Industrial Automation

(Automação de Processos Industriais)

Discrete Event Systems

http://www.isr.ist.utl.pt/~pjcro/courses/api1011/api1011.html

Prof. Paulo Jorge Oliveira
pjcro @ isr.ist.utl.pt
Tel: 21 8418053 or 2053 (internal)

Syllabus:

Chap. 5 – CAD/CAM and CNC [1 week]

…

Chap. 6 – Discrete Event Systems [2 weeks]

Discrete event systems modeling. Automata.

Petri Nets: state, dynamics, and modeling.

Extended and strict models. Subclasses of Petri nets.

…

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

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 Performance Analysis", 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

Generic characterization of systems resorting to input / output relations

State equations:

$$\dot{x}(t) = f(x(t), u(t), t)$$
$$y(t) = g(x(t), u(t), t)$$

in continuous time (or in discrete time)

Examples?

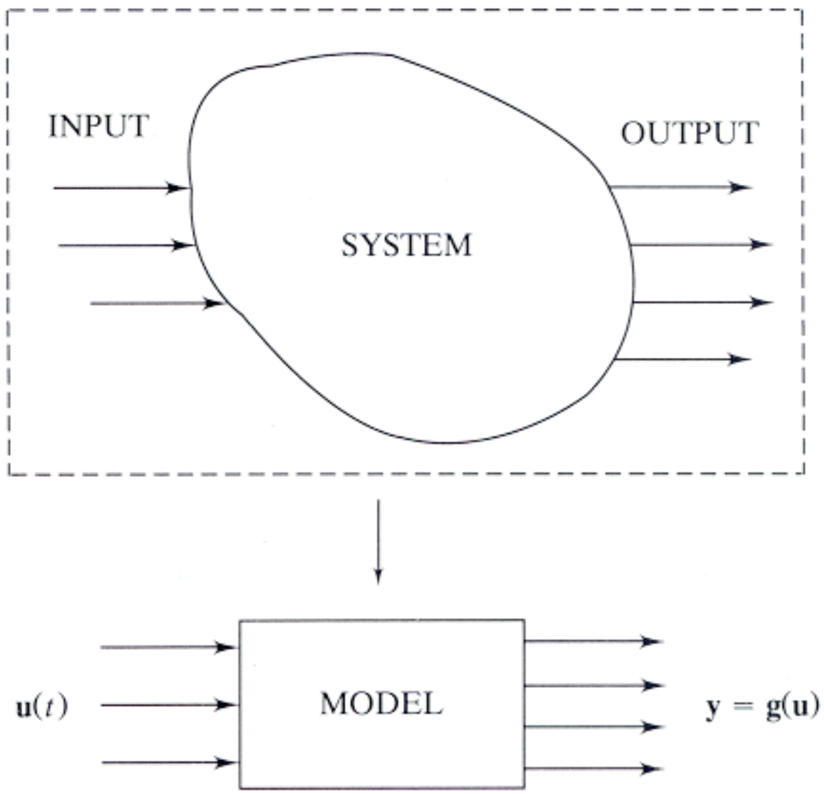


Figure 1.1. Simple modeling process.

Open loop vs close-loop (⇔ the use of feedback)

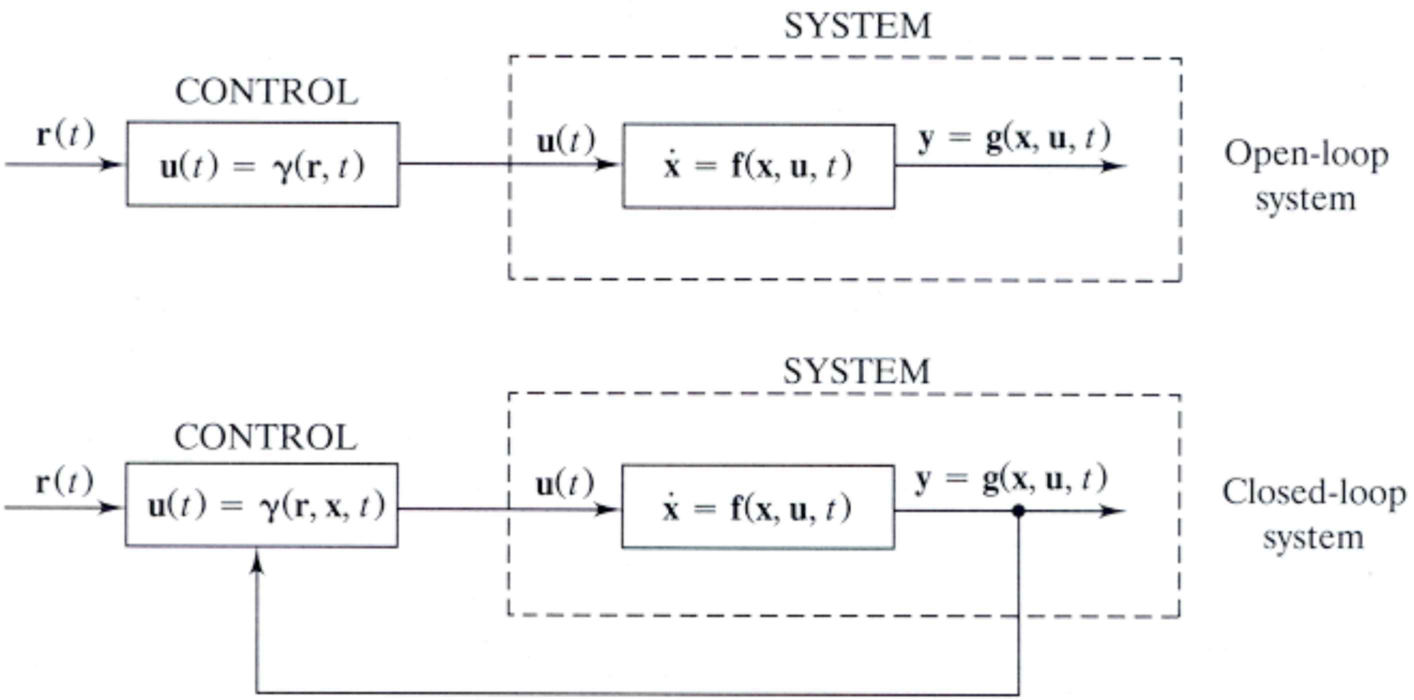


Figure 1.17. Open-loop and closed-loop systems.

Advantages of feedback?

(to revisit during SEDs supervision study)

Example of close-loop with feedback

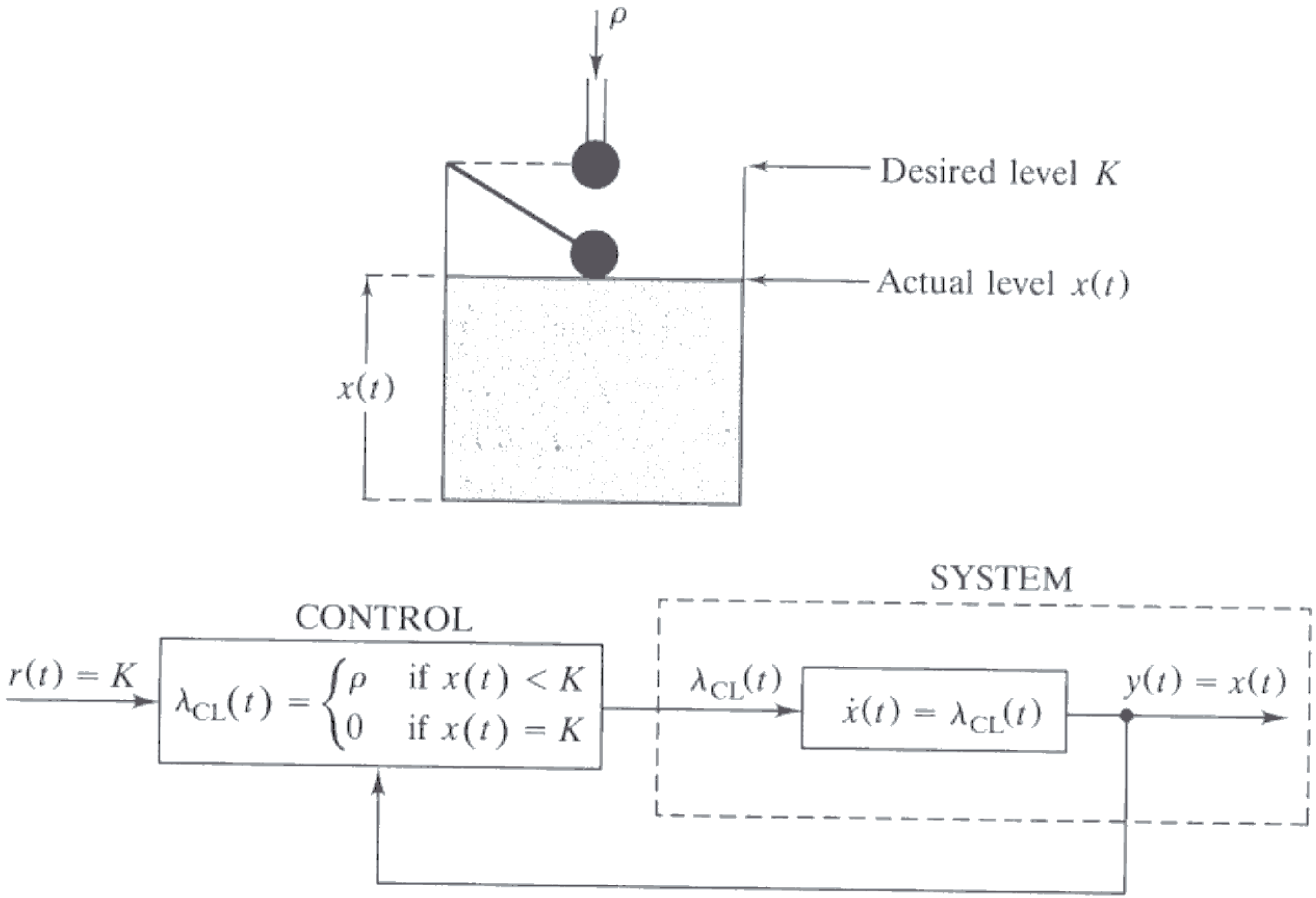


Figure 1.18. Flow system of Example 1.11 and closed-loop control model.

Discrete Event Systems: Examples

Set of events:

E={N, S, E, W}

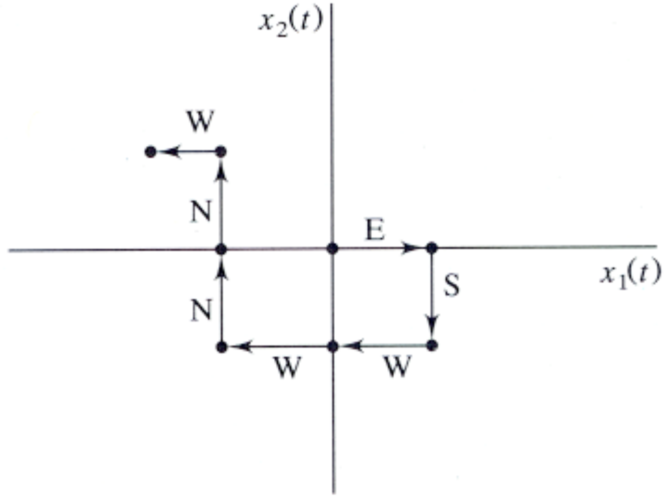


Figure 1.20. Random walk on a plane for Example 1.12.

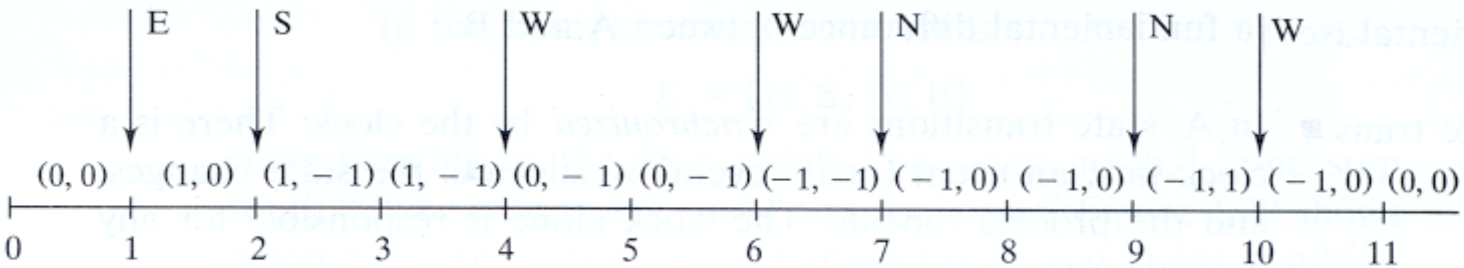


Figure 1.21. Event-driven random walk on a plane.

Characteristics of systems with continuous variables

1.State space is continuous

2. The state transition mechanism is *time-driven*

Characteristics of systems with discrete events

1.State space is discrete

2. The state transition mechanism is *event-driven*

Polling is avoided!

Taxonomy of Systems

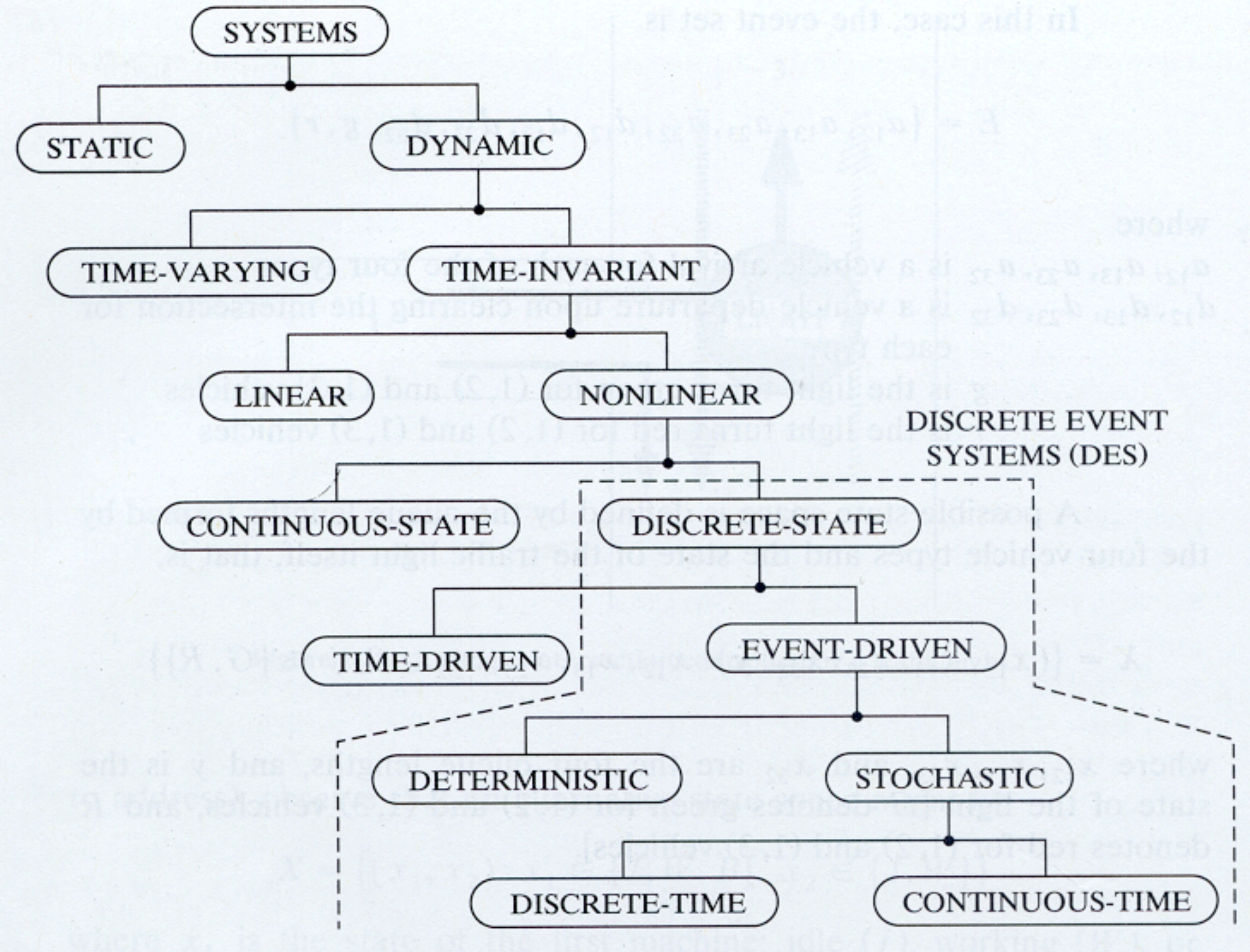


Figure 1.29. Major system classifications.

Levels of abstraction in the study of Discrete Event Systems

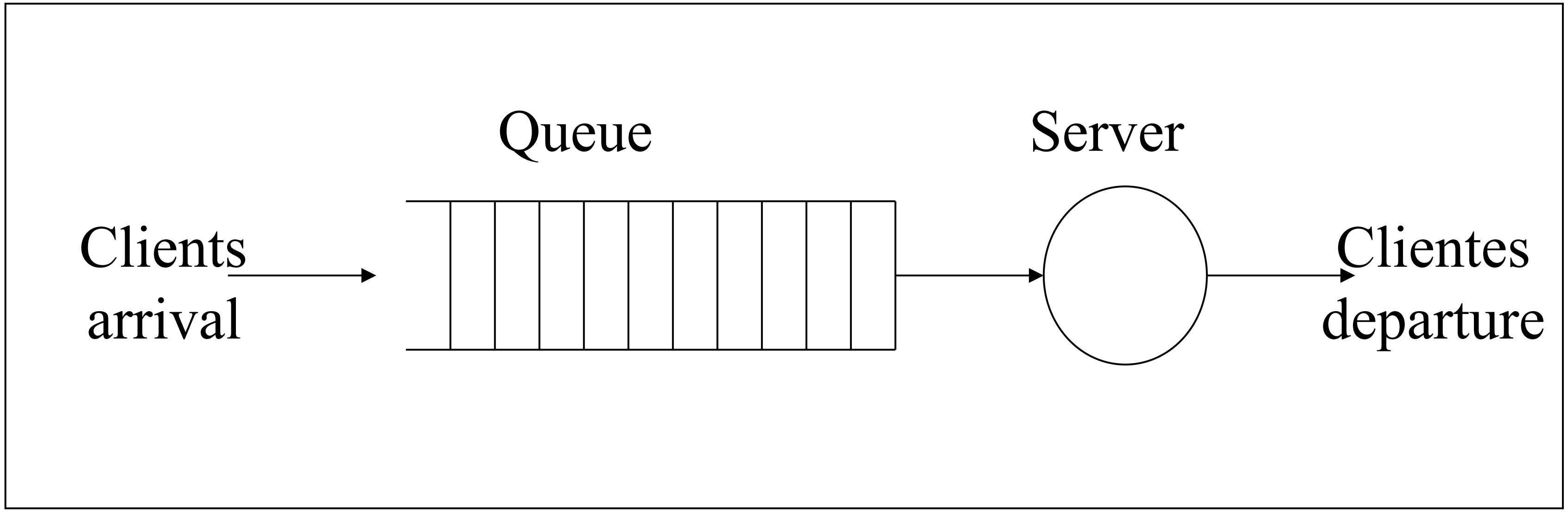
Languages

Timed languages

Stochastic timed languages

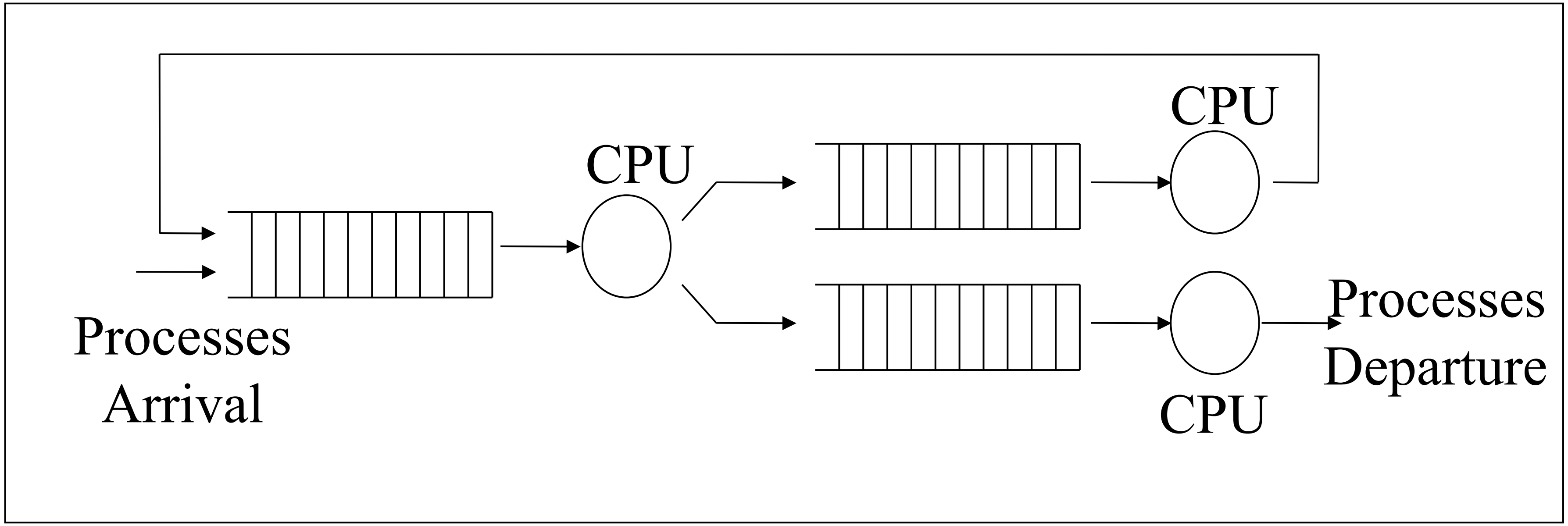
Discrete Event Systems: Examples

Queueing systems



Discrete Event Systems: Examples

Computational Systems



Systems' Theory Objectives

- Modeling and Analysis
- *Design* e synthesis
- Control / Supervision
- Performance assessment and robustness
- Optimization

Applications of Discrete Event Systems

- Queueing systems
- Operating systems and computers
- Telecommunications networks
- Distributed databases
- Automation

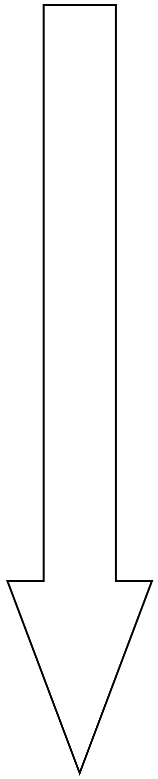
Discrete Event Systems

Typical modeling methodologies

Automata

GRAFCET

Petri nets



Augmenting
in
modeling
capacity
and
complexity

Automata Theory and Languages

Genesys of computation theory

Definition: A **language** L, defined over the alphabet **E** is a set of *strings* of finite length with events from **E**.

Exemplos: $\mathbf{E}=\{\alpha, \beta, \gamma\}$

$L_1=\{\varepsilon, \alpha\alpha, \alpha\beta, \gamma\beta\alpha\}$

$L_2=\{$all *strings* of length 3$\}$

How to build a machine that "talks" a given language?

or

What language "talks" a system?

Properties of languages

Kleene-closure E^*: set of all strings of finite length of E, including the null element ε.

Concatenation:

$$L_a L_b := \{ s \in E^* : s = s_a s_b, s_a \in L_a, s_b \in L_b \}$$

Prefix-closure:

$$\overline{L} := \{ s \in E^* : \exists_{t \in E^*} \, st \in L \}$$

Automata Theory and Languages

Definition: A deterministic automata is a 5-*tuple*

$$(\mathbf{E}, \mathbf{X}, \mathbf{f}, \mathbf{x}_0, \mathbf{F})$$

onde:

E	- finite alphabet (or possible events)	
X	- finite set of states	
f	- state transition function	$\mathbf{f}: \mathbf{X} \times \mathbf{E} \rightarrow \mathbf{X}$
$\mathbf{x}_0$	- initial state	$\mathbf{x}_0 \subset \mathbf{X}$
F	- set of final states or marked states	$\mathbf{F} \subset \mathbf{E}$

Example of a automata

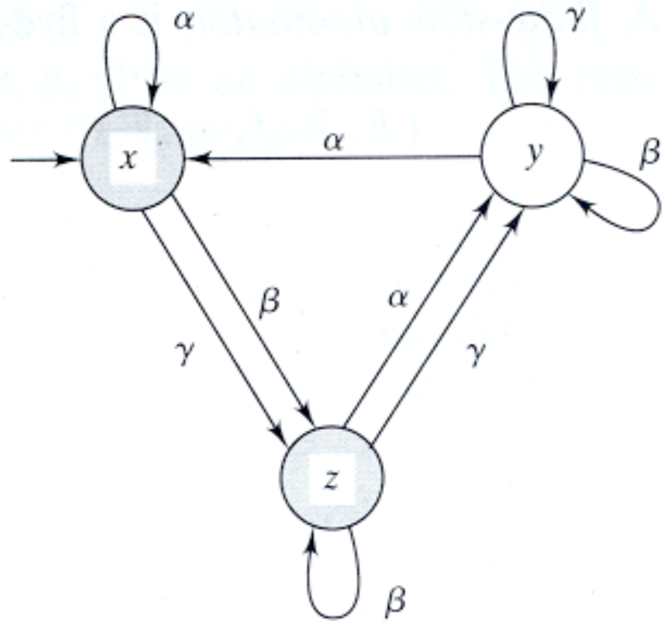


Figure 2.1. State transition diagram for Example 2.3.

(E, X, f, x_0, F)

E = $\{\alpha, \beta, \gamma\}$

X = $\{x, y, z\}$

$\mathbf{x_0} = x$

F = $\{x, z\}$

$f(x, \alpha)=x$	$f(x, \beta)=z$	$f(x, \gamma)=z$
$f(y, \alpha)=x$	$f(y, \beta)=y$	$f(y, \gamma)=y$
$f(z, \alpha)=y$	$f(z, \beta)=z$	$f(z, \gamma)=y$

Example of a stochastic automata

(E, X, f, x_0, F)

$\mathbf{E} = \{\alpha, \beta\}$

$\mathbf{X} = \{0, 1\}$

$\mathbf{x_0} = 0$

$\mathbf{F} = \{0\}$

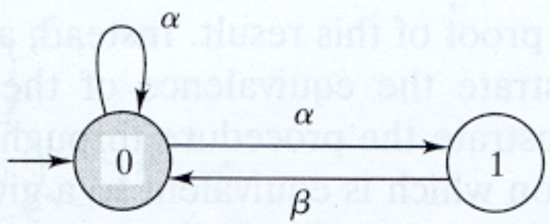


Figure 2.4. State transition diagram for the nondeterministic automaton of Example 2.7.

$f(0, \alpha) = \{0, 1\}$ $f(0, \beta) = \{\}$
$f(1, \alpha) = \{\}$ $f(1, \beta) = 0$

Given a language

$$\mathbf{G=(E, X, f, x_0 ,F)}$$

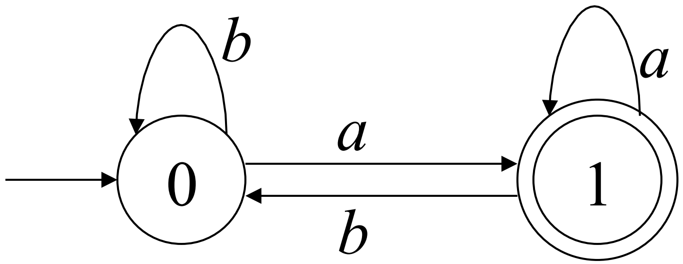
Generated language

$$\boldsymbol{L}(G) := \{s \in E^*: f(x_0,s) \text{ is defined}\}$$

Marked language

$$\boldsymbol{L_m}(G) := \{s \in E^*: f(x_0,s) \in F\}$$

Example: marked language of an automata



$$L_m(\mathrm{G}) := \{a, aa, ba, aaa, baa, bba, \ldots\}$$

Note: all strings with events *a* e *b*, followed by event *a*.

Automata equivalence:

The automata G_1 e G_2 are equivalent if

$$\boldsymbol{L}(\mathrm{G}_1) = \boldsymbol{L}(\mathrm{G}_2)$$

e

$$\boldsymbol{L}_{\mathrm{m}}(\mathrm{G}_1) = \boldsymbol{L}_{\mathrm{m}}(\mathrm{G}_2)$$

Example of an automata:

Objective: To validate a sequence of events

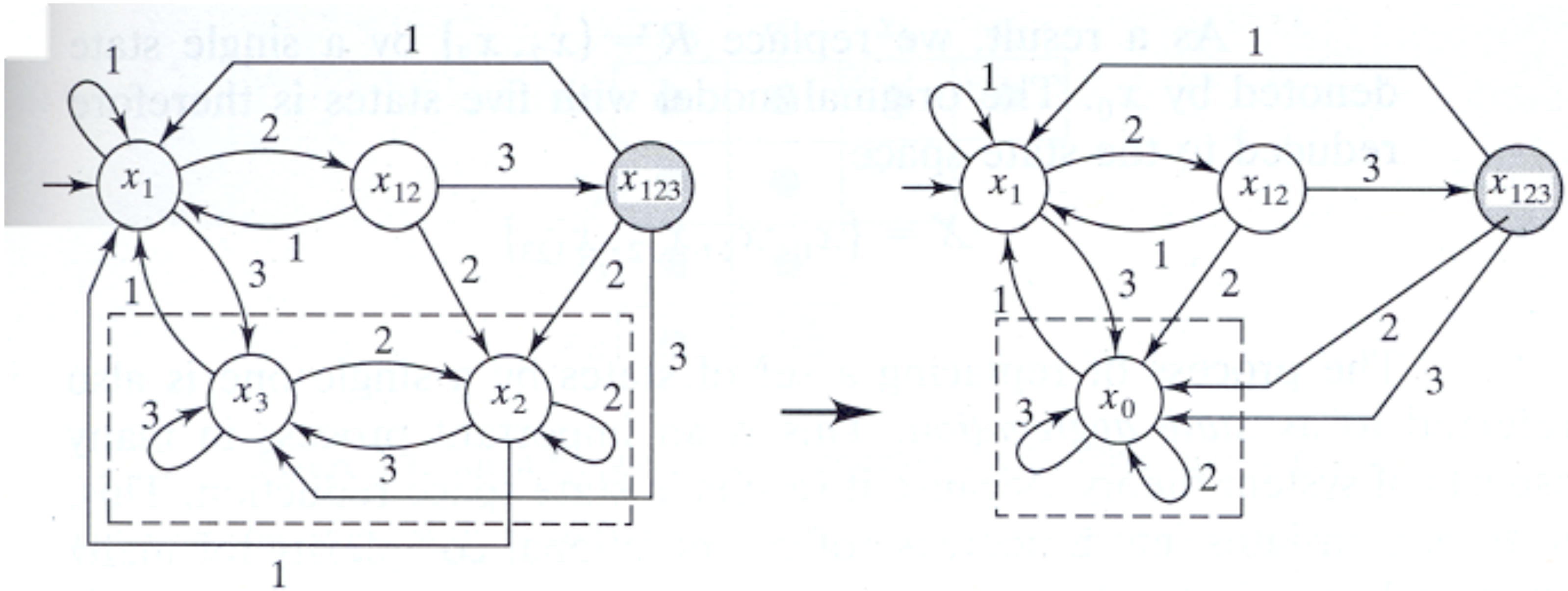
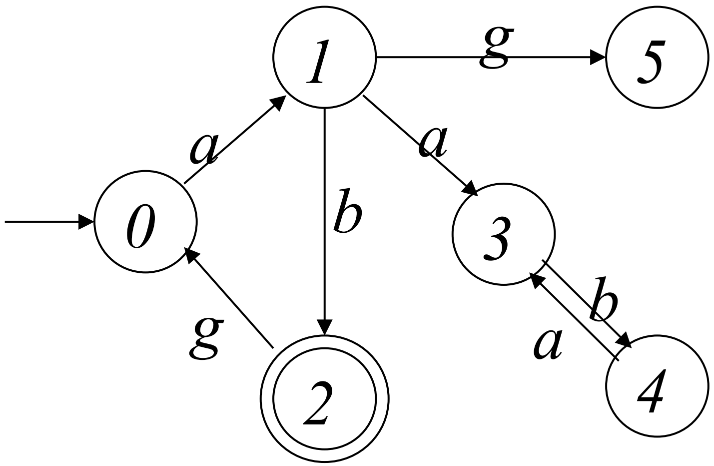


Figure 2.6. State transition diagrams for digit sequence detector in Example 2.9.

Deadlocks (*inter-blocagem*)

Example:



The state *5* is a *deadlock*.

The states *3* and *4* constitutes a *livelock*.

How to find the *deadlocks* and the *livelocks*?

Methodologies for the analysis Of Discrete Event Systems

Deadlock:

in general the following relations are verified

$$L_m(G) \subseteq \overline{L}_m(G) \subseteq L(G)$$

An automata G has a deadlock if

$$\overline{L}_m(G) \subset L(G)$$

and is not blocked when

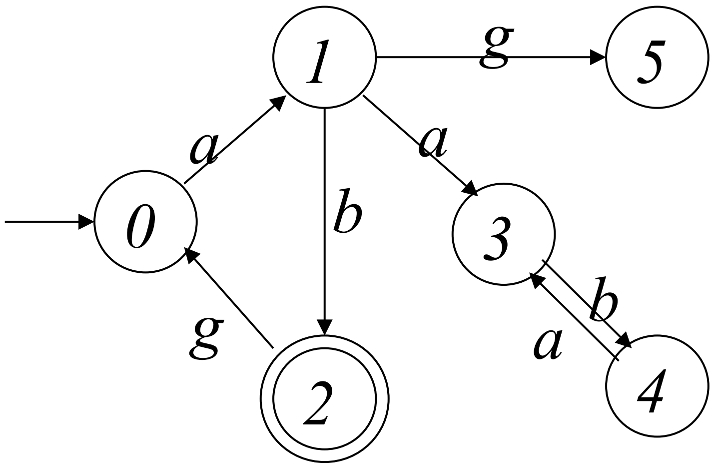
$$\overline{L}_m(G) = L(G)$$

Deadlock:

$$L_m(G)=\{ab, abgab, abgabgab, \ldots\}$$

Example:

$$L(G)=\left\{\begin{matrix}\varepsilon, a, ab, ag, aa, aab, \\ abg, aaba, abga, \ldots\end{matrix}\right\}$$



$$\left(L_m(G)\subset L(G)\right)$$

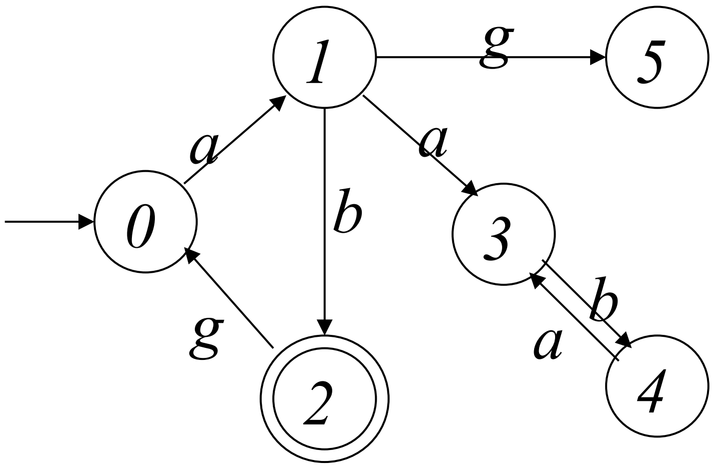
The state *5* is a *deadlock.*

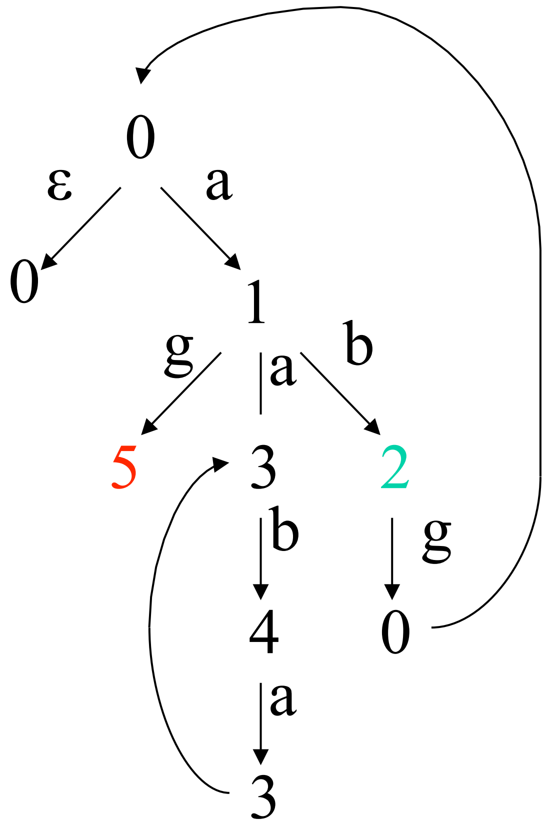
The states *3* and *4* constitutes a *livelock*.

$$\overline{L}_m(G)\neq L(G)$$

Alternative way to detect deadlocks:

Example:





The state *5* is a *deadlock.*

The states *3* and *4* constitutes a *livelock*.

Timed Discrete Event Systems

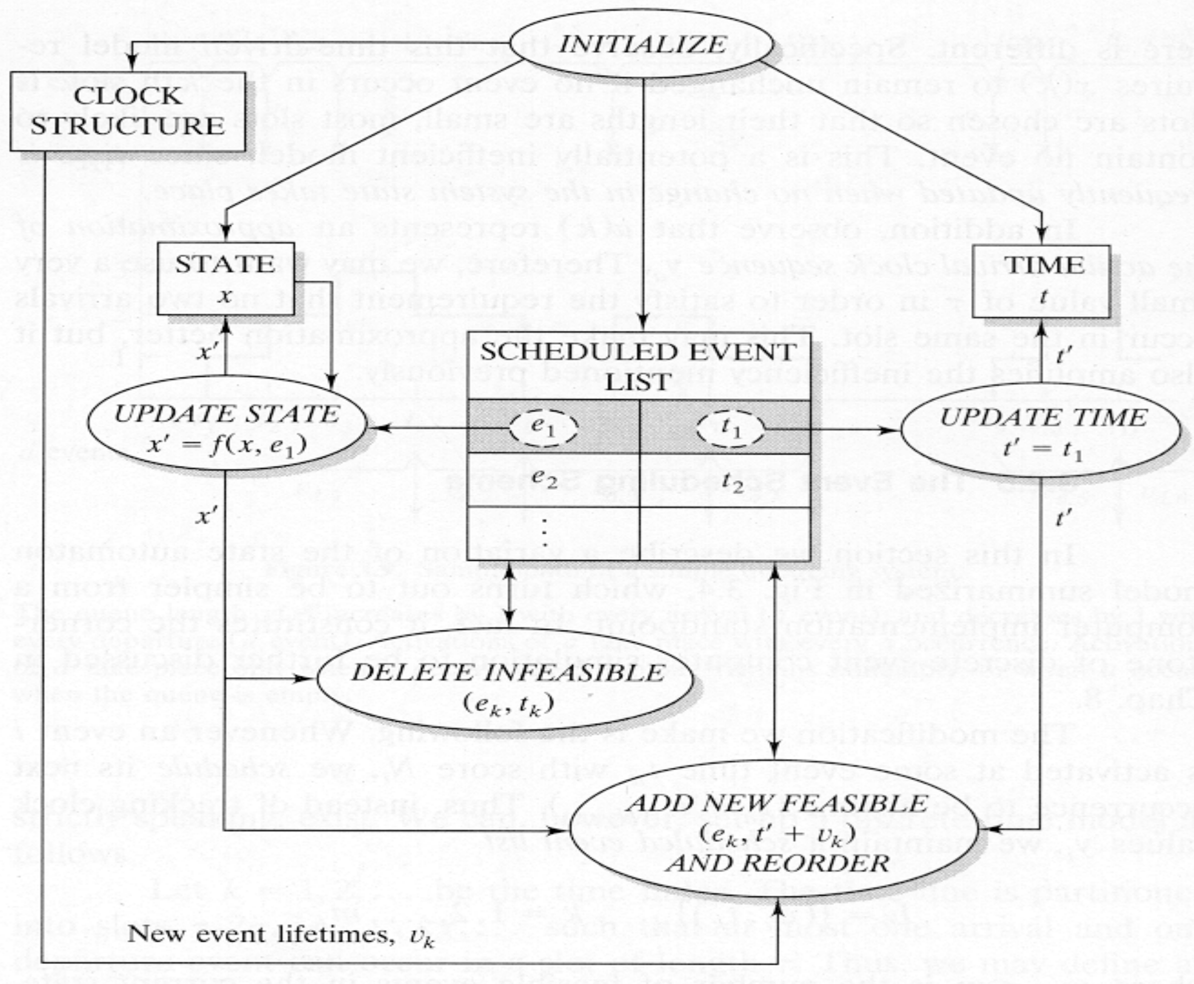


Figure 3.10. The event scheduling scheme.

Petri nets

Developed by Carl Adam Petri in his PhD thesis in 1962.

Definition: A marked Petri net is a *5-tuple*

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

where:

P	- set of places	
T	- set of transitions	
A	- set of arcs	$A \subset (P \times T) \cup (T \times P)$
w	- weight function	$w: A \to \mathbb{N}$
x_0	- initial marking	$x_0 : P \to \mathbb{N}$

Example of a Petri net

(P, T, A, w, x_0)

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

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

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

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

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

Petri nets

Rules to follow (mandatory):

- An arc (directed connection) can connect places to transitions
- An arc can connect transitions to places
- A transition can have no places as inputs (source)
- A transition can have no places as outputs (sink)
- The same happens with the input and output transitions for places

Example of a Petri net

(P, T, A, w, x_0)

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

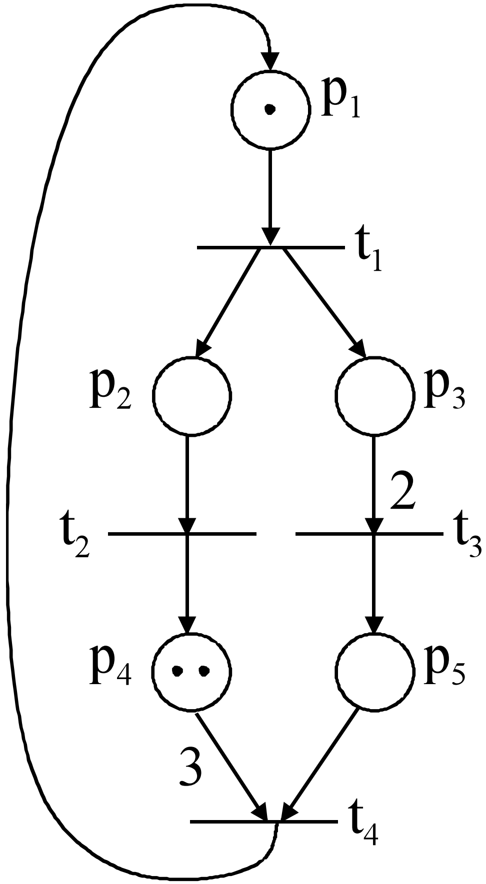
$T=\{t_1, t_2, t_3, t_4\}$

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

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

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

Petri net graph



Alternative definition of a Petri net

A marked Petri net is a *5-tuple*

$$(\mathbf{P}, \mathbf{T}, \mathbf{I}, \mathbf{O}, \mu_0)$$

where:

P	- set of places	
T	- set of transitions	
I	- transition input function	$\mathbf{I}: \mathbf{P} \rightarrow \mathbf{T}^{\infty}$
O	- transition output function	$\mathbf{O}: \mathbf{T} \rightarrow \mathbf{P}^{\infty}$
μ_0	- initial marking	$\mu_0 : \mathbf{P} \rightarrow \mathbf{N}$

Example of a Petri net and its graphical representation

Alternative definition

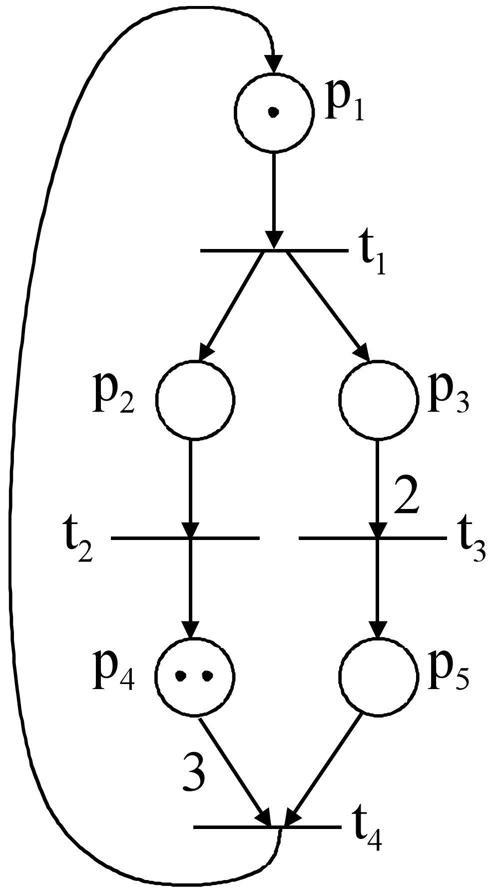
(P, T, I, O, μ_0)

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

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

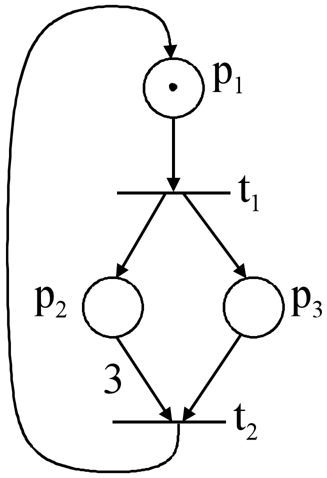
$I(t_1)=\{p_1\}$ $O(t_1)=\{p_2 , p_3\}$
$I(t_2)=\{p_2\}$ $O(t_2)=\{p_4\}$
$I(t_3)=\{p_3, p_3\}$ $O(t_3)=\{p_5\}$
$I(t_4)=\{p_4, p_4, p_4, p_5\}$ $O(t_4)=\{p_1\}$

$\mu_0 = \{1, 0, 0, 2, 0\}$



Petri nets

The state of a Petri net is characterized by the marking of all places.



The set of all possible markings of a Petri net corresponds to its state space.

How does the state of a Petri net evolves?

Dynamics of Petri nets

A transition t_j ε T is *enabled* if:

$$\forall p_i \in P: \quad \mu(p_i) \quad \geq \quad \#(p_i, I(t_j))$$

A transition t_j Î T is enabled to *fire*,
resulting in a new marking given by

$$\mu'(p_i) \;=\; \mu(p_i) \;-\; \#(p_i, I(t_j)) \;+\; \#(p_i, O(t_j))$$

Petri nets

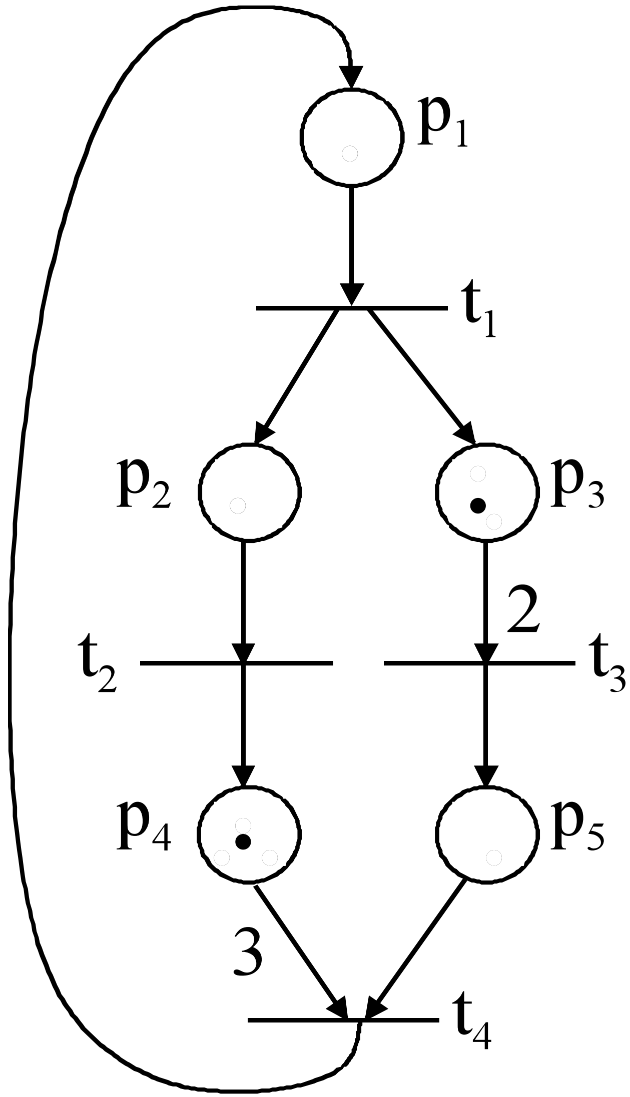
Example of evolution of a Petri net

Initial marking:

$\mu_0 = \{1, 0, 1, 2, 0\}$

This discrete event system can not change state.

It is in a *deadlock!*



Petri nets: Conditions and Events

Conditions:

a) The server is idle.
b) A job arrives and waits to be processed
c) The server is processing the job
d) The job is complete

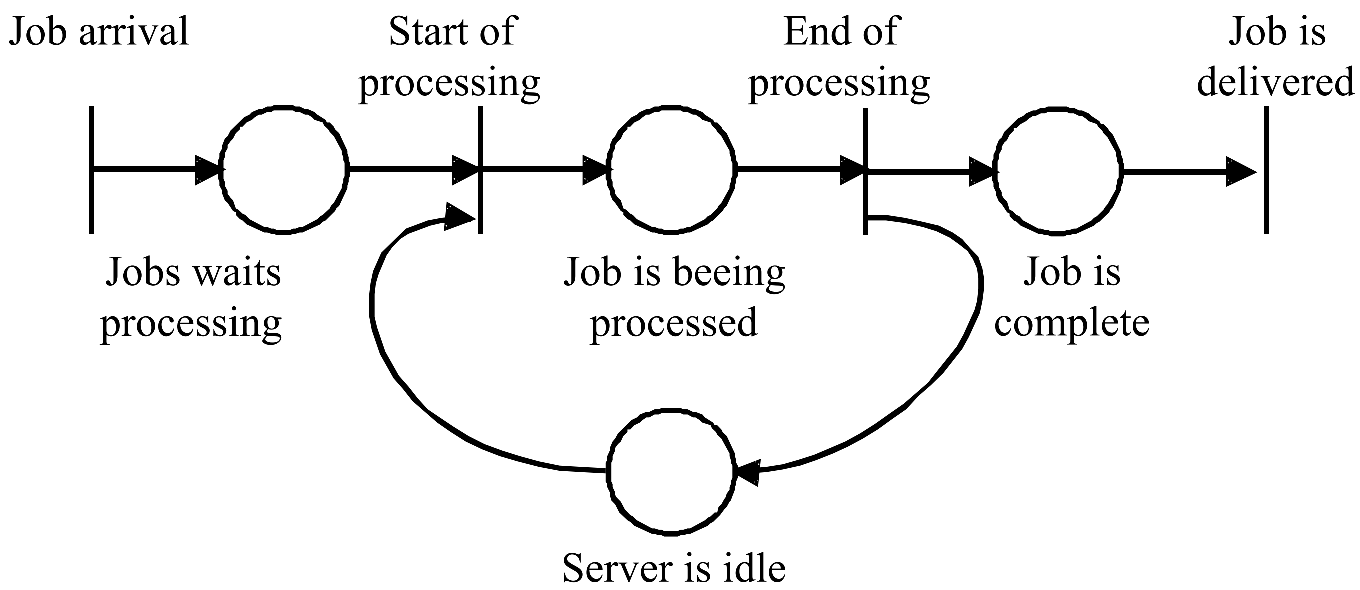
Events

1) Job arrival
2) Server starts processing
3) Server finishes processing
4) The job is delivered

Event	Pre-conditions	Pos-conditions
1	-	b
2	a,b	c
3	c	d,a
4	d	-

Petri nets: Conditions and Events

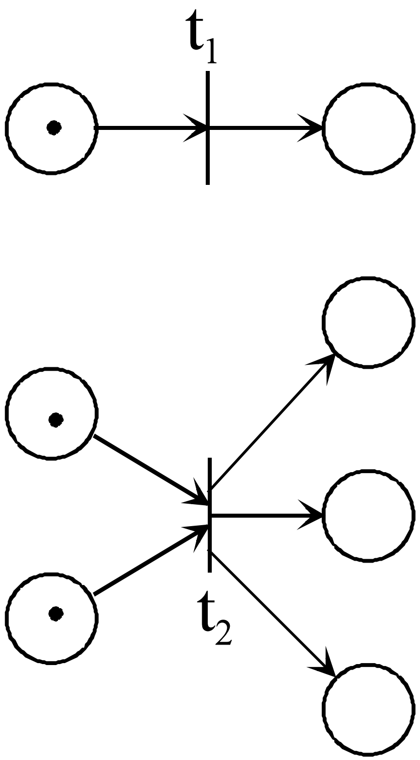
Event	Pre-conditions	Pos-conditions
1	-	b
2	a,b	c
3	c	d,a
4	d	-



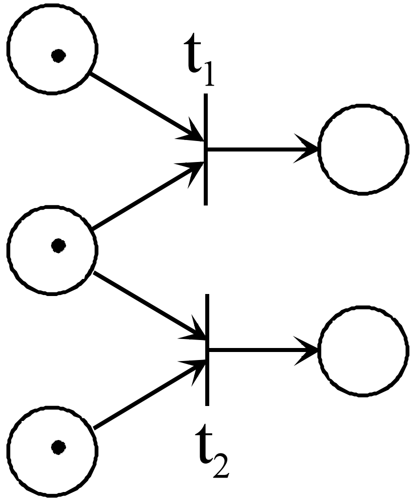
Petri nets

Modeling mechanisms

Concurrence



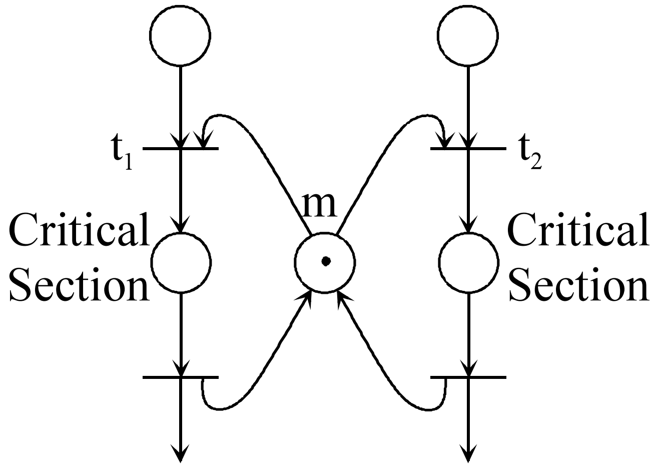
Conflict



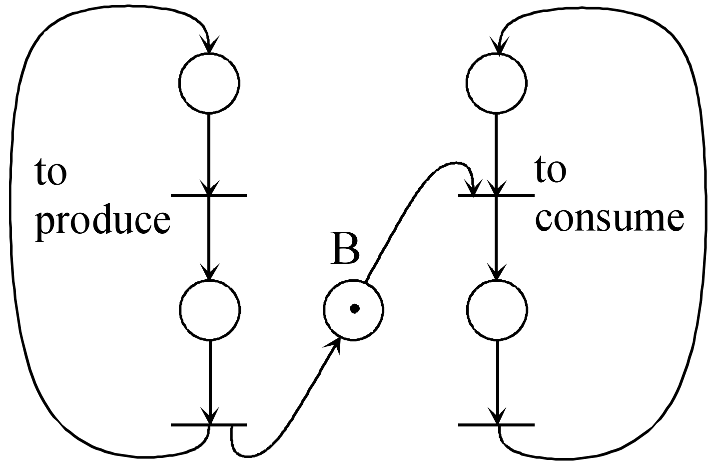
Petri nets

Modeling mechanisms

Mutual Exclusion



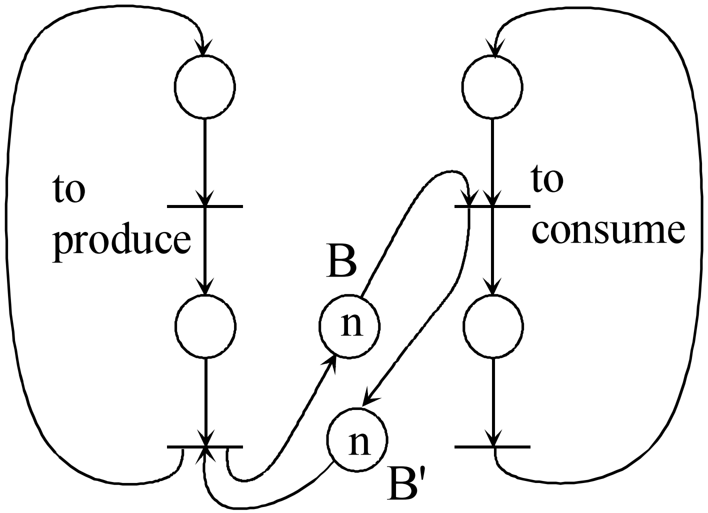
Producer / Consumer



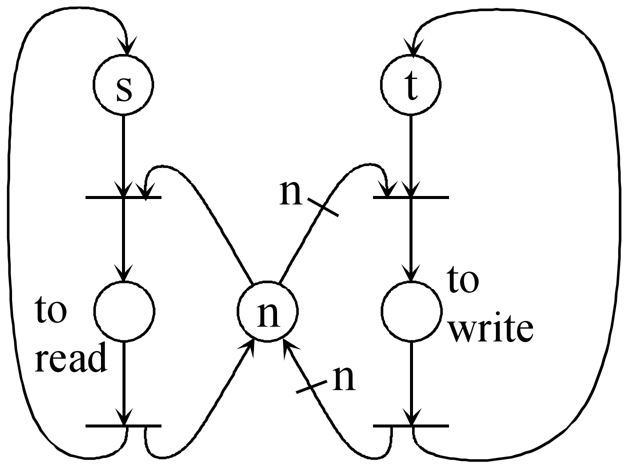
Petri nets

Modeling mechanisms

Producer / Consumer
with finite capacity



Readers / Writers

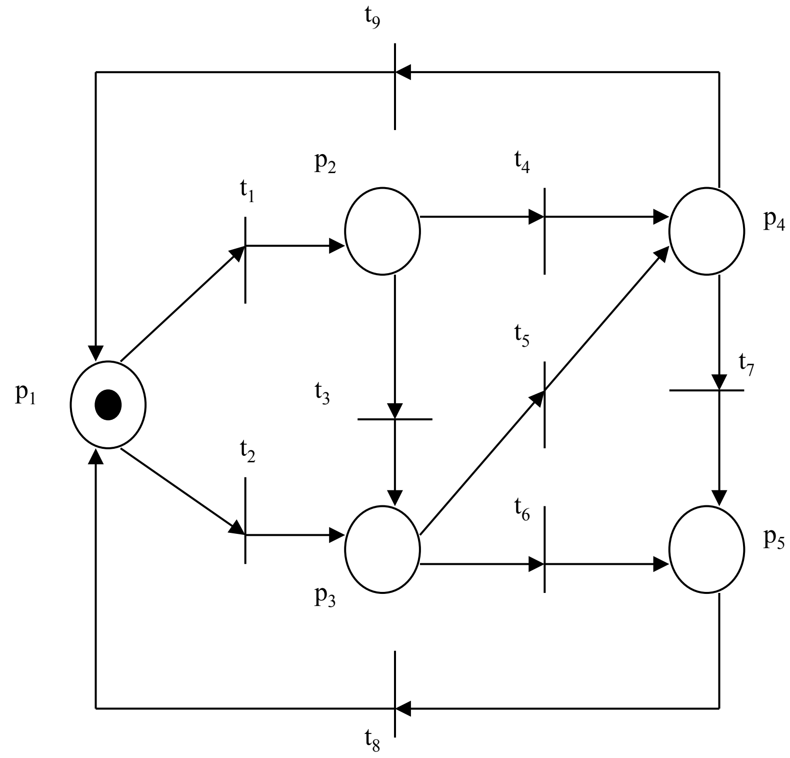


Discrete Event Systems

Example of a simple automation system modelled using PNs

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

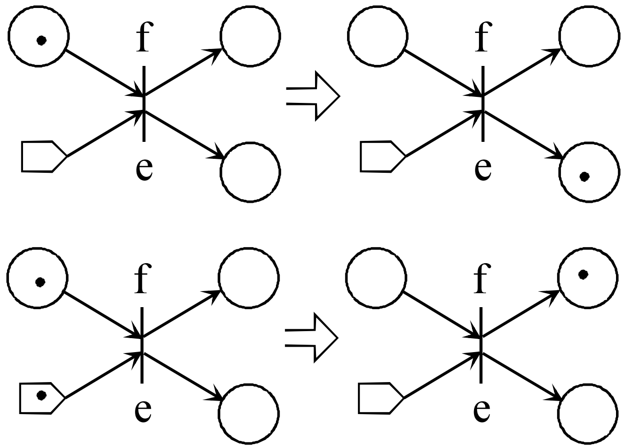
Assume that the money return operation is omitted.



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

Extentions to Petri nets

Switches [Baer 1973]



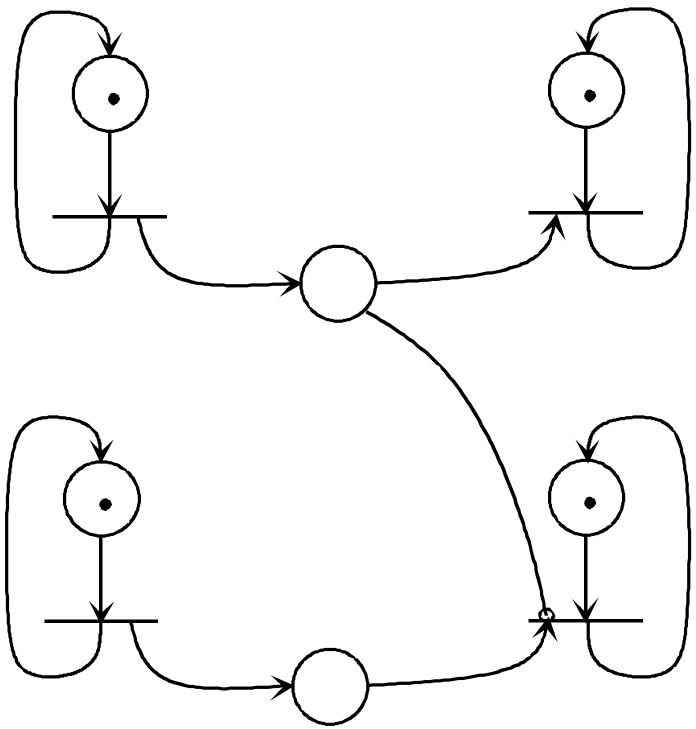
Possible to be implemented with restricted Petri nets.

Extentions to Petri nets

Inhibitor Arcs

Equivalent to

nets with priorities



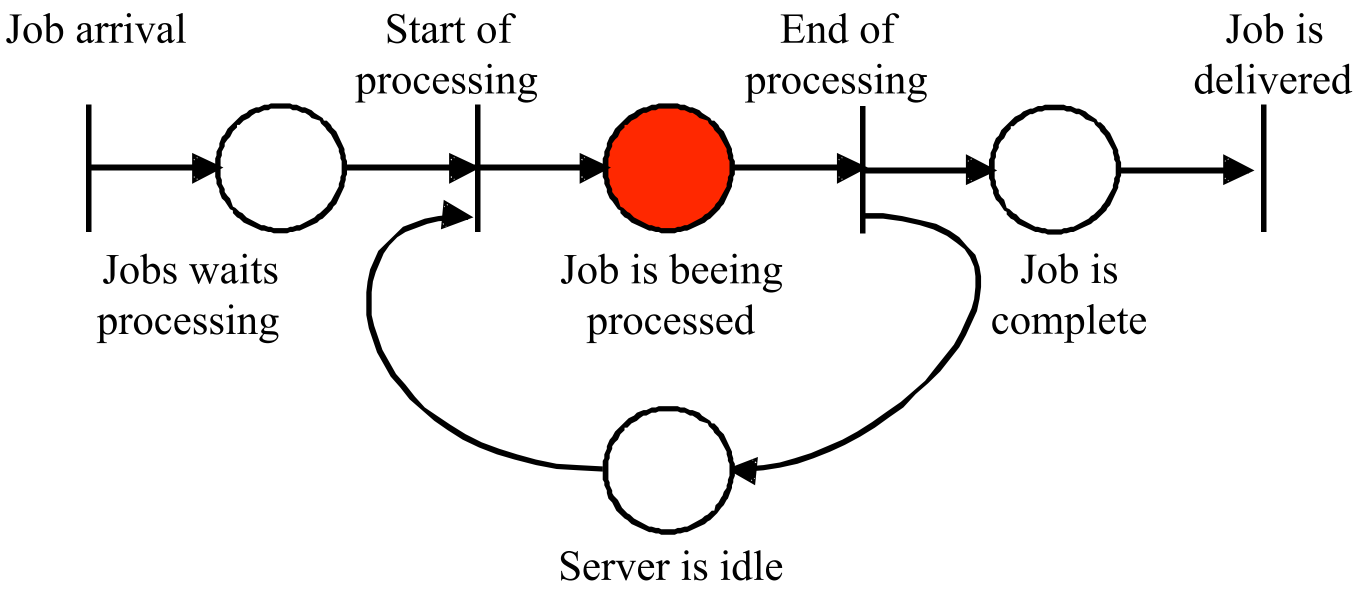
Can be implemented with restricted Petri nets?

Zero tests...

Infinity tests...

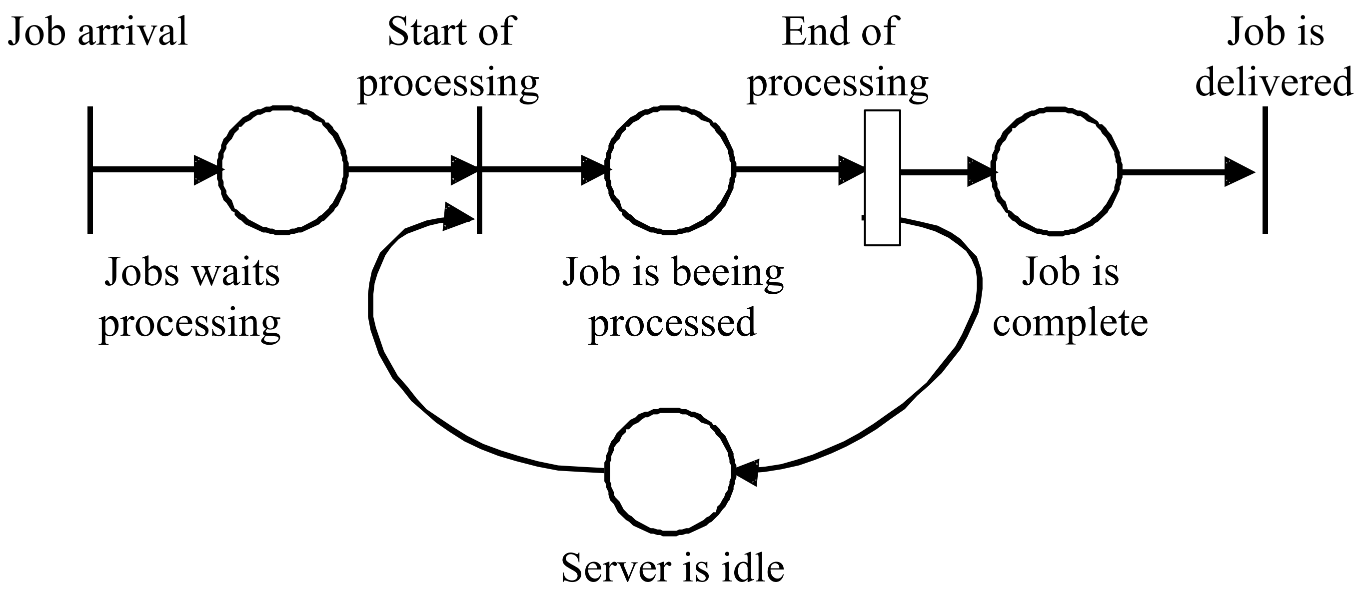
Extentions to Petri nets

P-Timed nets



Extentions to Petri nets

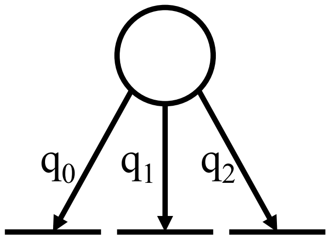
T-Timed nets



Extentions to Petri nets

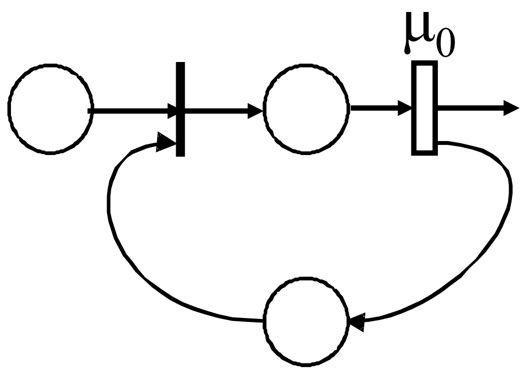
Stochastic nets

Stochastic switches



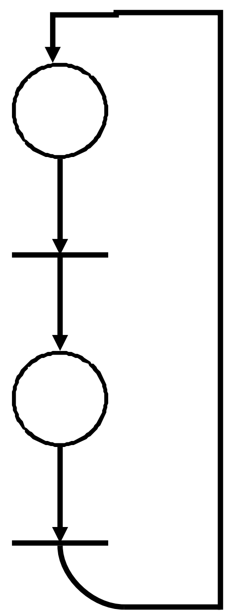
$q_0+q_1+q_2=1$

Transitions with stochastic timmings described by a stochastic variable with known pdf



Discrete Event Systems

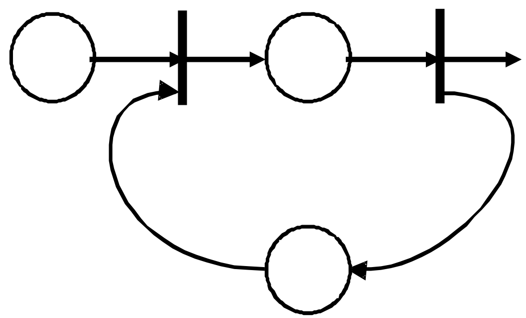
Sub-classes of Petri nets



State Machine:

Petri nets where each transition has exactly one input arc and one output arc.

Marked Graphs

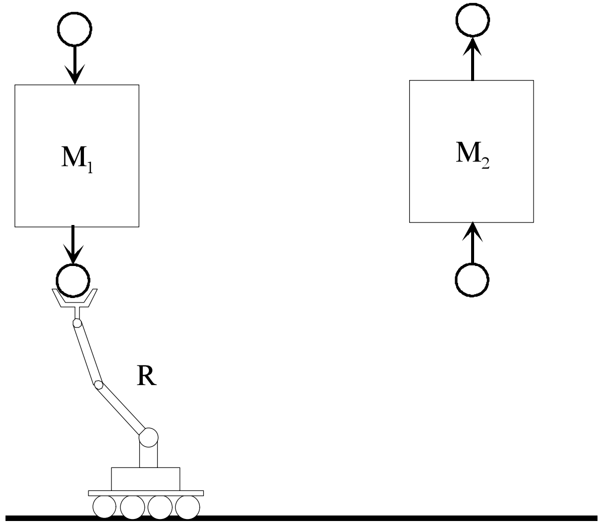


Petri nets where each place has exactly one input arc and one output arc.

Discrete Event Systems

Example of DES:

Manufacturing system composed by 2 machines (M_1 and M_2) and a robotic manipulator (R). This takes the finished parts from machine M_1 and transports them to M_2.



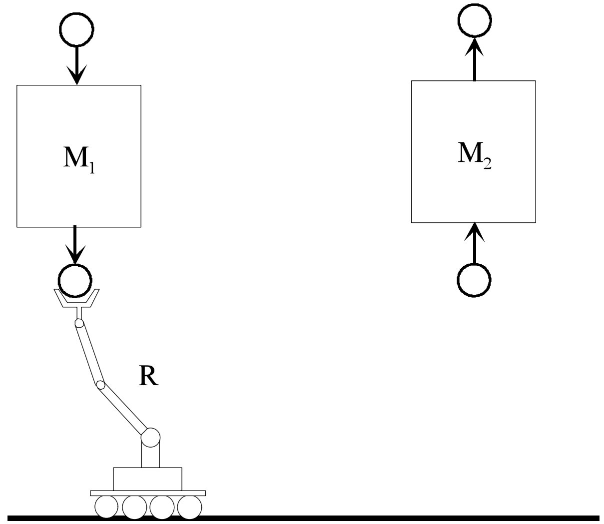
No buffers available on the machines. If R arrives near M_1 and the machine is busy, the part is rejected.

If R arrives near M_2 and the machine is busy, the manipulator must wait.

Machinning time: M_1=0.5s; M_2=1.5s; $R_{M1 \circledR M2}$=0.2s; $R_{M2 \circledR M1}$=0.1s;

Discrete Event Systems

Example of DES:



Variables of

M_1	x_1
M_2	x_2
R	x_3

Example of arrival of parts:

$$a(t)=\begin{cases} 1 \quad em \quad \{0.1, \quad 0.7, \quad 1.1, \quad 1.6, \quad 2.5\} \\ 0 \quad em \quad todos \quad os \quad outros \quad instantes \end{cases}$$

x_1={Idle, Busy, Waiting}
x_2={Idle, Busy}
x_3={Idle, Carrying, Returning}

Discrete Event Systems

Example of DES:

Definition of events:

a_1 - loads part in M_1

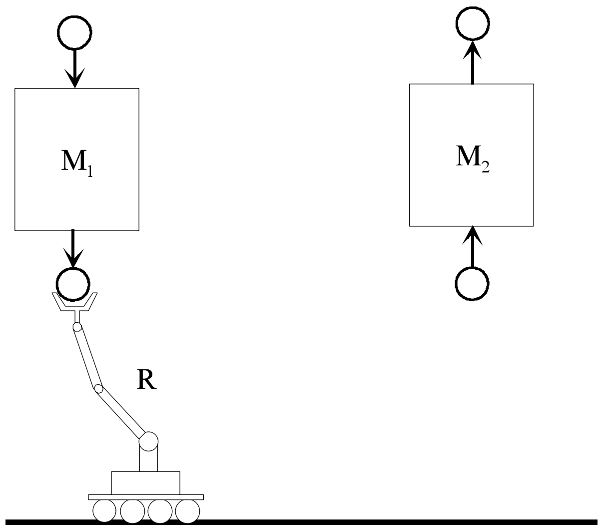
d_1 - ends part processing in M_1

r_1 - loads manipulator

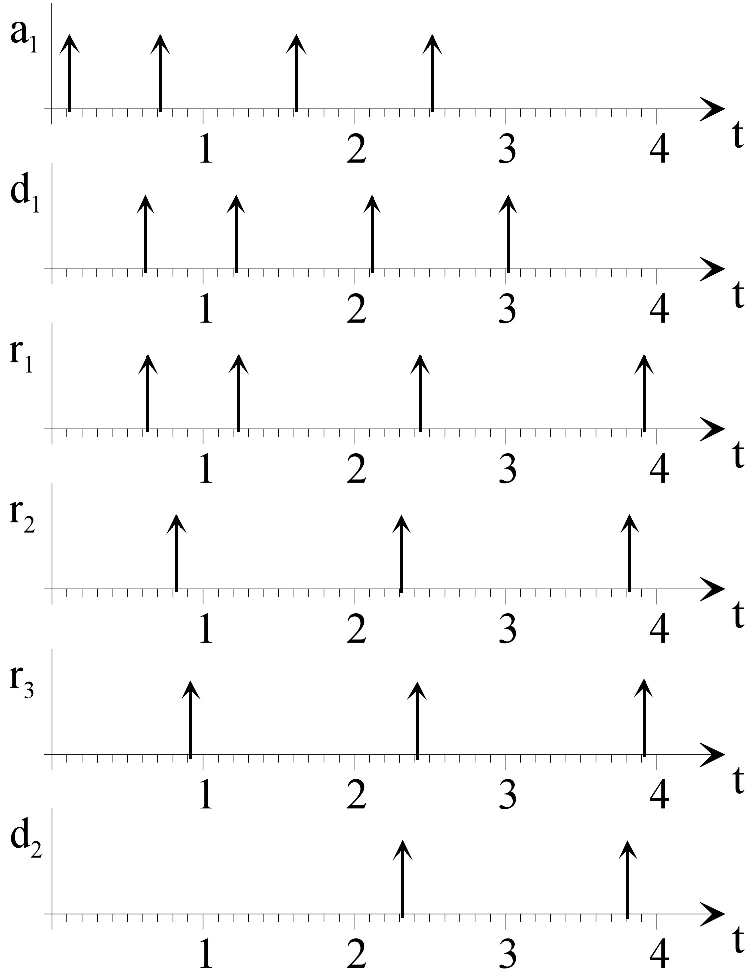
r_2 - unloads manipulator and loads M_2

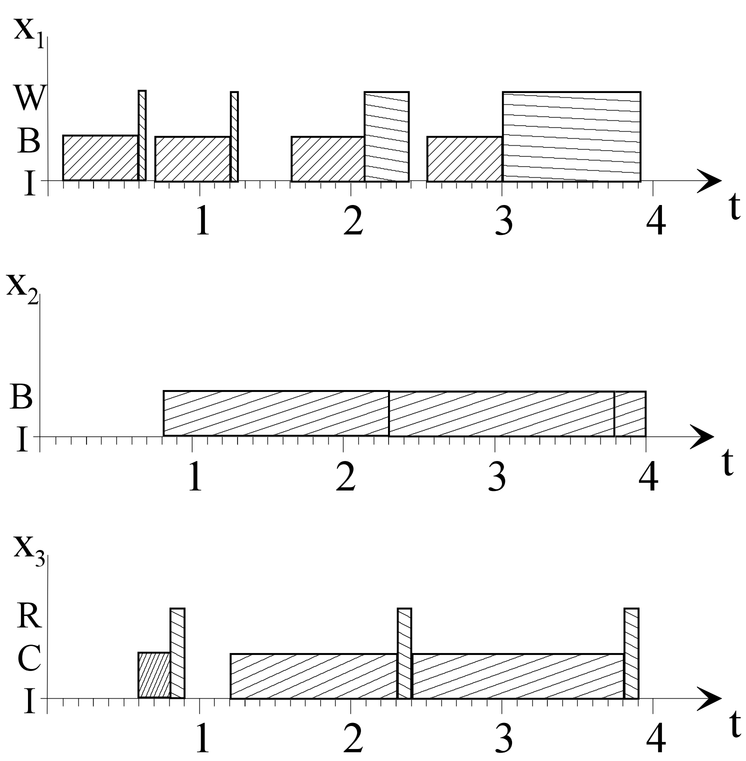
d_2 - ends part processing in M_2

r_3 - manipulator at base



Discrete Event Systems





Discrete Event Systems

Example of DES:

Events:

a_1 - loads part in M_1
d_1 - ends part processing in M_1
r_1- loads manipulator
r_2- unloads manipulator and loads M_2
d_2- ends part processing in M_2
r_3- manipulator at base