are t_2 and t_5. However, both also add new tokens to S_1.

S_1 is marked initially, and therefore in all reachable markings M the following inequality holds: $M(nc_1) + M(nc_2) \geq 1$

Traps can be useful in combination with place invariants to recapture information lost in the incidence matrix due to the cancellation of self-loop arcs.

Example 4

Consider the following attempt at a mutual exlusion algorithm for cr_1 and cr_2:



The idea is to achieve mutual exclusion by entering the critical section only if the other process is not already there.

Proving mutual exclusion properties using traps

In Example 4, we want to prove that in all reachable markings M, cr_1 and cr_2 cannot be marked at the same time. This can be expressed by the following inequality:

$$M(cr_1) + M(cr_2) \leq 1$$

The P-invariants we can derive in the net yield these equalities:

$$\begin{aligned} M(q_1) + M(pend_1) + M(cr_1) &= 1 && (2) \\ M(q_2) + M(pend_2) + M(cr_2) &= 1 && (3) \\ M(cr_1) + M(nc_1) &= 1 && (4) \\ M(cr_2) + M(nc_2) &= 1 && (5) \end{aligned}$$

However, these equalities are insufficient to prove the desired property!

Recall that $S_1 = \{nc_1, nc_2\}$ is a trap.

S_1 is marked initially and therefore in all reachable markings M. Thus:

$$M(nc_1) + M(nc_2) \geq 1 \tag{6}$$

Now, adding (4) and (5) and subtracting (6) yields $M(cr_1) + M(cr_2) \leq 1$, which proves the mutual exclusion property.

Unfoldings

Unfoldings

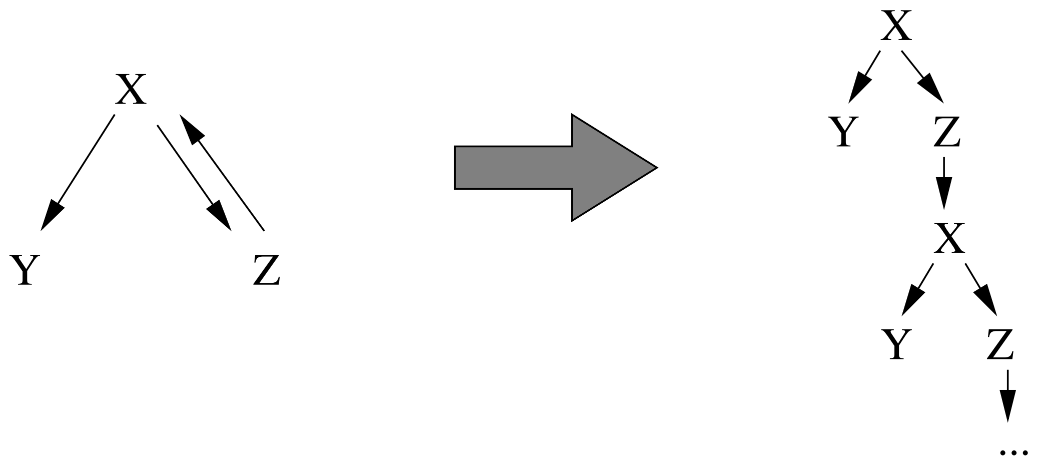
Unfoldings are a data structure that represents the behaviour of a Petri net.

We will study it for 1-safe nets.

Unfoldings represent a trade-off in terms of time/space requirements; their size is in between that of a net and its reachability graph, and checking whether a marking is reachable becomes easier than for the net, but more difficult than from the reachability graph.

Unfoldings for finite transition systems

Let $\mathcal{T}$ be a finite transition system with initial state X. One can define the acyclic unfolding $\mathcal{U}_{\mathcal{T}}$ (which is used for CTL model checking):

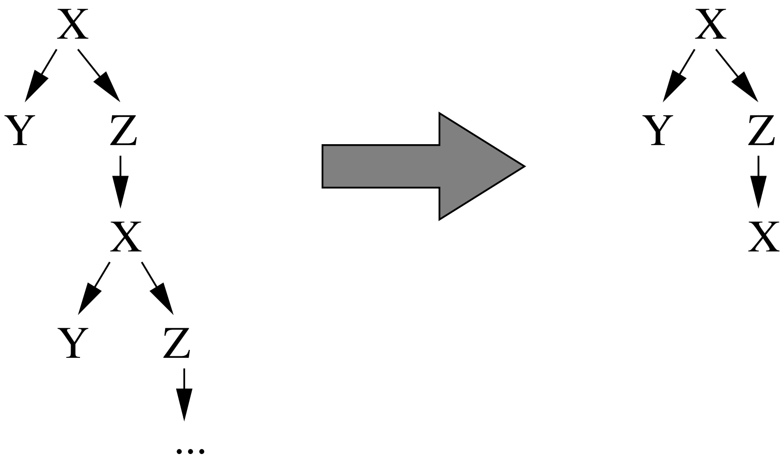


Remark: $\mathcal{U}_{\mathcal{T}}$ can be viewed as a structure in which every state is labelled by a state from $\mathcal{T}$. We denote this labelling by the function B.

$\mathcal{U}_{\mathcal{T}}$ contains the same behaviours as $\mathcal{T}$ (and the same reachable states). Additionally, $\mathcal{U}_{\mathcal{T}}$ has a simpler structure (acyclic, in fact, a tree). However, in general, $\mathcal{U}_{\mathcal{T}}$ is *infinite*.

Prefixes

$\mathcal{P}$ is called a prefix of $\mathcal{U}_\mathcal{T}$ if $\mathcal{P}$ is obtained by "pruning" arbitrary branches of $\mathcal{U}_\mathcal{T}$.
Example:



Observation: One can always find a *finite* prefix containing the same reachable states as the infinite unfolding (by unrolling loops exactly once). We shall call such a prefix complete.

Construction of complete prefixes

Let us discuss an algorithm to obtain a complete prefix of $\mathcal{U}_{\mathcal{T}}$.

The algorithm maintains a set $\mathcal{E}$, the set of states observed so far.

Some arcs in the prefix will be called cutoffs, we shall mark them red.

1. Initially, the prefix contains only the root, labelled by X. We set $\mathcal{E} := \{X\}$.

2. Select a node n on the prefix that is not the target of a cutoff edge. Let $B(n) = Y$ be the label of the node, and let Z be a state with $Y \to Z$ such that the prefix does not contain any edge from n to a Z-labelled node.

2a. If no such pair n, Z exists, we are done.

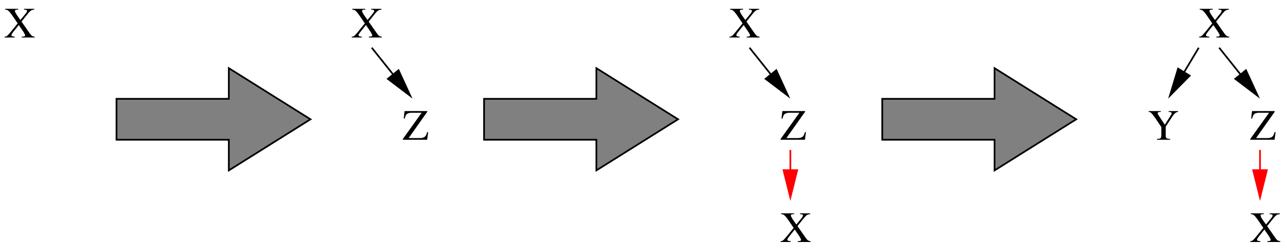
2b. Otherwise, add a new, Z-labelled node to the prefix and add an edge from n to it.

2c. If $Z \in \mathcal{E}$, then the new edge is a cutoff. Otherwise, set $\mathcal{E} := \mathcal{E} \cup \{Z\}$.

3. Continue at step 2.

Example

Step-by-step construction of the prefix in the previous example:



Observation (1): A complete prefix contains as many transitions as $\mathcal{T}$.

Observation (2): The shape of the prefix depends on the *order* in which edges are added!

The unfolding of a Petri net $\mathcal{P}$ (or, a prefix of the same) is an infinite *acyclic* Petri net $\mathcal{U}$. We shall be interested in computing a finite prefix $\mathcal{Q}$ of $\mathcal{U}$.

Remark: In the following, we call the places of $\mathcal{Q}$ conditions, the transitions of $\mathcal{Q}$ events. This merely serves to better distinguish the elements of $\mathcal{P}$ and $\mathcal{Q}$, functionally they are the same!

Every condition of $\mathcal{Q}$ is labelled by a place of $\mathcal{P}$,
every event of $\mathcal{Q}$ by a transition of $\mathcal{P}$.

Every event e is of the form (S, t), where S is the preset of e and t the label of e.

Let S be a set of conditions. $B(S)$ denotes the set of places labelling the elements of S.

Every condition has exactly one incoming arc.

Unfolding construction for Petri nets

We first discuss the construction of $\mathcal{U}$ (possibly infinite).

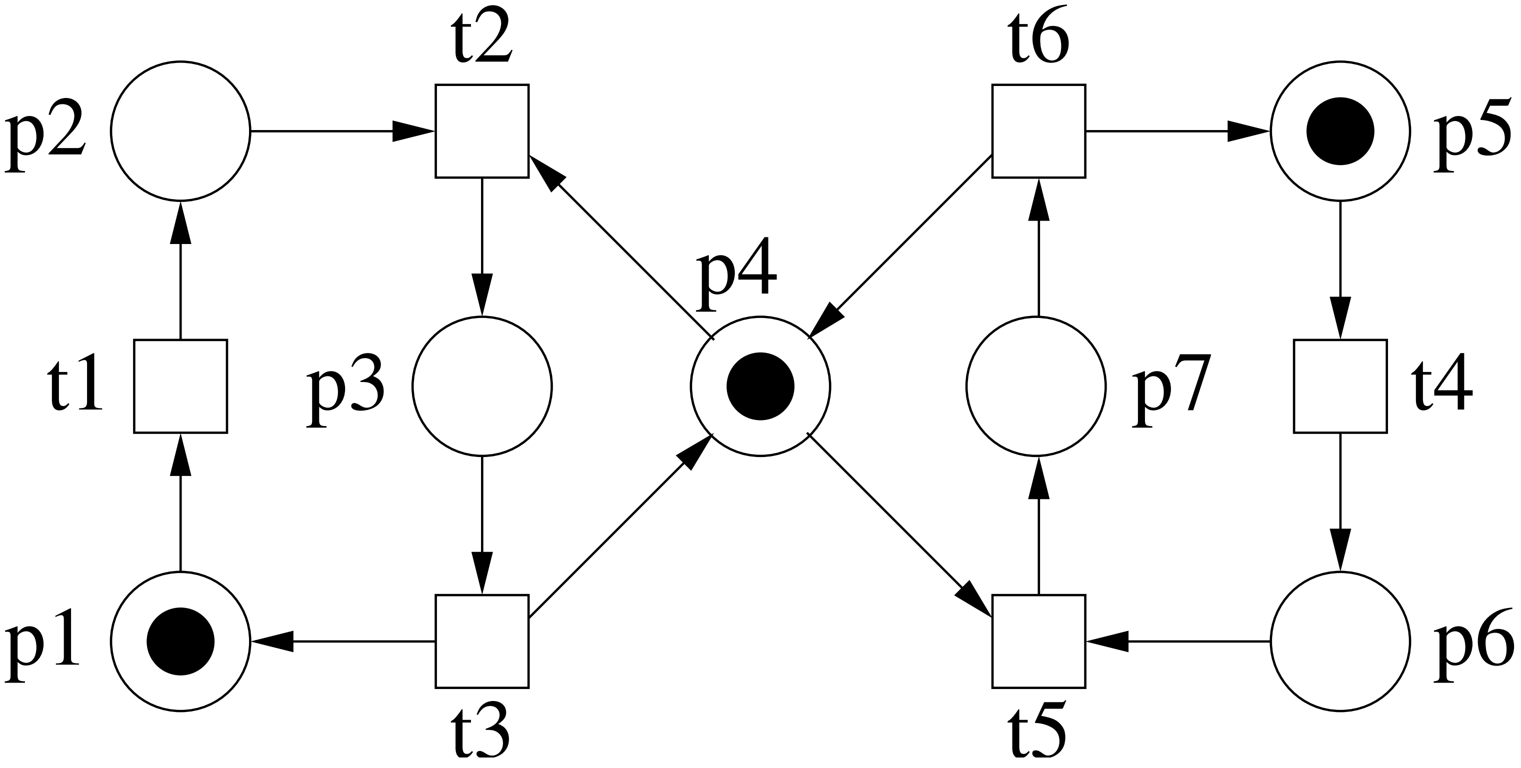
1. Let m_0 be the initial marking of $\mathcal{P}$. Then the initial marking of $\mathcal{U}$ contains exactly one condition for each place in m_0.

2. Let S the subset of a reachable marking in $\mathcal{U}$. Let $B(S) = {}^\bullet t$ for some transition t of $\mathcal{P}$ such that (S, t) is not yet contained in $\mathcal{U}$.

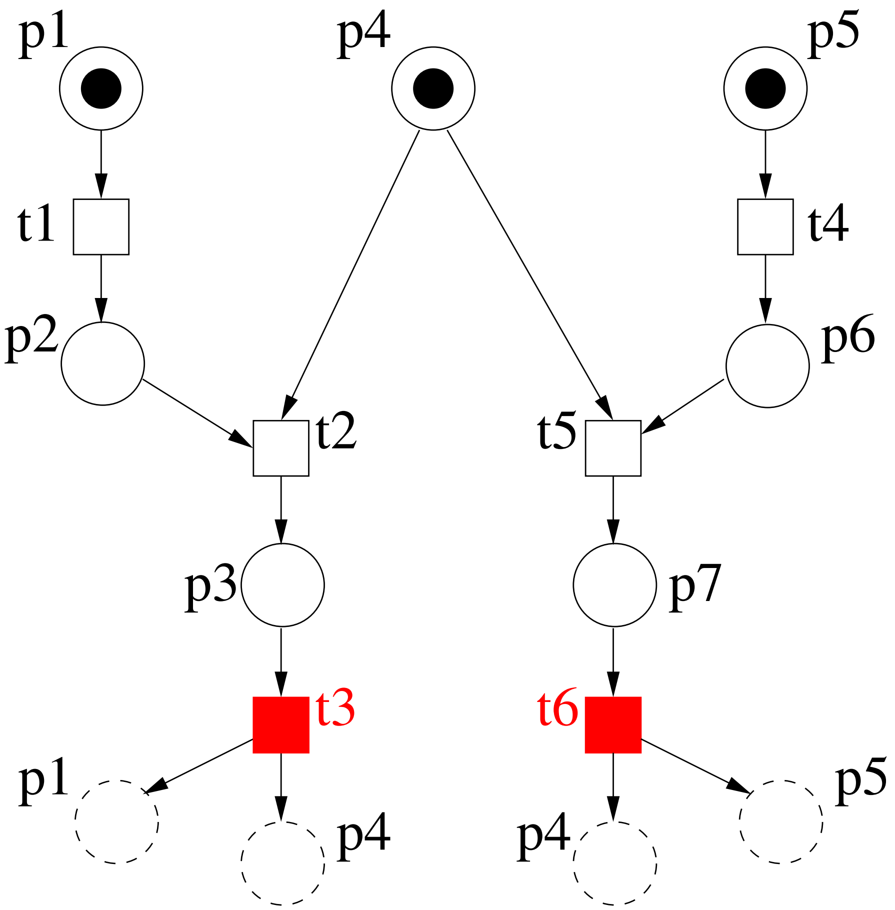
2a. If no such pair (S, t) exists, we are done.

2b. Add the event $e := (S, t)$ to the prefix (with S as preset and label t). Moreover, extend the prefix by one condition for every output place of t and make it an output place of e.

Example 1: Petri net. . .



. . . and a possible prefix of the unfolding



We shall now amend the construction so that it prunes certain branches of the unfolding, creating a *finite* prefix.

More precisely, certain events will be called cutoffs. These events lead to markings that we have already seen.

Prefix construction for Petri nets

1. Let m_0 be the initial marking of $\mathcal{P}$. Then the initial marking of $\mathcal{Q}$ contains exactly one condition for each place in m_0. We set $\mathcal{E} := \{m_0\}$.

2. Let S the subset of a reachable marking in $\mathcal{Q}$. Suppose that no element of S is the output place of a cutoff event. Let $B(S) = {}^{\bullet}t$ for some transition t of $\mathcal{P}$ such that (S, t) is not yet contained in $\mathcal{Q}$.

2a. If no such pair (S, t) exists, we are done.

2b. Add the event $e := (S, t)$ to the prefix (with S as preset and label t). Moreover, extend the prefix by one condition for every output place of t and make it an output place of e.

2c. We associate with e a marking m_e (which is reachable in $\mathcal{P}$) (see below). If $m_e \in \mathcal{E}$, then e is a cutoff. Otherwise $\mathcal{E} := \mathcal{E} \cup \{m_e\}$.

Determining m_e

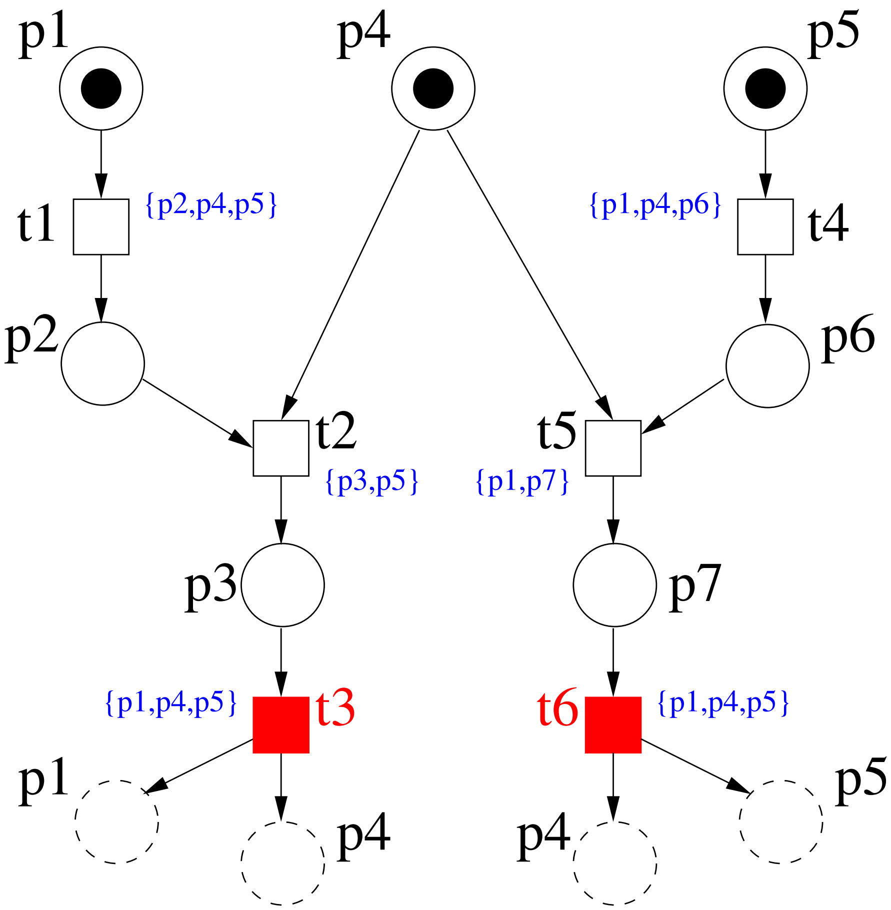
For the event $e = (S, t)$, we determine m_e, a marking of $\mathcal{P}$, as follows:

Idea: m_e is the label of the marking obtained by making the “minimal” effort to fire e.

Let x, y be two nodes (conditions *or* events) in $\mathcal{Q}$. Let $<$ be the smallest partial order where $x < y$ if there is an edge from x to y.

Let x be a node of $\mathcal{Q}$. We define $\lfloor x \rfloor := \{\, y \mid y \leq x \,\}$.

Let m_e be the labels of the marking obtained by firing the events of $\lfloor e \rfloor$ (in any order). Note: Such a firing sequence exists due to the properties of S.


p1
p4
p5
t1
{p2,p4,p5}
{p1,p4,p6}
t4
p2
p6
t2
t5
{p3,p5}
{p1,p7}
p3
p7
{p1,p4,p5}
t3
t6
{p1,p4,p5}
p1
p4
p4
p5

Complete prefixes

Let $\mathcal{P}$ be a Petri net and $\mathcal{Q}$ a prefix of its unfolding $\mathcal{U}$ with labelling B. We call $\mathcal{Q}$ complete if it satisfies the following property:

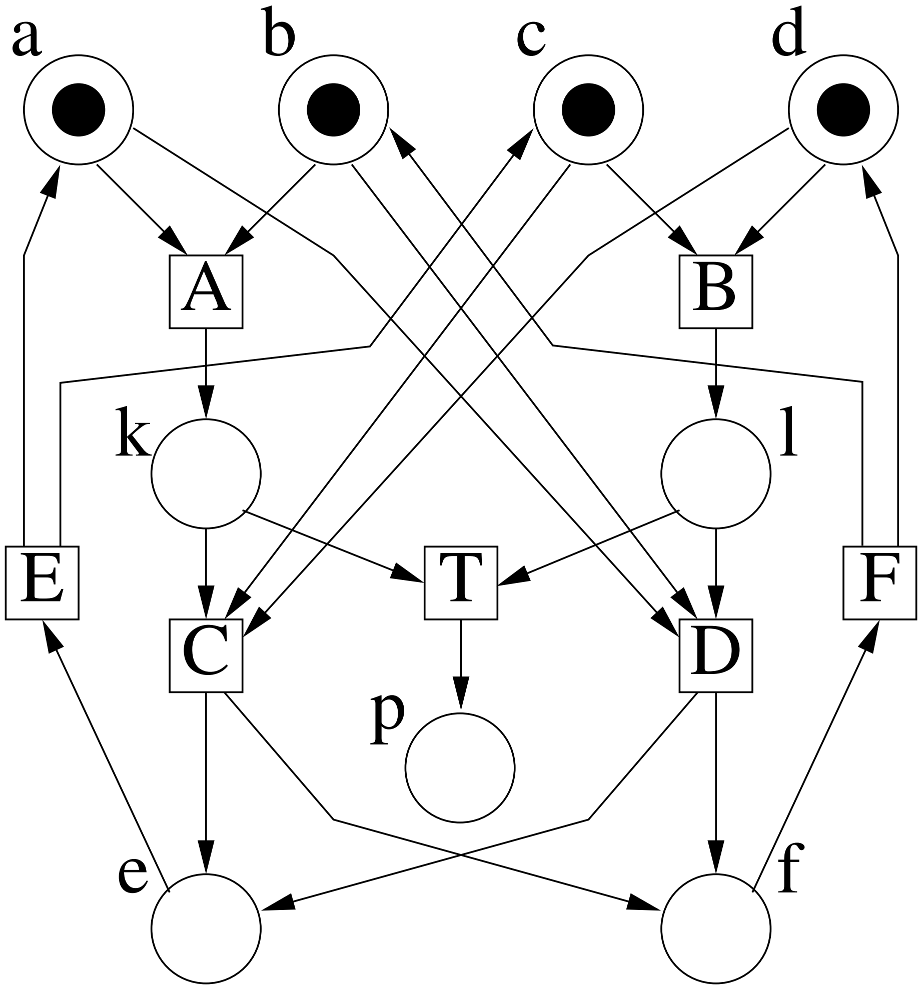
> A marking m is reachable in $\mathcal{P}$ *iff* a marking m' with $B(m') = m$ is reachable in $\mathcal{Q}$.

Thus, if $\mathcal{Q}$ is complete, we can decide reachability in $\mathcal{P}$ by examining $\mathcal{Q}$.

Unfortunately, the algorithm given previously does not always produce a complete prefix. Indeed, its shape depends on the order in which events are added. We shall discuss an example that demonstrates this effect.

Example 2

Consider the following Petri net:

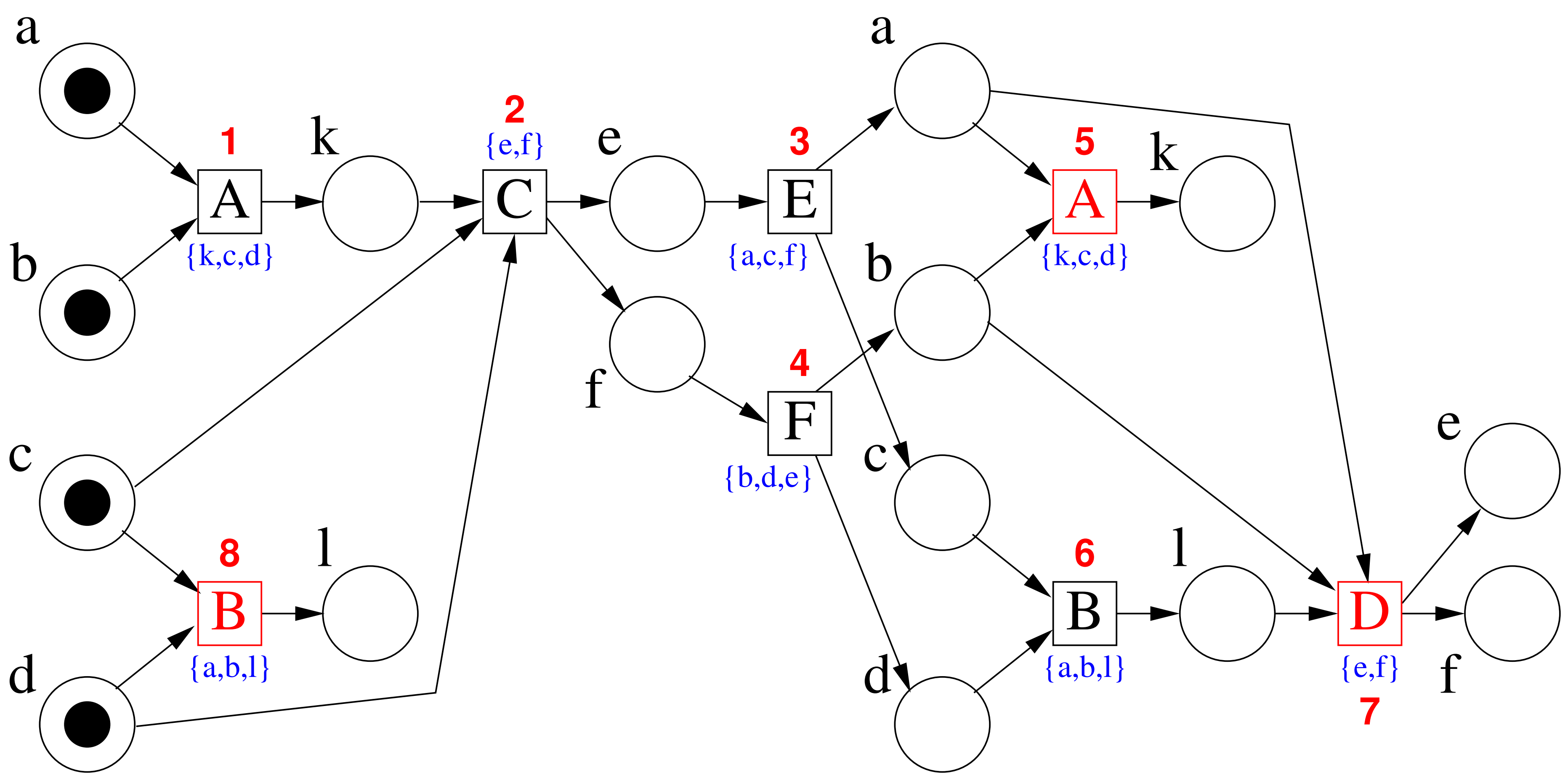


In Example 2 the marking $\{p\}$ is reachable, e.g. by firing $A\,B\,T$.

The net can also reach the marking $\{e, f\}$ by firing either $A\,C$ or $B\,D$, and then return by firing $E\,F$ to the initial marking.

We shall see that a prefix generated according to depth-first order will “overlook” the transition T.

Depth-first order generated the prefix shown below (order and cutoffs indicated in red):



Configurations

Let m be a reachable marking in $\mathcal{U}$ and let C be the set of events in $\bigcup_{c \in m} \lfloor c \rfloor$. Then we call C a configuration.

A configuration C represents a set of events that can all fire in one execution. Given C, we denote the marking (of $\mathcal{U}$) reached by such an execution by m_C.

Remark: For every event e, the set $\lfloor e \rfloor =: C_e$ is a configuration. We have $B(m_{C_e}) = m_e$.

We call E an extension of C iff $C \cap E = \emptyset$ and $C \cup E$ is a configuration. In this case, we write $C \oplus E$ to denote the configuration $C \cup E$.

Let C, C' be two configurations such that $B(m_C) = B(m_{C'})$. If E is an extension of C, then there is an extension E' of C' that is isomorphic to E.

Adequate orders

Let $\prec$ be a well-founded total order on configurations that refines $\subset$ (i.e. $C \subset C'$ implies $C \prec C'$).

Intuition: $\prec$ is a possible order in which the events of $\mathcal{U}$ can be generated; i.e., e is added before e' if $C_e \prec C_{e'}$.

Let $\mathcal{Q}_\prec$ be the prefix of $\mathcal{U}$ generated by adding events in the order given by $\prec$ as above.

We call $\prec$ adequate iff $\mathcal{Q}_\prec$ is complete.

A sufficient condition for adequate orders

The following condition guarantees that $\prec$ is adequate:

> Let C, C' be two configurations with $C \prec C'$ and $B(m_C) = B(m_{C'})$, and let E an extension of C and E' the extension of C' isomorphic to E. Then $D \prec D'$ must hold, where $D = C \oplus E$ and $D' = C' \oplus E'$.

Proof: Let $\prec$ be an order satisfying the above constraint. We show that $\mathcal{Q}_\prec$ is complete. So let m be a marking reachable in $\mathcal{P}$. Then there is a marking m' in $\mathcal{U}$ with $B(m') = m$. Let C' be the configuration containing the events in $\bigcup_{c \in m'} \lfloor c \rfloor$. Either C' is contained in $\mathcal{Q}_\prec$, or $C' = C_{e'} \oplus E'$ for some cutoff event e'. But then there is another event e with $m_e = m_{e'}$ and $C_e \prec C_{e'}$ and therefore a configuration $C := C_e \oplus E$, where E is isomorphic to E', and we have $B(m_C) = B(m_{C'}) = m$. Now, since $C_e \prec C_{e'}$ we have $C \prec C'$. Either C is contained in $\mathcal{Q}_\prec$, or one repeats the argument, but only finitely often since $\prec$ is well-founded.

Conflict, causality, concurrency

From the structure of the unfolding we can derive statements about the mutual relationships of conditions:

Let c, d be two (different) conditions of $\mathcal{Q}$.

c, d are called causally dependent if $c < d$ or $d < c$. (I.e., in every firing sequence containing both conditions, one condition must be consumed to generate the other.)

c, d are in conflict if there are events e, f (where $e \neq f$), $e \in \lfloor c \rfloor$, $f \in \lfloor d \rfloor$, and ${}^\bullet e \cap {}^\bullet f \neq \emptyset$. (I.e., c, d can *never* occur in a reachable marking of $\mathcal{Q}$!)

c, d are called concurrent if they are neither causally dependent nor in conflict with one another

Concurrent conditions are jointly reachable

Let $\mathcal{C}$ be a set of conditions. Then $\mathcal{C}$ is a subset of a reachable marking in $\mathcal{U}$ iff all conditions in $\mathcal{C}$ are mutually concurrent.

Proof (“$\Rightarrow$”): Obvious.

Proof (“$\Leftarrow$”): (sketch) Let $\mathcal{E}$ be the set of events in $\bigcup_{c \in \mathcal{C}} \lfloor c \rfloor$. Induction on the size of $\mathcal{E}$: obvious for $E = \emptyset$, otherwise remove a maximal event e from $\mathcal{E}$ and prove that $(\mathcal{C} \setminus e^\bullet) \cup {}^\bullet e$ is mutually concurrent.

Reachability checking using complete prefixes

Theorem: Let $\mathcal{P}$ be a Petri net and $\mathcal{Q}$ a complete unfolding prefix. Given $\mathcal{Q}$ and a marking m of $\mathcal{P}$, it is NP-complete to determine whether m is reachable in $\mathcal{P}$.

Proof: Membership in NP: guess a marking m' of $\mathcal{Q}$ such that $B(m') = m$, check if it does not contain causally dependent or conflicting conditions.

Hardness: polynomial reduction from SAT (proof on blackboard)

Reducing reachability to SAT

In the other direction, we can, given m and $\mathcal{Q}$, produce a propositional logic formula, of polynomial size in $|m| + |\mathcal{Q}|$, that is satisfiable iff m is reachable in $\mathcal{P}$. This makes sense because extremely efficient SAT solvers are available.

The formula uses one boolean variable for each event and each condition. Its satisfying assignments are those that correspond to a reachable marking m' (i.e. concurrent sets of conditions) in $\mathcal{Q}$.

The formula assigns "true" to the conditions and events in $\bigcup_{c \in m'} \lfloor c \rfloor$ and false to all others; then it checks that no condition in m' is consumed by one of the events in that set and that no condition is consumed twice.

Finally, one demands that the image of m' in $\mathcal{P}$ is m.

Remarks

Remark (1): Notice that the unfolding and most of the formula is independent of m and needs to be generated from $\mathcal{P}$ only once for an arbitrary number of reachability queries.

Remark (2): In a very similar way, one can check whether $\mathcal{P}$ contains a deadlock, i.e. a reachable marking that does not enable any transition.