Discrete and continuous dynamic systems

Petri Nets
Definition and operation

Anna Ibolya Pózna

University of Pannonia
Faculty of Information Technology
Department of Electrical Engineering and Information Systems

pozna.anna@virt.uni-pannon.hu

April 2021

Lecture overview

1. Previous notions
   - Discrete event systems
   - Automata models
   - Simple examples
2. Petri net models
   - Description forms
   - Operation (dynamics) of Petri nets
   - Parallel and conflicting execution steps
3. Solution of Petri net models
   - The reachability graph

Discrete event systems

Characteristic properties:

- the *range space* of the signals (input, output, state) is **discrete**: $x(t) \in \mathbf{X} = \{x_0, x_1, ..., x_n\}$
- *event*: the occurrence of change in a discrete value
- *time is also* ***discrete***: $T = \{t_0, t_1, ..., t_n\} = \{0, 1, ..., n\}$

Only the **order of the events** is considered

- description of sequential and parallel events
- **application area**: scheduling, operational procedures, resource management

Discrete time linear state space models

$$x(k+1) = \Phi x(k) + \Gamma u(k) \qquad (state\ equation)$$
$$y(k) = Cx(k) + Du(k) \qquad (output\ equation)$$

given initial condition $x(0)$;
vector valued signals

$$x(k) \in \mathcal{R}^n \ , \ y(k) \in \mathcal{R}^p \ , \ u(k) \in \mathcal{R}^r$$

system parameters:

$$\Phi \in \mathcal{R}^{n \times n} \ , \ \Gamma \in \mathcal{R}^{n \times r} \ , \ C \in \mathcal{R}^{p \times n} \ , \ D \in \mathcal{R}^{p \times r}$$

(Not necessarily) equidistant $(t_k - t_{k-1} = \Delta h)$

$$x(k) = x(t_k) \quad , \quad u(k) = u(t_k) \quad , \quad y(k) = y(t_k)$$

Discrete event systems – discrete time state space models

Generalization of discrete time linear state space models

$$x(k+1) = \Psi(x(k), u(k)) \qquad (state\ equation)$$
$$y(k) = h(x(k), u(k)) \qquad (output\ equation)$$

with given initial condition $x(0)$ and nonlinear state Ψ and output function h.

Discrete event system:

1. discrete time with non-equidistant sampling
2. the range space of the signals is discrete
3. event: change in the discrete value of a signal

Automaton - abstract model: $\mathbf{G} = (X, U, Y, f, g, x_0)$

- **finite set of states**: $X = \{x_1, x_2, ... x_n\}$
- **finite set of input events**: $U = \{\varepsilon; u_1, u_2, ..., u_m\}$
- **finite set of output events**: $Y = \{\varepsilon; y_1, y_2, ..., y_k\}$
- **(partial) state transition function**:
  $f : X \times U \to X$ e.g. $f(x_1, u_3) = x_2$
- **output function**:
  $g : X \times U \to Y$ e.g. $g(x_1, u_3) = y_1$ (Mealy automaton)
  $g : X \to Y$ e.g. $g(x_1) = y_2$ (Moore automaton)
- *initial state*: x_0

Graphical description: weighted directed graph

- **Vertices**: states (X)
- **Edges**: state transitions (f)
- **Edge weights**: input/output symbols (Mealy), input symbols (Moore)

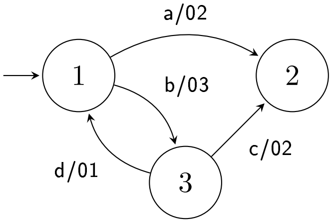
Operation of automata

Given

- Initial state: x_0
- The content of the input tape: $U = [u_1, u_2, \ldots, u_n], u_i \in U$

Compute

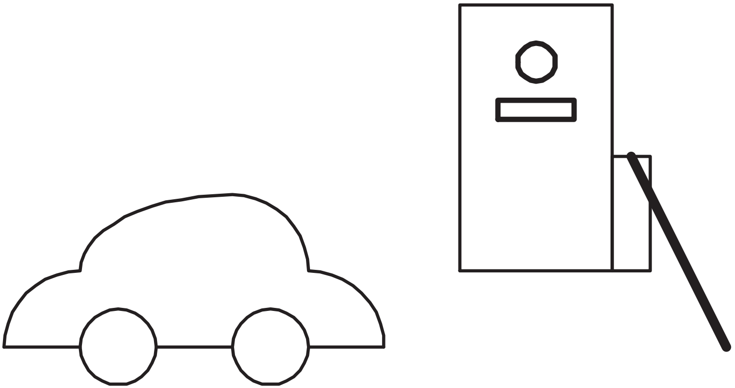
- The content of the output state: $Y = [y_1, y_2, ..., y_n]$, $y_i \in Y$



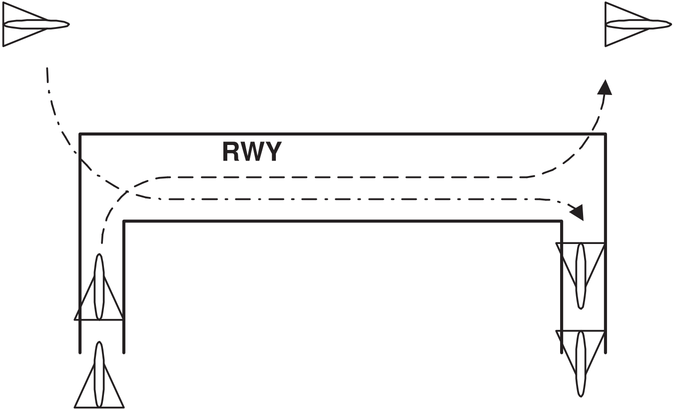
Automata - discrete event systems

	Automaton model	Discrete event state space model
State space	X	$\mathcal{X} \in \mathbb{Z}^n$
Input u	string from U	discrete time discrete valued signal
Output y	string from Y	discrete time discrete valued signal
State equation	$x(k+1) = f(x(k), u(k))$	$x(k+1) = \Psi(x(k), u(k))$
Output equation	$y(k) = g(x(k), u(k))$ (Mealy) $y(k) = g(x(k))$ (Moore)	$y(k) = h(x(k), u(k))$

Introductory example: Garage gate



Simple example: Runway



Overview - Petri nets: modelling and dynamics

1. Previous notions
2. Petri net models
   - Description forms
   - Operation (dynamics) of Petri nets
   - Parallel and conflicting execution steps
3. Solution of Petri net models

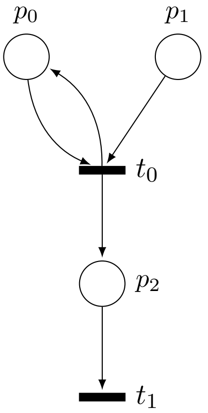
(ordinary) Petri net - abstract description: $\mathbf{PN} = (P, T, I, O)$

Static description (structure)

- set of **places (conditions)**: P
- set of **transitions (events)**: T
- **Input (pre-condition) function**: $I : T \to P^{\infty}$
- **Output (consequence) function**: $O : T \to P^{\infty}$

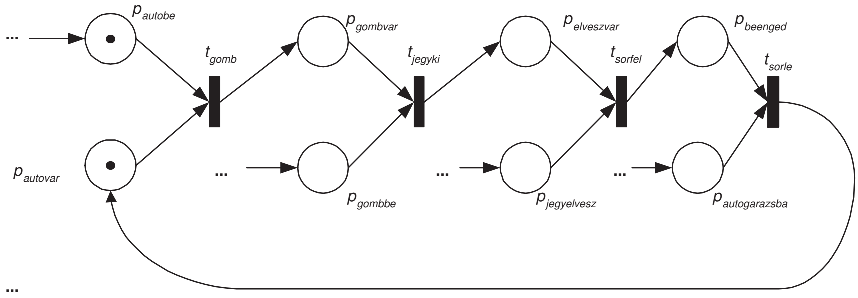
Graphical description: bipartite directed graph

- **Vertices**: places (P) and transitions (T) (partitions)
- **Edges**: input and output functions (I, O)



Example: garage gate – 1

Petri net model - graphical description



Example: garage gate – 2

Petri net model - formal description

Places (states; inputs):

$$P = \{p_{autovar}, p_{gombvar}, p_{elveszvar}, p_{beenged}\ ;$$

$$p_{autobe}, p_{gombbe}, p_{jegyelevesz}, p_{autogarazsba}\}$$

Transitions:

$$T = \{t_{gomb}, t_{jegyki}, t_{sorfel}, t_{sorle}\}$$

Input function:

$$I(t_{gomb}) = \{p_{autobe}, p_{autovar}\} \quad , \quad I(t_{jegyki}) = \{p_{gombbe}, p_{gombvar}\}$$
$$I(t_{sorfel}) = \{p_{jegyelvesz}, p_{elveszvar}\} \quad , \quad I(t_{sorle}) = \{p_{beenged}, p_{autogarazsba}\}$$

Output function:

$$O(t_{gomb}) = \{p_{gombvar}\} \quad , \quad O(t_{jegyki}) = \{p_{elveszvar}\}$$
$$O(t_{sorfel}) = \{p_{beenged}\} \quad , \quad O(t_{sorle}) = \{p_{autovar}\}$$

Dynamics of Petri nets

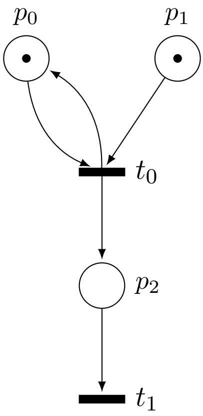
- **tokens** in places represent that the place is "active" (condition is "true")
- the **marking function** assigns tokens to each place:

$$\mu : \mathbf{P} \rightarrow \mathbb{N} \quad , \quad \mu(p_i) = \mu_i \geq 0$$

- the **marking vector** denotes the number of tokens on the places

$$\underline{\mu}^T = [\mu_1, \mu_2, \ldots, \mu_n] \quad , \quad n = |\mathbf{P}|$$

- **marked** Petri net : $PN = (P, T, I, O, \underline{\mu}^{(0)})$
  - $\underline{\mu}^{(0)}$ is the initial marking
- example: $\underline{\mu} = [1, 1, 0]^T$



Dynamics of Petri nets

A transition t is **enabled** when its pre-conditions are "true" (there is at least one **token** on its input places)

$$\mu(p) \geq 1 \; \forall p, \text{ where } I(t,p) \text{ exists}$$

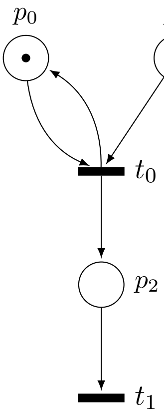
An enabled transition may **fire** (operate): it

"consumes" tokens from all of its input places and produces tokens in each output places

Notion: $\underline{\mu}^{(i)}[t_j > \underline{\mu}^{(i+1)}$

Firing (operation) sequence

$$\underline{\mu}^{(0)}[t_{j0} > \underline{\mu}^{(1)}[t_{j1} > \dots[t_{jk} > \underline{\mu}^{(k+1)}$$



Dynamics of Petri nets

A transition t is **enabled** when its pre-conditions are "true" (there is at least one **token** on its input places)

$$\mu(p) \geq 1 \; \forall p, \text{ where } I(t, p) \text{ exists}$$

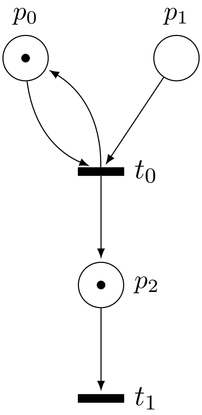
An enabled transition may **fire** (operate): it

"consumes" tokens from all of its input places and produces tokens in each output places

Notion: $\underline{\mu}^{(i)}[t_j > \underline{\mu}^{(i+1)}$

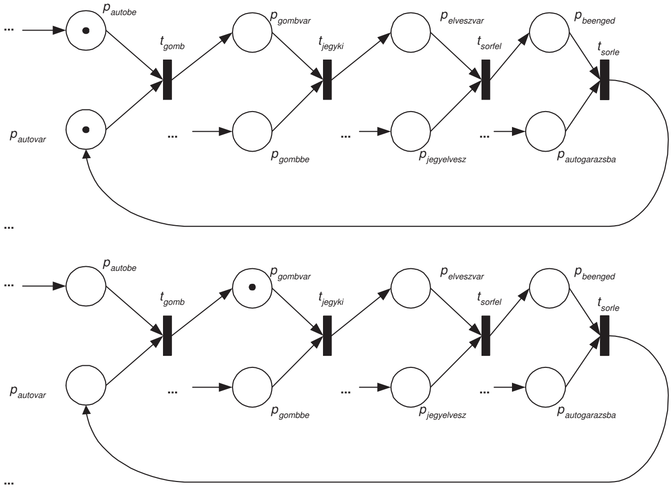
Firing (operation) sequence

$$\underline{\mu}^{(0)}[t_{j0} > \underline{\mu}^{(1)}[t_{j1} > \ldots[t_{jk} > \underline{\mu}^{(k+1)}$$



Example: garage gate – 3

One operation steps



Example: garage gate – 4

Formal description of an operation step

Marking vector

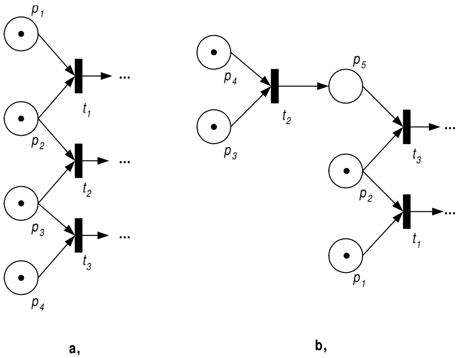
$$\underline{\mu}^T = [\mu_{autovar}, \mu_{gombvar}, \mu_{elveszvar}, \mu_{beenged} ; \mu_{autobe}, \mu_{gombbe}, \mu_{jegyelevesz}, \mu_{autogarazsba}]$$

Operation (firing) of transition t_{gomb}

$$\underline{\mu}^{(1)}[t_{gomb} > \underline{\mu}^{(2)}$$
$$\underline{\mu}^{(1)} = [1,\ 0,\ 0,\ 0\ ;\ 1,\ 0,\ 0,\ 0]^T$$
$$\underline{\mu}^{(2)} = [0,\ 1,\ 0,\ 0\ ;\ 0,\ 0,\ 0,\ 0]^T$$

Parallel events

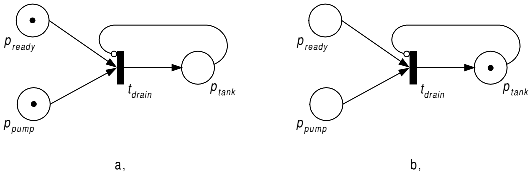
More than one enabled (fireable) transition:
concurrency (independent conditions), conflict, confusion



Conflict resolution

Using **inhibitor edges**:
- priority given by the user
- test edges

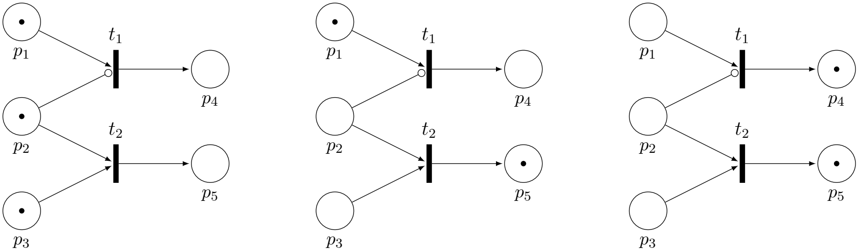
Other solutions:
- capacity of the places



Conflict resolution

Inhibitor edges - Example

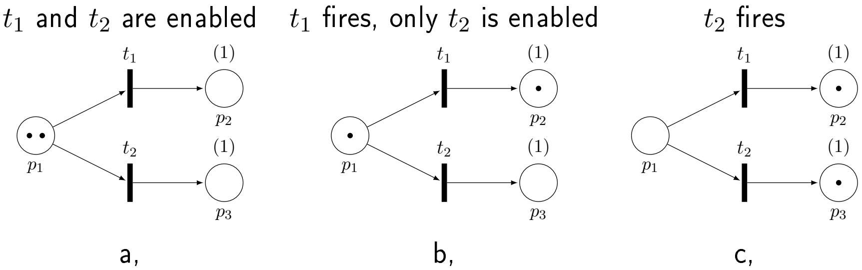
t_2 is enabled | t_2 fires, t_1 becomes enabled | t_1 fires



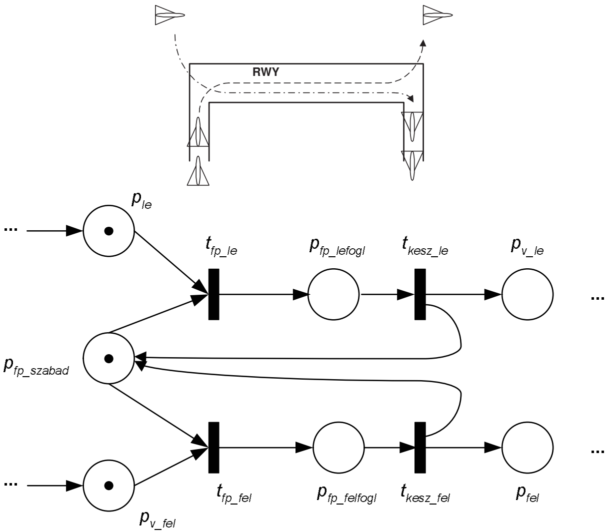
a, b, c,

Conflict resolution

Capacity of places - Example

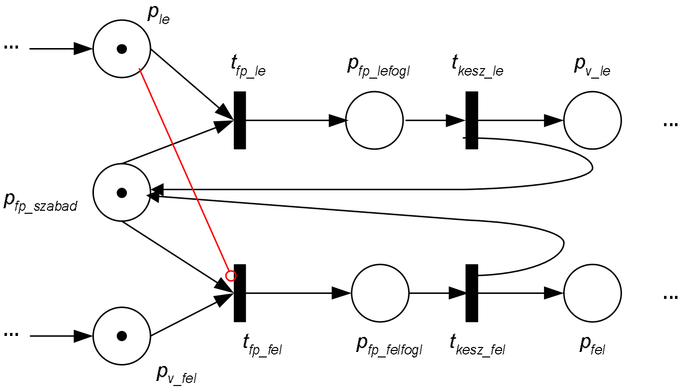


Petri net model of a runway – 1



Petri net model of a runway – 2

Conflict resolution: landing aircraft has priority



Overview - Solution of Petri net models

The solution problem

Abstract problem statement

Given:

- a *formal description* of a discrete event system model
- *initial state(s)*
- *external events*: system inputs

Compute:

- the sequence of *internal (state and output) events*

The solution is **algorithmic**! **The problem is NP-hard**!

Petri net models – reachability graph

Solution: marking (systems state) sequences
reachability graph (tree) (weighted directed graph)

- *vertices*: markings
- *edges*: if exists transition the firing of which connects them
- *edge weights*: the transition and the external events

Construction:

1. *start*: at the given initial state (marking)
2. *adding a new vertex*: by firing an enabled transition (with the effect of inputs!)

May be NP-hard (in conflict situation or non-finite operation)

The state space of Petri net models

State vector: marking in *internal* places

in- and out-degree is at least 1

$$x(k) \quad \sim \quad \underline{\mu}_x^{(k)}$$

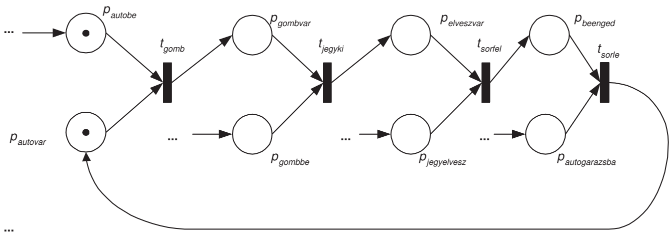
Inputs: marking in *input* places

in-degree is zero

$$u(k) \quad \sim \quad \underline{\mu}_u^{(k)}$$

Example: garage gate

Petri net model



$$\underline{\mu}_x^T = [\mu_{autovar}, \mu_{gombvar}, \mu_{elveszvar}, \mu_{beenged}]$$

$$\underline{\mu}_u^T = [\mu_{autobe}, \mu_{gombbe}, \mu_{jegyelevesz}, \mu_{autogarazsba}]$$