1 PLACE/TRANSITION NETS

- Let us generalize the basic notion of a Petri net:
  - Several tokens are allowed in each place.
  - There can be several arcs between a single place and a transition.
  - Tokens can now be interpreted as a resource.

©Johan Lilius 1997, Keijo Heljanko 1998, Esa Kettunen 1999, 2000

The Net

Definition 1 The 3-tuple $N =< S, T; F >$ is called a *net* iff

1. S and T are disjoint sets ($S \cap T = \emptyset$), comprising the sets called S elements and T elements, respectively, and
2. $F \subseteq (S \times T) \cup (T \times S)$ is the binary *flow relation*.

- For a given element $a \in S \cup T$, the set ${}^\bullet a = \{a' \mid a'Fa\}$ is called the *preset* of the element a, and, correspondingly the set $a^\bullet = \{a' \mid aFa'\}$ is called the *postset* of a.

©Johan Lilius 1997, Keijo Heljanko 1998, Esa Kettunen 1999, 2000

Place/Transition Net

Definition 2 6-tuple $N =< S, T; F, K, M_0, W >$ is a place/transition net (P/T net, German *Stellen/Transition-Netz*) iff:

1. $< S, T; F >$ is a finite Petri net; the elements of the sets S and T are called *places* and *transitions*.
2. Function $K : S \to (\mathbb{N} \uplus \{\omega\})$ associates a (possibly infinite) *capacity* with each place (infinite [unbounded] capacity is denoted with $\{\omega\}$),
3. Function $W : F \to (\mathbb{N} \setminus \{0\})$ assocatiates an *arc weight* with each arc, and
4. $M_0 \in \mathsf{M_N}$ is the *initial marking* if the net, where

$$\mathsf{M_N} = \{M : S \to \mathbb{N} \uplus \{\omega\} \mid \forall s \in S : M(s) \leq K(s)\},$$

is the set of *markings* of the net N.

©Johan Lilius 1997, Keijo Heljanko 1998, Esa Kettunen 1999, 2000

Place/Transition Net

1. $< S, T; F >=< \{p_1, p_2, p_3\}, \{t\}; \{(p_1, t), (p_2, t), (t, p_3)\} >$
2. $K = \{p_1 \mapsto 4, p_2 \mapsto \omega, p_3 \mapsto 5\}$
3. $W = \{(p_1, t) \mapsto 2, (p_2, t) \mapsto 1, (t, p_3) \mapsto 2\}$
4. $M_0 = \{p_1 \mapsto 2, p_2 \mapsto 5, p_3 \mapsto 0\}$

©Johan Lilius 1997, Keijo Heljanko 1998, Esa Kettunen 1999, 2000

Firing rule

Definition 3 Let $N =< S, T; F, K, M_0, W >$ be a P/T net. Let M be a marking of the net N.

1. The transition t is in M-*concession* (denoted $M[t >$) iff

$$\begin{aligned} \forall s \in {}^\bullet t \quad &: \quad M(s) \geq W(s,t) \wedge \\ \forall s \in t^\bullet \quad &: \quad M(s) \leq K(s) - W(t,s). \end{aligned}$$

   (Firing rule)

2. The transition t in M-concession can *fire* producing a *follower marking* M' (denoted $M[t > M'$), where the following holds

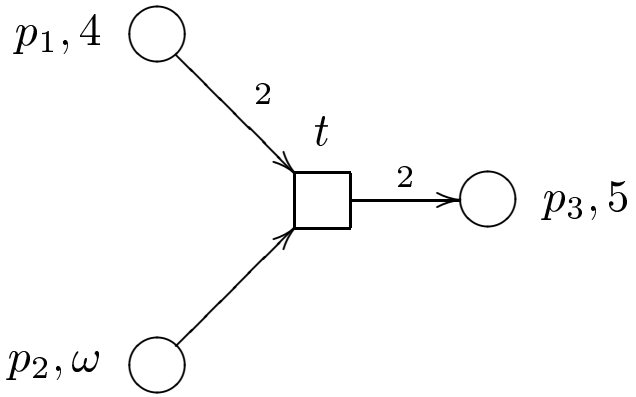
$$\forall s \in S : M'(s) = M(s) - W(s,t) + W(t,s).$$

   (Firing rule)

3. Let us use the notation $[M_0 >= \{M \in \mathsf{M_N} \mid M_0[>^* M\}$, for the set of markings which can be reached from marking M_0 with a finite sequence of transitions.

©Johan Lilius 1997, Keijo Heljanko 1998, Esa Kettunen 1999, 2000

Firing rule



In the initial marking:

- $M_0 = \{p_1 \mapsto 2, p_2 \mapsto 5, p_3 \mapsto 0\}$ the transition t is in concession.
- $M_0 = \{p_1 \mapsto 2, p_2 \mapsto 5, p_3 \mapsto 5\}$ the transition t is not in concession.
- $M_0 = \{p_1 \mapsto 1, p_2 \mapsto 5, p_3 \mapsto 0\}$ the transition t is not in concession.

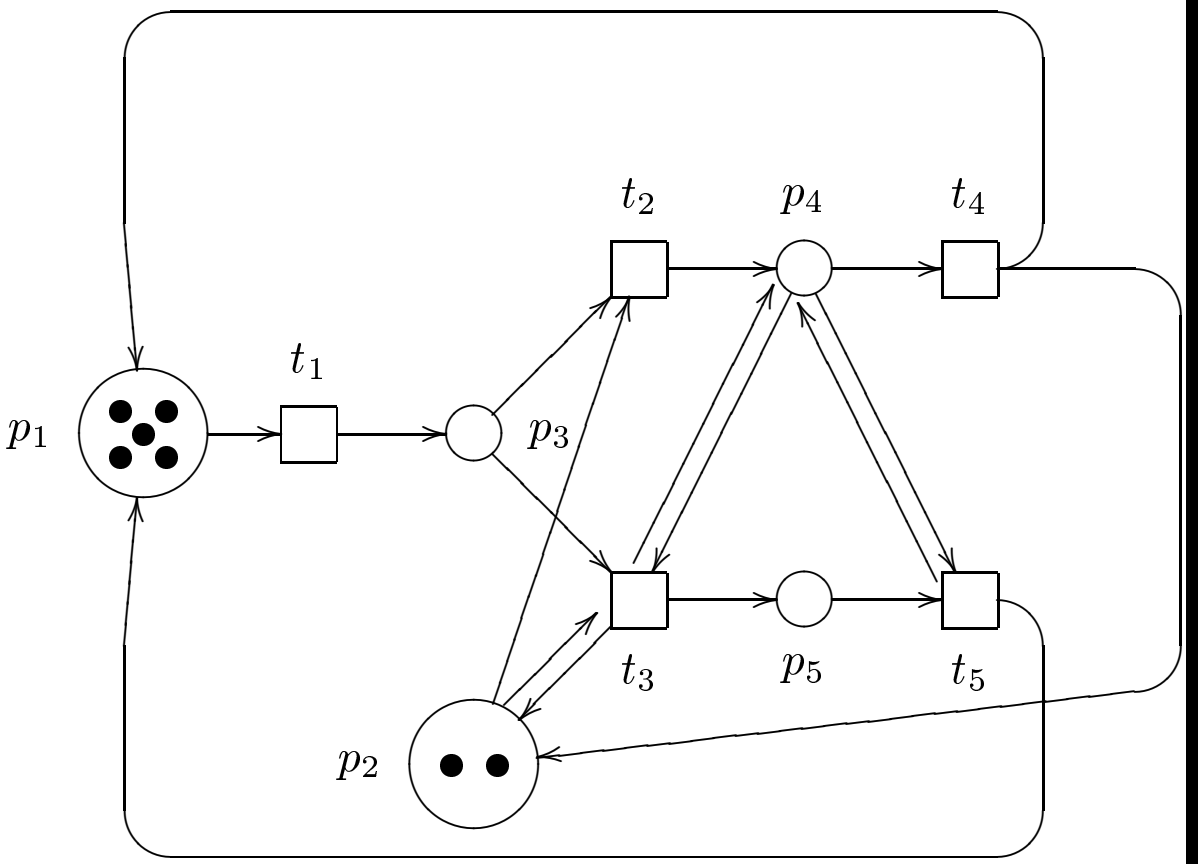
©Johan Lilius 1997, Keijo Heljanko 1998, Esa Kettunen 1999, 2000

Example: Multi-processor System

- 5 processors
- 3 shared memory units
- 2 busses
- Modelling: Level of abstraction is at the use of resources in the system.
- Processor states:
  - Active,
  - Waiting for bus to become available,
  - Waiting for memory to become available, or
  - Writing/reading memory.
- Processor states are modelled as places.
- Additionally, a place is is needed for the bus resources.

©Johan Lilius 1997, Keijo Heljanko 1998, Esa Kettunen 1999, 2000

Example: Multi-processor System



- t_1: Access request,
- p_3: Processor waiting for bus access,
- p_4: Processor has bus access,
- p_5: Processor waiting for memory access,
- t_5: End of queuing (processor has memory access),
- t_2, t_3: Memory usage; already allocated memory (t_3) or free memory (t_2),
- t_4: There are no access requests for memory, and
- Capacity for all places is ω.

©Johan Lilius 1997, Keijo Heljanko 1998, Esa Kettunen 1999, 2000

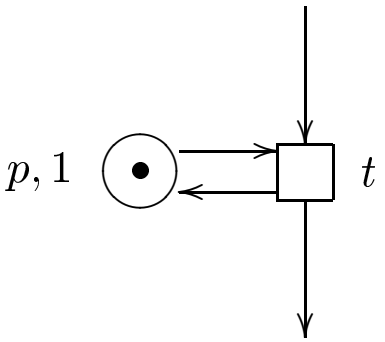
Example: Multi-processor System

- From the model we can conclude:
  - When one of the processors makes a memory access reservation, another bus can be used to convey access requests to queue up for the memory.
  - If two processors select free memory all the others wait.
  - For all markings M the following holds:
    1. $M(p_1) + M(p_3) + M(p_4) + M(p_5) = 5$, and
    2. $M(p_2) + M(p_4) = 2$
  - What we describe here are the *invariants* of the net.
- Please note!
  - The *identity* of an individual processor, bus, and memory is not known (we will discuss this later).
  - An arbitrary number of processors can be added to the model.
  - How about busses?
  - The model is independent of the number of memories.
  - By giving probabilities to the firings of transitions the quantitative behaviour of the system can be modelled (we will discuss this later). E.g. we might want to know how many processor are waiting for a resource (in a place) on average.

©Johan Lilius 1997, Keijo Heljanko 1998, Esa Kettunen 1999, 2000

The Tight Loop

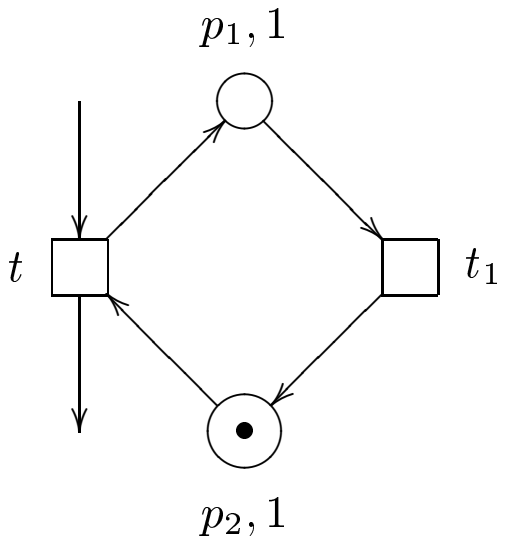
- In certain situations it is natural to use the following construct as part of the net:



  (compare with the multi-processor system)
- According to the firing rule t does not have concession.
- The firing rule can be changed to the following form:

$$\forall s \in {}^\bullet t \quad : \quad M(s) \geq W(s,t) \wedge$$
$$\forall s \in t^\bullet \quad : \quad M(s) - W(s,t) + W(t,s) \leq K(s).$$

- Alternatively the net can be changed:



©Johan Lilius 1997, Keijo Heljanko 1998, Esa Kettunen 1999, 2000

Reachability graph

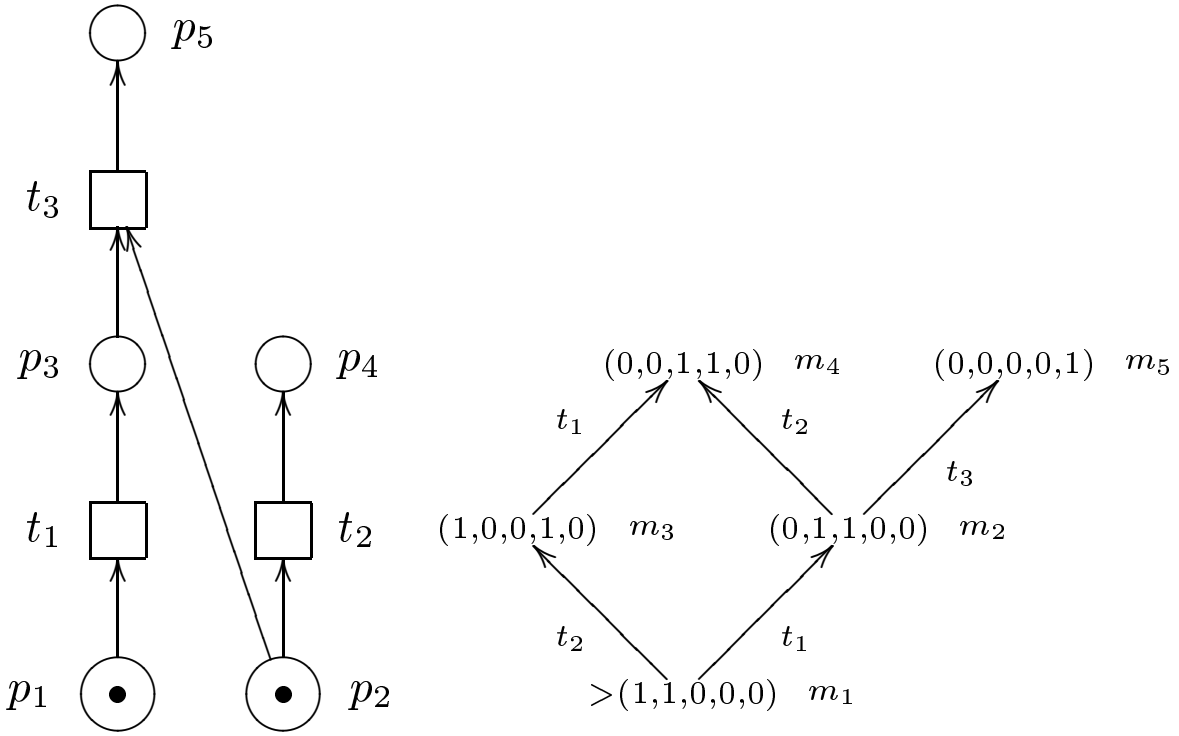
Definition 4 The *reachability graph* of a P/T net $N =< S, T; F, K, M_0, W >$ is a directed, rooted, arc-annotated graph $SG =< V; A, r >$, where $r = M_0, V = [M_0 >$, and

$$A = \{< M, t, M' > \mid M \in V \wedge M[t > M'\}$$

- Practically all analysis tools and methods have the the computation of the reachability graph as a prerequisite.
- The reachability graph makes it possible to consider a concurrent system as an ordinary automaton.
- On the other hand, it is impossible to deduce from a reachability graph what events are independent and what enents are concurrent.

©Johan Lilius 1997, Keijo Heljanko 1998, Esa Kettunen 1999, 2000

Reachability graph



- All capacities are arc weights $= 1$.
- Often the notation $\{p_1, p_2\}$ is used instead of $(1, 1, 0, 0, 0)$.
- The independent transitions t_1 and t_2 are *interleaved*.

©Johan Lilius 1997, Keijo Heljanko 1998, Esa Kettunen 1999, 2000

Reachability graph generation

- Given are the following functions:
  1. $enabled(M)$: return the set of transitions having concession in the marking M,
  2. $fire(t, M)$: return the new marking obtained by firing the transition t in marking M.

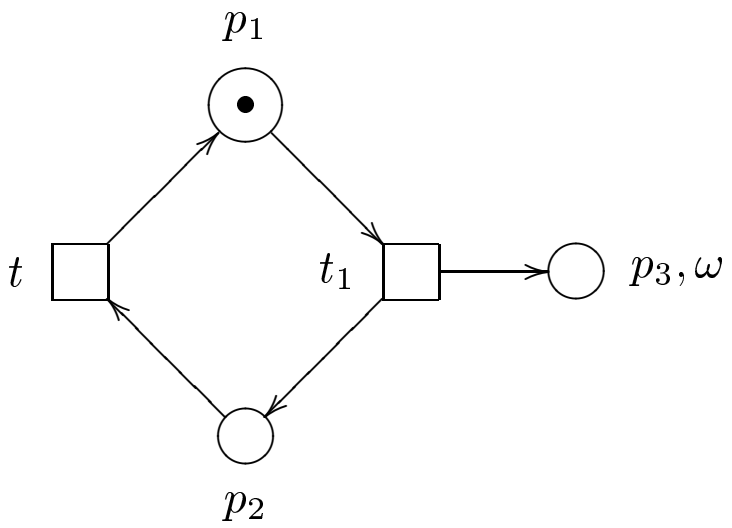
Reachability-Graph($N :< S, T; F, K, M_0, W >$)

```
1  G : graph; Markings : stack;
2  G ← ∅;
3  Markings.push(M_0);
4  while Markings ≠ ∅
5  do M ← pop(Markings);
6      for t in enabled(M)
7      do M' ← fire(t, M);
8          if M' ∉ G
9              then push(M', Markings);
10         G ← G ∪ {< M, t, M' >};
```

©Johan Lilius 1997, Keijo Heljanko 1998, Esa Kettunen 1999, 2000

Finite reachability graphs

- Infinite reachability graph:



- Reachability graph:

$$\underset{(1,0,0)}{m_1} \xrightarrow{t_1} \underset{(0,1,1)}{m_2} \xrightarrow{t} \underset{(1,0,1)}{m_3} \xrightarrow{t_1} \underset{(0,1,2)}{m_4} \quad \cdots$$

- The marking of the place p_3 is unbounded.
- Is it possible to calculate the boundedness of a net?

©Johan Lilius 1997, Keijo Heljanko 1998, Esa Kettunen 1999, 2000

Finite reachability graphs

Definition 5 A given P/T net N is *k-bounded* iff

$$\forall M \in [M_0 >, s \in S : M(s) \leq k.$$

A P/T net is *bounded* iff there exists $k \geq 0$ for which N is k-bounded. Otherwise the net is *unbounded*.

- The boundedness of a net can be decided by modifying the algorithm for generating the reachability graph.
  - The marking M *covers* the marking M' iff $M \geq M'$.
  - Before inserting the marking M to the graph it is checked if there is a marking M' on the trace leading to M such that $M \geq M'$. If such a marking is found then instead of the marking M another marking M'' will be added to the graph, when the following holds
    - $M''(s) = \omega$, if $M(s) > M'(s)$,
    - $M''(s) = M(s)$ otherwise.

©Johan Lilius 1997, Keijo Heljanko 1998, Esa Kettunen 1999, 2000

Coverability Tree

Coverability Tree(N)

```
 1  T : Tree; Markings : Stack;
 2  T ← ∅;
 3  Markings ← M_0;
 4  while Markings ≠ ∅
 5  do M ← pop(Markings);
 6      for t in enabled(M)
 7      do M' ← fire(t, M);
 8          M'' ← M;
 9          while (M'' ≠ nil) ∧ ¬(M'' ≤ M')
10          do M'' ← T.pred(M'');
11          if M'' ≠ nil
12            then M' ← M' + (M' − M'') · ω;
13          T.addLeaf(M, t, M');
14          M'' ← M;
15          while (M'' ≠ nil) ∧ (M'' ≠ M')
16          do M'' ← T.pred(M'');
17          if M'' = nil
18            then push(M', Markings);
```

©Johan Lilius 1997, Keijo Heljanko 1998, Esa Kettunen 1999, 2000

- Coverability
  - $M \geq M'$ iff $\forall s \in S : M(s) \geq M'(s)$, which is to say

$$\begin{aligned} (1,1,0) &\geq (1,0,0) \\ (1,0,0) &\ngeq (0,0,1) \end{aligned}$$

  - N.B.: $M \ngeq M'$ does not imply $M < M'$.
- Given are the following functions:
  - $T.pred(M)$ returning the node preceding M in the tree T,
  - $T.addleaf(M, t, M')$ starting from the node M, connect the new leaf M' to the tree with the arc t
- On lines 8–11, traverse the tree backwards until the root is reached or a marking M'' covered by M' is found.

©Johan Lilius 1997, Keijo Heljanko 1998, Esa Kettunen 1999, 2000

- E.g., on line 13:

$$
\begin{aligned}
(1,1,0) &= (1,1,0) + ((1,1,0) - (1,1,0)) \cdot \omega \\
&= (1,1,0) + (0,0,0) \cdot \omega \\
&= (1,1,0) + (0 \cdot \omega, 0 \cdot \omega, 0 \cdot \omega) \\
&= (1,1,0) + (0,0,0) \\
(1,\omega,0) &= (1,1,0) + ((1,1,0) - (1,0,0)) \cdot \omega \\
&= (1,1,0) + (0,1,0) \cdot \omega \\
&= (1,1,0) + (0 \cdot \omega, 1 \cdot \omega, 0 \cdot \omega) \\
&= (1,1,0) + (0,\omega,0)
\end{aligned}
$$

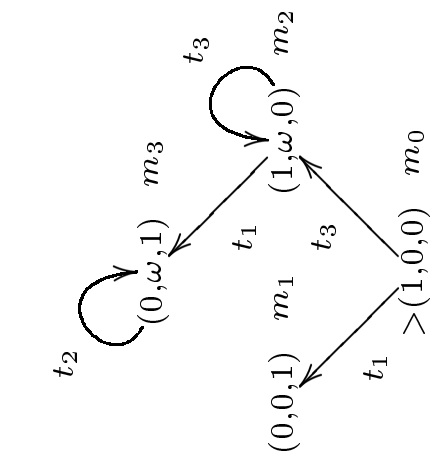
©Johan Lilius 1997, Keijo Heljanko 1998, Esa Kettunen 1999, 2000

Since

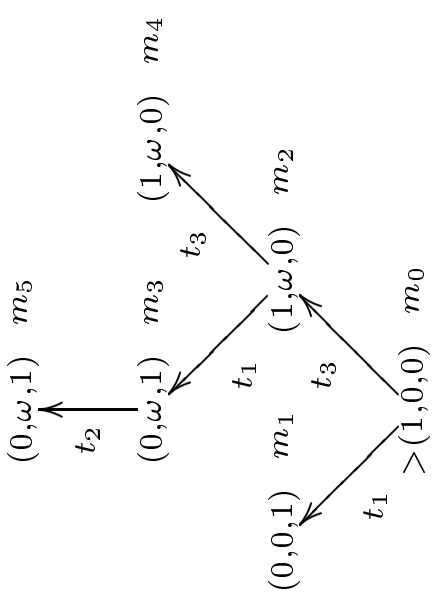
- $\forall n \in \mathbb{N} : \omega > n, \omega + n = \omega - n = \omega, \omega \geq \omega, n \cdot \omega = \omega \ \ (n > 0), 0 \cdot \omega = 0.$
- On line 15, the marking M' is added as a new leaf even though such markings had already been found.
- On lines 16–19, traverse the tree backwards until the generated marking is found or the root is reached.
- The latter indicates that the marking has not been visited before so it can be pushed to the stack (lines 20–22).
- The coverability tree is converted to a coverability graph by combining the covering nodes. Of course, this can be done during the generation of the tree. However, the loops on lines 9–11 and 17–19 will have to changed to graph traversals.
- N.B.: The coverability graph does not provide a solution to the reachibility problem of infinite P/T nets because the existence of a covering marking does not guarantee the reachability of a certain marking.

©Johan Lilius 1997, Keijo Heljanko 1998, Esa Kettunen 1999, 2000

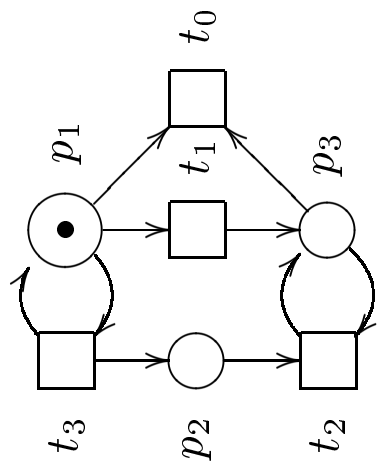
EXAMPLE



- Coverability graph.



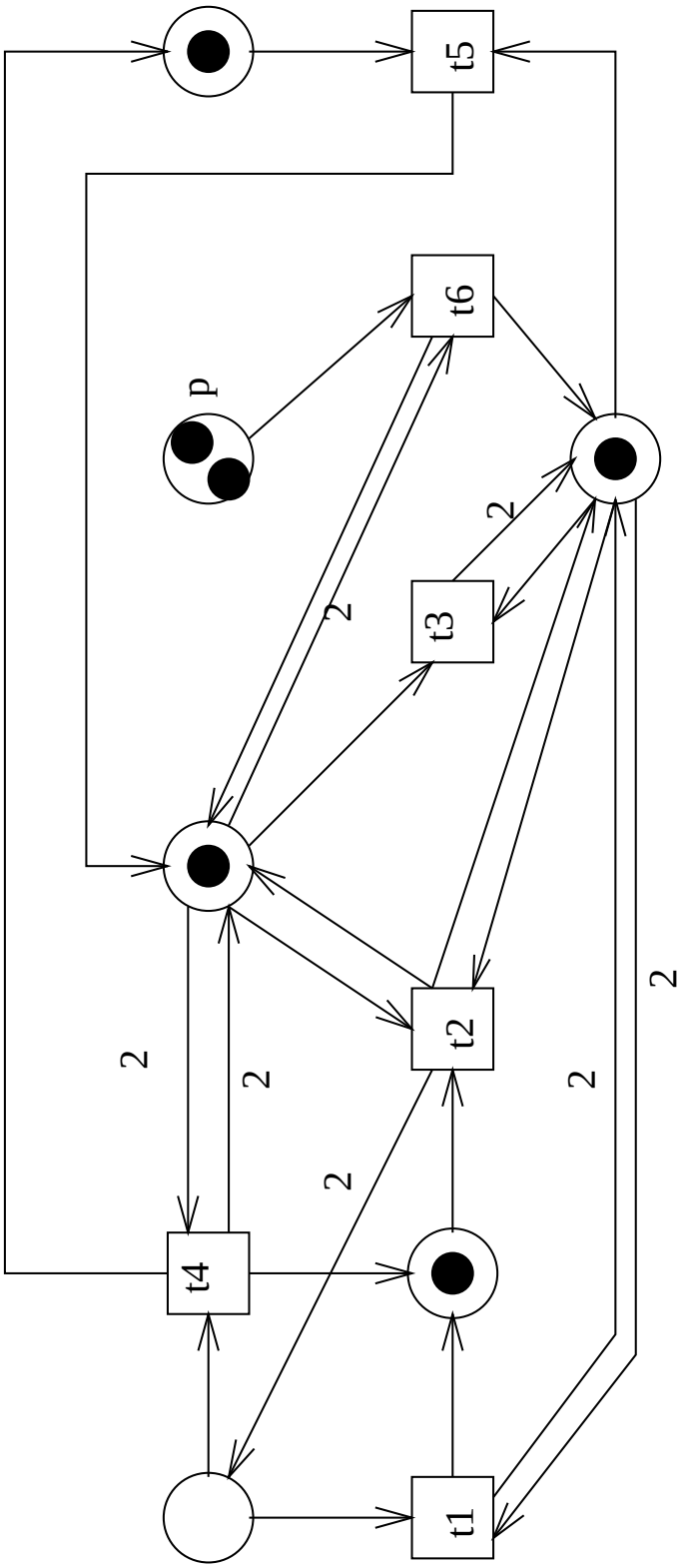
- Coverability tree.



- Unbounded net.

©Johan Lilius 1997, Keijo Heljanko 1998, Esa Kettunen 1999, 2000

COMPLEXITY



©Johan Lilius 1997, Keijo Heljanko 1998, Esa Kettunen 1999, 2000

Complexity

- The net is bounded for all initial markings where the place p is allowed contain an arbitrary number of tokens.
- The size of the initial marking is thus $M_0(p) + 4$, and the size of the net is maximal when $M_0(p) = k$. The following holds:

$$\begin{aligned} Z(k) &= 2 \cdot f(k) + 2 \\ f(0) &= 2 \\ f(k+1) &= f(k) \cdot 2^{f(k)} \end{aligned}$$

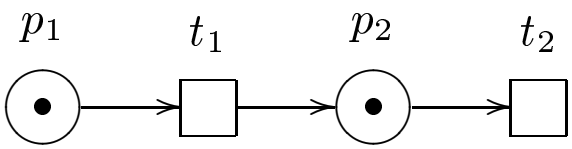
- $Z(0) = 6, Z(1) = 18, Z(2) = 4098, Z(3) = 2^{2060} + 2, \ldots$.
- The number of reachable markings: $M(0) = 30, M(1) = 427, \ldots$.
- In the general case, the time complexity may be superexponential, i.e. the values of the function increase faster than those of any exponential function (Cf. Ackermann function).
- However, for a 1-bounded net

$$|[M_0 >| \leq 2^S$$

©Johan Lilius 1997, Keijo Heljanko 1998, Esa Kettunen 1999, 2000

Contact

- In a net whose places have finite capacities, it is possible to find a marking with a phenomenon called the *contact*.
- This means that a transition does not have concession because one of its post-places is 'full'. (Compare with tight loop.)



- A practical interpretation of the contact phenomenon:
  - $p_1 =$ has money,
  - $t_1 =$ start making phone call,
  - $p_2 =$ making the phone call, and
  - $t_2 =$ hang up.

  Here the net above describes the following situation: You have the money so you can make a phone call unless someone is making a phone call, i.e. the phone is unavailable.

©Johan Lilius 1997, Keijo Heljanko 1998, Esa Kettunen 1999, 2000

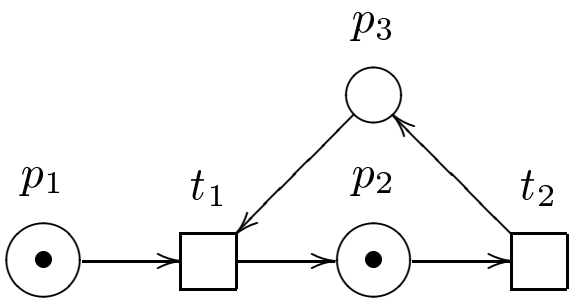
Contact

- A net can be rendered contact-free with the help of the *complement construction*:

  For each place s with a *finite capacity* a new place s' is added to the net, where

  1. The arc $< s', t >$ to the transition t iff there exists an arc tFs $(W(s', t) = W(t, s))$ in the original net,
  2. The arc $< t, s' >$ from the transitions t iff there exists an arc sFt $(W(t, s') = W(s, t))$ in the original net, and
  3. The initial marking is defined to maintain $M_0(s) + M_0(s') = K(s)$ for all pairs (s, s') of places and their complement places.

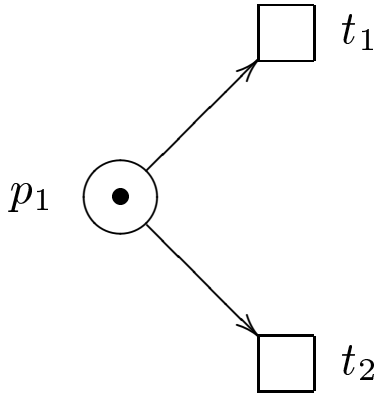
- In our example we will have the following:



©Johan Lilius 1997, Keijo Heljanko 1998, Esa Kettunen 1999, 2000

Confusion

- Two transitions are independent iff they have no pre-places or post-places in common. This means they can fire in an arbitrary order (or concurrently). The transitions are *concurrent*.
- A phenomenon called the *conflict* arises when two or more transitions share pre-places (**Note: Now it's really correct.**)

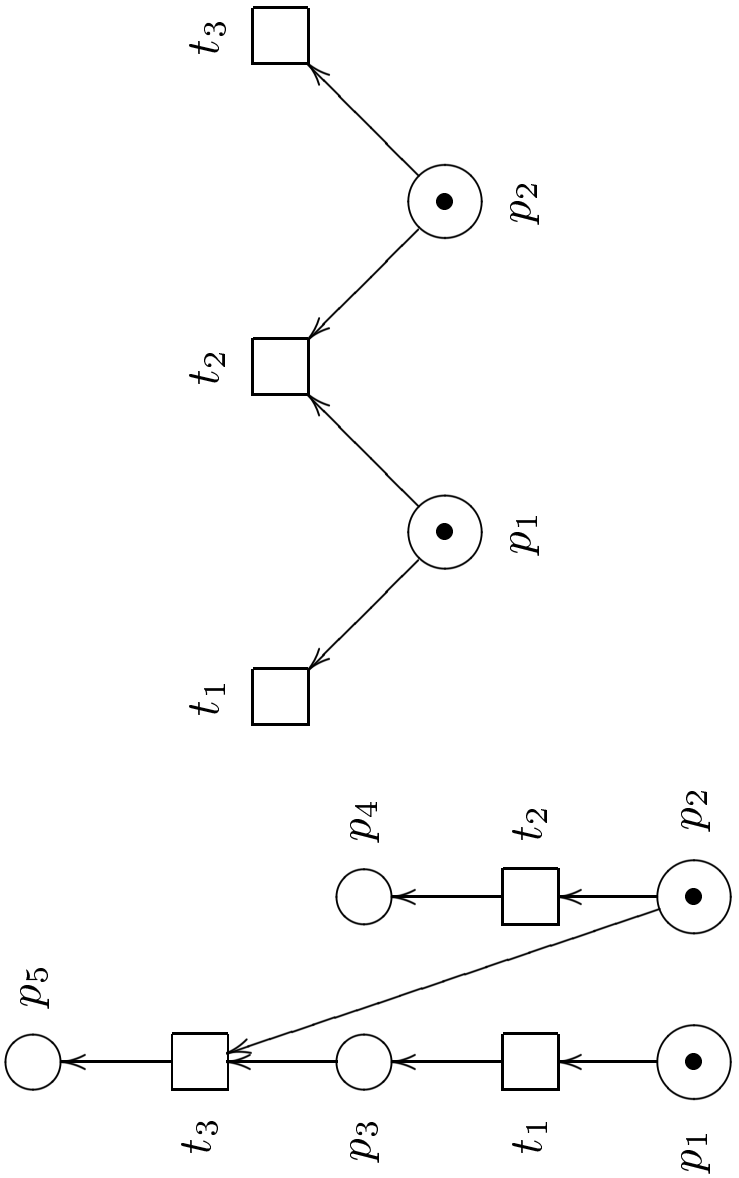


  This means only one of the transitions can fire. A conflict is a non-deterministic aspect of the system.
- A situation with both concurrency and conflict is called a *confusion*.

©Johan Lilius 1997, Keijo Heljanko 1998, Esa Kettunen 1999, 2000

CONFUSION

- Asymmetrical and symmetrical confusion:



©Johan Lilius 1997, Keijo Heljanko 1998, Esa Kettunen 1999, 2000

Confusion

- In the left-hand case confusion stems from the fact that it is not clear if t_2 and t_3 are in conflict:
  - Let us assume that the system has two observers H_1 and H_2 who can only see the firing of a single transition at a time.
  - H_1 observes $\{p_1, p_2\}[t_1; t_2 > \{p_3, p_4\}$
  - H_2 observes $\{p_1, p_2\}[t_2; t_1 > \{p_3, p_4\}$
  - H_1 observes, that t_2 and t_3 are in a conflict which is resolved in favour of t_2.
  - H_2 does not observe a conflict.
- Correspondingly, in the right-hand case it is not clear which conflict is resolved first:
  - H_1 observes $\{p_1, p_2\}[t_1; t_3 >$
  - H_2 observes $\{p_1, p_2\}[t_3; t_1 >$
  - According to H_1, the conflict between t_1 and t_2 is resolved in favour of t_1.
  - According to H_2, the conflict between t_2 and t_3 is resolved in favour of t_3.
- Confusion is a source of problems:
  1. When there is an attempt to analyse whether the system is *fair* or not, and
  2. When trying to apply heuristics to reduce the size of the reachability graph.

©Johan Lilius 1997, Keijo Heljanko 1998, Esa Kettunen 1999, 2000