# Industrial Automation

## (Automação de Processos Industriais)

## Analysis of Discrete Event Systems

http://users.isr.ist.utl.pt/~jag/courses/api1617/api1617.html

Prof. Paulo Jorge Oliveira, original slides
Prof. José Gaspar, rev. 2016/2017

# Syllabus:

**Chap. 6 – Discrete Event Systems [2 weeks]**

**…**

**Chap. 7 – Analysis of Discrete Event Systems [2 weeks]**

Properties of DESs.

Methodologies to analyze DESs:

* The Reachability tree.
* The Method of Matrix Equations.

**…**

**Chap. 8 – DESs and Industrial Automation [1 week]**

# Some pointers to Discrete Event Systems

History: http://prosys.changwon.ac.kr/docs/petrinet/1.htm

Tutorial: http://vita.bu.edu/cgc/MIDEDS/
http://www.daimi.au.dk/PetriNets/

Analyzers, and Simulators:
http://www.ppgia.pucpr.br/~maziero/petri/arp.html (in Portuguese)
http://wiki.daimi.au.dk:8000/cpntools/cpntools.wiki
http://www.informatik.hu-berlin.de/top/pnk/download.html

Bibliography:

* Cassandras, Christos G., **"Discrete Event Systems - Modeling and PerformanceAnalysis"**, Aksen Associates, 1993.
* Peterson, James L., **"Petri Net Theory and the Modeling of Systems"**, Prentice-Hall,1981
* **Petri Nets and GRAFCET: Tools for Modelling Discrete Event Systems** R. DAVID, H. ALLA, New York : PRENTICE HALL Editions, 1992

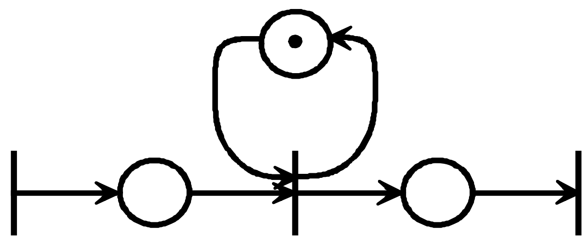
# Properties of Discrete Event Systems

## 1. Reachability

Given a Petri net C=($P, T, I, O, \mu_0$) with initial marking $\mu_0$ , the **set of all markings** that can be obtained starting from μ is the **Reachable Set**, $R$(C, μ).

Note: **in general** $R$(C, μ) **is infinite!**

How to describe and compute $R$(C, μ)?



**Reachability problem:** Given a Petri net C with initial marking $\mu_0$ , does the marking μ' belong to the set of all markings that can be obtained, i.e. μ' ∈ $R$(C, μ)?

**Property usage:** State μ **belongs / does not belong** to R(C, $\mu_0$).

usage2: State μ **is / is not reachable**.

usage3: Net C has a **finite / infinite** Reachable Set.

# Properties of Discrete Event Systems

## 2. Coverability

Given a Petri net C=($P, T, I, O, \mu_0$) with initial marking $\mu_0$ and states μ, μ' ∈ $R$(C, $\mu_0$) then **μ' is covered by μ** if **$\mu'(i) \leq \mu(i)$**, for all places $p_i \in P$.
Equivalently, one says **μ covers μ'.**

**Property usages:**

State μ' **covers / does not cover** state μ.
State μ **is / is not covered** by state μ'.
State μ **is / is not coverable** by other reachable states.

Note, μ' not covered by μ does not imply μ' covers μ.

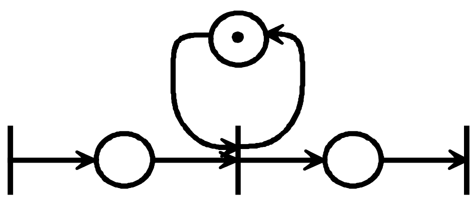
*Is it possible to use this property to help on the search for the reachable set? Yes! Details after some few slides.*

# Properties of Discrete Event Systems

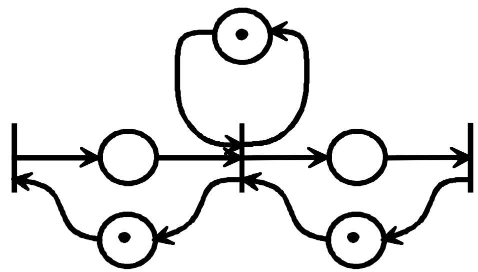
## 3. Safeness

A place $p_i \in P$ of the Petri net $C=(P, T, I, O, \mu_0)$ **is safe** if
for all $\mu' \in R(C, \mu_0)$: $\mu_i' \leq 1$.

A Petri net **is safe** if all its places are safe.



Petri net not safe



Petri net safe

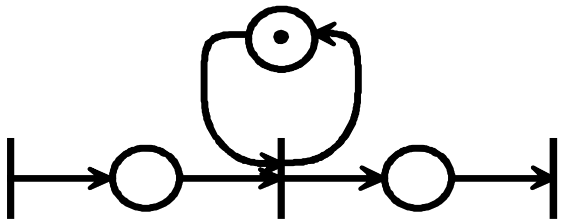
**Property usage:** Place $p_i$ / Net C **is / is not** **safe**.

# Properties of Discrete Event Systems

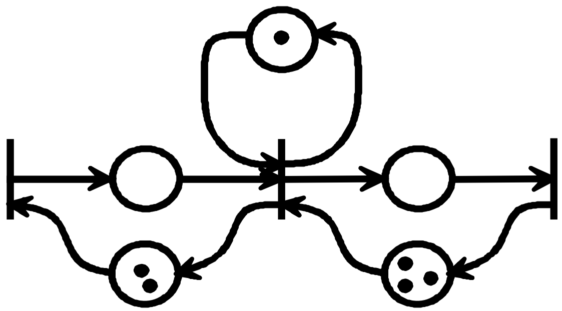
## 4. Boundedness

Given a Petri net C=($P, T, I, O, \mu_0$), a **place $p_i$** $\in P$ is **k-bounded** if $\mu_i' \leq k$ for all $\mu'=(\mu_1', \ldots, \mu_i', \ldots, \mu_N') \in R(C, \mu_0)$.

A Petri net is **k-bounded** if all places are k-bounded.



Petri net not bounded



Petri net 3-bounded

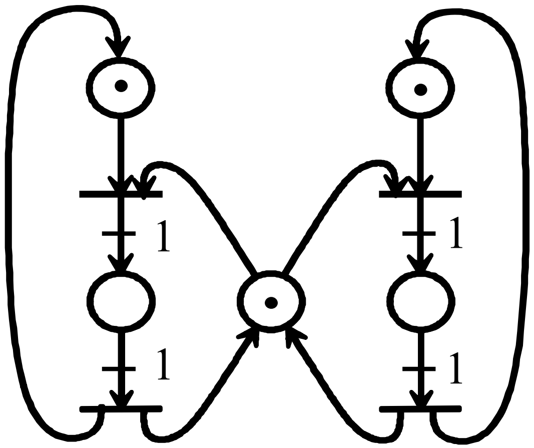
**Property usage:** Place $p_i$ / Net C **is / is not** **k-bounded**.

# Properties of Discrete Event Systems

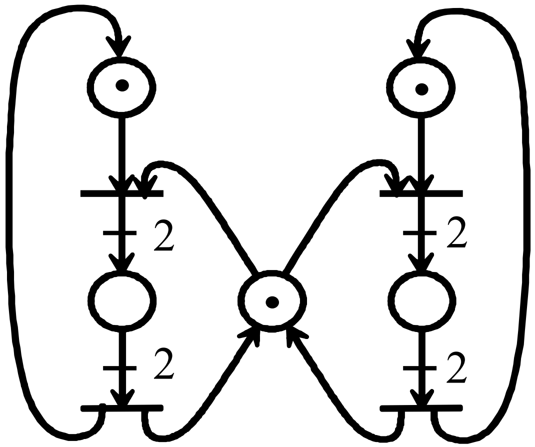
## 5. Conservation

A Petri net C=($P$, $T$, $I$, $O$, $\mu_0$) is **strictly conservative** if for all $\mu' \in R(C, \mu)$

$$\sum_{p_i \in P} \mu'(p_i) = \sum_{p_i \in P} \mu(p_i)$$




Petri net **not strictly conservative**




Petri net **strictly conservative**

**Property usage:** Net C **is / is not (strictly) conservative**.

# Properties of Discrete Event Systems

## 6. Liveness

A transition $t_j$ is live of

**Level 0 - if it can never be fired (transition is *Dead*).**

**Level 1 - if it is potentially firable** an upper-bounded number of times, **i.e. if there exists** $\mu' \in R(C, \mu)$ **such that** $t_j$ **is enabled in** $\mu'$.

**Level 2 - if for every integer *n*, there exists a firing sequence such that** $t_j$ **occurs *n* times.**

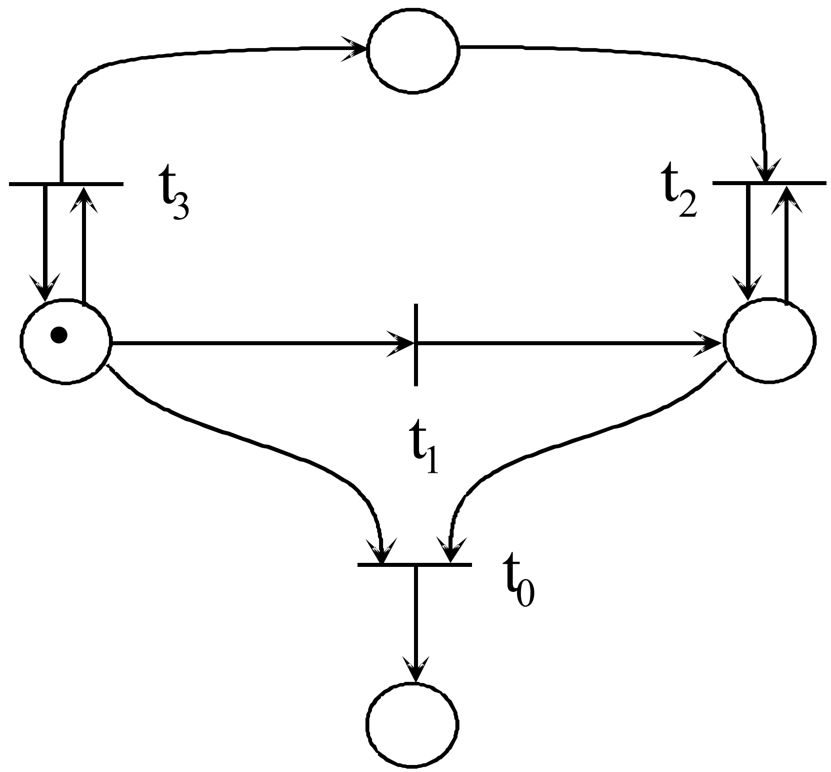
**Level 3 - if there exists an infinite firing sequence such that** $t_j$ **occurs infinite times.**

**Level 4 - if for each** $\mu' \in R(C, \mu)$ **there exist a sequence** $\sigma$ **such that the transition** $t_j$ **is enabled (transition is *Live*).**

# Properties of Discrete Event Systems

## Example of liveness of transitions

- $t_0$ is of level 0.
- $t_1$ is of level 1.
- $t_2$ is of level 2.
- $t_3$ is of level 3.
- *this net does not have level 4 transitions.*



# Properties of Discrete Event Systems

## Reachability problem

Given a Petri net $C=(P, T, I, O, \mu_0)$ with initial marking $\mu_0$ and a marking $\mu'$, is $\mu' \in R(C, \mu_0)$ reachable?

**Analysis methods:**

0- Brute force...

1- Reachability tree

2- Matrix equations

# Analysis Methods, 1- Reachability Tree

## Reachability Tree - construction *[Peterson81, §4.2.1]*

A reachability tree is a tree of reachable markings.
**Tree nodes are states**. The root node is the initial state (marking).

It is constituted by three types of nodes:

- **Terminal** no state changes after a terminal state
- **Interior** state can change after
- **Duplicated** state already found in the tree

The infinity marking symbol (ω) is introduced whenever a marking covers other. This symbol allows obtaining finite trees.

*The reachability tree is useful to study properties previously introduced.*
*Some examples later.*

# Analysis Methods

## Reachability Tree - construction *[Peterson81, §4.2.1]*

Algebra of the infinity symbol ($\omega$):

For every positive integer *a* the following relations are verified:

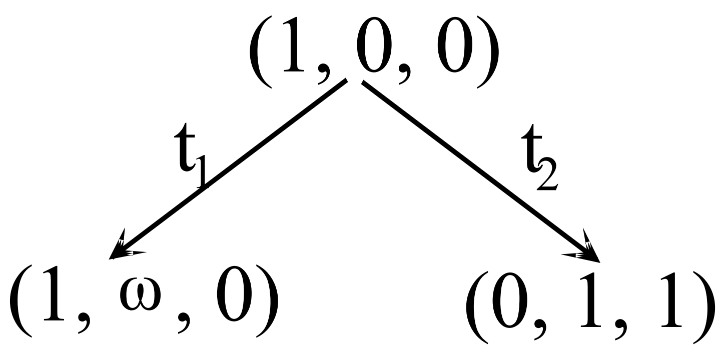
1. $\omega + a = \omega$
2. $\omega - a = \omega$
3. $a < \omega$
4. $\omega \leq \omega$

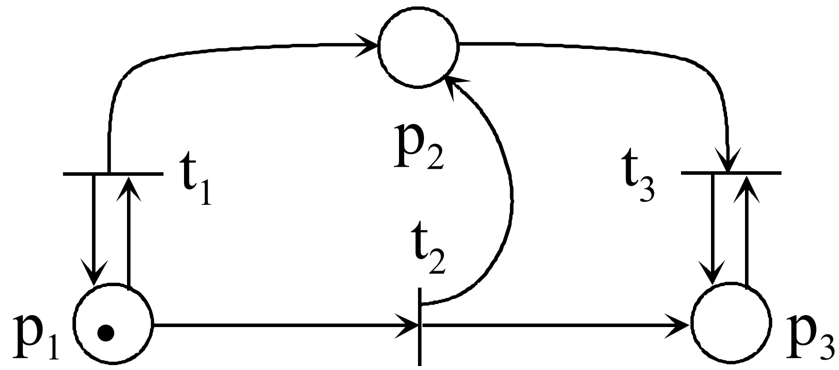
## Reachability Tree and Deadlocks

**Theorem** - If there exist terminal nodes in the reachability tree then the corresponding Petri net has ***deadlocks***.

# Analysis Methods

## Example of reachability tree:

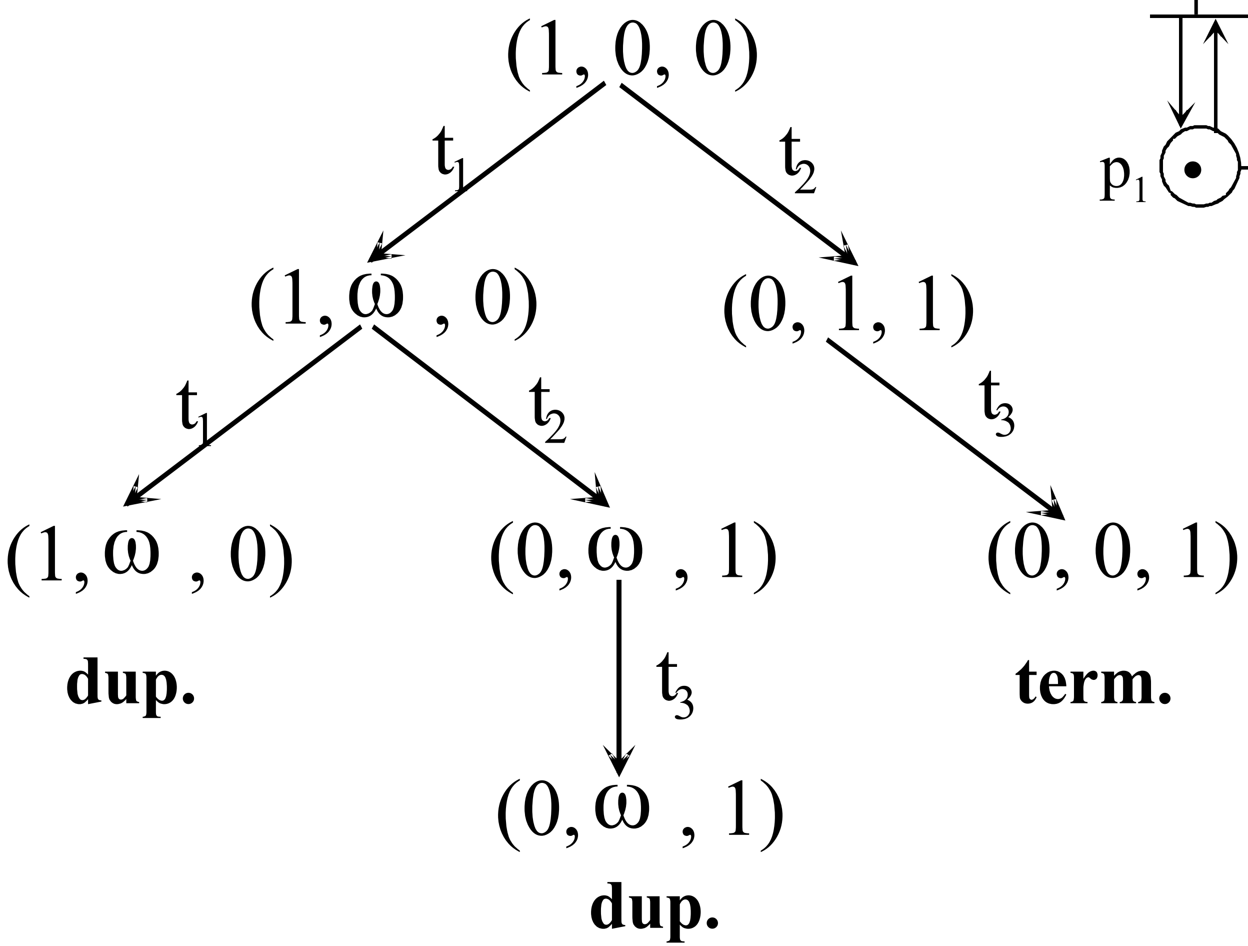


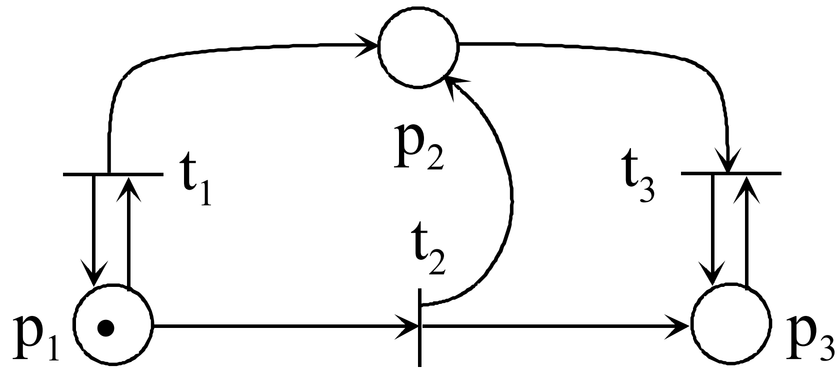




After t1 one obtains **(1, 0, 0)** which **is covered by (1, 1, 0).** Hence one introduces the **infinity symbol, ω** and writes the state as **(1, ω, 0)**.

# Analysis Methods

## Example of reachability tree:



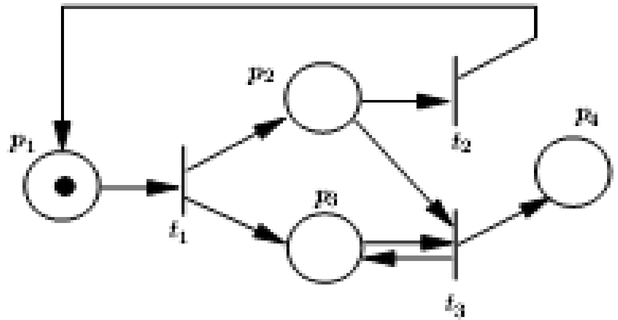

**We can conclude immediately that there are**

**DEADLOCKS!**

# Analysis Methods

## Reachability Tree vs Coverability Tree

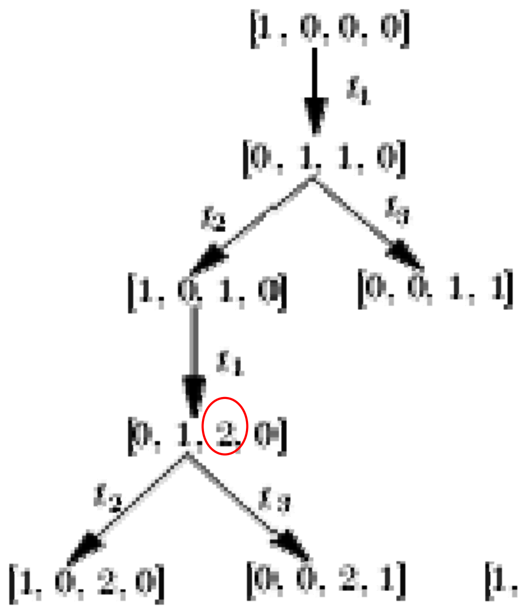
*[Cassandras08, §4.4.2]*

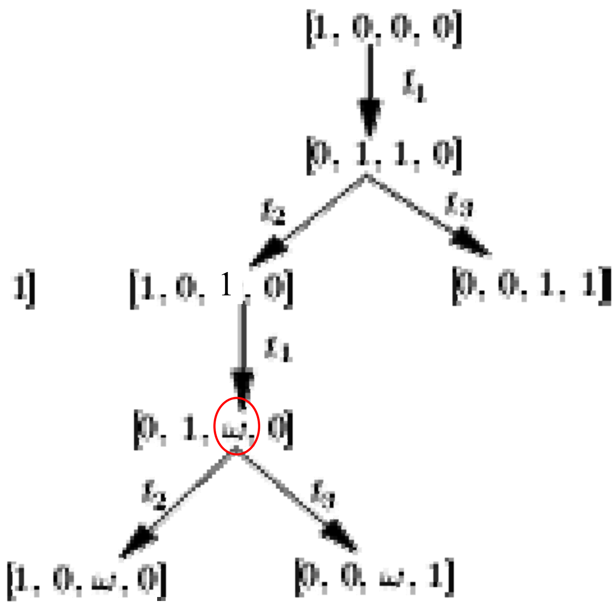



Considering a Petri net the **reachability tree** is "a tree whose root node is (...), then examine all transitions that can fire from this state, define new nodes in the tree, and repeat until all possible reachable states are identified."

"The reachability tree (...) may be infinite. A finite representation (...) is possible, but at the expense of losing some information. The finite version of an infinite reachability tree will be called a **coverability tree**."

Reachability tree , Coverability tree




*(In this course we use Peterson's terminology, i.e. "reachability tree" in both cases)*

# Example1: simple Petri net, properties?

$(P, T, A, w, x_0)$

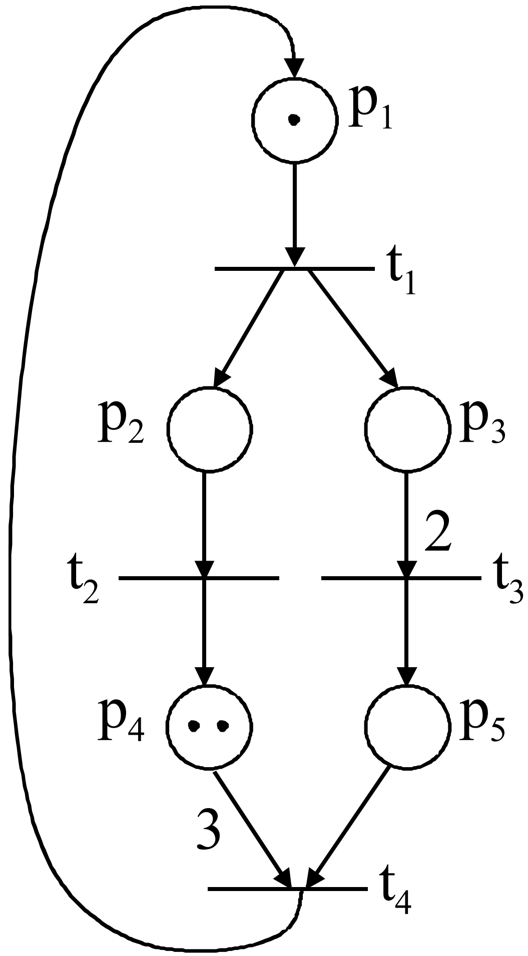
$P=\{p_1, p_2, p_3, p_4, p_5\}$

$T=\{t_1, t_2, t_3, t_4\}$

$A=\{(p_1, t_1), (t_1, p_2), (t_1, p_3), (p_2, t_2), (p_3, t_3),$
$(t_2, p_4), (t_3, p_5), (p_4, t_4), (p_5, t_4), (t_4, p_1)\}$

$w(p_1, t_1)=1, w(t_1, p_2)=1, w(t_1, p_3)=1, w(p_2, t_2)=1$
$w(p_3, t_3)=2, w(t_2, p_4)=1, w(t_3, p_5)=1, w(p_4, t_4)=3$
$w(p_5, t_4)=1, w(t_4, p_1)=1$

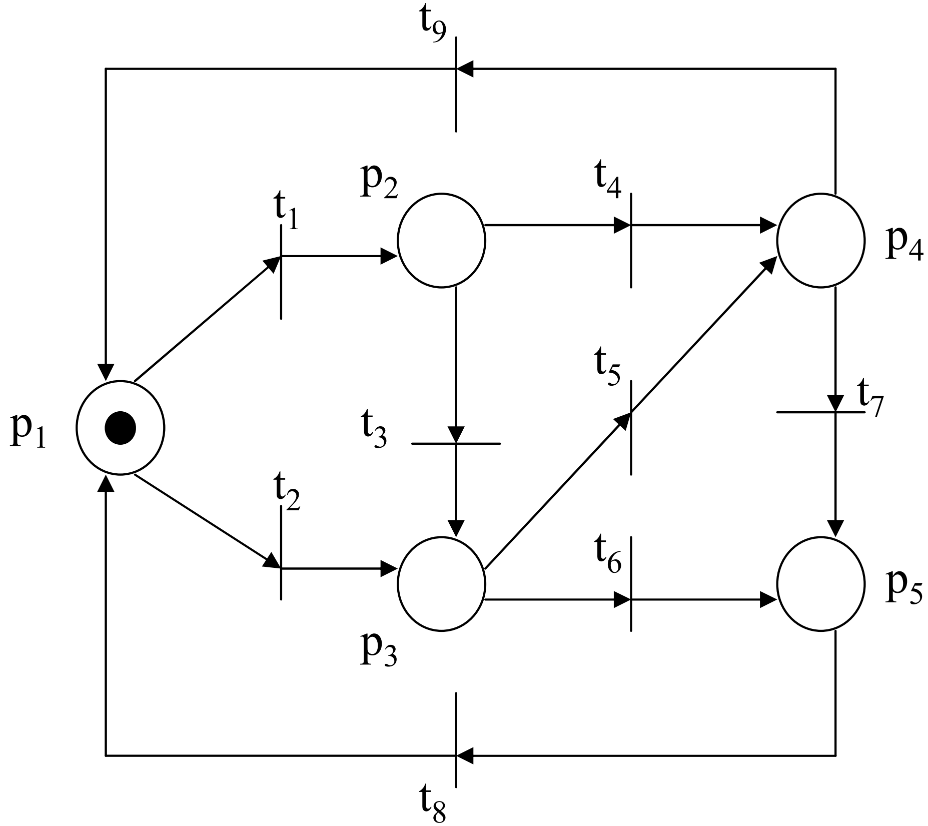
$x_0 = \{1, 0, 0, 2, 0\}$



# Example2: simple automation system modeled using PNs, properties?

An automatic soda selling machine accepts
50c and $1 coins and
sells 2 types of products:
SODA A, that costs $1.50 and
SODA B, that costs $2.00.

Assume that the money return operation is omitted.




$p_1$: machine with $0.00;
$t_1$: coin of 50 c introduced;
$t_8$: SODA B sold.

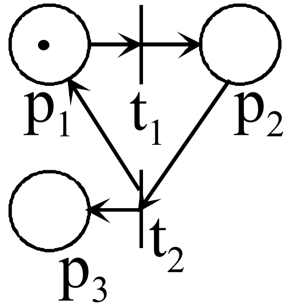
# Example3:

**(counter-example)**

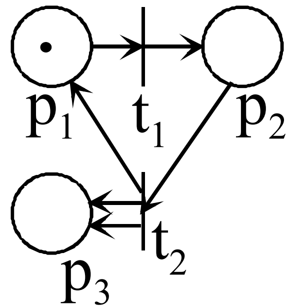
**Different reachable sets**

but the

**same reachability tree**




(1, 0, 0)
↓ $t_1$
(0, 1, 0)
↓ $t_2$
(1, 0, ω)
↓ $t_1$
(0, 1, ω)
↓ $t_2$
(1, 0, ω)
dup.




(1, 0, 0)
↓ $t_1$
(0, 1, 0)
↓ $t_2$
(1, 0, ω)
↓ $t_1$
(0, 1, ω)
↓ $t_2$
(1, 0, ω)
dup.

## Decidability Problem:

Can one reach (1,0,1)? **Yes** in one net, **No** in the other one. Simple to answer in this net, but undecidable in general due to the symbol ω.

*The reachability tree does not ensure decidability of state reachability.*

# Analysis Methods, 2- MME

## Method of the Matrix Equations (MME) of State Evolution

The dynamics of the Petri net state can be written in compact form as:

$$\mu(k+1) = \mu(k) + Dq(k)$$

*This methodology can also be used to study the other properties previously introduced.*
*Requires some thought... ;)*

where:

- $\mu(k+1)$ - marking to be reached
- $\mu(k)$ - initial marking
- $q(k)$ - firing vector (transitions)
- $D$ - incidence matrix. Accounts the balance of tokens, giving the transitions fired.

# Analysis Methods

## How to build the Incidence Matrix, D ?

For a Petri net with *n* places and *m* transitions

$$\mu \in N_0^{\ n}$$

$$q \in N_0^{\ m}$$

$$\boxed{D = D^+ - D^-}\ , \quad D \in Z^{n\times m}, \quad D^+ \in N_0^{n\times m}, \quad D^- \in N_0^{n\times m}$$

The enabling firing rule is $\boxed{\mu \geq D^- q}$

Can also be written in compact form as the inequality $\mu + Dq \geq 0$, interpreted element-by-element.

*Note: unless otherwise stated in this course all vector and matrix inequalities are read element-by-element.*

# Analysis Methods

## Properties that can be studied immediately with the Method of Matrix Equations:

- Reachability

> **Theorem - *No Reachability Sufficient Condition*** – if the problem of finding the transition firing vector that drives the state of a Petri net from μ to state μ’ has no solution, resorting to the method of matrix equations, then the problem of reachability of μ’ does not have solution.

- Conservation – the firing vector is a by-product of the MME.

- Temporal invariance – cycles of operation can be found.

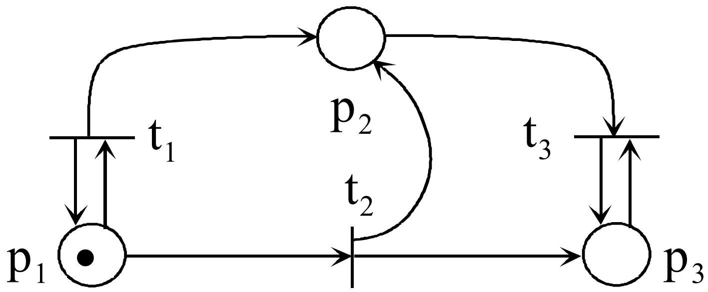
# Analysis Methods

## 1. Reachability

**Reachability problem:** Given a Petri net C with initial marking $\mu_0$, does the marking μ' belong to the set of all markings that can be obtained, i.e. $\mu' \in R(C, \mu)$?

**Example using the method of matrix equations** $\mu(k+1) = \mu(k) + Dq(k)$

**Given** the net:




$$D = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 1 & -1 \\ 0 & 1 & 0 \end{bmatrix} \quad \mu(k) = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

**Problem**:
is μ(k+1) reachable?
e.g.

$$\mu(k+1) = \begin{bmatrix} 1 \\ 3 \\ 0 \end{bmatrix},$$

**Solution**, find q(k):
*(note using $\sigma_{ti}$ to avoid using $q_i$ and confusing with q(i); this is to drop soon)*

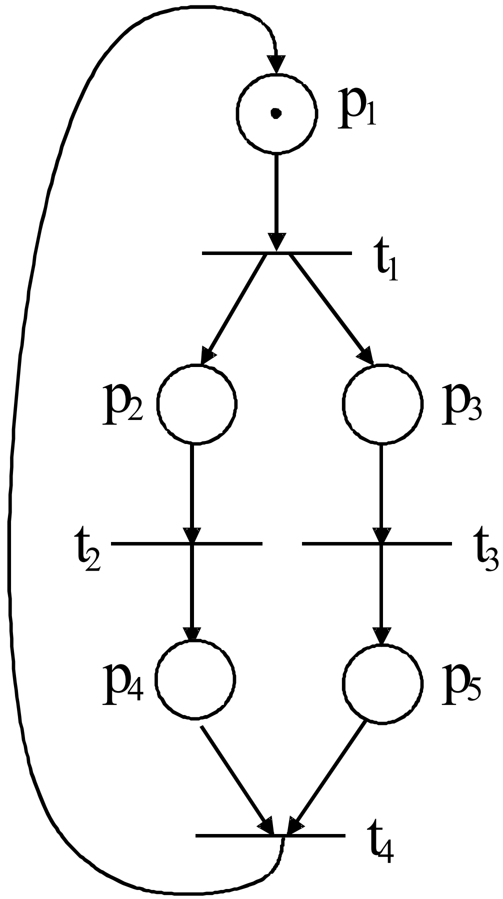
$$q(k) = \begin{bmatrix} \sigma_{t1} \\ \sigma_{t2} \\ \sigma_{t3} \end{bmatrix} \qquad \begin{cases} 1 = 1 - \sigma_{t2} \\ 3 = \sigma_{t1} + \sigma_{t2} - \sigma_{t3} \\ 0 = \sigma_{t2} \end{cases} \qquad \begin{cases} \sigma_{t2} = 0 \\ \sigma_{t1} - \sigma_{t3} = 3 \end{cases}$$

*Verify!*

*∃ q such that Dq(k)= μ(k+1)-μ(k) is a **necessary** but **not sufficient** condition.*

## Example of a Petri net

# 2. Conservation




To maintain the (weighted) number of tokens one writes:

$$w^T \mu' = w^T \mu + w^T Dq$$

and therefore:

$$\boxed{w^T D = 0}$$

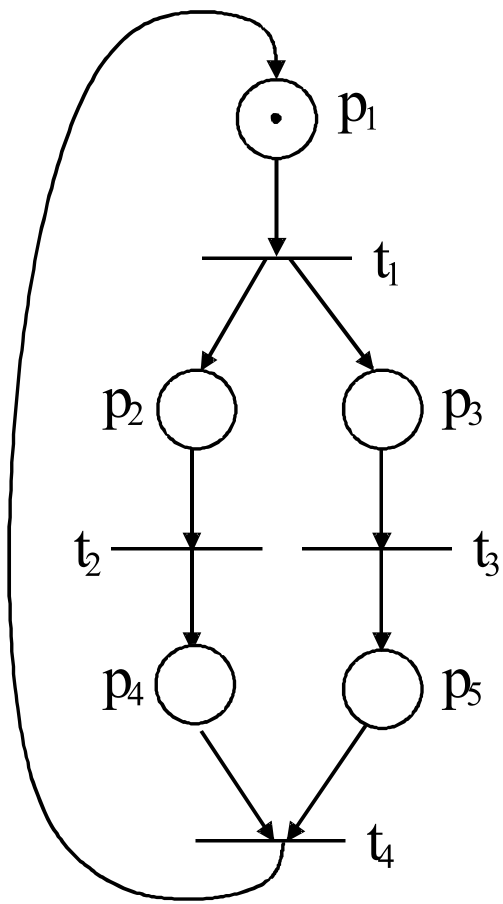
*∃x>0 is a **necessary and sufficient** condition*

$$D = \begin{bmatrix} -1 & 0 & 0 & 1 \\ 1 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & -1 \end{bmatrix} \quad \begin{cases} -w_1 + w_2 + w_3 = 0 \\ -w_2 + w_4 = 0 \\ -w_3 + w_5 = 0 \\ w_1 - w_4 - w_5 = 0 \end{cases} \quad \begin{cases} w_1 = w_2 + w_3 \\ w_2 = w_4 \\ w_3 = w_5 \\ - \end{cases}$$

This example has a solution in the form of an undetermined system of equations, where we can choose:

$$w^T = [2 \quad 1 \quad 1 \quad 1 \quad 1].$$

## Example of a Petri net




## 3. Temporal invariance

To determine the transition firing vectors that make the Petri net return to the same state(s):

$$Dq = 0$$

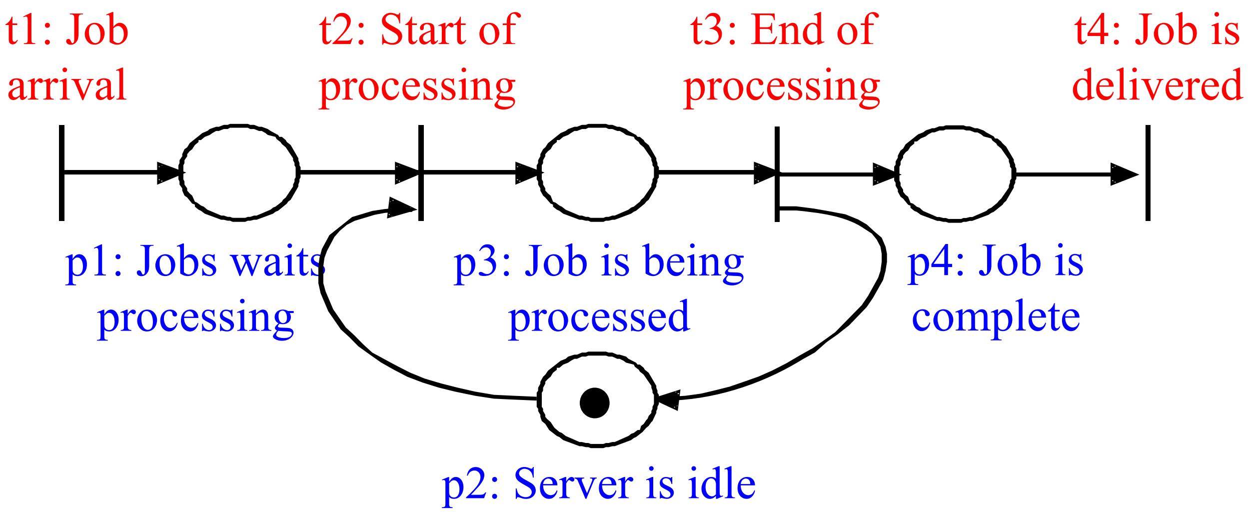
*∃ q is a necessary (not sufficient) condition*

$$D = \begin{bmatrix} -1 & 0 & 0 & 1 \\ 1 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & -1 \end{bmatrix}, \quad q = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} \quad \begin{cases} -q_1 + q_4 = 0 \\ q_1 - q_2 = 0 \\ q_1 - q_3 = 0 \\ q_2 - q_4 = 0 \\ q_3 - q_4 = 0 \end{cases}$$

This example has a solution in the form of an undetermined system of equations from which we can choose e.g.:

$$q = [1 \ \ 1 \ \ 1 \ \ 1]^T .$$

# Example for the analysis of properties:




| Event | Pre-conditions | Pos-conditions |
|---|---|---|
| t1 | - | p1 |
| t2 | p1, p2 | p3 |
| t3 | p3 | p4, p2 |
| t4 | p4 | - |

*Q: Exists* ***conservation*** *?*

*A: Find w such that* $\mathbf{w^T.D=0}$

*if* $\exists w>0$ *then net is conservative*
*else it is not conservative*

$$D = \begin{bmatrix} 1 & -1 & & \\ & -1 & 1 & \\ & 1 & -1 & \\ & & 1 & -1 \end{bmatrix}$$

$$w^T = [w_1\ w_2\ w_3\ w_4] = ?$$

*Q2: What changes if initial marking in p2 is zero?*

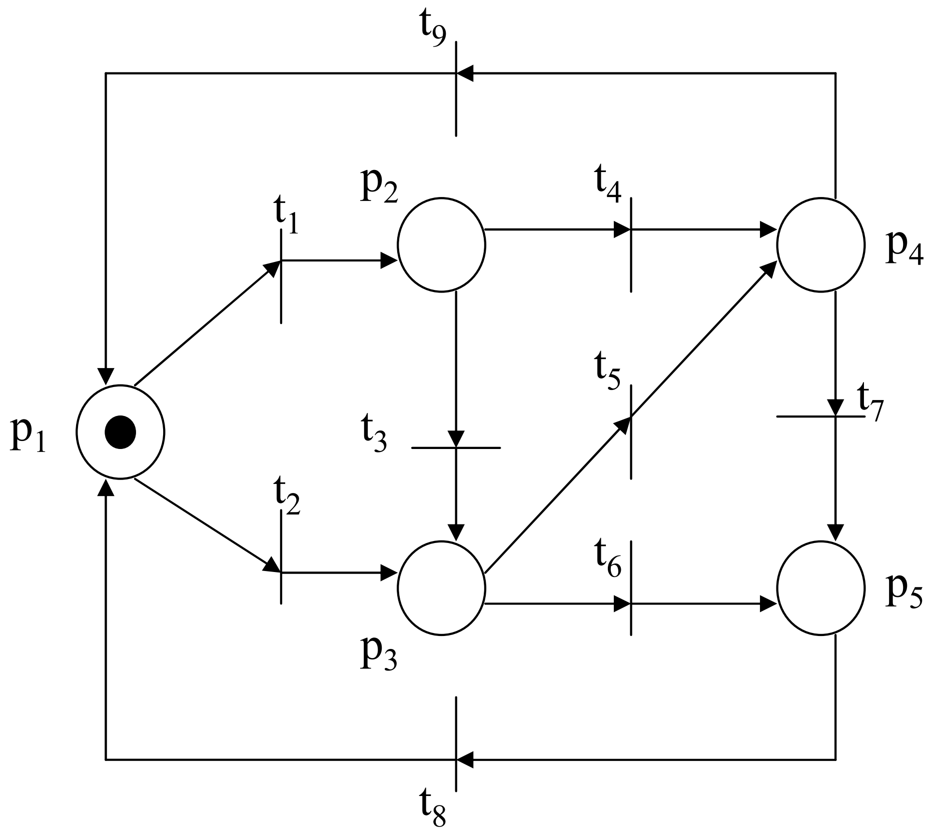
# Discrete Event Systems

## Example of a simple automation system modeled using PNs

An automatic soda selling machine accepts
50c and $1 coins and
sells 2 types of products:
SODA A, that costs $1.50 and
SODA B, that costs $2.00.

Assume that the money return operation is omitted.

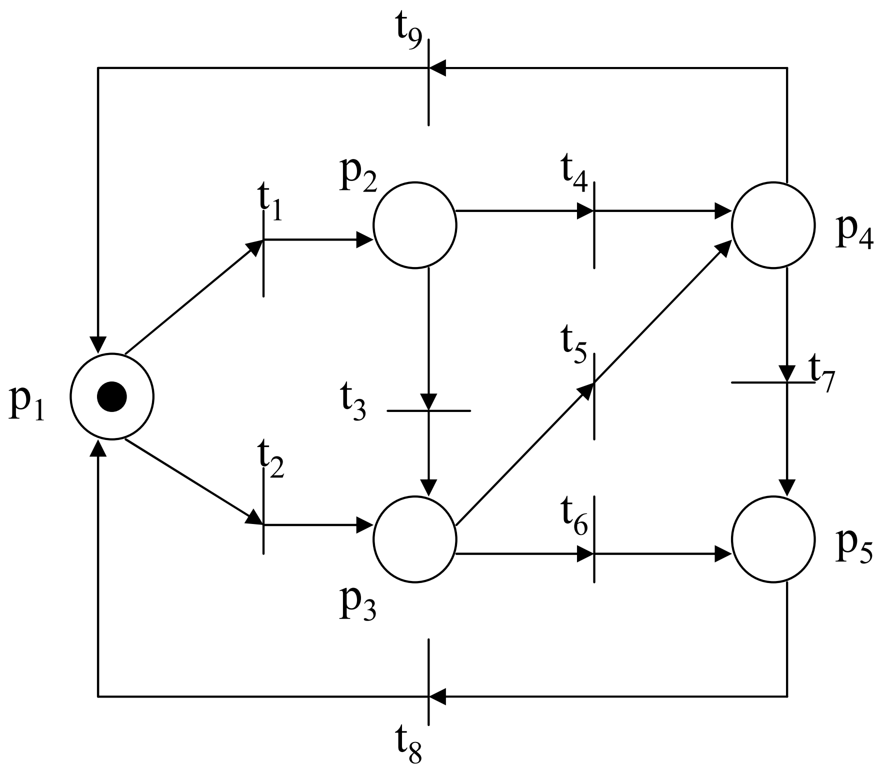
*Q: Are there transition firing vectors that make the Petri net return to the same state?*




$p_1$: machine with $0.00;
$t_1$: coin of 50 c introduced;
$t_8$: SODA B sold.

# Discrete Event Systems

## Example of a simple automation system modeled using PNs




```
D= [ -1  -1   0   0   0   0   0   1   1
      1   0  -1  -1   0   0   0   0   0
      0   1   1   0  -1  -1   0   0   0
      0   0   0   1   1   0  -1   0  -1
      0   0   0   0   0   1   1  -1   0 ]
```

***Time invariance*** *? Find q such that. D.q=0*

```
>> q= null( D, 'r' )
q =
     1   -1    1    0    1
    -1    1   -1    1    0
     1    0    0    0    0
     0   -1    1    0    1
     0    1    0    0    0
     0    0   -1    1    0
     0    0    1    0    0
     0    0    0    1    0
     0    0    0    0    1

>> q(:,1)= q(:,1)+q(:,4);
>> q(:,2)= q(:,2)+q(:,5);
>> q(:,3)= q(:,3)+q(:,4);
q =
     1    0    1    0    1
     0    1    0    1    0
     1    0    0    0    0
     0    0    1    0    1
     0    1    0    0    0
     1    0    0    1    0
     0    0    1    0    0
     1    0    1    1    0
     0    1    0    0    1
```