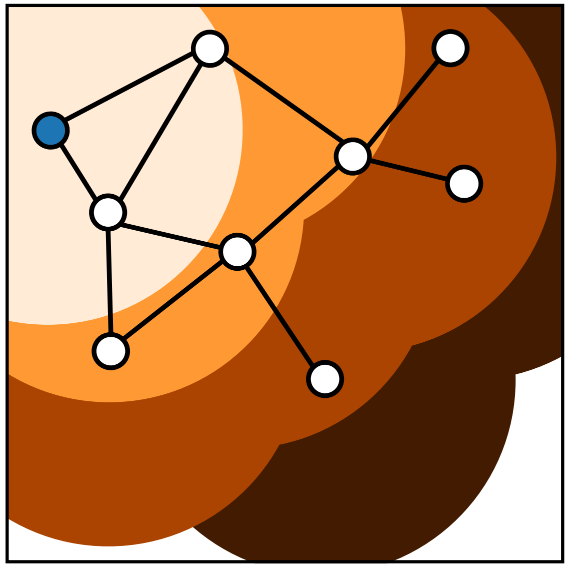
Discrete Event Systems
Petri Nets



Romain Jacob

www.romainjacob.net

ETH Zurich (D-ITET)

December 9, 2021

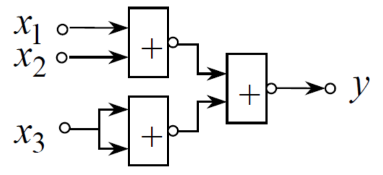
Most materials from Lothar Thiele

Last week in Discrete Event Systems

Verification Scenarios

Example

$y = (x_1 + x_2) \cdot x_3$



Comparison of specification and implementation

$$\begin{tikzcd}
\text{reference system} \arrow[r] & \text{data structure} \arrow[r] & \text{comparison} \\
\text{system under test} \arrow[r] & \text{data structure} \arrow[ur] &
\end{tikzcd}$$

Proving properties

"The device can always be switched off."

$$\begin{tikzcd}
\text{property} \arrow[rr] & & \text{fixed-point calculation} \\
\text{system under test} \arrow[r] & \text{data structure} \arrow[ur] &
\end{tikzcd}$$

So... what is model-checking exactly?

Model-checking is an algorithm which takes two inputs

- a DES model $\boldsymbol{M}$
- a formula $\boldsymbol{\phi}$

Finite automato
Petri nets
Kripke machine
...

CTL, LTL, ...

It explores the state space of $\boldsymbol{M}$ such as to either

- prove that $\boldsymbol{M} \vDash \boldsymbol{\phi}$, or
- return a trace where the formula does not hold in $\boldsymbol{M}$.

a counter-example

Extremely useful!

- Debugging the model
- Searching a specific execution sequence

Let’s see how it works in practice...

UPPAAL model-checker

- free for academia
- (much) more general than what we show here
- can verify the timed behavior of communicating finite automata

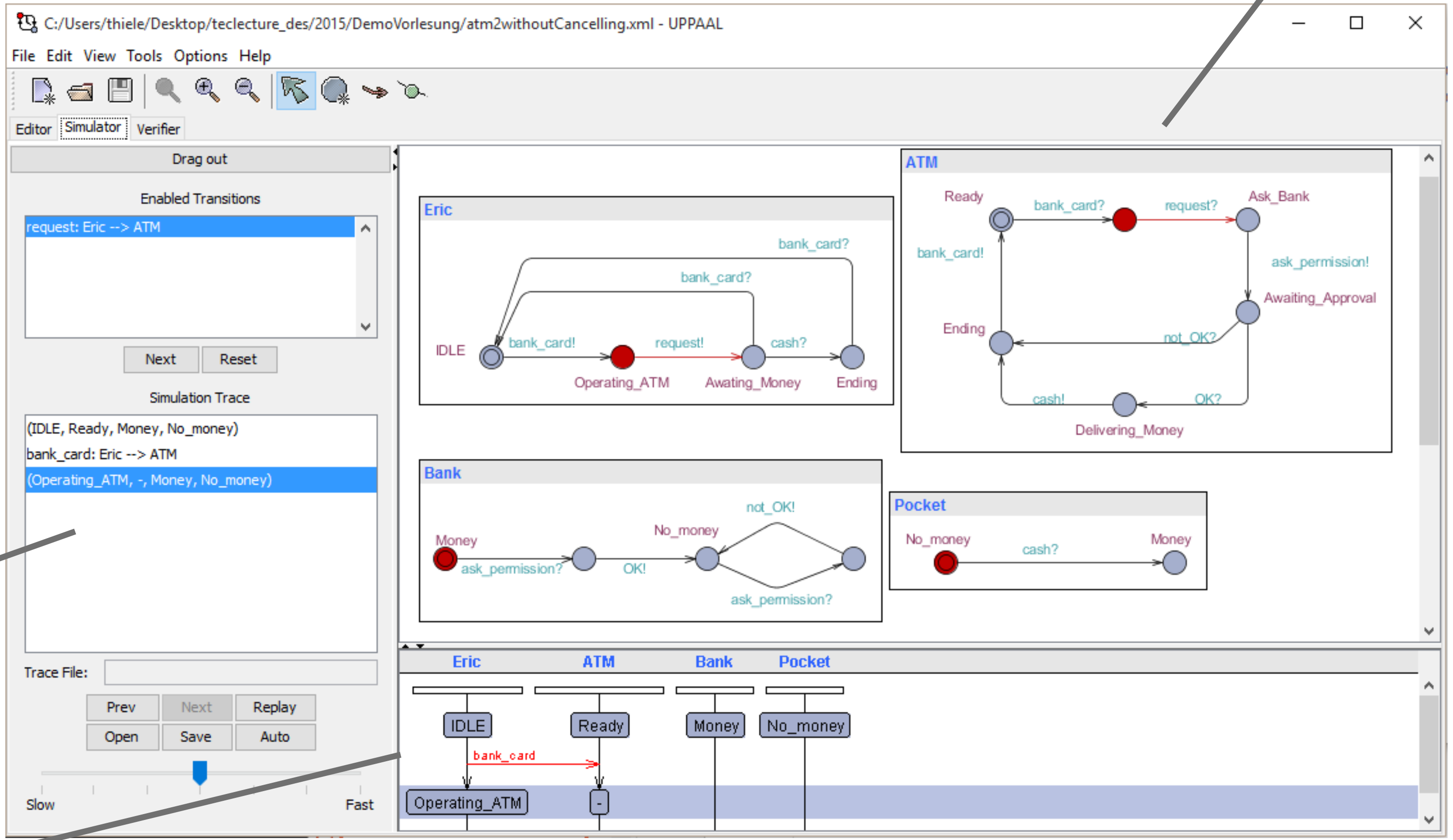
Example

Modeling and verification of a simple protocol for ATM-Money-Withdrawal

communicating finite automata

simulation trace

sequence diagram


C:/Users/thiele/Desktop/teclecture_des/2015/DemoVorlesung/atm2withoutCancelling.xml - UPPAAL

Enabled Transitions

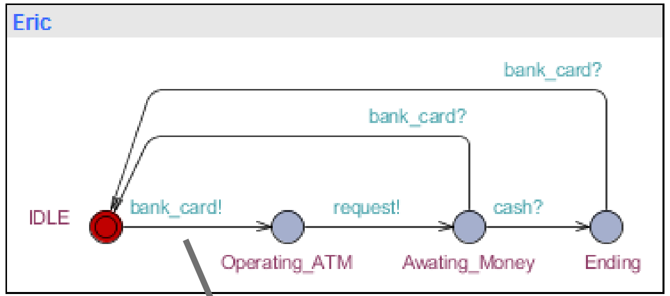
request: Eric --> ATM

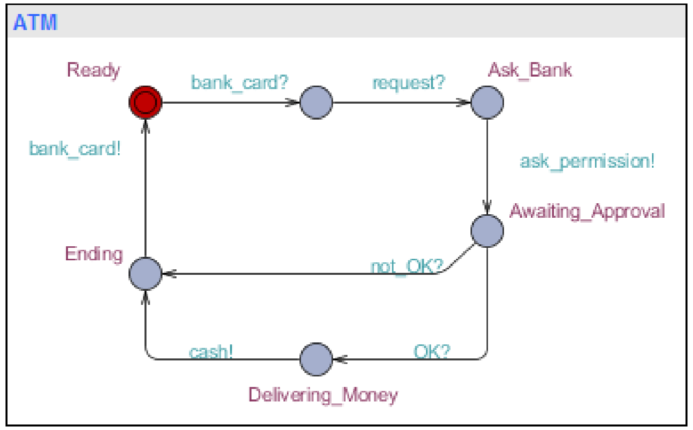
Simulation Trace

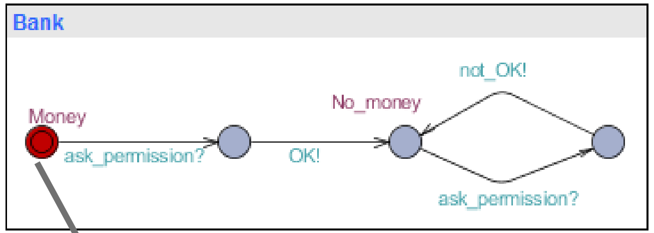
```
(IDLE, Ready, Money, No_money)
bank_card: Eric --> ATM
(Operating_ATM, -, Money, No_money)
```

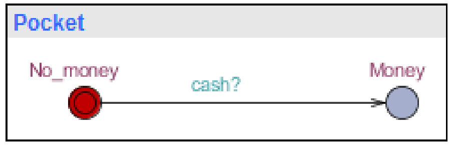
Step 1. ATM without Cancel

communicating finite automata









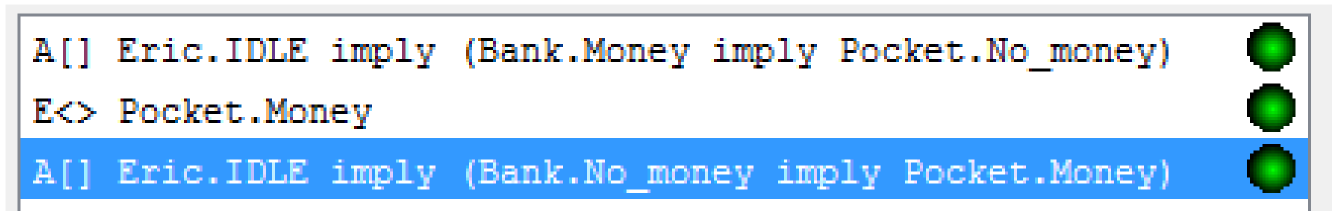
send event "bank_card"

initial state

enabled by event "cash"

AG

EF



```
A[] Eric.IDLE imply (Bank.Money imply Pocket.No_money)
E<> Pocket.Money
A[] Eric.IDLE imply (Bank.No_money imply Pocket.Money)
```

This week in Discrete Event Systems

Petri Nets – Motivation

In contrast to state machines, state transitions in Petri nets are asynchronous.
The ordering of transitions is partly uncoordinated; it is specified by a partial order.

Therefore, Petri nets can be used to model concurrent distributed systems.

Many flavors of Petri nets are in use, e.g.

- Activity charts (UML)
- Data flow graphs, signal flow graphs and marked graphs
- GRAFCET (programming language for programming logic controllers)
- Specialized languages for workflow management and business processes

Invented by Carl Adam Petri in 1962 in his thesis "Kommunikation mit Automaten"

Definition	- Semantics - Token game
Properties	- Safety - Liveness
Analysis	- Coverability tree - Incidence matrix

Definition

- Semantics
- Token game

Properties

- Safety
- Liveness

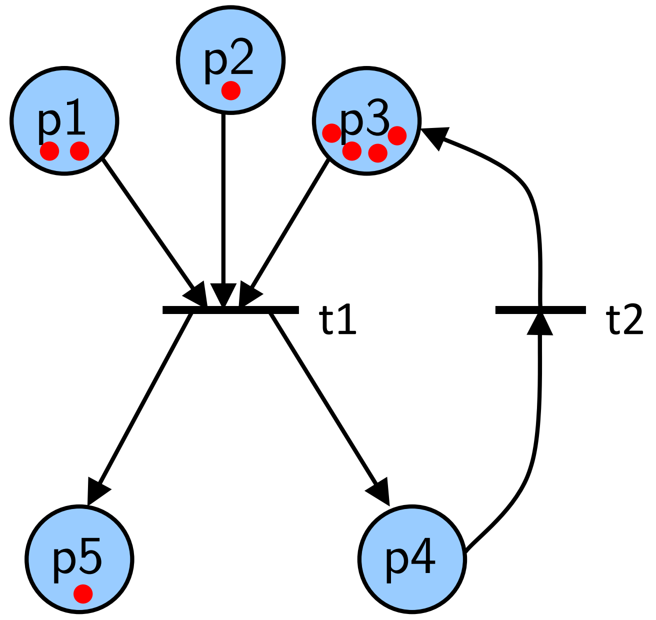
Analysis

- Coverability tree
- Incidence matrix

Petri Net – Definition

A Petri net is a bipartite, directed graph defined by a 4-tuple (S, T, F, M_0), where

- S is a set of places p
- T is a set of transitions t
- F is a set of edges (flow relations) f
- M0 : S $\rightarrow$ N; the initial marking



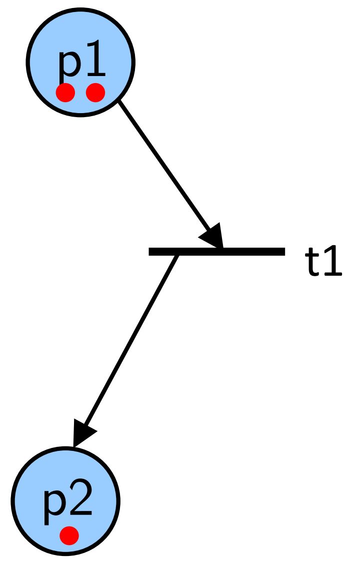
$\{p1, p2, p3, p4, p5\} \in$ S

$\{t1, t2\} \in$ T

$\{(p1, t1), (p2, t1), (t1, p5), \dots\} \in$ F

Token Marking

- Each place p_i is marked with a certain number of tokens.
- The initial distribution of the tokens is given by M_0 .
- M(s) denotes the marking of a place s.
- The distribution of tokens on places defines the state of a Petri net.
- The dynamics of a Petri net is defined by a token game.



Token Game of Petri Nets

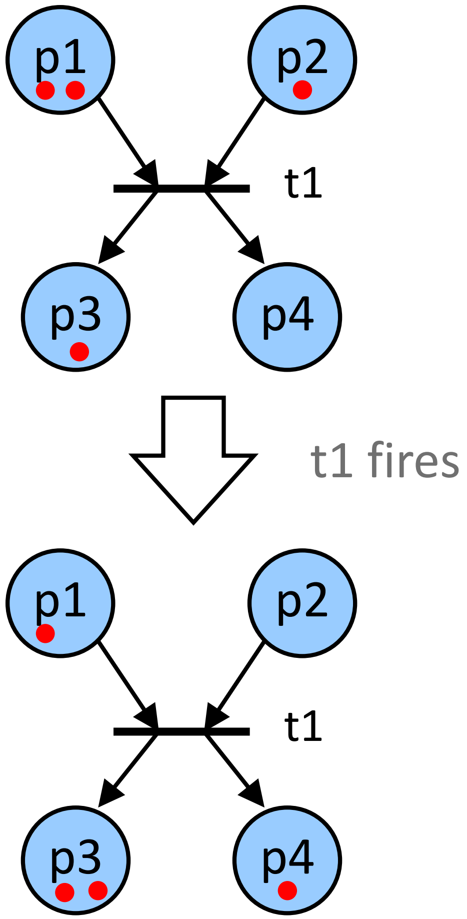
A marking M activates a transition t ∊ T if each place p connected through an edge f towards t contains at least one token.

If a transition t is activated by M,
a state transition to M’ fires (happens) eventually.

Only one transition is fired at any time.

When a transition fires

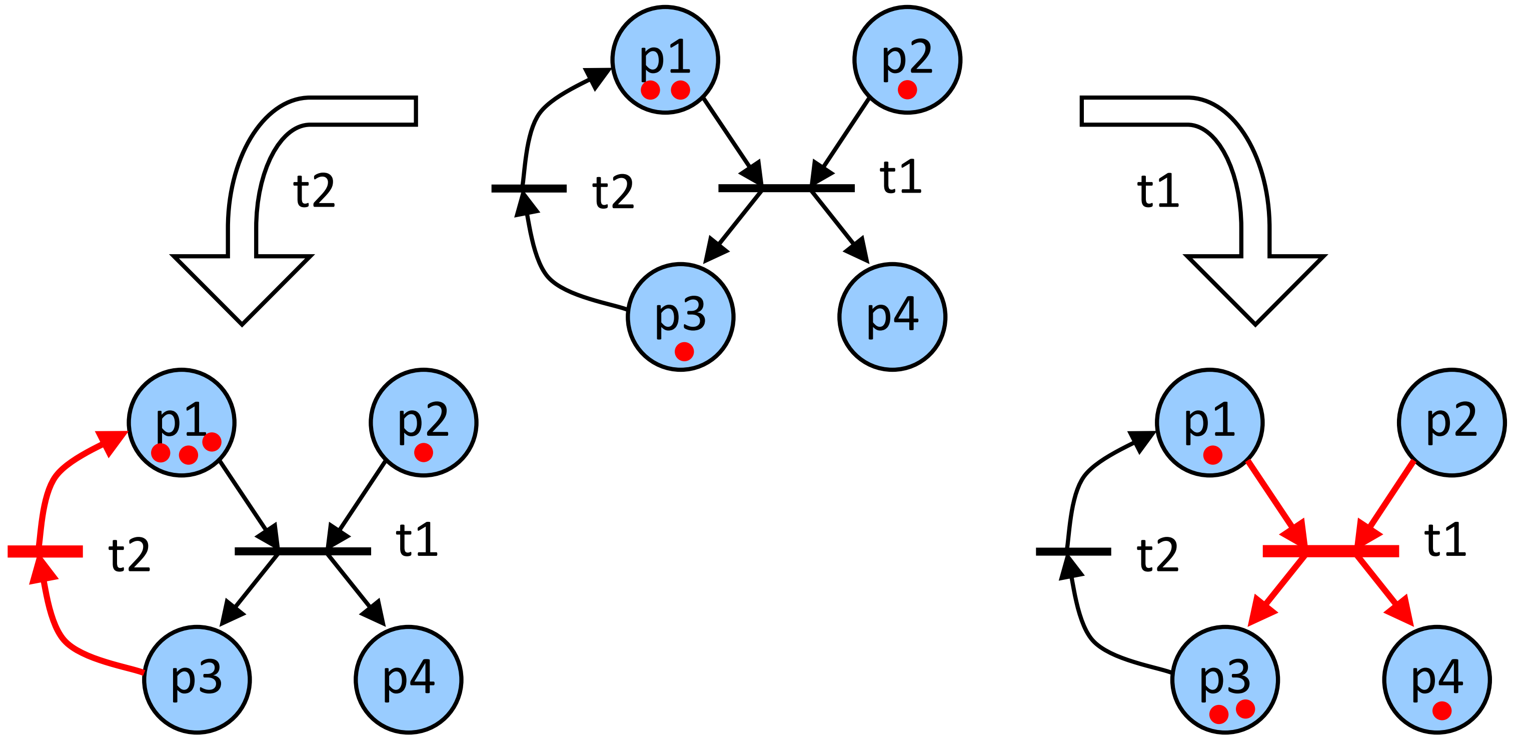
- it consumes a token from each of its input places,
- it adds a token to each of its output places.



Non-Deterministic Evolution

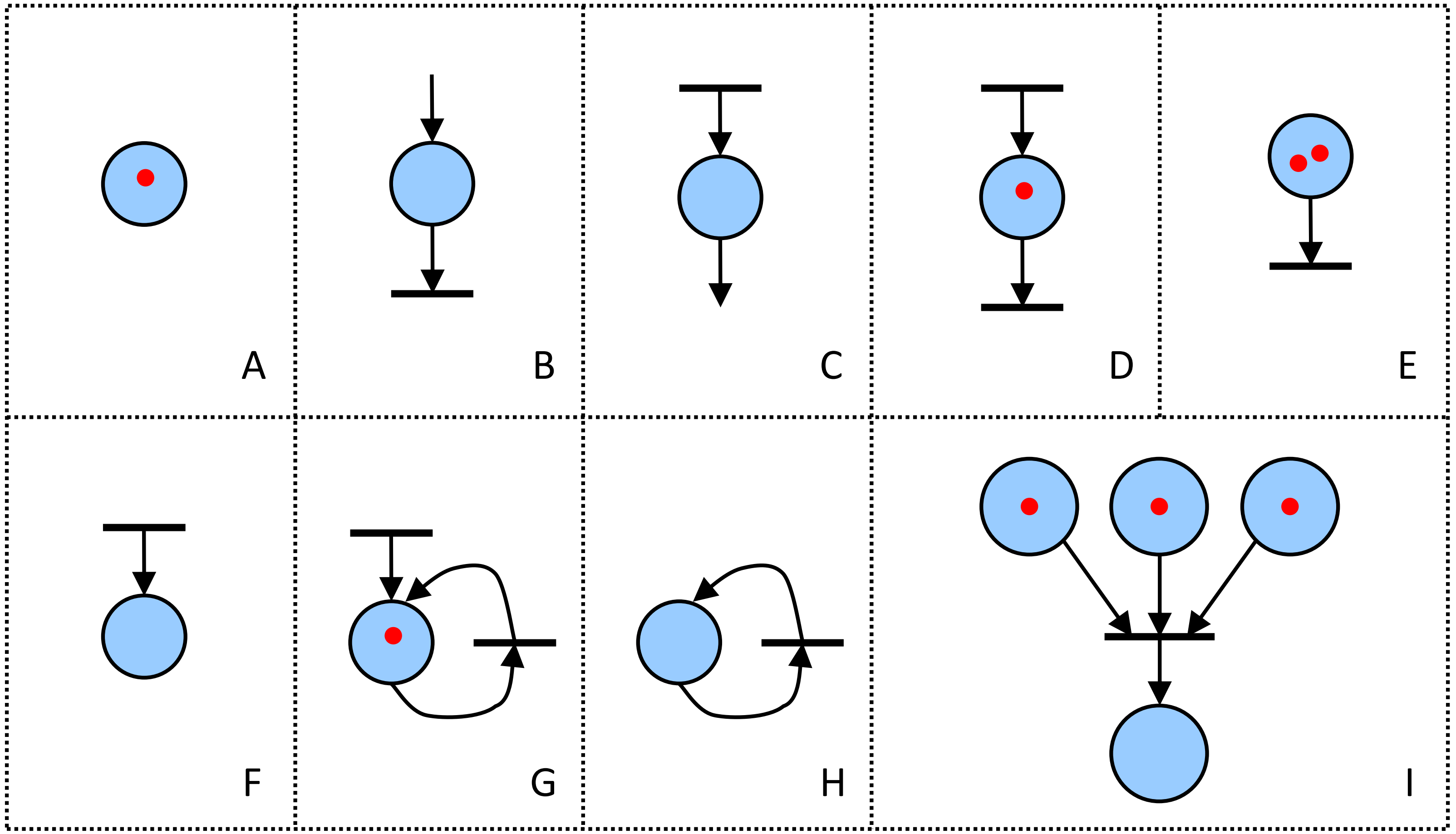
The evolution of
Petri nets is
not deterministic.

Any activated transactions can fire.



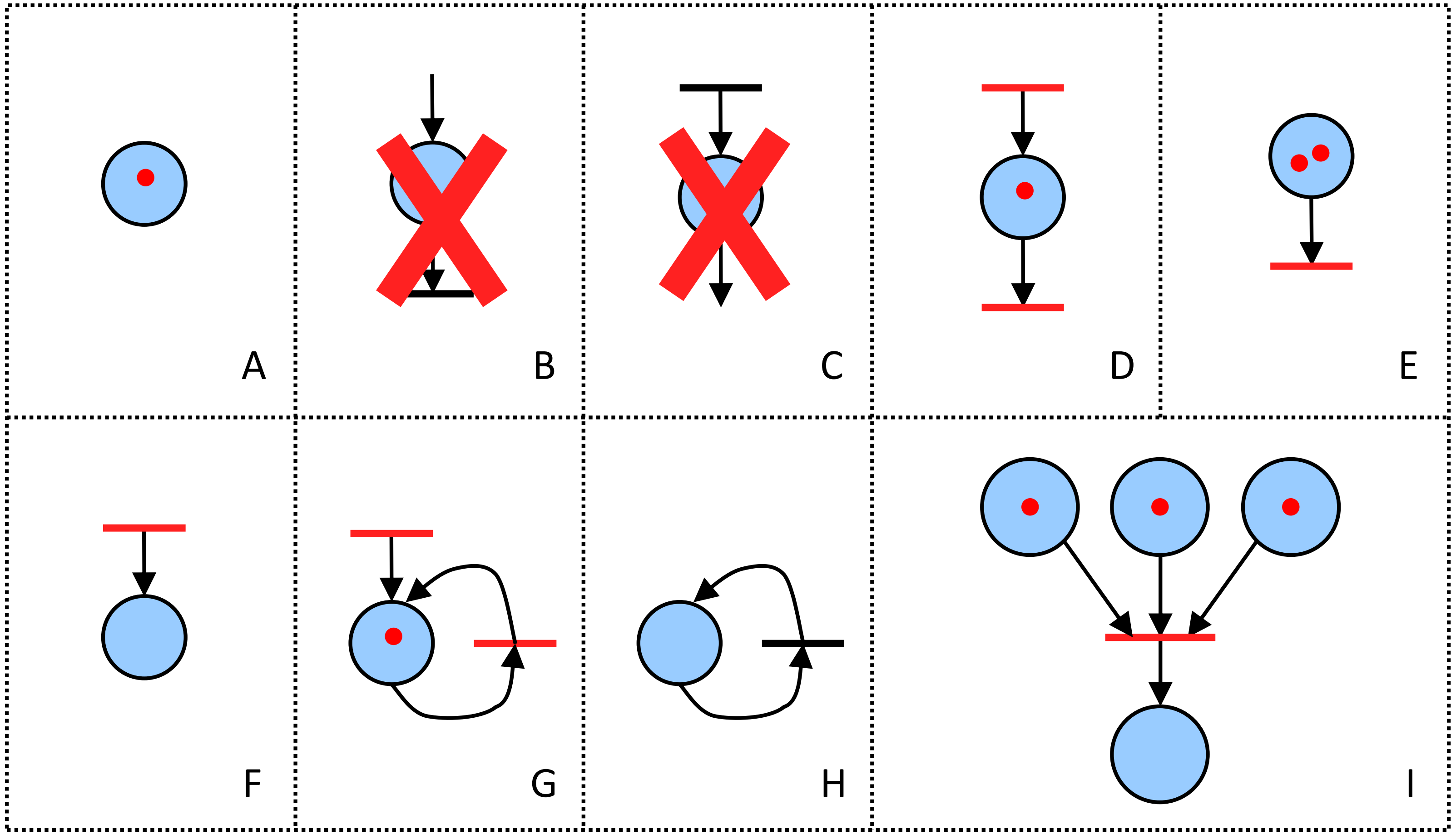
Syntax Exercise (1)

- Is it a valid Petri Net?
- Which transitions are activated?
- What is the marking after firing?



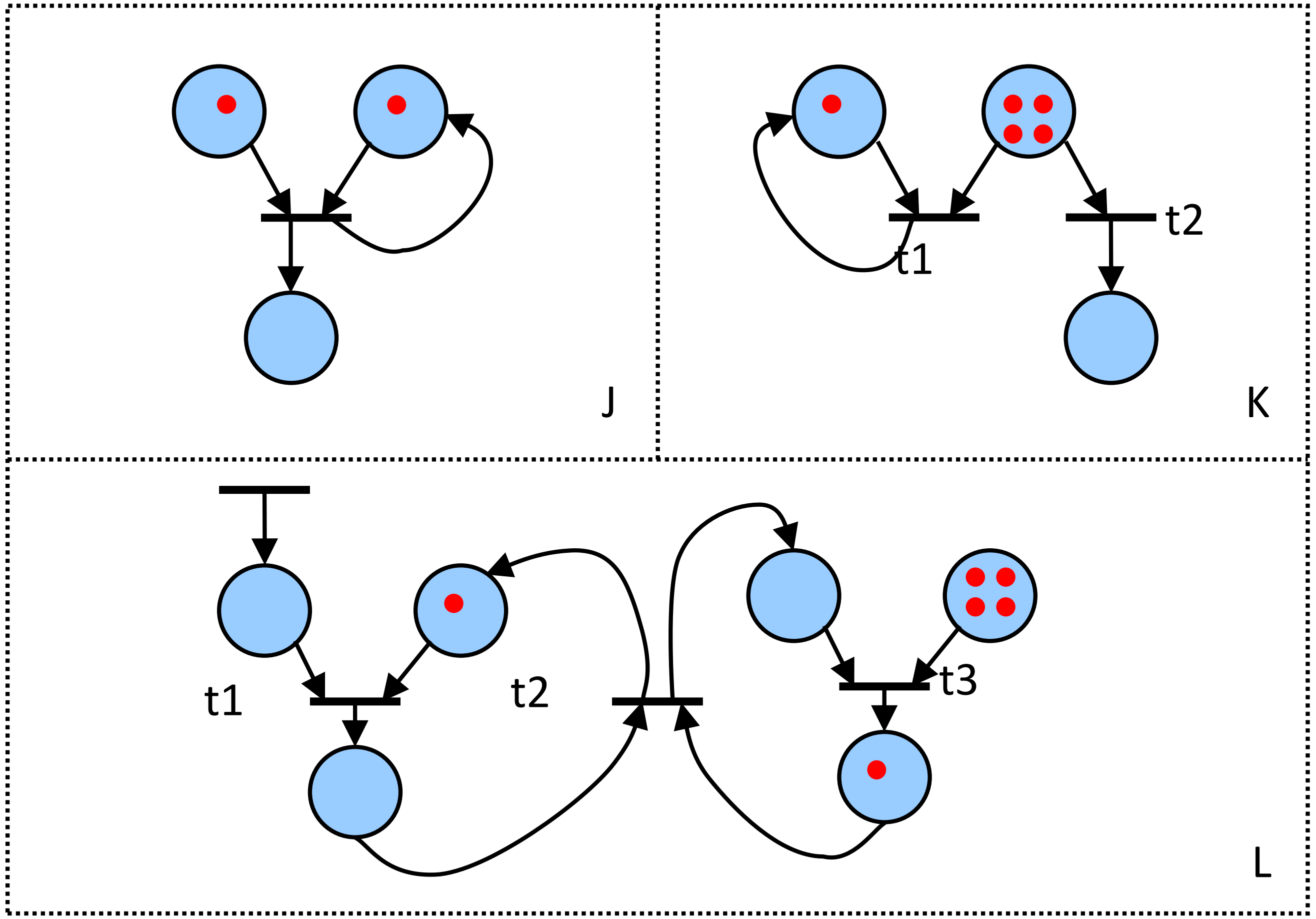
Syntax Exercise (1)

- Is it a valid Petri Net?
- Which transitions are activated?
- What is the marking after firing?



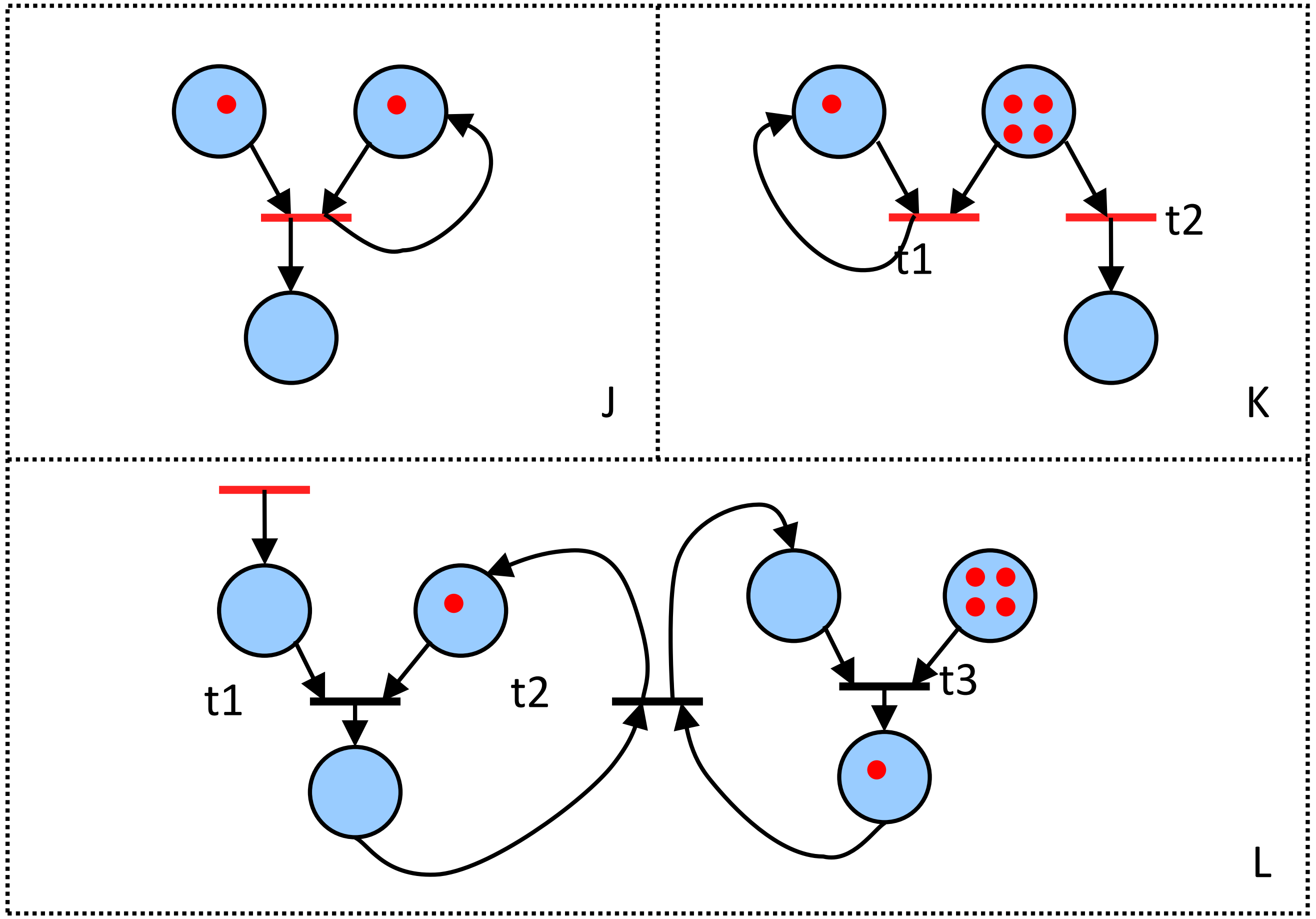
Syntax Exercise (2)

- Is it a valid Petri Net?
- Which transitions are activated?
- What is the marking after firing?



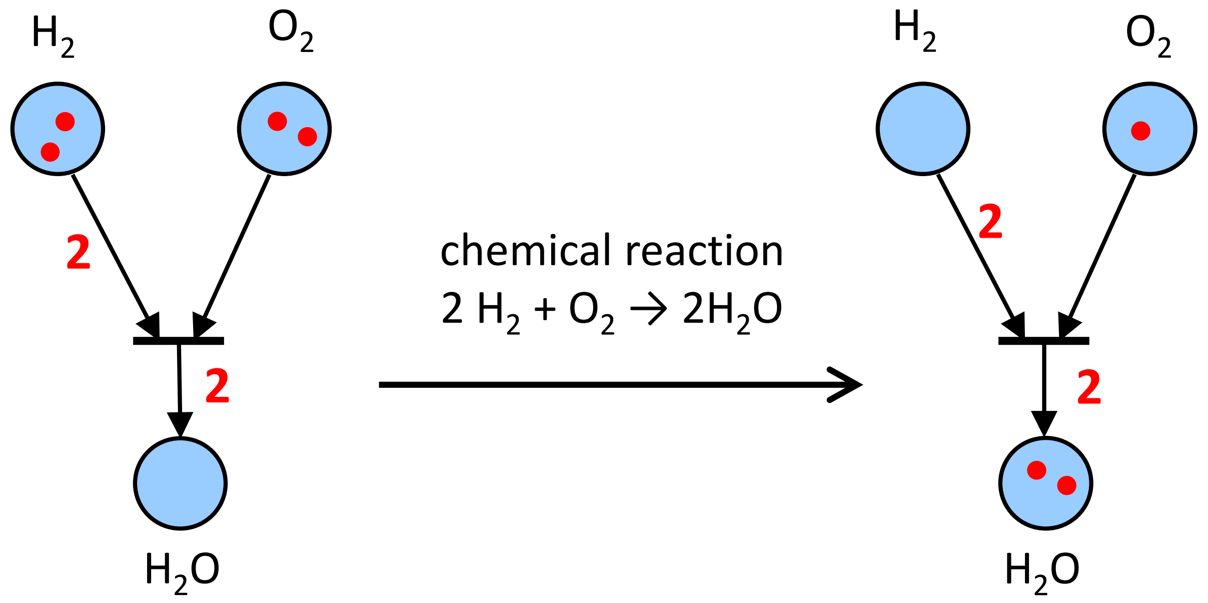
Syntax Exercise (2)

- Is it a valid Petri Net?
- Which transitions are activated?
- What is the marking after firing?



Weighted Edges

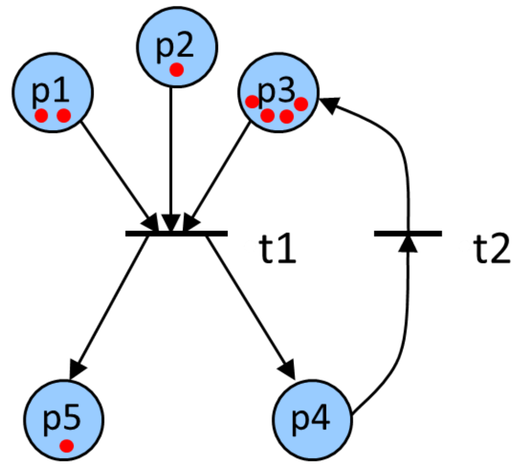
- Weights can be associated to edges.
- Each edge f has an associated weight W(f) (defaults to 1).
- A transition t is activated if each place p connected through an edge f to t contains at least W(f) token.
- When transition t fires, then W(f) token are transferred.



State Transition Function

- Using the previous definitions, we can now define the state transitions function δ of a Petri net:
  - Suppose that in a given Petri net (S, T, F, W, M_0) the transition t is activated. Before firing the marking is M.
  - Then after firing t, the new marking is $M' = \delta(M, t)$ with

$$M'(p) = \begin{cases} M(p) - W(p, t) & \text{if } (p, t) \in F \text{ and } (t, p) \notin F \\ M(p) + W(t, p) & \text{if } (t, p) \in F \text{ and } (p, t) \notin F \\ M(p) - W(p, t) + W(t, p) & \text{if } (t, p) \in F \text{ and } (p, t) \in F \\ M(p) & \text{otherwise} \end{cases}$$

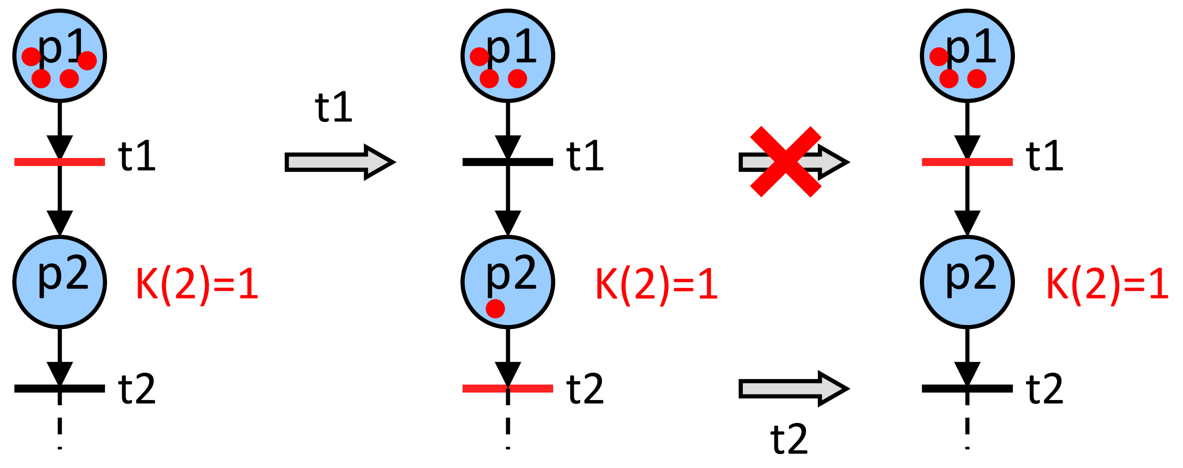


$\{p1, p2, p3, p4, p5\} \in S$
$\{t1, t2\} \in T$
$\{(p1, t1), (p2, t1), (t1, p5), \ldots\} \in F$

  - We also write sometimes $M' = M \cdot t$ instead of $M' = \delta(M, t)$.

Finite Capacity Petri Net

- Each place p can hold maximally K(p) token.
- A transition t is only active if all output places pi of t cannot exceed K(pi) after firing t.

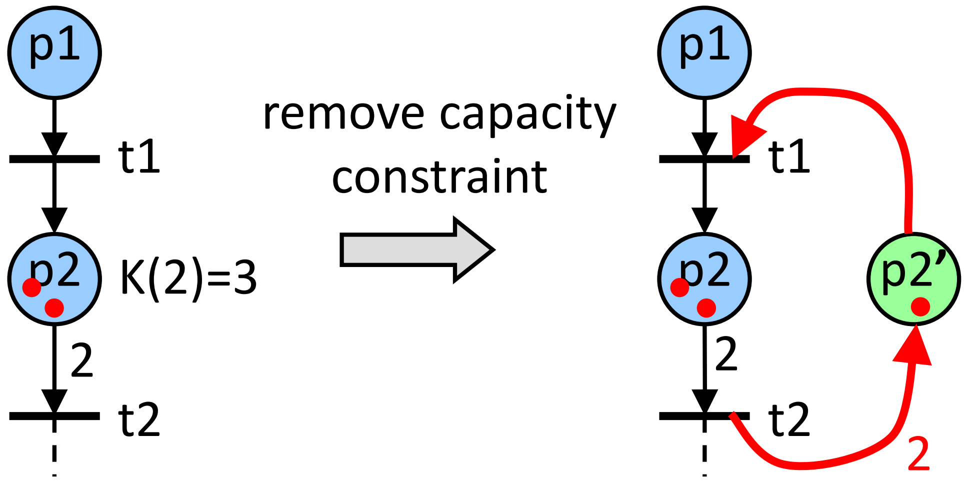


- Finite capacity Petri Nets can be transformed into equivalent infinite capacity Petri Nets (without capacity restrictions)

where "equivalent" means "Both nets have the same set of possible firing sequences."

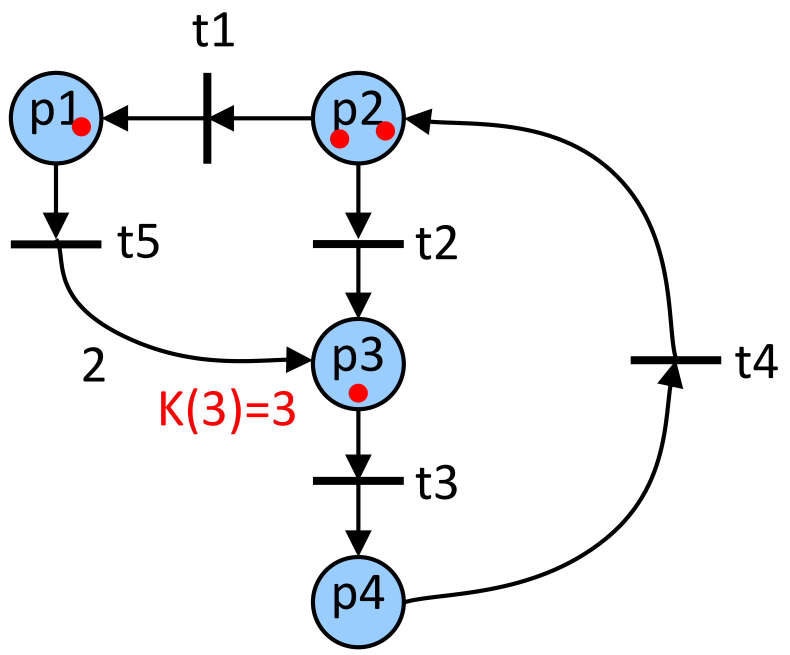
Removing Capacity Constraints

- For each place p with K(p) > 1, add a complementary place p' with initial marking M0(p') = K(p) – M0(p).
- For each outgoing edge f = (p, t), add an edge f' from t to p' with weight W(f).
- For each incoming edge f = (t, p), add an edge f' from p' to t with weight W(f).



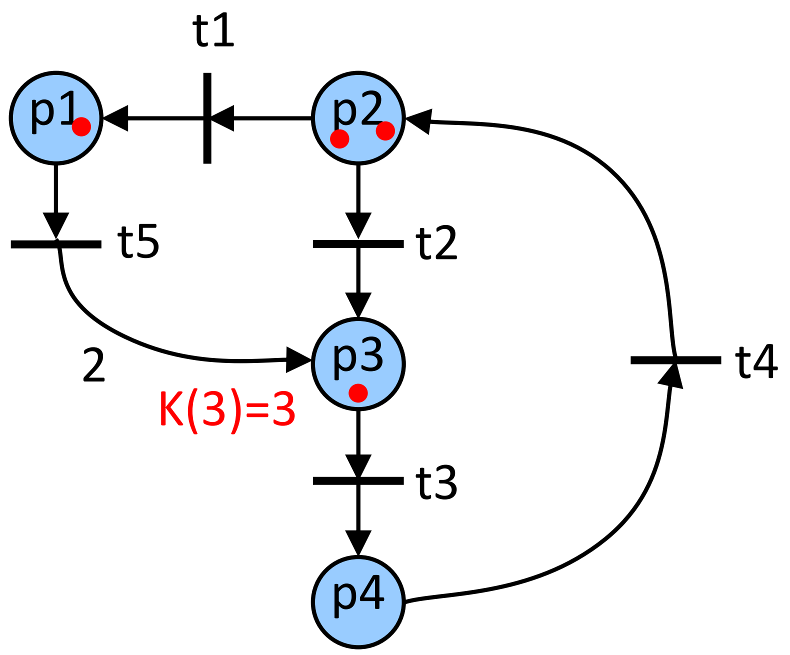
Your turn!

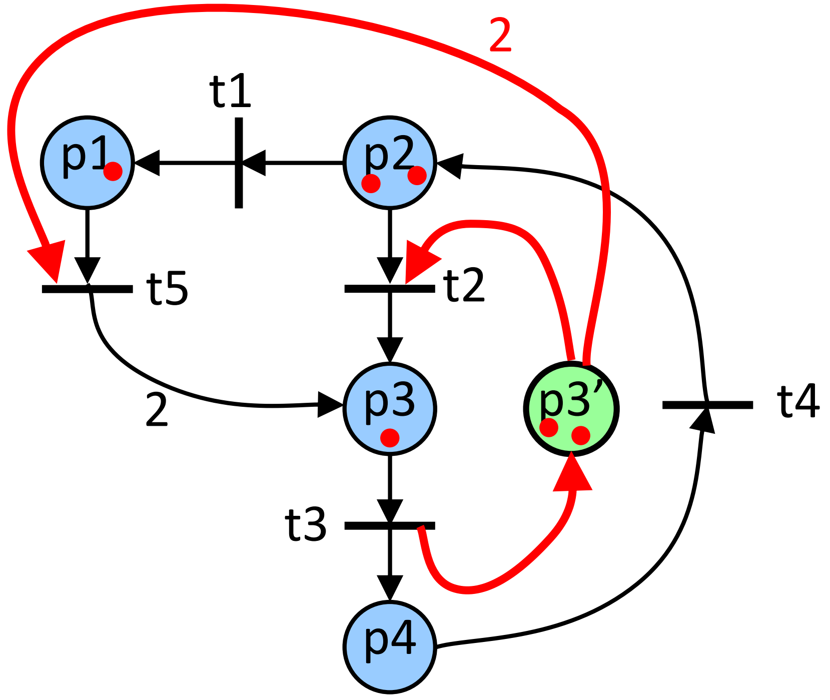
Remove the capacity constraint from place p3.



Your turn!

Remove the capacity constraint from place p3.





Modeling Finite Automata

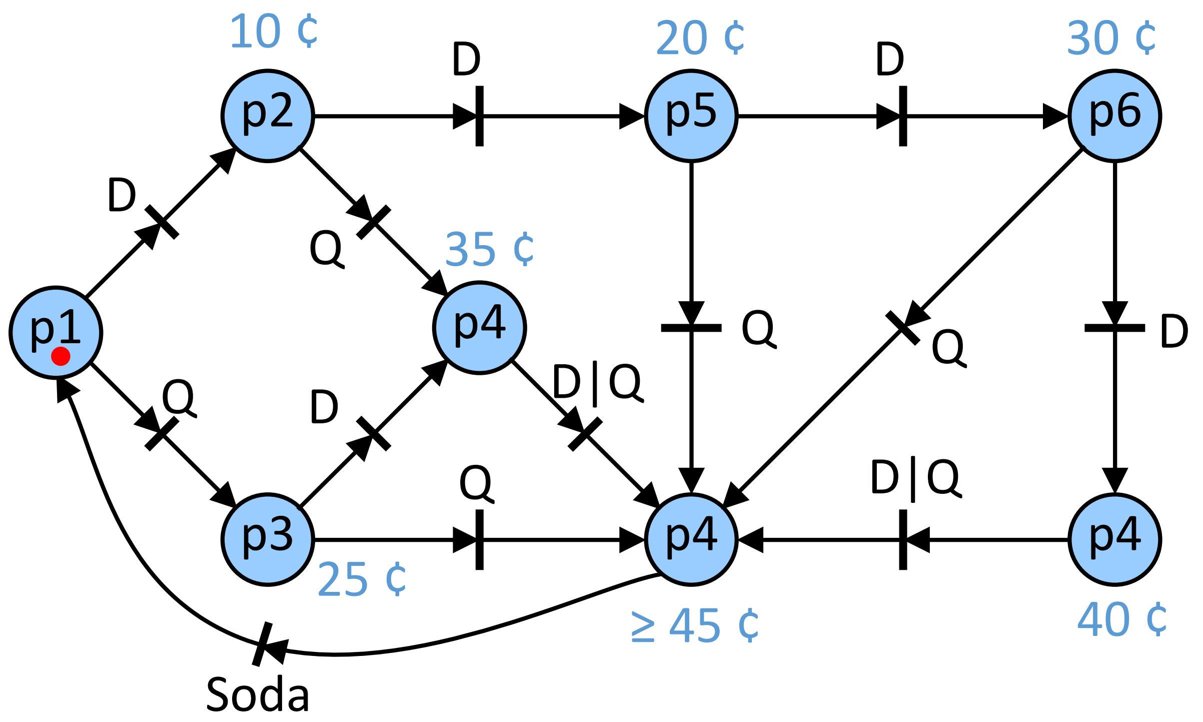
Finite automata can be represented by a subclass of Petri nets,

where each transition has exactly one incoming edge and one outgoing edge.

Coke vending machine

Coke costs 45 ¢.
Customer pays with
- Dime (10 ¢) or
- Quarter (25 ¢).

Overpaid money is lost.

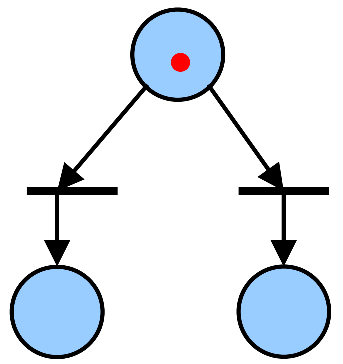


Concurrent Activities

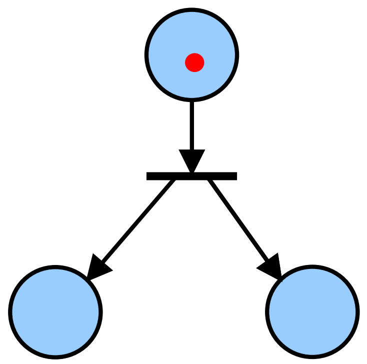
Finite Automata allow the representation of decisions, but no concurrency.

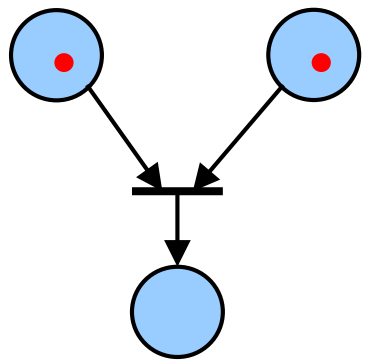
Petri nets support concurrency with intuitive notations:

Decision


decision / conflict

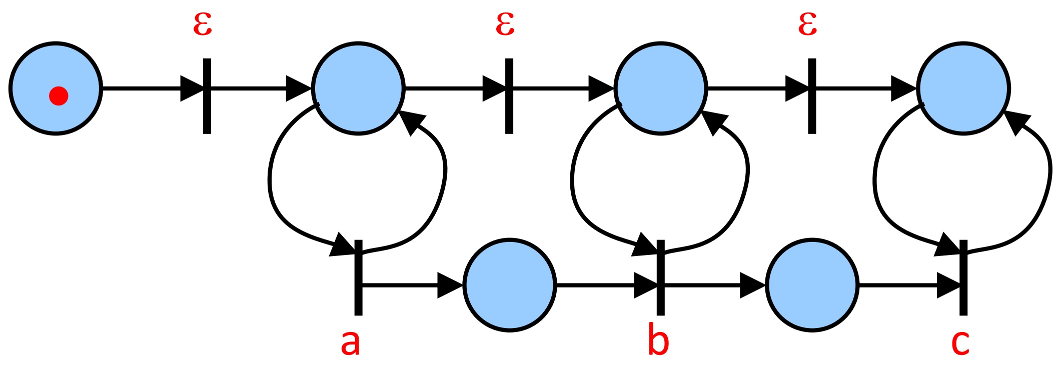
Concurrency


fork


join / synchronization

Petri Net Languages

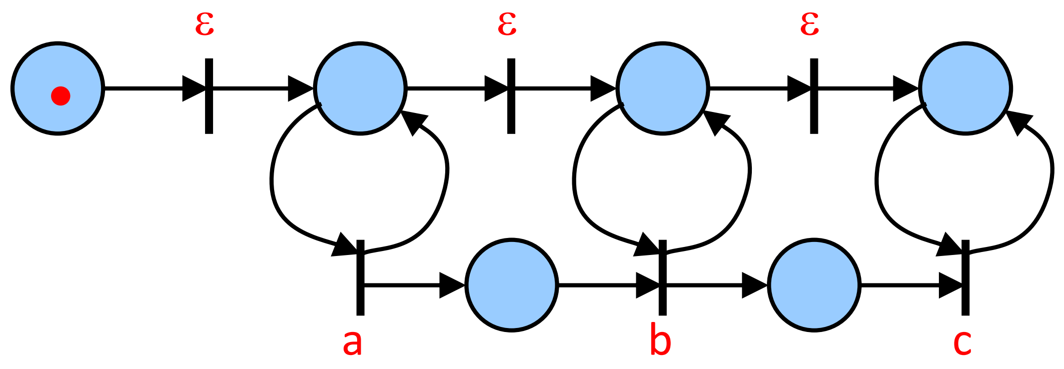
- Transitions are labeled with (not necessarily distinct) symbols.
- Final state is reached if no transition is activated.
- Any sequence of firing generates a string of symbols, i.e. a word of the language.



$L(M_0) = ?$

Petri Net Languages

- Transitions are labeled with (not necessarily distinct) symbols.
- Final state is reached if no transition is activated.
- Any sequence of firing generates a string of symbols, i.e. a word of the language.



▶ $L(M_0) = \{a^n\, b^m\, c^m \mid n \geq m \geq 0\}$

- Every finite-state machine can be modeled by a Petri net. ▶

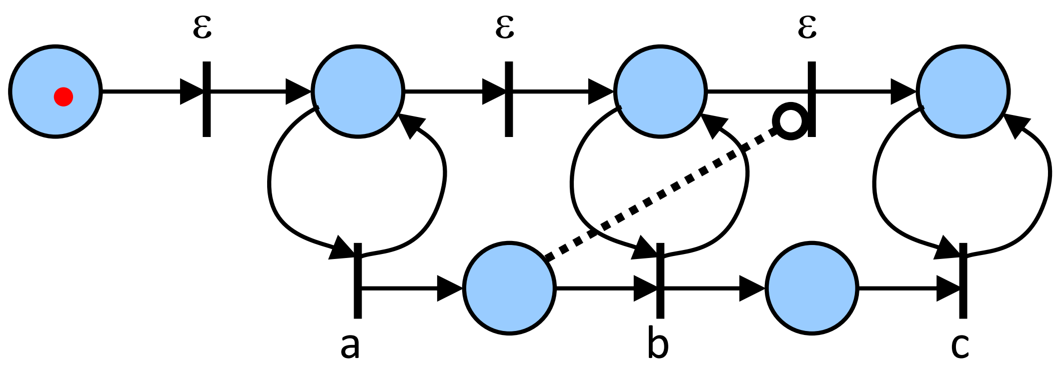
Every regular language is a Petri net language.

Not every Petri net language is regular.

Common Extensions

Timed Petri nets
See next week

Colored Petri nets	Tokens carry values (colors). A Petri net with finite number of colors can be transformed into a regular Petri net.
Continuous Petri nets	The number of tokens can be a real number (not only an integer). Cannot be transformed into a regular Petri net.
Inhibitor Arcs	Enable a transition if a place contains no tokens. Cannot be transformed to a regular Petri net



$L(M_0) = \{a^n\, b^n\, c^n \mid n \geq 0\ \}$

Definition

- Semantics
- Token game

Properties

- Safety
- Liveness

Analysis

- Coverability tree
- Incidence matrix

Behavioral Properties (1)

Reachability

A marking M_n is reachable from M_0 iff there exists a sequence of firings

$\{t_1, t_2, \dots t_n\}$ such that $M_n = M_0 \cdot t_1 \cdot t_2 \cdot \dots \cdot t_n$

K-Boundedness

A Petri net is K-bounded if the number of tokens in every place never exceeds K. The number of states is finite in this case.

Safety

1-Boundedness: Every node holds at most 1 token at any time.

Behavioral Properties (2)

Liveness

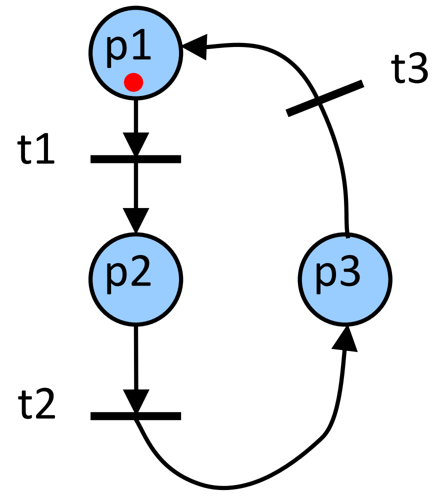
A transition t in a Petri net is

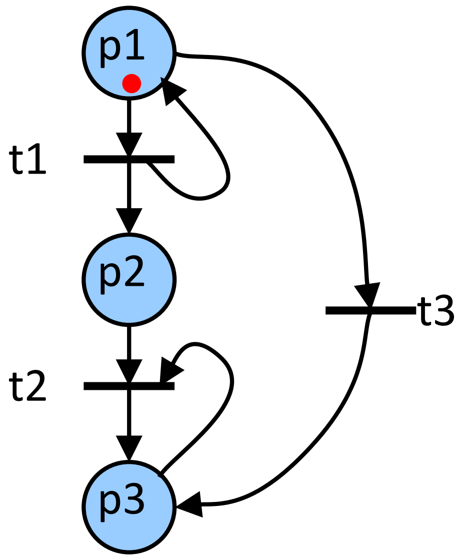
dead	iff t cannot be fired in any firing sequence,
L_1-live	iff t can be fired at least once in some firing sequence,
L_2-live	iff, $\forall$ k $\in$ $\mathbf{N}^+$, t can be fired at least k times in some firing sequence,
L_3-live	iff t appears infinitely often in some infinite firing sequence,
L_4-live	(live) iff t is L_1-live for every marking that is reachable from M_0 .

L_{j+1}-liveness implies L_j-liveness.

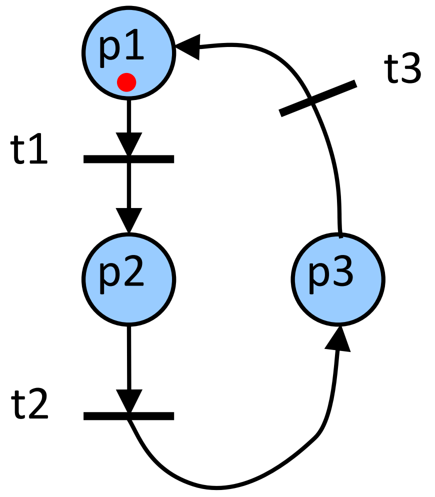
A Petri net is free of deadlocks iff there is no reachable marking from M_0 in which all transitions are dead.

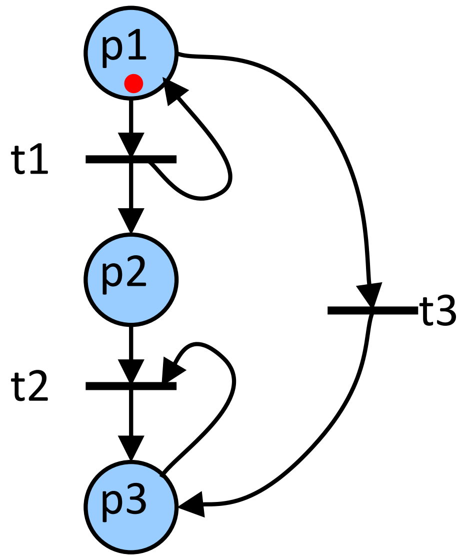
Liveness Examples





Liveness Examples





- All transitions are L4-live.
- The Petri net is free of deadlocks.

- t1 is L3-live.
- t2 is L2-live.
- t3 is L1-live.
- The Petri net is not free of deadlocks.

Definition

- Semantics
- Token game

Properties

- Safety
- Liveness

Analysis

- Coverability tree
- Incidence matrix

Analysis Methods

Coverability tree

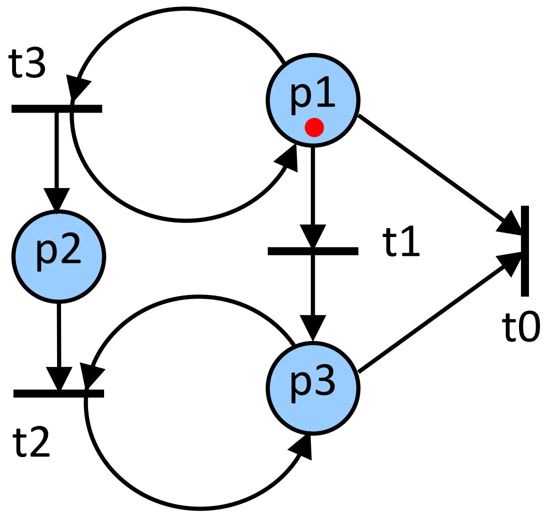
Enumeration of all reachable markings, limited to small nets if done by explicit enumeration.
Reachability analysis similar to that of finite automata can be done if the net is bounded.

Incidence Matrix

Describes the token-flow and state evolution by a set of linear equations.
This method allows to derive necessary but not sufficient conditions for reachability.

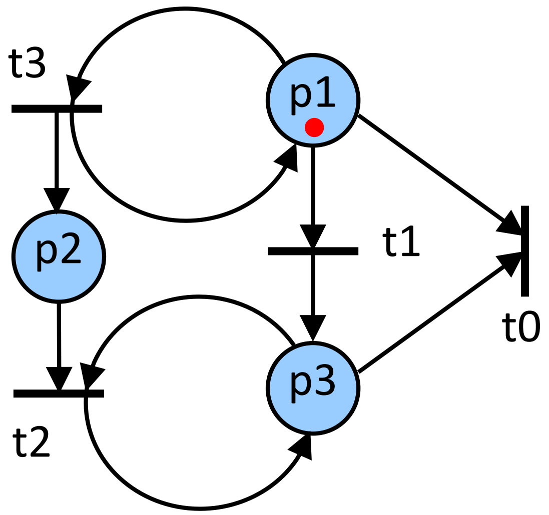
Coverability Tree

Question	What token distributions are reachable?
Problem	There might be infinitely many reachable markings, but we must avoid an infinite tree.
Solution	Introduce a special symbol ω to denote an arbitrary number of tokens.



Coverability Tree

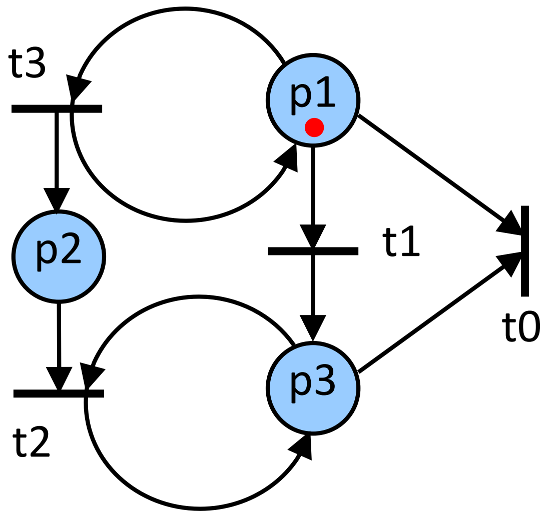
Question	What token distributions are reachable?
Problem	There might be infinitely many reachable markings, but we must avoid an infinite tree.
Solution	Introduce a special symbol ω to denote an arbitrary number of tokens.



$$\begin{tikzcd}
 & M_0 = [1\ 0\ 0] \arrow[dl, "t1"'] \arrow[dr, "t3"] & & \\
\begin{matrix} M_1 = [0\ 0\ 1] \\ \text{deadlock} \end{matrix} & & M_3 = [1\ \omega\ 0] \arrow[dl, "t1"'] \arrow[dr, "t3"] & \\
 & M_4 = [0\ \omega\ 1] \arrow[d, "t2"] & & \begin{matrix} M_6 = [1\ \omega\ 0] \\ \text{old} \end{matrix} \\
 & M_5 = [0\ \omega\ 1]\ \ \text{old} & &
\end{tikzcd}$$

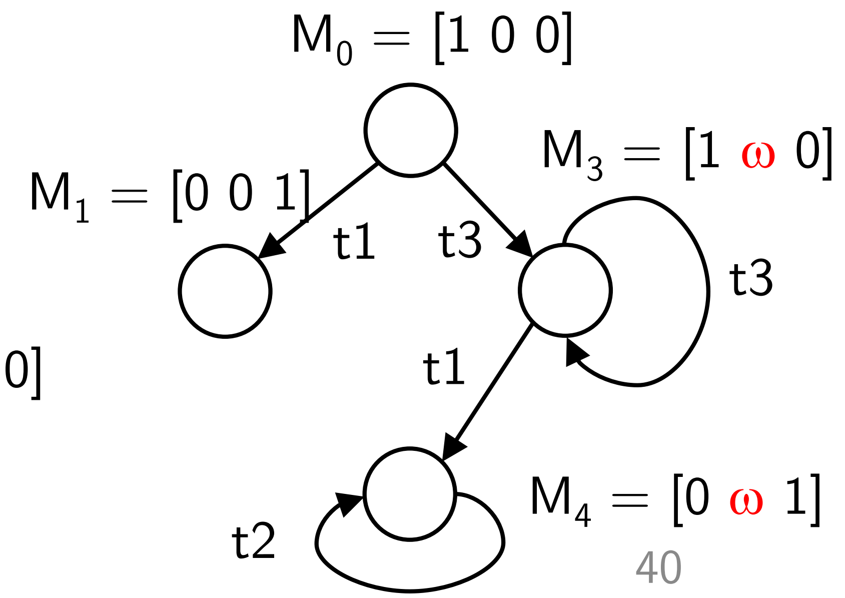
Coverability Tree

Question	What token distributions are reachable?
Problem	There might be infinitely many reachable markings, but we must avoid an infinite tree.
Solution	Introduce a special symbol ω to denote an arbitrary number of tokens.



$$\begin{tikzcd}
 & M_0 = [1\ 0\ 0] \arrow[dl, "t1"'] \arrow[dr, "t3"] & & \\
M_1 = [0\ 0\ 1] \text{ deadlock} & & M_3 = [1\ \omega\ 0] \arrow[dl, "t1"'] \arrow[dr, "t3"] & \\
 & M_4 = [0\ \omega\ 1] \arrow[d, "t2"] & & M_6 = [1\ \omega\ 0] \text{ old} \\
 & M_5 = [0\ \omega\ 1] \text{ old} & &
\end{tikzcd}$$

tree



graph

Coverability Tree – Algorithm

Special symbol ω, similar to ∞: $\forall n \in N$: $\omega > n$; $\omega = \omega \pm n$; $\omega \geq \omega$

Label initial marking M_0 as root and tag it as new

while new tags exist, pick one, say M

- Remove tag new from M;
- If M is identical to an already existing marking, tag it as old; continue;
- If no transitions are enabled at M, tag it as deadlock; continue;
- For each enabled transition t at M do
  - Obtain marking $M' = M \cdot t$
  - If there exists a marking M'' on the path from the root to M s.t. $M'(p) \geq M''(p)$ for each place p and $M' \neq M''$, replace $M'(p)$ with ω for p where $M'(p) > M''(p)$.
  - Introduce M' as a node, draw an arc with label t from M to M' and tag M' new.

Results from the Coverability Tree T

- The net is bounded iff ω does not appear in any node label of T.
- The net is safe iff only ‘0’ and ‘1’ appear in the node labels of T.
- A transition t is dead iff it does not appear as an arc in T.

- If M is reachable from M_0, then there exists a node M' s.t. $M \leq M'$.
  This is a necessary, but not sufficient condition for reachability.

- For bounded Petri nets, the coverability tree T does not contain ω
  is also called reachability tree, as all reachable markings are contained in it.

Definition

- Semantics
- Token game

Properties

- Safety
- Liveness

Analysis

- Coverability tree
- Incidence matrix

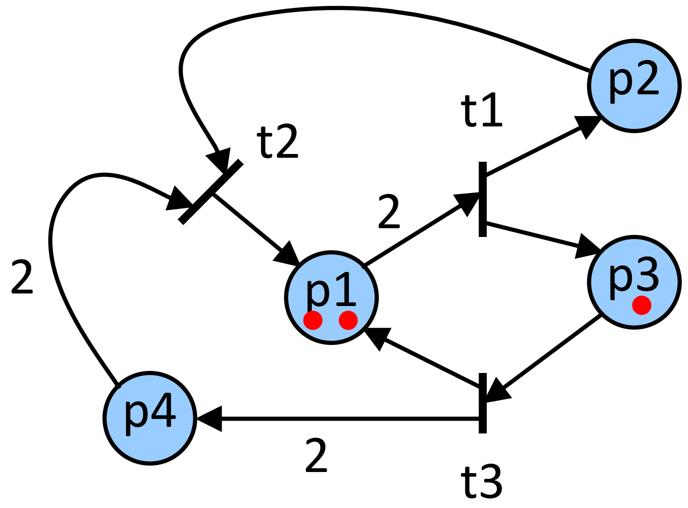
Incidence Matrix

Approach

Describe a Petri net with a set of linear equations

- A marking M is written as a m × 1 column vector.
- The incidence matrix A describes the token-flow for a Petri net with n transitions and m places in a m × n matrix.

A_{ij} corresponds to the "gain" of tokens at place p_i when transition t_j fires.



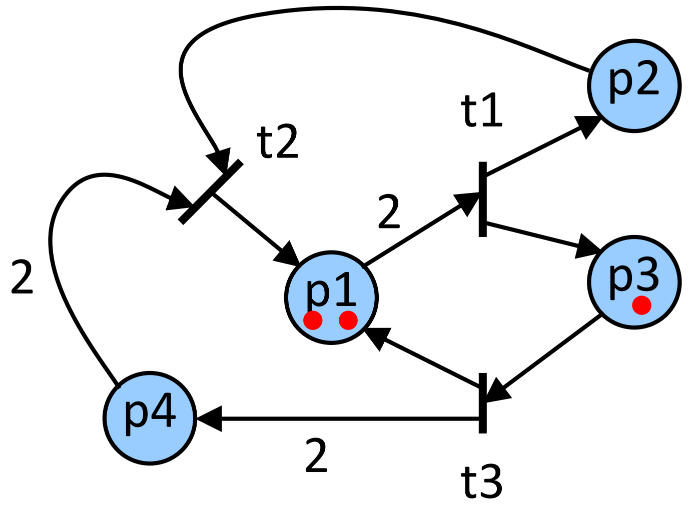
Incidence Matrix

Approach

Describe a Petri net with a set of linear equations

- A marking M is written as a m × 1 column vector.
- The incidence matrix A describes the token-flow for a Petri net with n transitions and m places in a m × n matrix.

A_{ij} corresponds to the "gain" of tokens at place p_i when transition t_j fires.



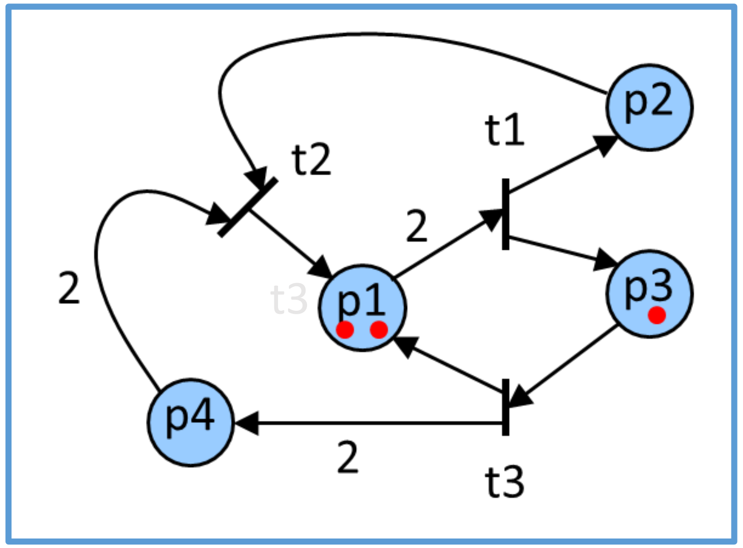
$A_{ij} = W(t_j , p_i) - W(p_i , t_j)$
with $W(p,t) = 0$ or $W(t, p)=0$
when the corresponding edges do not exist

$$M_0 = \begin{bmatrix} 2 \\ 0 \\ 1 \\ 0 \end{bmatrix} \qquad A = \begin{bmatrix} -2 & 1 & 1 \\ 1 & -1 & 0 \\ 1 & 0 & -1 \\ 0 & -2 & 2 \end{bmatrix}$$

State Equation

- The firing vector u describes the firing of a transition t. If transition ti fires, then ui consists of all '0', except for the i-th row, where it has a '1':

$$u_1 = \begin{bmatrix}1\\0\\0\end{bmatrix} \quad u_2 = \begin{bmatrix}0\\1\\0\end{bmatrix} \quad u_3 = \begin{bmatrix}0\\0\\1\end{bmatrix}$$



- A state transition from M to M' due to firing ti is written as

$$M' = \delta(M, ti) = M + A \cdot ui$$

- For example, M1 is obtained from M0 by firing t3:

$$\begin{bmatrix}3\\0\\0\\2\end{bmatrix} = \begin{bmatrix}2\\0\\1\\0\end{bmatrix} + \begin{bmatrix}-2 & 1 & 1\\1 & -1 & 0\\1 & 0 & -1\\0 & -2 & 2\end{bmatrix}\begin{bmatrix}0\\0\\1\end{bmatrix}$$

State Equation: Reachability

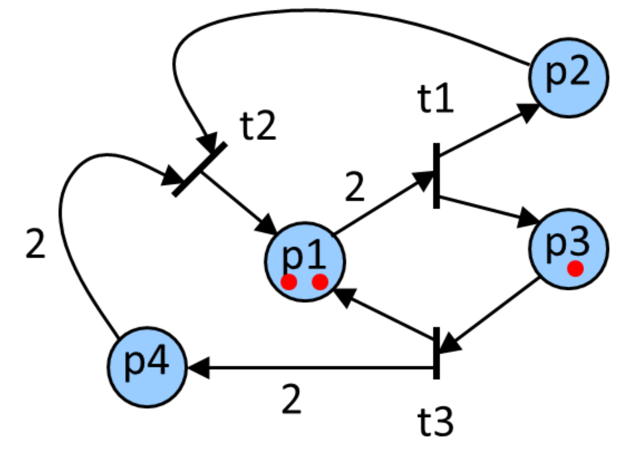
- A marking M_k is reachable from M_0 if there is a sequence σ of k transitions $\{t_\sigma[1], t_\sigma[2], \ldots, t_\sigma[k]\}$ such that $M_k = M_0 \cdot t_\sigma[1] \cdot t_\sigma[2] \cdot \ldots \cdot t_\sigma[k]$.
- Expressed with the incidence matrix:

$$M_k = M_0 + A \sum_{i=1}^{k} u_{\sigma[i]} \qquad (1)$$

which can be rewritten as

$$M_k - M_0 = \Delta M = Ax \qquad (2)$$

- If M_k is reachable from M_0, eq. (2) must have a solution where all components of x are non-negative integers.

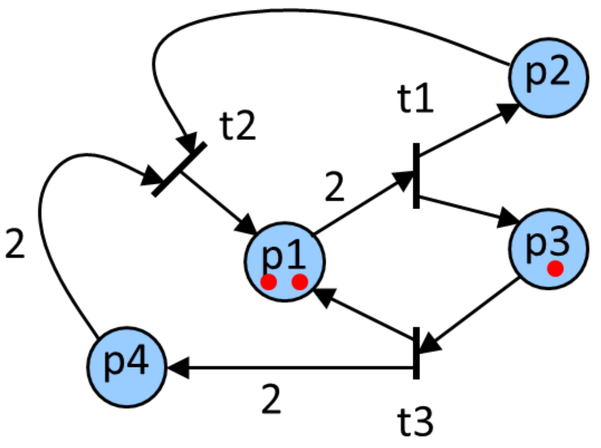


$$M_0 = \begin{bmatrix} 2 \\ 0 \\ 1 \\ 0 \end{bmatrix} \quad A = \begin{bmatrix} -2 & 1 & 1 \\ 1 & -1 & 0 \\ 1 & 0 & -1 \\ 0 & -2 & 2 \end{bmatrix}$$

This is a necessary but not sufficient condition for reachability.

Reachability - Example

- Is $M_k = \begin{bmatrix} 3 \\ 0 \\ 0 \\ 2 \end{bmatrix}$ reachable?



$$M_0 = \begin{bmatrix} 2 \\ 0 \\ 1 \\ 0 \end{bmatrix} \quad A = \begin{bmatrix} -2 & 1 & 1 \\ 1 & -1 & 0 \\ 1 & 0 & -1 \\ 0 & -2 & 2 \end{bmatrix}$$

- Is $M_k = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 2 \end{bmatrix}$ reachable?

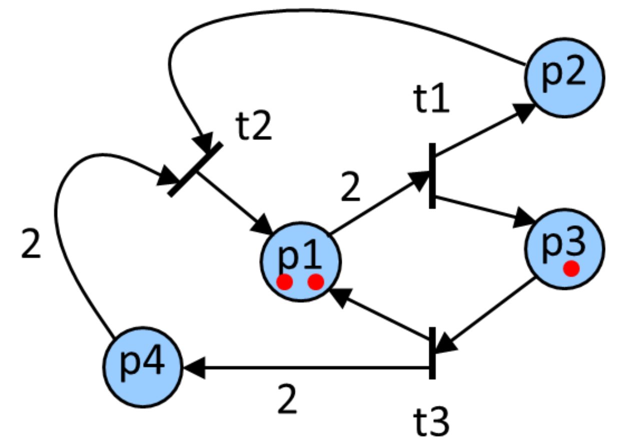
Reachability - Example

- Is $M_k = \begin{bmatrix} 3 \\ 0 \\ 0 \\ 2 \end{bmatrix}$ reachable? Possibly yes.

$x = \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix}$ is a solution to $M_k - M_0 = \Delta M = Ax$

with $\Delta M = \begin{bmatrix} 1 \\ 0 \\ -1 \\ 2 \end{bmatrix}$

▶ It is actually reachable,
e.g., with the sequence $\{t_1, t_3, t_3, t_2\}$.



$M_0 = \begin{bmatrix} 2 \\ 0 \\ 1 \\ 0 \end{bmatrix}$ $A = \begin{bmatrix} -2 & 1 & 1 \\ 1 & -1 & 0 \\ 1 & 0 & -1 \\ 0 & -2 & 2 \end{bmatrix}$

- Is $M_k = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 2 \end{bmatrix}$ reachable? No.

There are no solutions to $M_k - M_0 = \Delta M = Ax$ with $\Delta M = \begin{bmatrix} -1 \\ 0 \\ -1 \\ 2 \end{bmatrix}$

Invariants

AG p

From the incidence matrix, one can derive some system' invariants.

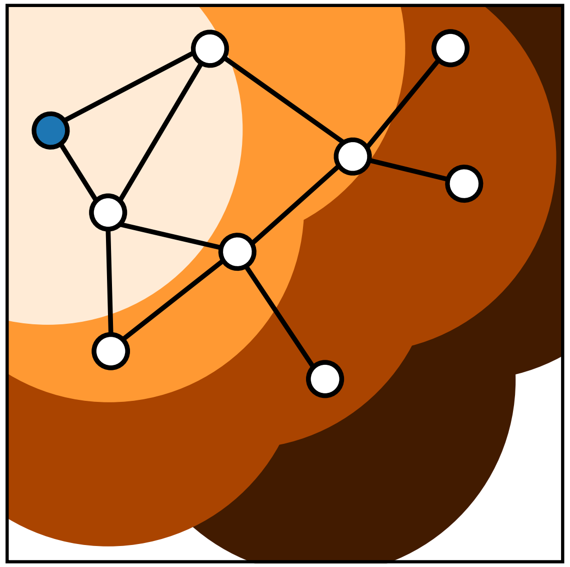
- A linear combinaison of transitions
  that does not change the net's marking
- A linear combinaison of places' marking
  that sums up to the same amount of tokens

Definition	- Semantics - Token game
Properties	- Safety - Liveness
Analysis	- Coverability tree - Incidence matrix

Your turn to practice!
after the break

1. Familiarise yourself with the token game
2. Use Petri Nets to model simple computation structures (mutual exclusion)
3. Analyse Petri Nets with using coverability graphs and incidence matrices

See you next week!
in Discrete Event Systems



Romain Jacob

www.romainjacob.net

ETH Zurich (D-ITET)

December 9, 2021

Most materials from Lothar Thiele