Timed DES

Pedro U. Lima

Instituto Superior Técnico (IST)
Instituto de Sistemas e Robótica (ISR)
Av.Rovisco Pais, 1
1049-001 Lisboa
PORTUGAL

November 2002

TIMED DES

- Sample paths can no longer be specified as event sequences $\{e_1, e_2, \ldots\}$ or state sequences $\{x_0, x_1, \ldots\}$ but must include timing information
- Let t_k denote the time instant when the k^{th} transition occurs (with t_0 given) – a timed sample path of a DES may now be described by the sequence

$$\{(x_0,t_0), (x_1,t_1), \ldots\} \text{ or } \{(e_0,t_0), (e_1,t_1), \ldots\}$$

- This way, we can answer questions like:
  - *How many events of a particular type can occur in a given time interval*?
  - *How long does the system spend in a given state*?
- The language generated by a timed model consists of a *single* string. Probabilistic strings are introduced by *stochastic timed* models. Random time associated to transitions is typically described in the literature.

TIMED AUTOMATA

First, let's slightly change our definition of automaton
(no definition of timed automaton here yet!)

An automaton is a tuple (X, E, f, Γ, x_0) , where

X is a *countable* state space

E is a *countable* event set

$f : X \times E \rightarrow X$ is a (possibly partial) state transition function

$\Gamma : X \rightarrow 2^E$ active event function

x_0 is the initial state

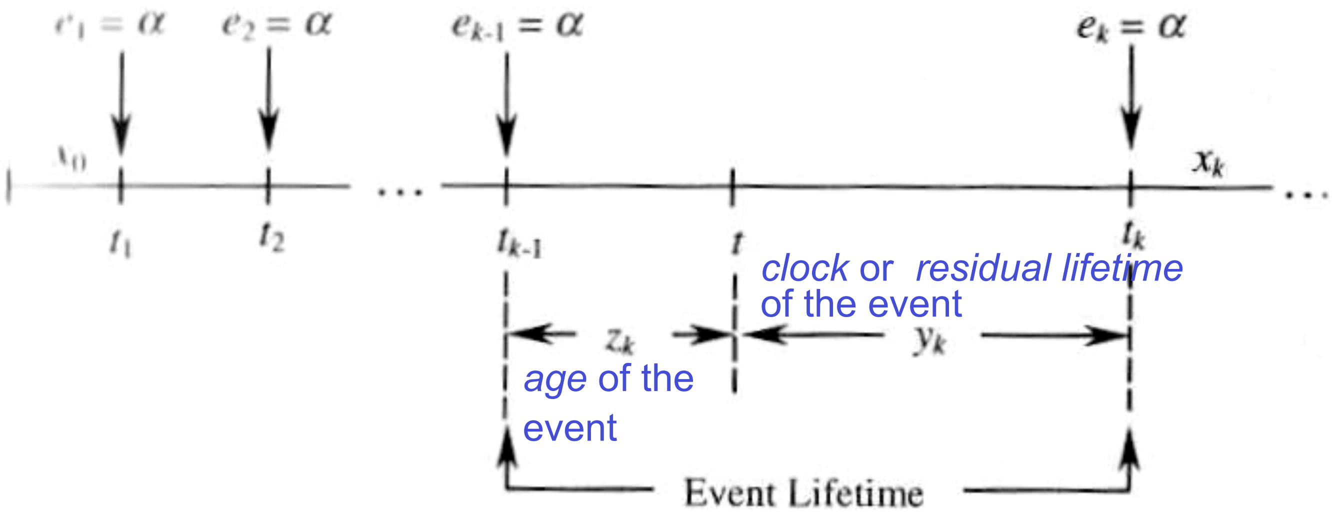
~~X_M is a set of marked states~~ **we will not be concerned with blocking issues**

TIMED AUTOMATA

The Clock Structure

A DES with a single event

$$E = \{\alpha\},\ \Gamma(x) = \{\alpha\}, \forall_{x \in X}$$



(reprinted from [Cassandras, Lafortune])

$v_k = t_k - t_{k-1} \quad v_k \in \Re^+,\ k = 1,2,\ldots$ *K*th *lifetime* of event α

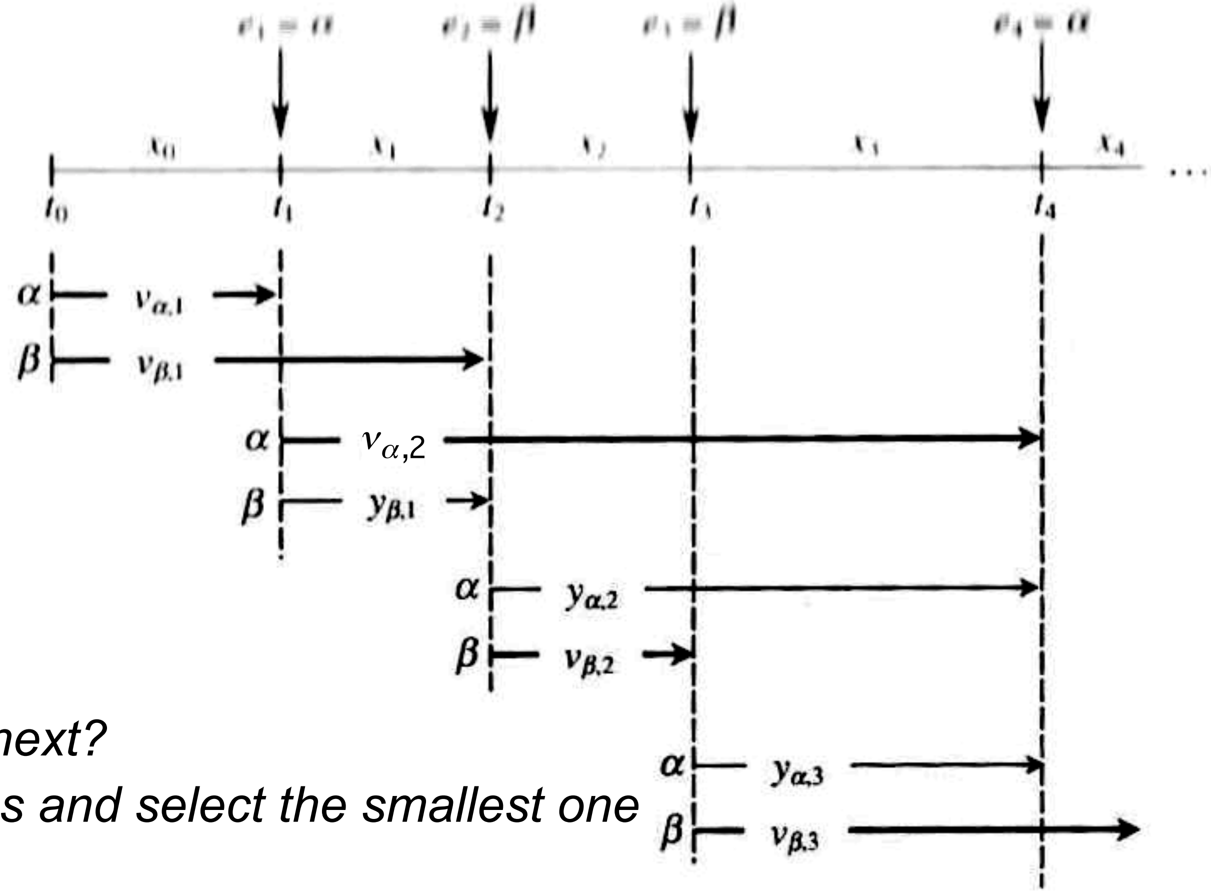
- **at t_{k-1}, the event becomes activated or enabled, if it is in $\Gamma(x)$**
- **during the time interval $]t_{k-1}, t_k]$, the event is active**
- **at t_k, the event occurs**

TIMED AUTOMATA

The Clock Structure

A DES with two permanently active events

$E = \{\alpha, \beta\}, \ \Gamma(x) = \{\alpha, \beta\}, \forall_{x \in X}$



$$\mathbf{v}_\alpha = \{v_{\alpha,1}, v_{\alpha,2}, \ldots\}$$
$$\mathbf{v}_\beta = \{v_{\beta,1}, v_{\beta,2}, \ldots\}$$

Q: *Which event occurs next?*

A: *Compare clock values and select the smallest one*

(reprinted from [Cassandras, Lafortune])

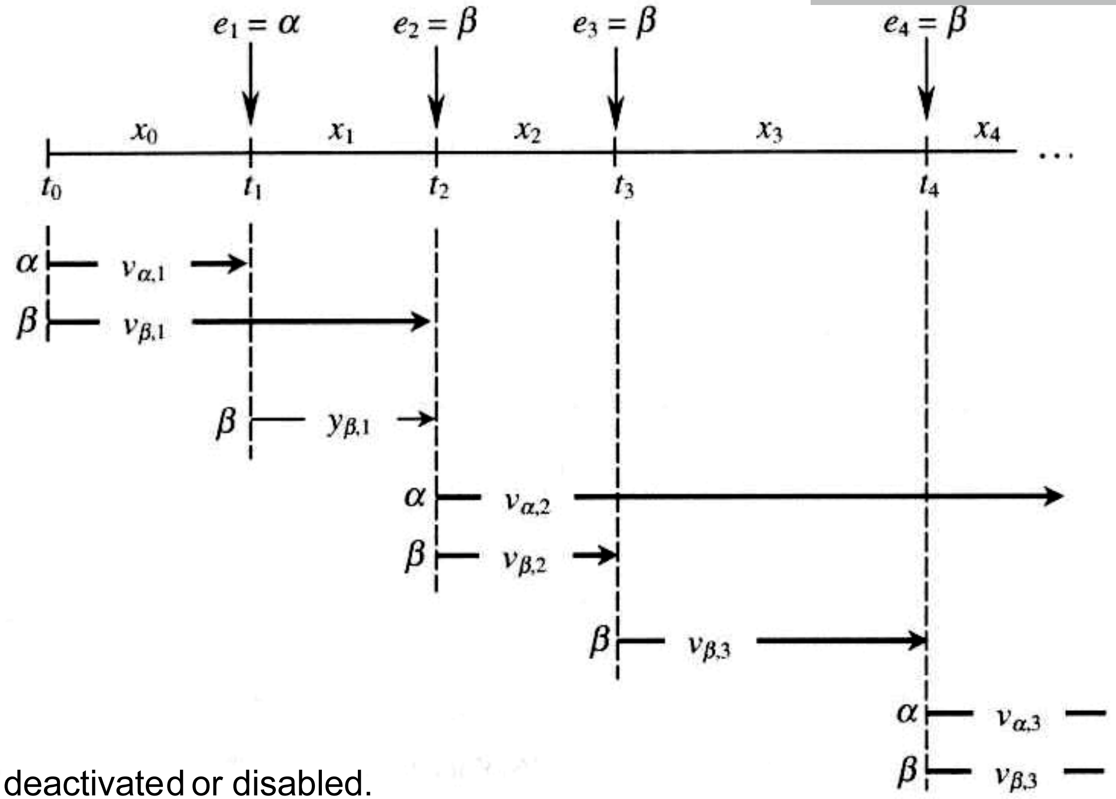
TIMED AUTOMATA

The Clock Structure

A DES with two not permanently active events

$E = \{\alpha, \beta\},\ \Gamma(x) = \{\alpha, \beta\}$ for some $x \in X$

$\Gamma(x) = \{\beta\}$ for the remaining $x \in X$



$e.g.,\ \Gamma(x_3) = \{\beta\} \Rightarrow \alpha$ is deactivated or disabled.

Its clock becomes irrelevant in the determination of the next event and it is, therefore, ignored.

$v_{\alpha,2}$ is discarded when α is deactivated at t_3.

(reprinted from [Cassandras, Lafortune])

TIMED AUTOMATA

The Clock Structure

General mechanism for selecting the "next event":

Rule 1: To determine the next event, compare the clock values of all feasible events at the current state and select the smallest one.

Rule 2: An event *e* is activated when:

- *e* has just occurred and remains feasible in the new state
- a different event occurs while *e* was not feasible, causing a transition to a new state where *e* is feasible

Rule 3: An event *e* is deactivated when a different event occurs causing a transition to a new state where *e* is not feasible.

TIMED AUTOMATA

Definition: Clock Structure or **Timing Structure** associated with an event set E is a set

$$V = \{\mathbf{v}_i : i \in E\}$$

of clock (or lifetime) sequences

$$\mathbf{v}_i = \{v_{i,1}, v_{i,2}, \ldots\}, \; i \in E, \; v_{i,k} \in \Re^+, \; k = 1, 2, \ldots$$

Definition: The **Score**, $N_{i,k}$ of an event $i \in E$ after the k^{th} state transition on a given sample path is the number of times that i has been activated in the interval $[t_0, t_k]$.

The score of an event i serves as a pointer to its clock sequence $\mathbf{v}_i$, which specifies the next lifetime to be assigned to its clock when i is activated.

TIMED AUTOMATA
Event Timing Dynamics

Untimed view

$$\{e_1, e_2, \ldots, e_{k+1}, \ldots\} \rightarrow \boxed{\mathbf{x}_{k+1} = f(\mathbf{x}_k, e_{k+1}),\ k = 0,1,\ldots} \xrightarrow{?}$$

Timed view

assuming *m* events

$$\mathbf{V} = \{\mathbf{v}_i \ : \ i = 1,\ldots,m\} \rightarrow \boxed{\begin{array}{c} e_{k+1} = h(\mathbf{x}_k, \mathbf{v}_1, \ldots, \mathbf{v}_m) \\ \mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{v}_1, \ldots, \mathbf{v}_m) \end{array}} \xrightarrow{?}$$

Q.: Is it enough?...

A.: No. We need some other information •

$$e_{k+1} = h(\mathbf{x}_k, \mathbf{v}_1, \ldots, \mathbf{v}_m, \bullet)$$

TIMED AUTOMATA
Event Timing Dynamics

x is the current state
e is the most recent event, which caused transition into *x*
t is the most recent event time (corresponding to *e*)

N_i is the current *score* of event *i*, $N_i \in \{0,1,...\}$
y_i is the current *clock* value of event *i*, $y_i \in \Re^+$

e' is the next event, or the *triggering event* ($e' \in \Gamma(x)$)
t' is the next event time (corresponding to *e'*)
x' is the next state ($x'=f(x,e')$)

N_i' is the next *score* of event *i*, after *e'* occurs
y_i', is the next *clock* value of event *i*, after *e'* occurs

Note: *x* corresponds to x_k, *x'* to x_{k+1}, and similarly for (e,e'), (t,t')

TIMED AUTOMATA
Event Timing Dynamics

Step 1. Since x is known, we can evaluate the feasible event set $\Gamma(x)$.

Step 2. Associated with each event $i \in \Gamma(x)$ is a clock value y_i. We can then determine the smallest clock value among those, denoted by y^*:

$$y^* = \min_{i \in \Gamma(x)} \{y_i\} \qquad (1)$$

Step 3. Determine the triggering event, e', as the value of i in the previous equation that defines y^*. We express this as

$$e' = \arg \min_{i \in \Gamma(x)} \{y_i\} \qquad (2)$$

Step 4. With e' defined by the previous equation, determine the next state:

$$\mathbf{x}' = f(\mathbf{x}, e') \qquad (3)$$

Step 5. With y^* defined by (1) determine the next event time:

$$t' = t + y^* \qquad (4)$$

TIMED AUTOMATA
Event Timing Dynamics

This process then repeats with *x'*, *e'* and *t'* specified. Step 2, however, requires the new clock values y_i'. Therefore, at least one more step is needed:

Step 6. Determine the new clock values *for* all new feasible events $i \in \Gamma(x')$:

$$y_i' = \begin{cases} y_i - y^* & \text{if } (i \neq e') \wedge i \in \Gamma(x) \\ v_{i,N_i+1} & \text{if } (i = e') \vee i \notin \Gamma(x) \end{cases} \qquad i \in \Gamma(x') \qquad (5)$$

Step 7. Determine the new scores values for all new feasible events $i \in \Gamma(x')$:

$$N_i' = \begin{cases} N_i + 1 & \text{if } (i = e') \vee i \notin \Gamma(x) \\ N_i & \text{if } (i \neq e') \wedge i \in \Gamma(x) \end{cases} \qquad i \in \Gamma(x') \qquad (6)$$

Note: y^* is the *interevent* time.

TIMED AUTOMATA

Definition: A *Timed Automaton* is a six-tuple

$$(X,E,f,\Gamma,x_0,\mathbf{V})$$

where $\mathbf{V} = \{\mathbf{v}_i : i \in E\}$ is a clock structure, and $(\mathbf{X},E,f,\Gamma,x_0)$ is an automaton. The automaton generates a state sequence $x'=f(x,e')$ driven by an event sequence $\{e_1,e_2,\ldots\}$ generated through

$$e' = \arg\min_{i \in \Gamma(x)} \{y_i\}$$

with the *clock values* y_i, $i \in E$, defined by

$$y_i' = \begin{cases} y_i - y^* & \text{if } (i \neq e') \wedge i \in \Gamma(x) \\ v_{i,N_i+1} & \text{if } (i = e') \vee i \notin \Gamma(x) \end{cases} \qquad i \in \Gamma(x')$$

where the *interevent time* y^* is defined as

$$y^* = \min_{i \in \Gamma(x)} \{y_i\}$$

TIMED AUTOMATA

and the *event scores* N_i, $i \in E$, are defined by

$$N_i' = \begin{cases} N_i + 1 & \text{if } (i = e') \vee i \notin \Gamma(x) \\ N_i & \text{if } (i \neq e') \wedge i \in \Gamma(x) \end{cases} \qquad i \in \Gamma(x')$$

In addition, initial conditions are: $y_{i,0} = v_{i,1}$ and $N_{i,0} = 1$ for all $i \in \Gamma(x_0)$. If $i \notin \Gamma(x_0)$, then y_i is undefined and $N_{i,0} = 0$.

$$\mathbf{v}_1 = \{v_{1,1}, v_{1,2}, \ldots\} \quad \vdots \quad \mathbf{v}_m = \{v_{m,1}, v_{m,2}, \ldots\}$$

$$\mathbf{x}' = f^x(\mathbf{x}, \mathbf{y}, \mathbf{N}, \mathbf{v}_1, \ldots, \mathbf{v}_m)$$

$$y' = f^y(\mathbf{x}, \mathbf{y}, \mathbf{N}, \mathbf{v}_1, \ldots, \mathbf{v}_m)$$

$$N' = f^N(\mathbf{x}, \mathbf{y}, \mathbf{N}, \mathbf{v}_1, \ldots, \mathbf{v}_m)$$

$$\{(e_1, t_1), (e_2, t_2), \ldots\}$$

INSTITUTO SUPERIOR TÉCNICO

TIMED PN MODELS

- A positive real number v_{jk} is assigned to t_j, meaning that, when t_j is *enabled* for the k^{th} time, it does not fire immediately, but incurs a firing delay given by v_{jk} ; during this delay, tokens are kept in the input places of t_j .
- $T = T_0 \cup T_D$

immediate (thin bar) — **timed** (thick bar)

Def.: The *clock structure* associated with $T_D \subseteq T$ of a marked PN (P,T,A,w,x) is a set $\mathbf{V} = \{\mathbf{v}_j : t_j \in T_D\}$ of *lifetime sequences*
$\mathbf{v}_j = \{v_{j1}, v_{j2}, \ldots\}$, $t_j \in T_D$, $v_{jk} \in \mathrm{R}^+$, $k=1,2,\ldots$

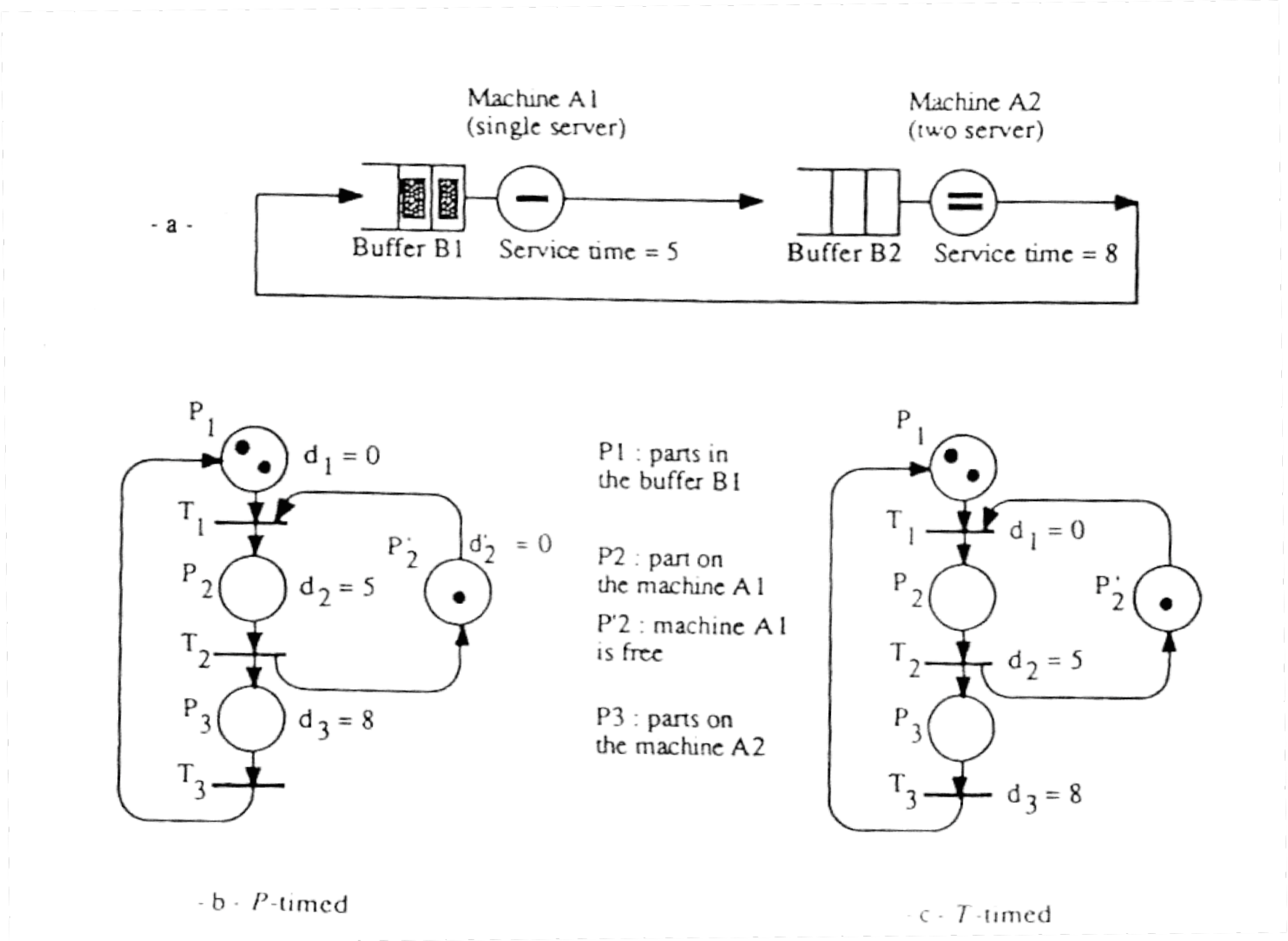
Def.: A *Timed PN* is a six-tuple $(P,T,A,w,x,\mathbf{V})$ where (P,T,A,w,x) is a marked PN, and $\mathbf{V} = \{\mathbf{v}_j : t_j \in T_D\}$ is a *clock structure*.

TIMED PN MODELS

P-Timed	***T-Timed***
firing delay $d_i \geq 0$ associated to place p_i, $\forall\, p_i \in P$	firing delay $d_i \geq 0$ associated to transition t_i, $\forall\, t_i \in T$
when a token is deposited in p_i, it must stay there for at least a time d_i	a token has 2 possible states: • *reserved* • *non-reserved* regarding firing of t_i.
the token becomes *available* after time d_i expires	
only *available* tokens enable transitions	only *non-reserved* tokens enable transitions
A transition is fired as soon as it becomes *enabled*	A transition t_i is fired after a firing delay d_i that starts when the token becomes reserved by t_i.

P-timed and *T-timed* PNs are equivalent

TIMED PN MODELS



Two equivalent PNs modeling a manufacturing system.

Reprinted from [2].

TIMED PN MODELS

- v_{jk} is the lifetime of the event associated to t_j , when t_j becomes ready to be enabled (as soon as input tokens become non-reserved) for the k^{th} time , $k = 1, 2, \ldots$
- τ_{jk} is the k^{th} firing time of t_j
- π_{ik} is the time instant when place p_i receives its k^{th} token.

Goal: To express τ_{jk} as a function of τ_{jk-1} and v_{jk} .

TIMED PN MODELS

For Marked Graphs:

p_i has only one input transition t_r and $x(p_i)=x_{i0}$ initially:

$$\pi_{ik} = \tau_{r,k-x_{i0}}, \; p_i \in O(t_r), \; k = x_{i0}+1, x_{i0}+2, \ldots$$

p_i has only one output transition t_j :

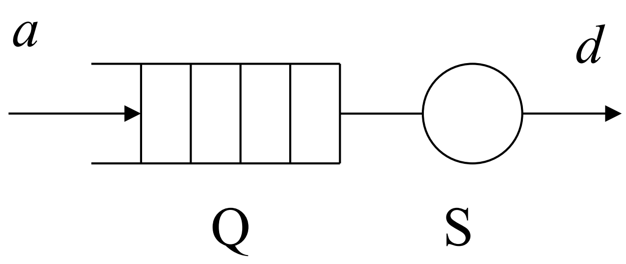
$$\tau_{j,k} = \pi_{ik}, \; p_i \in I(t_j), \; t_j \in T_0, \; k = 1,2,\ldots$$

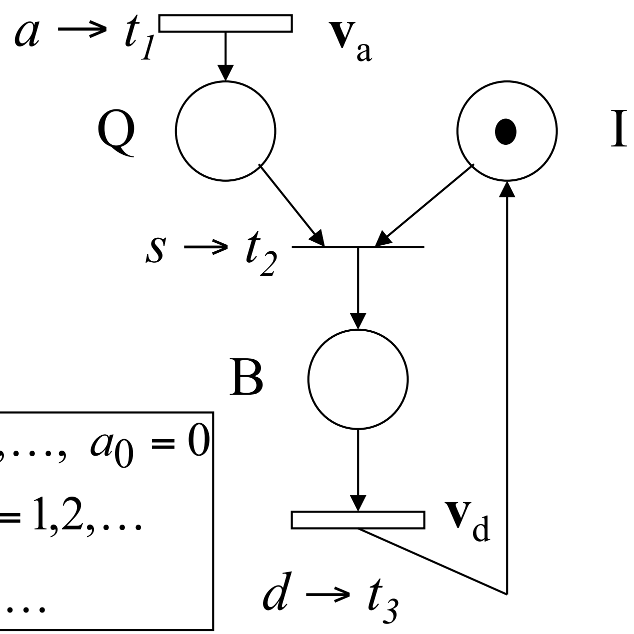
$$\tau_{j,k} = \pi_{ik} + \nu_{jk}, \; p_i \in I(t_j), \; t_j \in T_D, \; k = 1,2,\ldots$$

p_i is not the only input place of t_j :

$$\tau_{j,k} = \max_{p_i \in I(t_j)} \{\pi_{ik}\} + \nu_{jk}, \; t_j \in T_D, \; k = 1,2,\ldots$$

TIMED PN MODELS OF TIMED QUEUEING SYSTEM





$$\pi_{Qk} = a_k, k = 1,2,\ldots$$
$$\pi_{Ik} = d_{k-1}, k = 2,3,\ldots, \pi_{i1} = 0$$
$$\pi_{Bk} = s_k, k = 1,2,\ldots$$

$$a_k = a_{k-1} + v_{ak},\ k = 1,2,\ldots,\ a_0 = 0$$
$$s_k = \max\{\pi_{Qk}, \pi_{Ik}\},\ k = 1,2,\ldots$$
$$d_k = \pi_{Bk} + v_{dk},\ k = 1,2,\ldots$$

$$\tau_{1k} = a_k := k\text{th arrival time}$$
$$\tau_{2k} = s_k := k\text{th service start time}$$
$$\tau_{3k} = d_k := k\text{th departure time}$$

$$a_k = a_{k-1} + v_{ak},\ k = 1,2,\ldots,\ a_0 = 0$$
$$d_k = \max\{a_{k-1} + v_{ak}, d_{k-1}\} + v_{dk},\ k = 1,2,\ldots\ d_0 = 0$$

(max,+) algebra

- dioid algebra (two operations):
  - ***addition:*** $a \oplus b = \max\{a,b\}$
  - ***multiplication:*** $a \otimes b = a + b$
- $\oplus$ and $\otimes$ are commutative
- $\oplus$ and $\otimes$ are associative
- $\otimes$ is distributive over $\oplus$
- $\eta = -\infty$ is the null element of $\oplus$
- $\eta = -\infty$ is the absorbing null element of $\otimes$
- Idempotency in $\oplus$:

$$a \oplus a = \max\{a, a\} = a$$

TIMED PN MODELS OF TIMED QUEUEING SYSTEM

(max,+) algebra model of queueing system

$$a_k = a_{k-1} + v_{ak}, \quad a_0 = 0$$
$$d_k = \max\{a_{k-1} + v_{ak}, d_{k-1}\} + v_{dk}, \quad d_0 = 0$$

$-L$ such that

$\max\{a_{k-1} \otimes v_{ak}, d_{k-1} \otimes -L\} = a_{k-1} \otimes v_{ak}$

$$a_k = \max\{a_{k-1} \otimes v_{ak}, d_{k-1} \otimes -L\}, \quad a_0 = 0$$
$$d_k = \max\{a_{k-1} \otimes v_{ak}, d_{k-1}\} \otimes v_{dk}, \quad d_0 = 0$$

$$a_k = (a_{k-1} \otimes v_{ak}) \oplus (d_{k-1} \otimes -L), \quad a_0 = 0$$
$$d_k = (a_{k-1} \otimes v_{ak} \otimes v_{dk}) \oplus (d_{k-1} \otimes v_{dk}), \quad d_0 = 0$$

$$\begin{bmatrix} a_k \\ d_k \end{bmatrix} = \begin{bmatrix} v_{ak} & -L \\ v_{ak} + v_{dk} & v_{dk} \end{bmatrix} \begin{bmatrix} a_{k-1} \\ d_{k-1} \end{bmatrix}$$

$$\mathbf{x}_k = \mathbf{A}\mathbf{x}_{k-1}, \quad \mathbf{x}_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

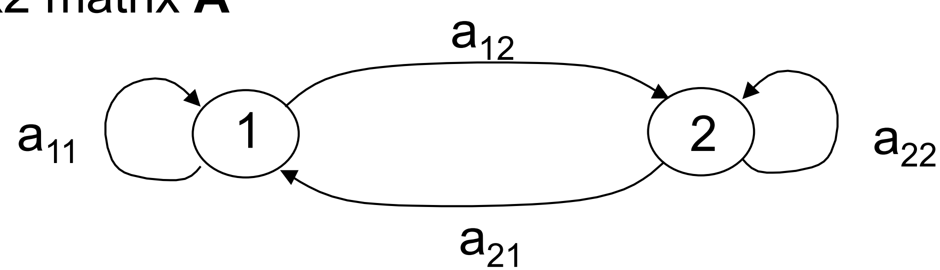
handling the system matrix

Form a graph where:

Number of nodes is equal to the dimension of the square matrix **A**.

Each arc corresponds to a matrix entry, and has an associated weight equal to that entry value.

Example for a 2x2 matrix **A**



Every closed loop in the graph forms a *circuit*.

Simple circuits are those where each node is only visited once (e.g., (1,1), (1,2,1), (2,2)).

The *length* of a circuit is the number of arcs forming the circuit.

The *weight* of a circuit is the sum of arc weights in the circuit.

Average weight of a circuit = *weight / length* of the circuit.

TIMED PN MODELS OF TIMED QUEUEING SYSTEM

critical circuit of the system matrix

Def.: The *critical circuit* of a matrix **A** is the circuit with maximum average weight.

Matrix **A** has an **eigenvalue** λ equal to the average weight of its critical circuit.

periodicity of the system matrix

Def.: Under the (max,+) algebra, a matrix **M** is said to be n-periodic iff there exists an integer k^* such that $\mathbf{M}^{k+n} = \mathbf{M}^k$, for all $k > k^*$

If there is an unique critical circuit whose length is n, then the matrix $\mathbf{B} = \mathbf{A}/w$ is n-periodic, where w is the average weight of this critical circuit.

Note: in the (max,+) algebra, $\mathbf{A}/w$ is defined so that $w\mathbf{B} = \mathbf{A}$.

TIMED DES

Further reading

- Queueing systems as timed automata
- Event scheduling scheme
- More on (max,+) algebra

Other references

• G. Cohen, P. Moller, J.-P. Quadrat, M. Viot, "Algebraic Tools for Performance Evaluation in Discrete Event Systems", *Proceedings of the IEEE,* Jan 1989 - *original paper on the (max,+) algebra*