System Modeling with Petri Nets

Andrea Bobbio and Kishor Trivedi
Dipartimento di Informatica
Università del Piemonte Orientale, 15100 Alessandria (Italy)
bobbio@unipmn.it - URL: www.mfn.unipmn.it/~bobbio

Center for Advanced Computing and Communication (CACC)
Department of Electrical and Computer Engineering
Duke University, Durham, NC 27708-0291(USA)
kst@ee.duke.edu - URL: www.ee.duke.edu/~kst

Duke University, November 2003

Outline

- What are Petri Nets;
- Definitions and basic concepts;
- Examples;
- Stochastic Petri Net (SPN);
- Generalized SPN and Stochastic Reward Net (SRN).

A Monograph on this subject is: *http://www.mfn.unipmn.it/~bobbio/BIBLIO/PAPERS/ANNO90/kluwerpetrinet.pdf*

Petri Nets

Petri Nets (PN) are a graphical paradigm for the formal description of the logical interactions among parts or of the flow of activities in complex systems.

PN are particularly suited to model:

- Concurrency and Conflict;
- Sequencing, conditional branching and looping;
- Synchronization;
- Sharing of limited resources;
- Mutual exclusion.

Petri Nets

Petri Nets (PN) originated from the Phd thesis of Carl Adam Petri in 1962.

A web service on PN is managed at the University of Aarhus in Denmark, where a bibliography with more that 8,500 items can be found.

http://www.daimi.au.dk/PetriNets/

Regular International Conferences:

⇒ ATPN - Application and Theory of PN

⇒ PNPM – PN and Performance Models

Petri Nets

The original PN did not convey any notion of time.

For performance and dependability analysis it is necessary to introduce the duration of the events associated to PN transitions.

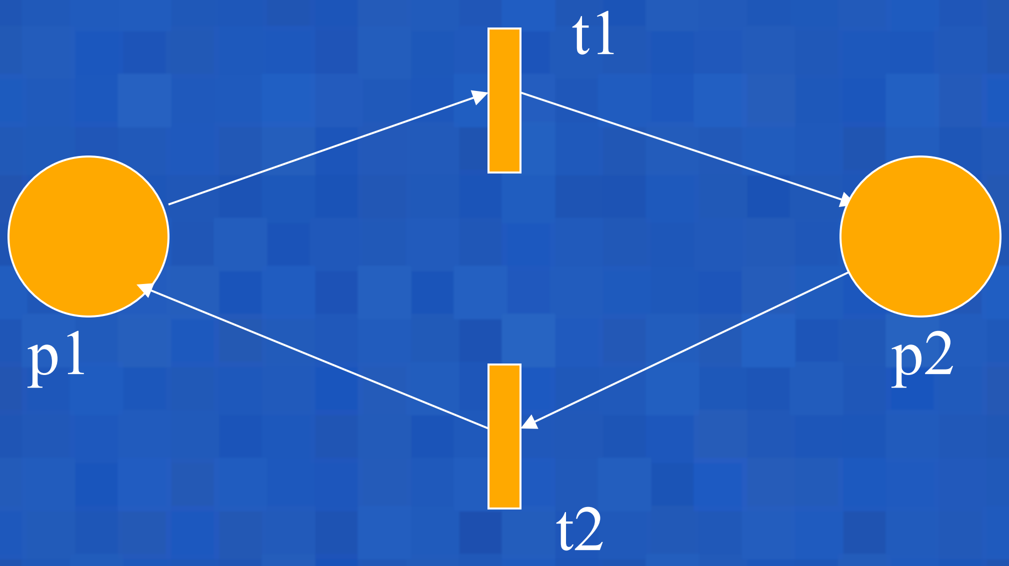
Timed model were subsequently extensively explored, following two main lines:

- □ Random durations : → Stochastic PN (SPN)
- □ Deterministic or interval: → Timed PN (TPN)

Definitions

- A Petri net (PN) is a *bipartite directed* graph consisting of two kinds of nodes: *places* and *transitions*
    - Places typically represent conditions within the system being modeled ○
    - Transitions represent events occurring in the system that may cause change in the condition of the system |
    - Arcs connect places to transitions and transitions to places (never an arc from a place to a place or from a transition to a transition)

Example of a PN



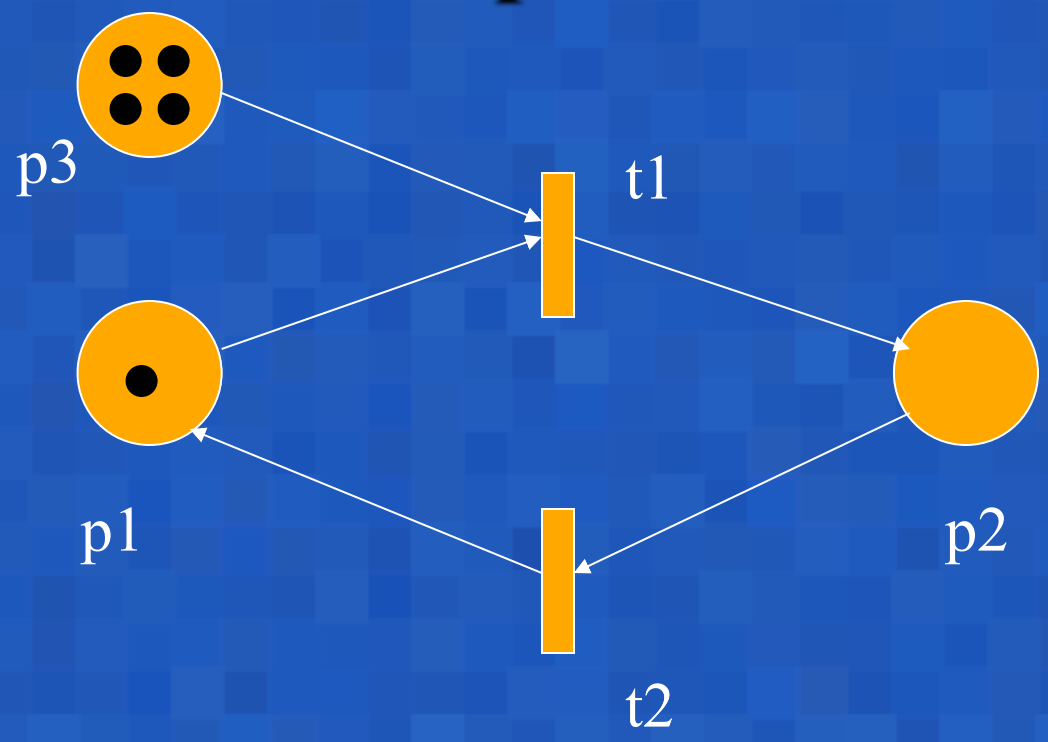
p1 – resource idle

p2 – resource busy

t1 – task arrives

t2 – task completes

Example of a PN



p1 – resource idle

p2 – resource busy

p3 – user

t1 – task arrives

t2 – task completes

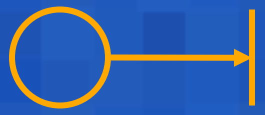
Definition of PN

A PN is a n-tuple (P,T,I,O,M)

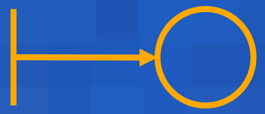
P	set of places
T	set of transitions
I	input arcs
O	output arcs
M	marking

PN Definitions

- Input arcs are directed arcs drawn from places to transitions, representing the conditions that need to be satisfied for the event to be activated

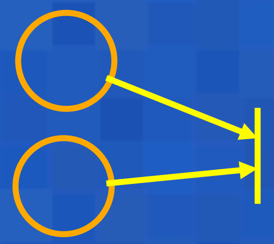


- Output arcs are directed arcs drawn from transitions to places, representing the conditions resulting from the occurrence of the event

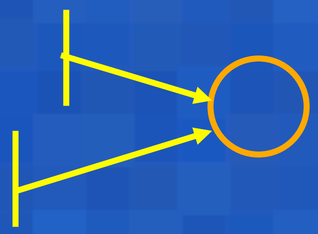


PN Definitions

- Input places of a transition are the set of places that are connected to the transition through input arcs

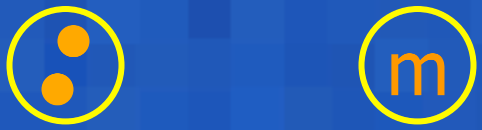


- Output places of a transition are the set of places to which output arcs exist from the transition



PN Definitions

- Tokens are dots (or integers) associated with places; a place containing tokens indicates that the corresponding condition holds

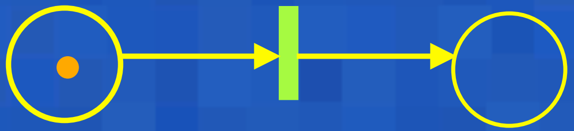


- Marking of a Petri net is a vector listing the number of tokens in each place of the net

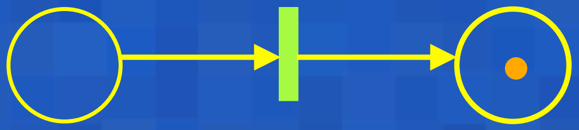
$(m_1\ m_2\ \dots\ m_P)$; P = # of Places

PN Definitions

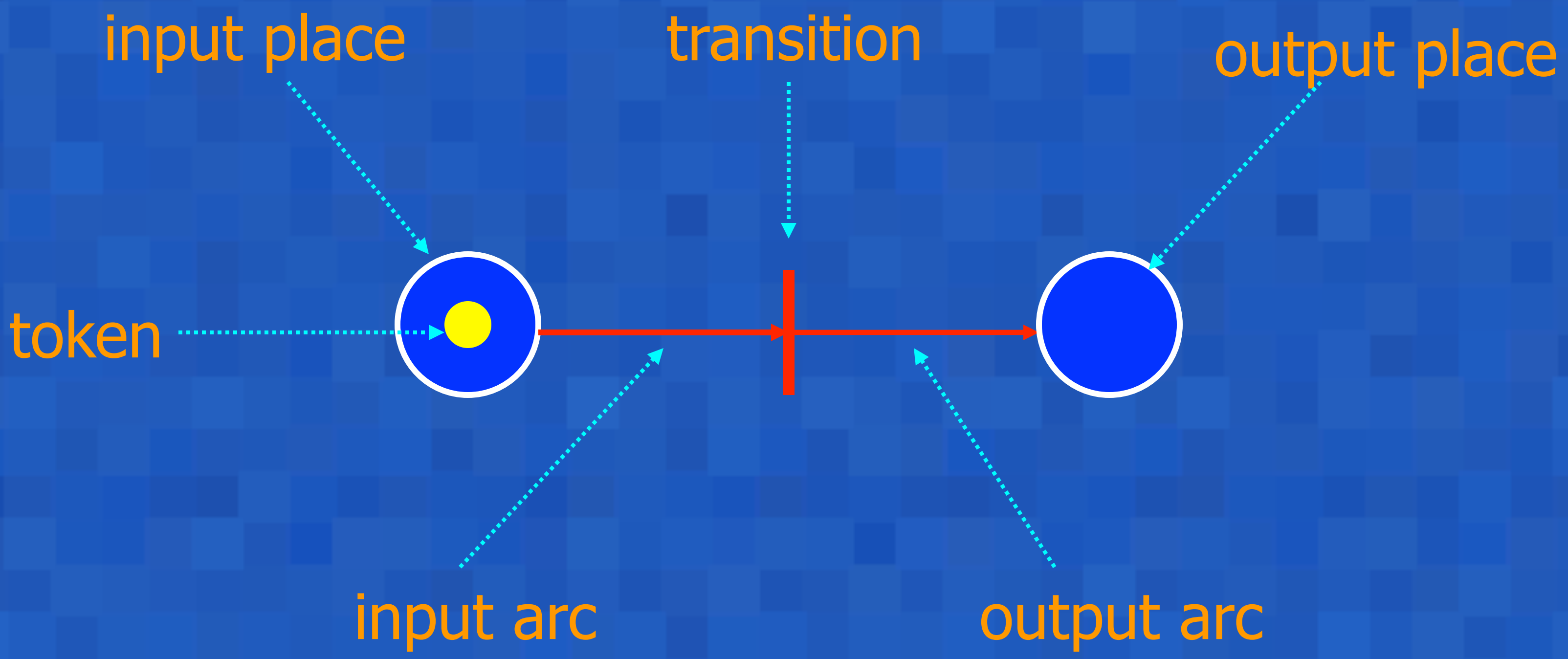
- When input places of a transition have the required number of tokens, the transition is enabled.



- An enabled transition may fire (event happens) removing one token from each input place and depositing one token in each of its output place.

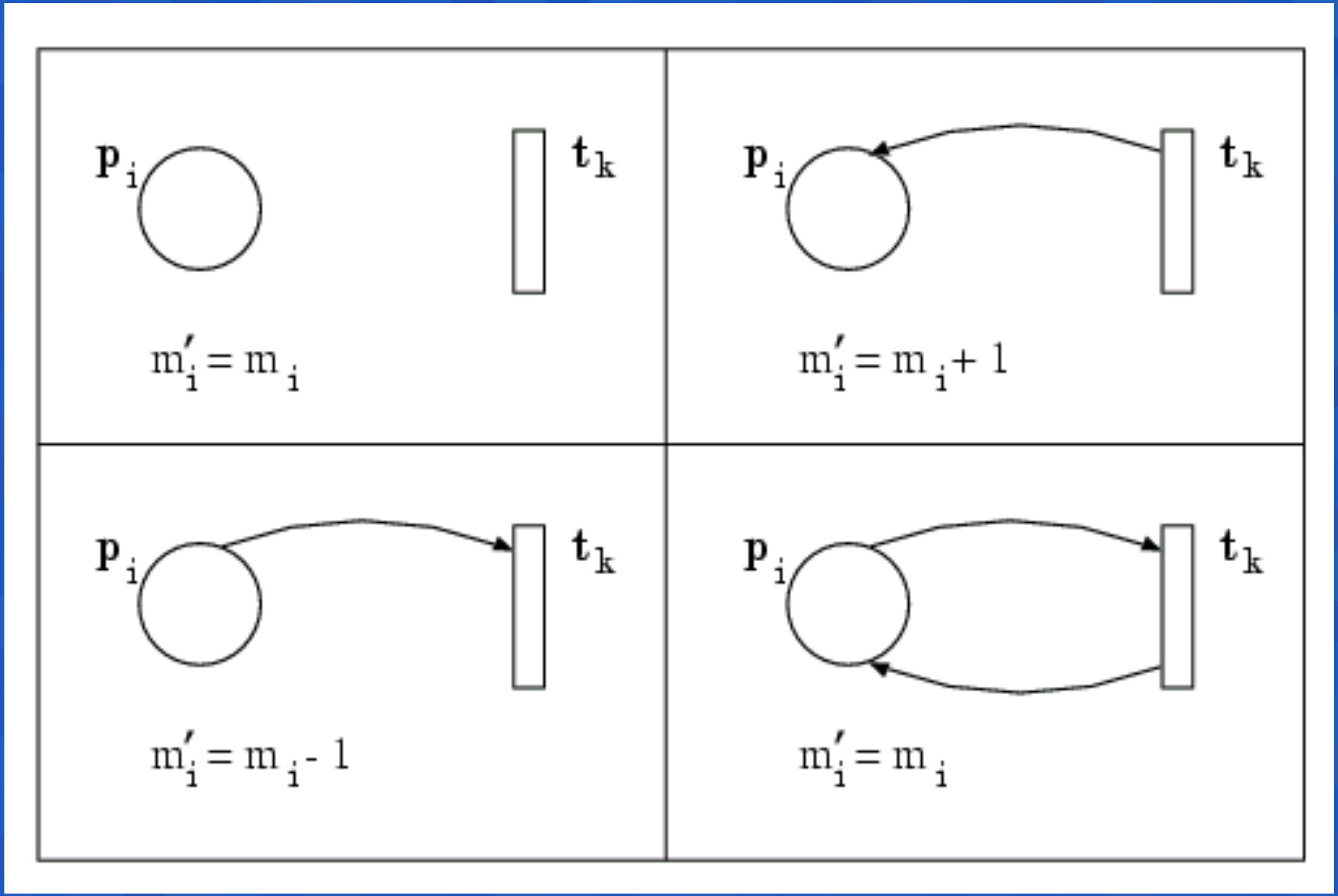


Basic Components of PN

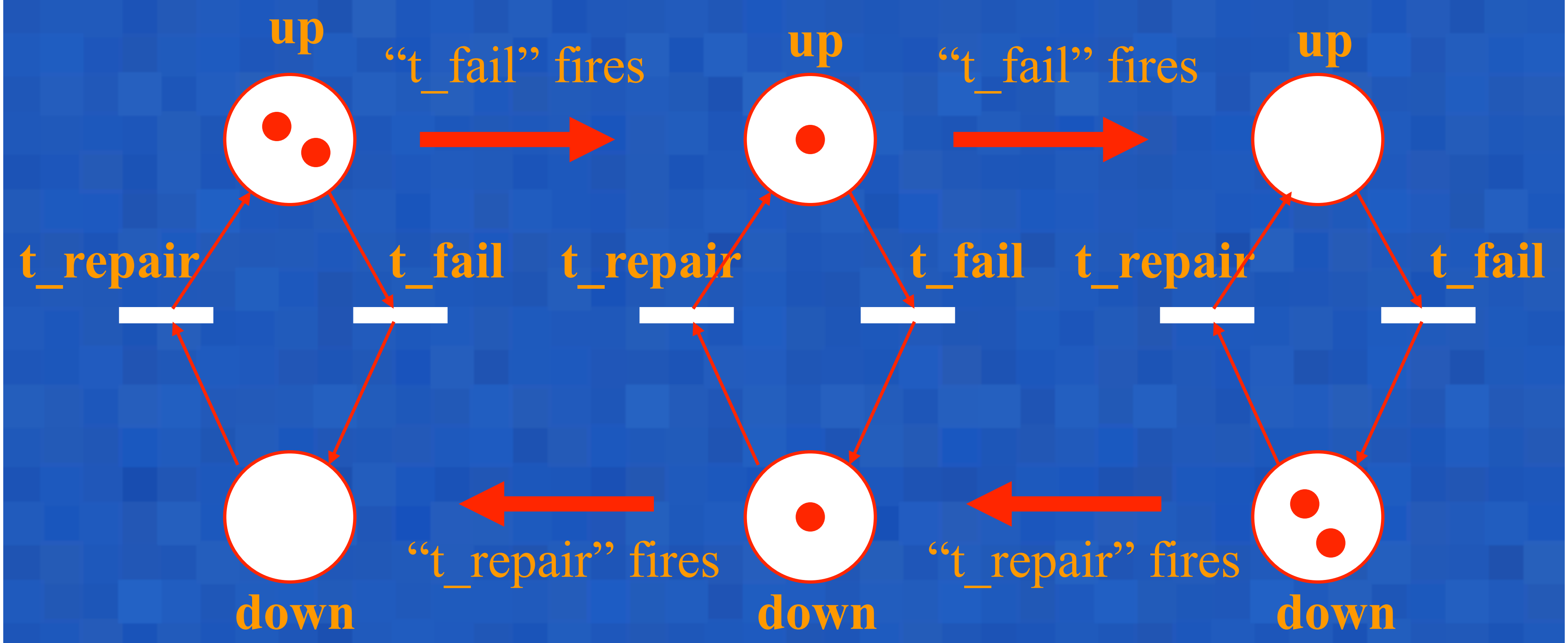


The firing rules of a PN

$$m \rightarrow t_k \rightarrow m'$$

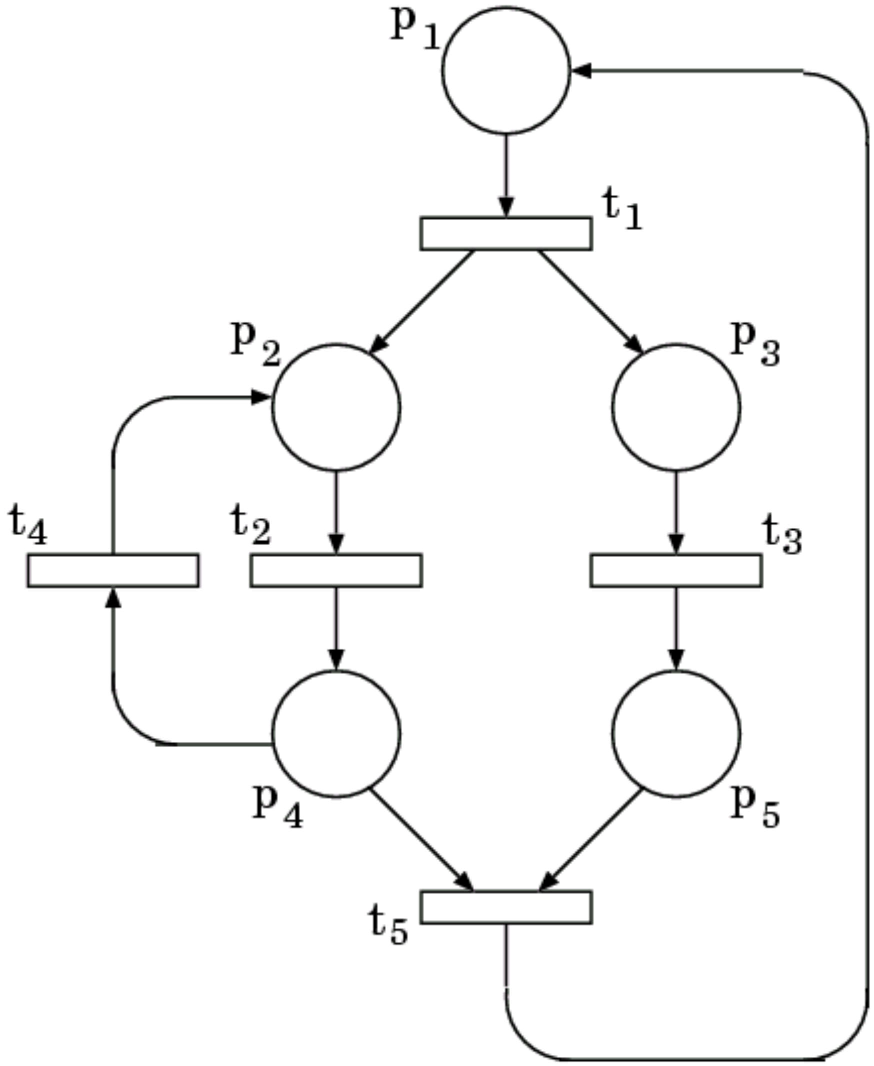


Enabling & Firing of Transitions



A 2-processor failure/repair model

Example of PN



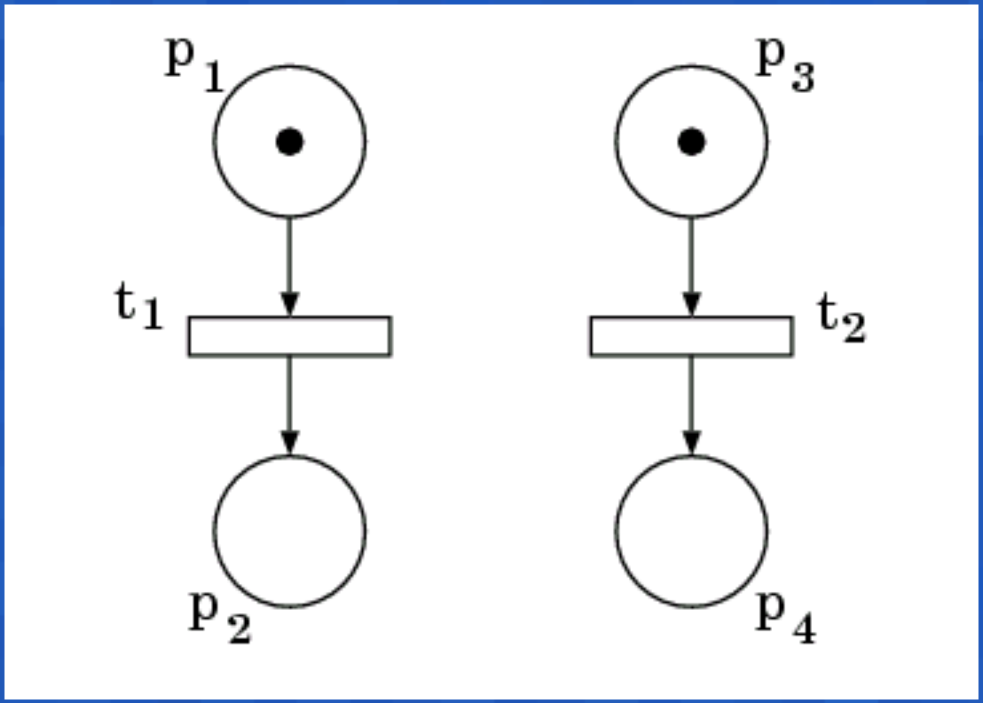
$P = \{p_1\ p_2\ p_3\ p_4\ p_5\}$

$T = \{t_1\ \ t_2\ \ t_3\ \ t_4\ \ t_5\ \}$

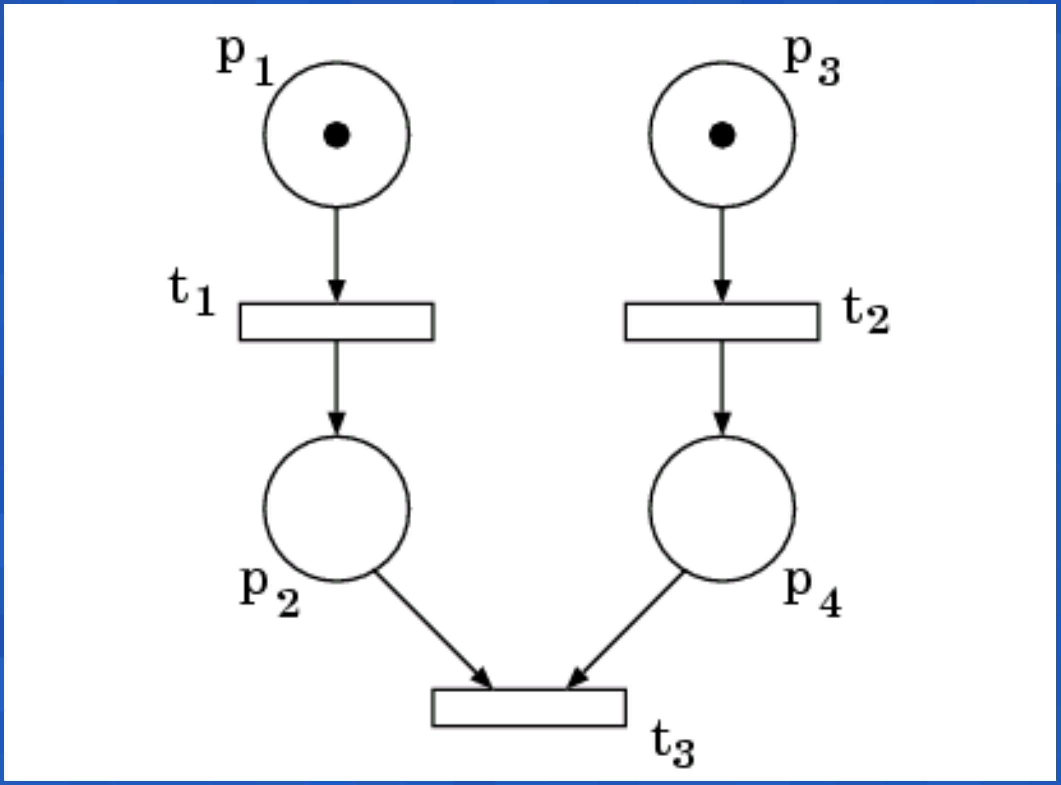
$I(t_1) = \{p_1\}$	$O(t_1) = \{p_2\ p_3\}$
$I(t_2) = \{p_2\}$	$O(t_2) = \{p_4\}$
$I(t_3) = \{p_3\}$	$O(t_3) = \{p_5\}$
$I(t_4) = \{p_4\}$	$O(t_4) = \{p_2\}$
$I(t_5) = \{p_4\ p_5\ \}$	$O(t_5) = \{p_1\}$

$M_1 = (1, 0, 0, 0, 0)$

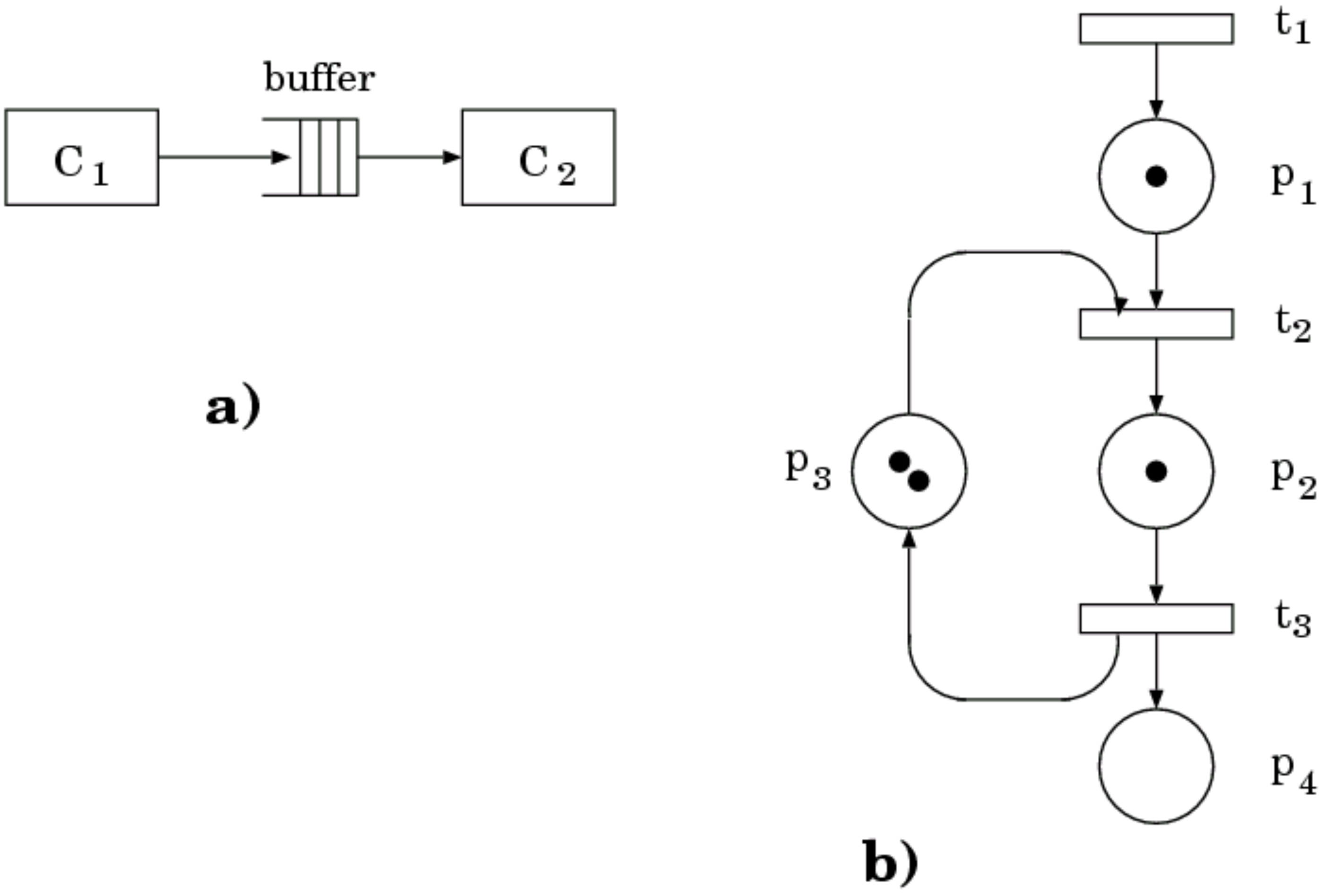
Concurrency (or Parallelism)



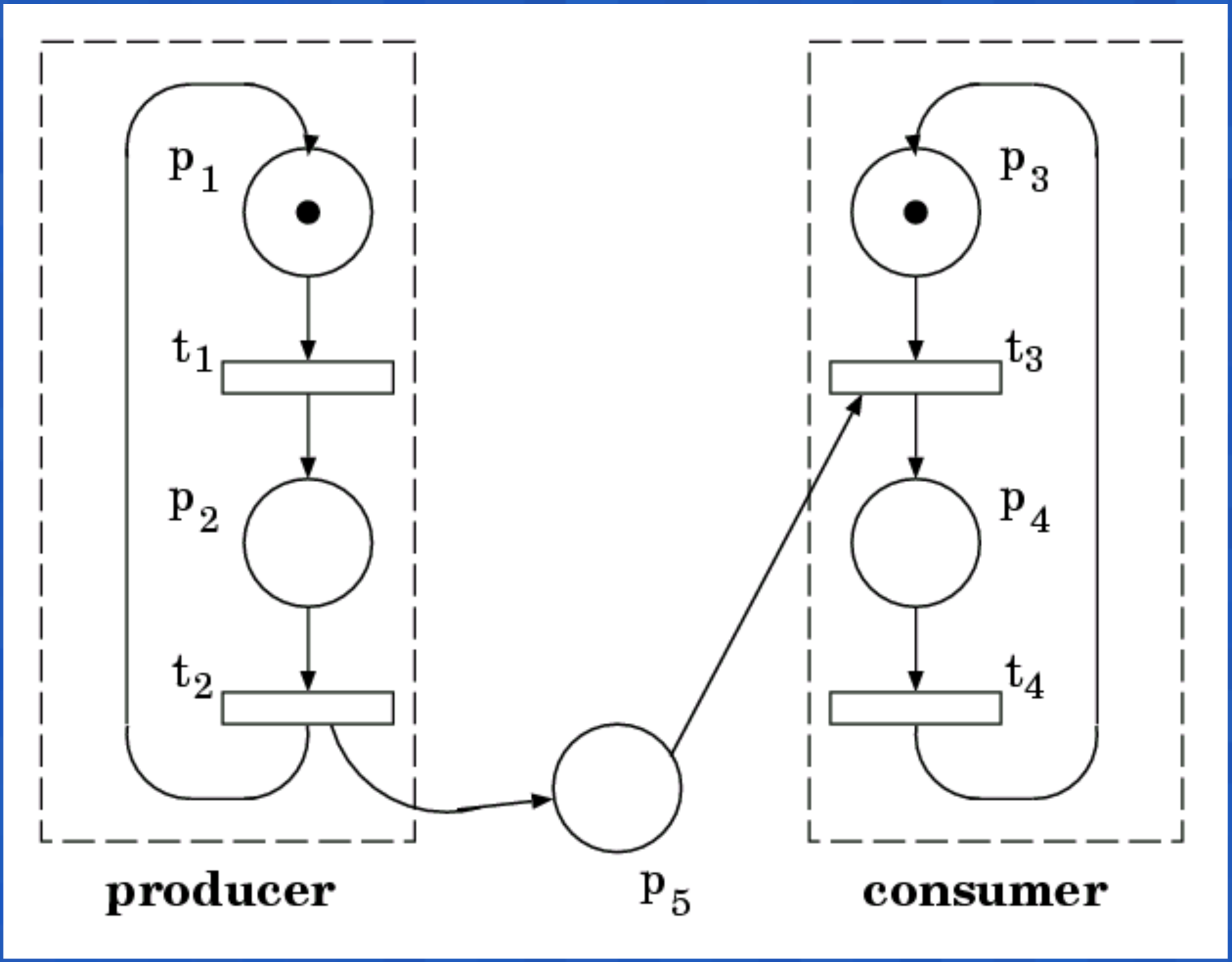
Synchronization



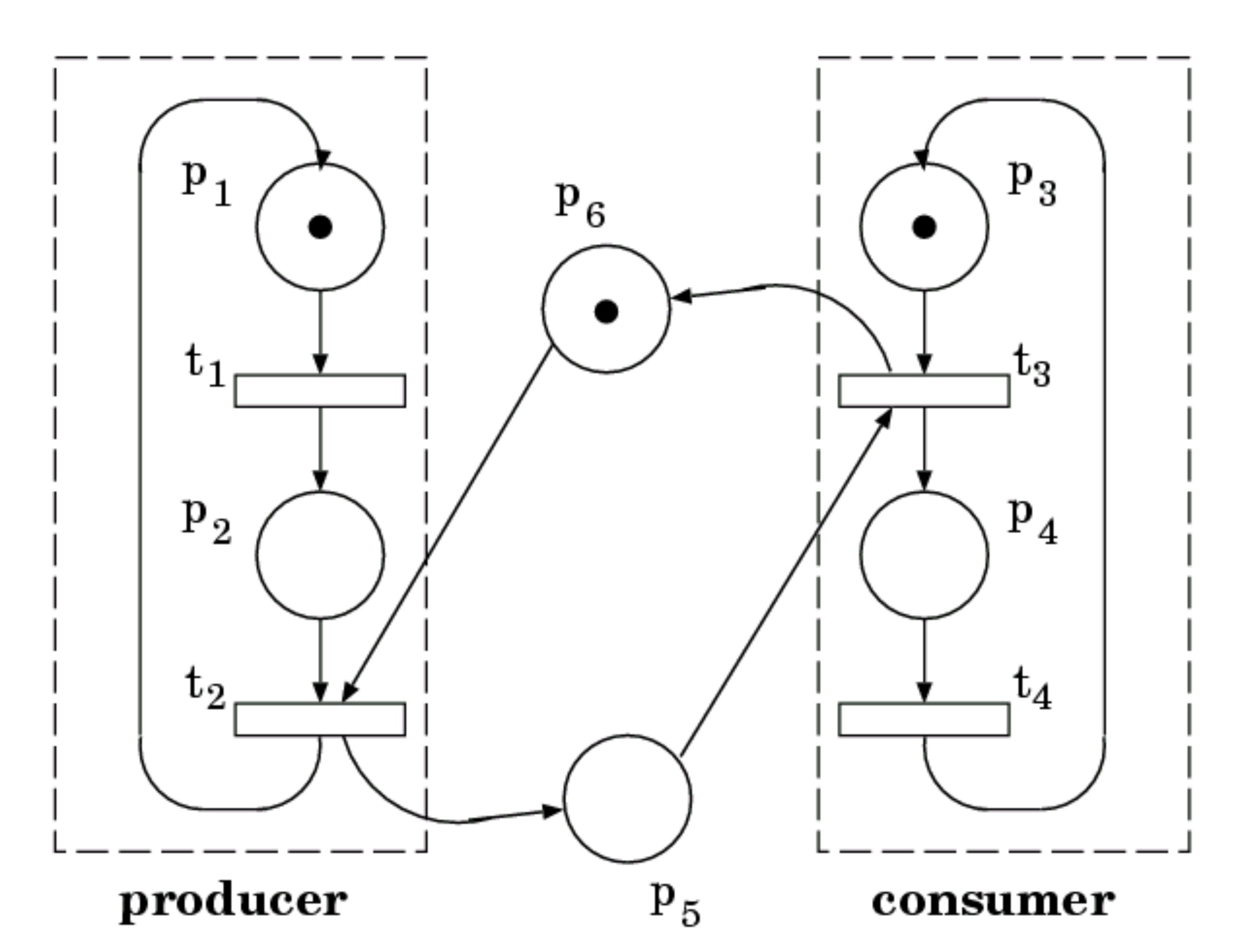
Limited Resources



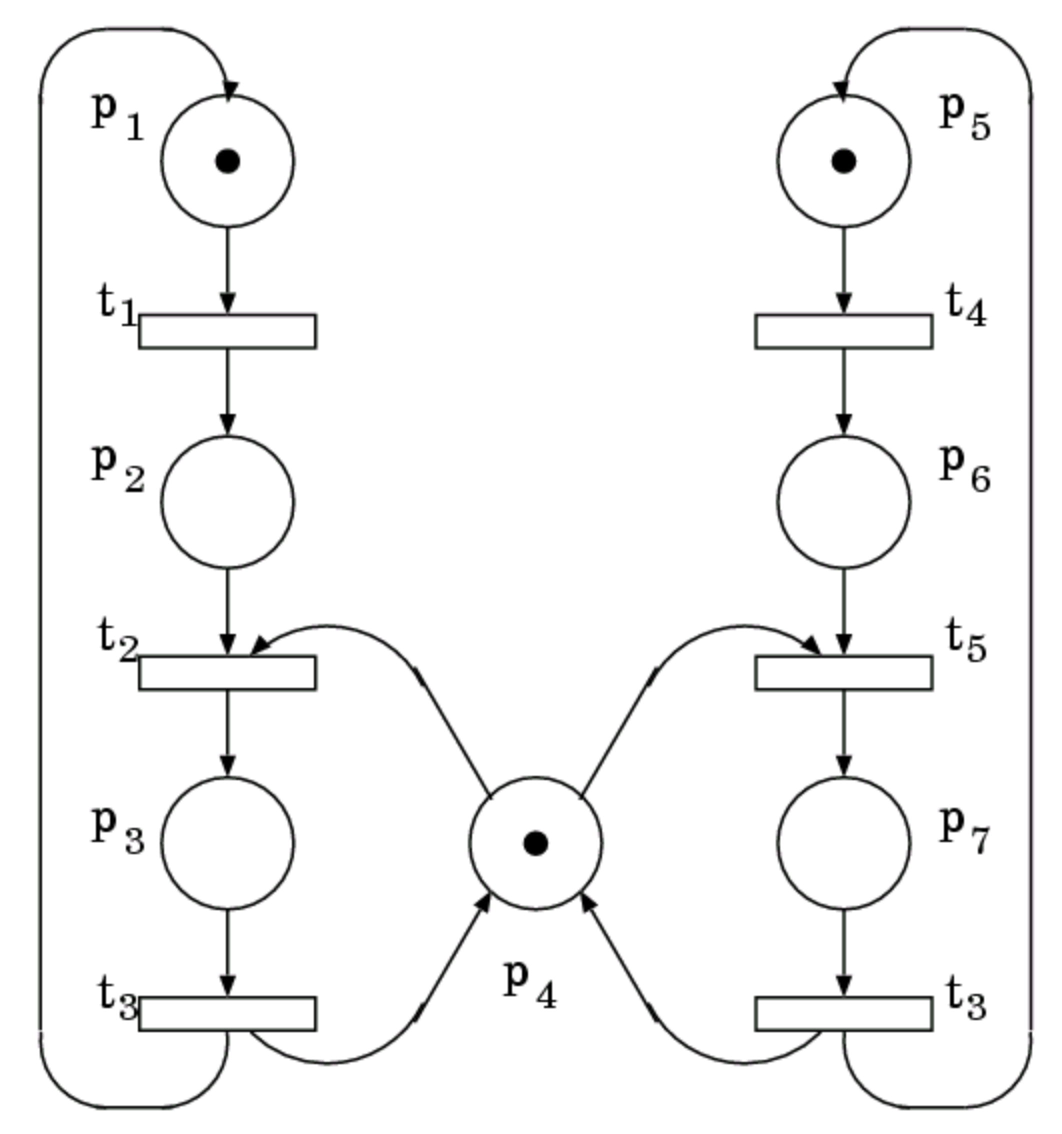
Producer/consumer



Producer/consumer with buffer



Mutual exclusion



Reachability Analysis

- A marking is reachable from another marking if there exists a sequence of transition firings starting from the original marking that results in the new marking

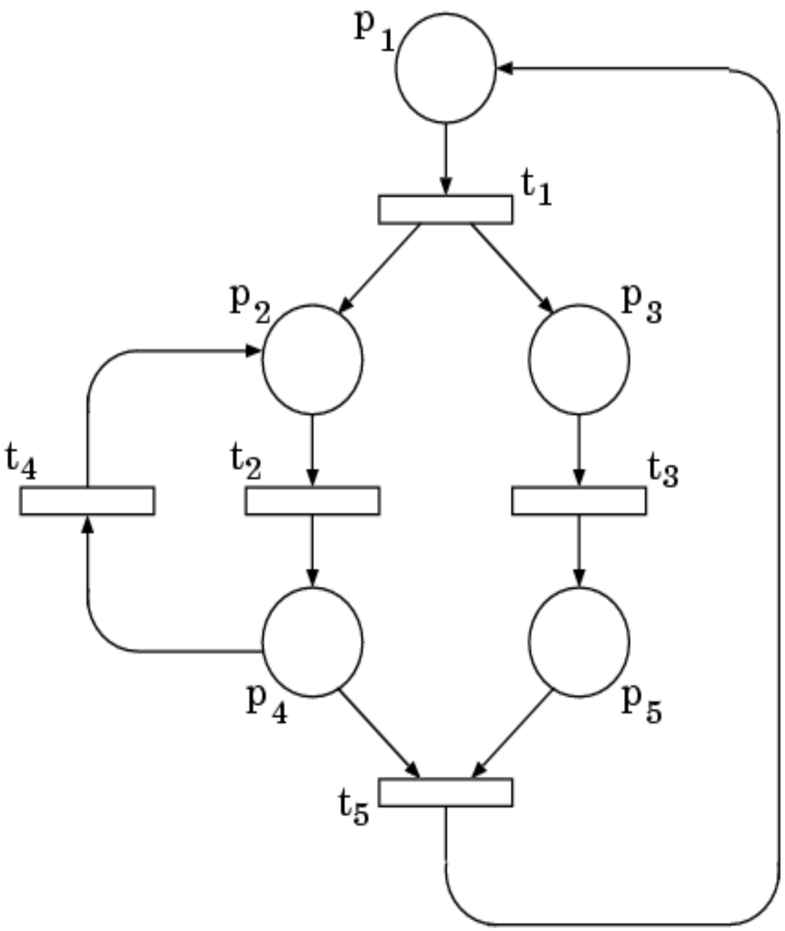
- The *reachability set* of a PN is the set of all markings that are reachable from its *initial* marking

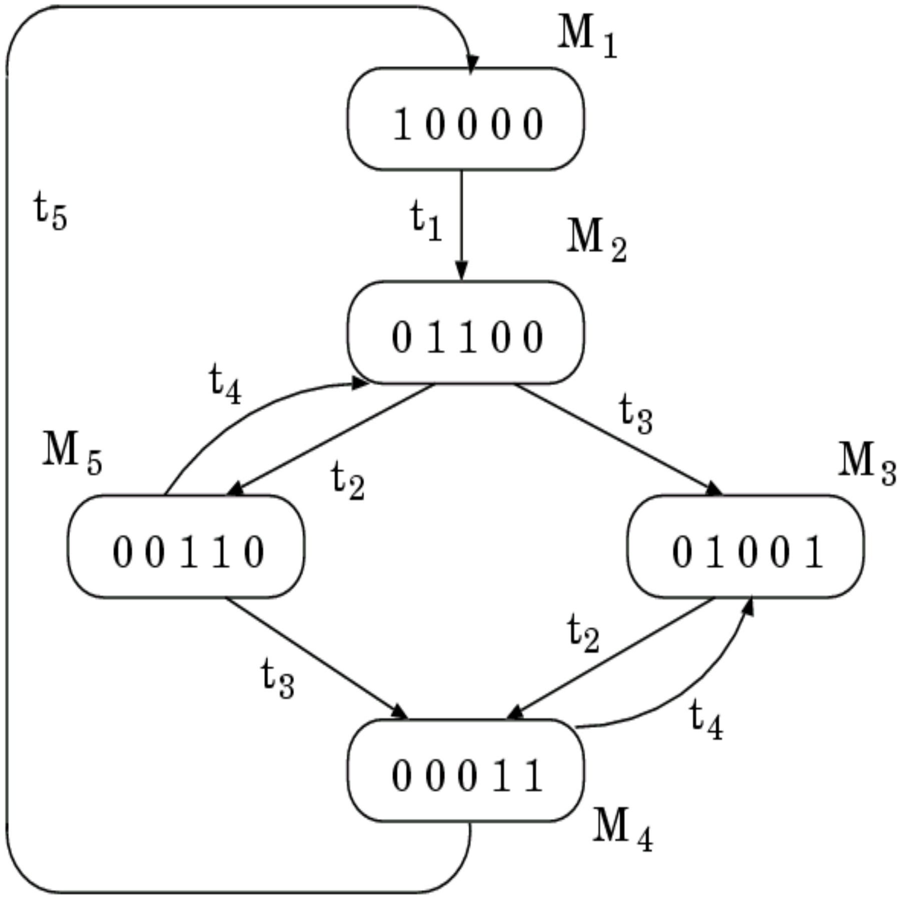
Reachability Analysis

- A reachability graph is a *directed graph* whose nodes are the markings in the reachability set, with directed arcs between the markings representing the marking-to-marking transitions

- The directed arcs are labeled with the corresponding transition whose firing results in a change of the marking from the original marking to the new marking

Generation of the reachability graph





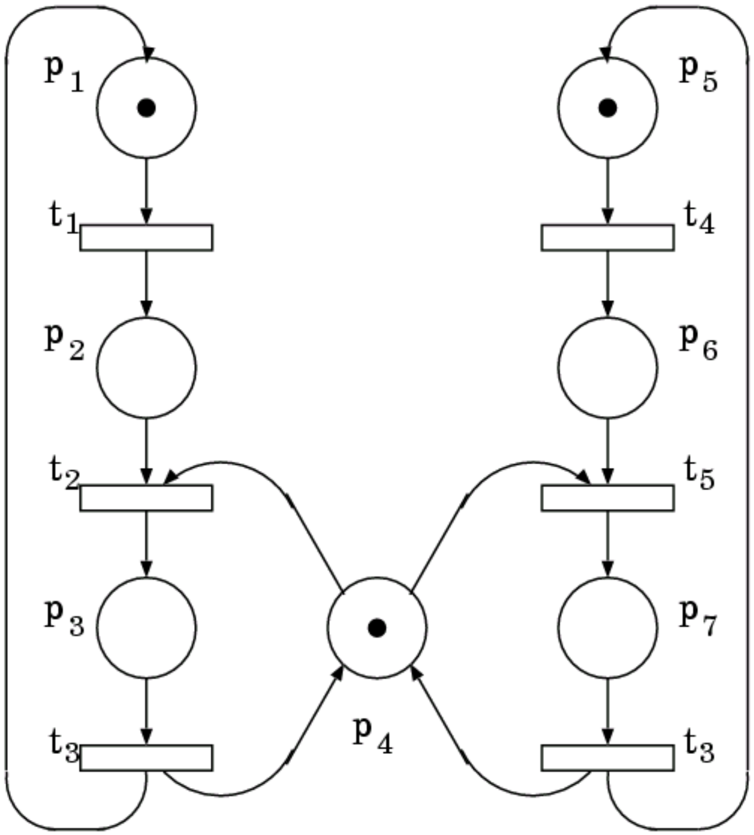
Generation of the reachability graph

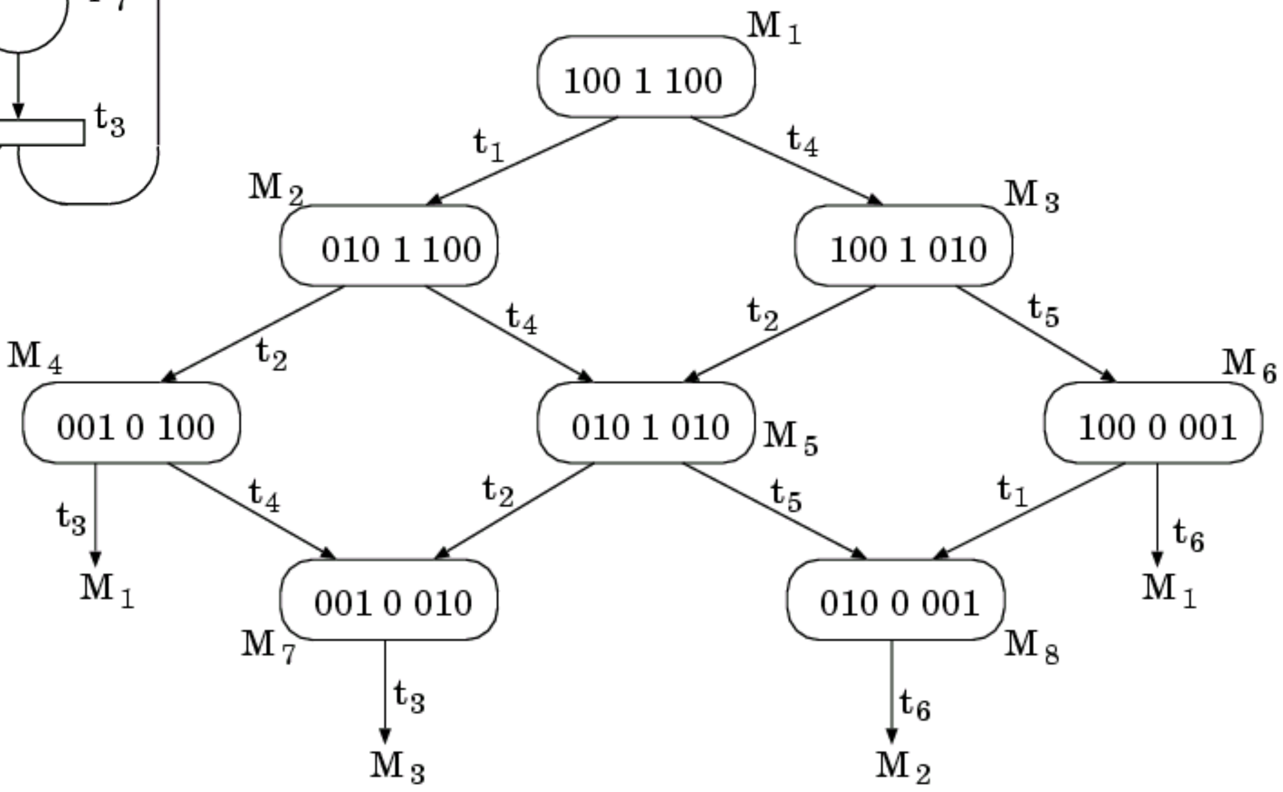
By properly identifying the frontier nodes, the generation of the reachability graph involves a finite number of steps, even if the PN is unbounded.

Three type of frontier nodes:

- terminal (dead) nodes: no transition is enabled;
- duplicate nodes: already generated;
- infinitely reproducible nodes.

Generation of the reachability graph





Infinitely reproducible nodes

A marking M´´ is an infinitely reproducible node if:

$$M'' \} M'$$

$$m_i'' \; □ \; m_i' \qquad (i = 1,2 \; \ldots., \; n_{place})$$

where M´ is a marking already generated.

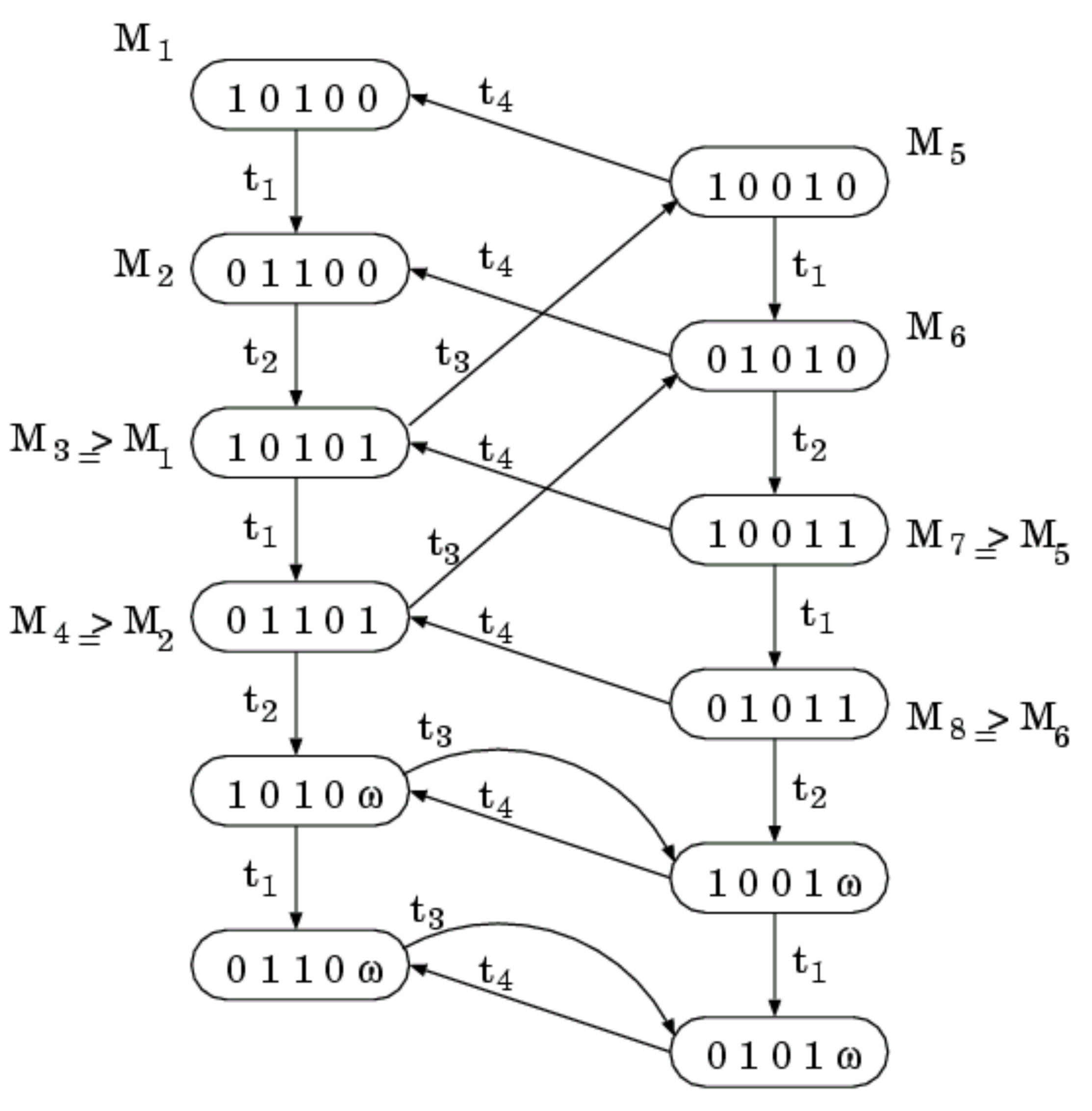
In fact, the sequence M´ □ M´´ is firable from M´´ and then is infinitely reproducible.

An arbitrarily large number of tokens is represented by a special symbol □

Generation of an unbounded RG

Producer/consumer

$$\begin{aligned} \omega + a &= \omega \\ \omega - a &= \omega \\ a &< \omega \end{aligned}$$



Extensions of PN models

➔ arc multiplicity

è inhibitor arcs

è priority levels

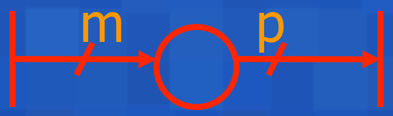
è enabling functions (guards)

Note: The last three extensions destroy the infinitely reproducible property.

Petri Net: Arc Multiplicity

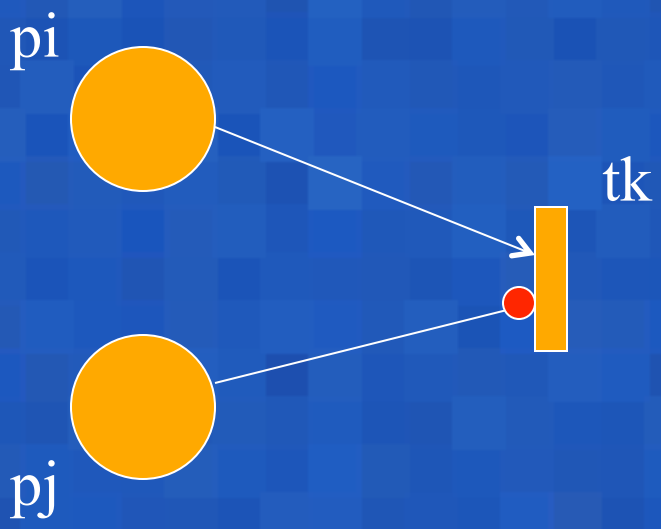
- An arc cardinality (or multiplicity) may be associated with input and output arcs, whereby the enabling and firing rules are changed as follows:

  – Each input place must contain at least as many tokens as the cardinality of the corresponding input arc.



  – When the transition fires, it removes as many tokens from each input place as the cardinality of the corresponding input arc, and deposits as many tokens in each output places as the cardinality of the corresponding output arc.

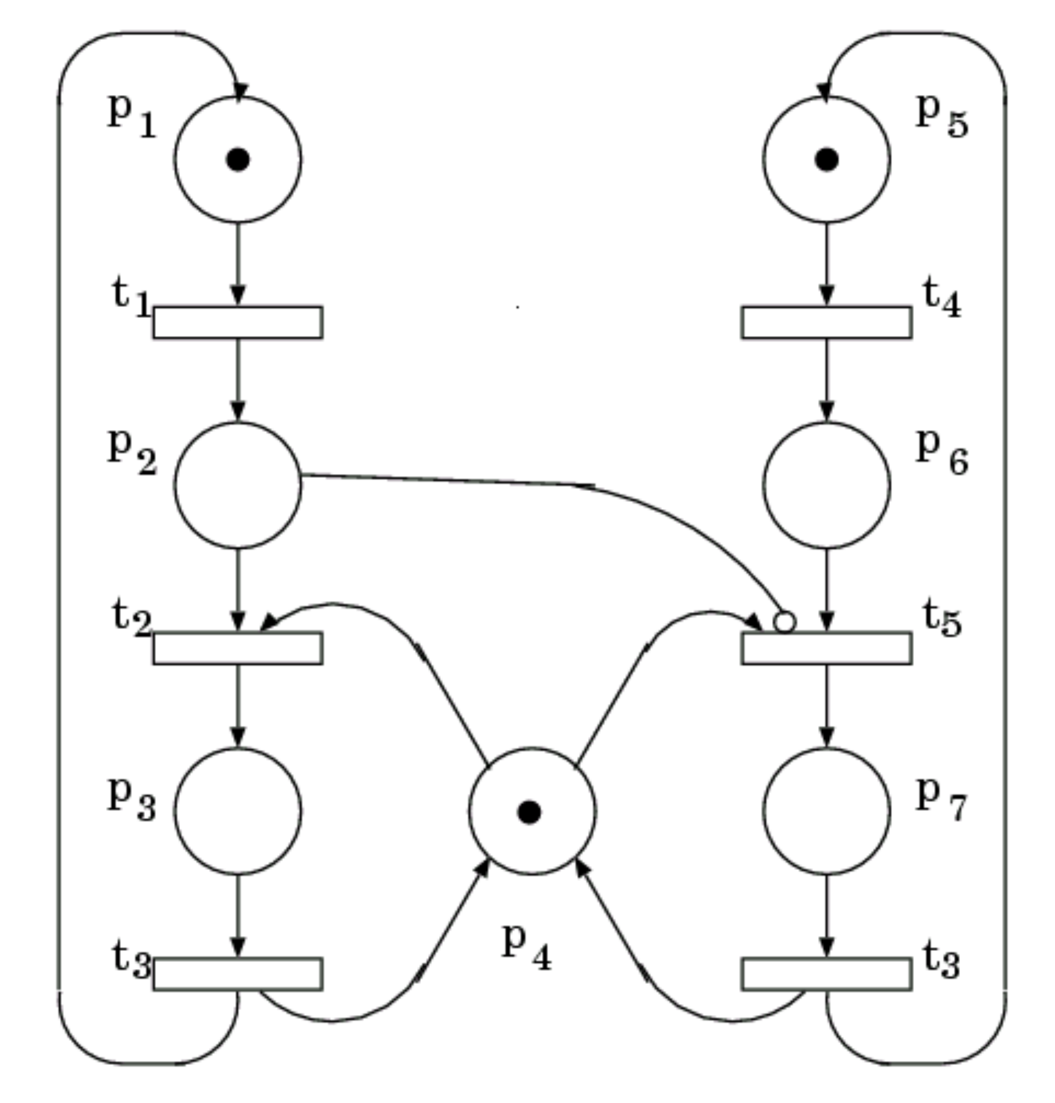
Petri Net : Inhibitor Arc



Inhibitor arcs are represented with a circle-headed arc.

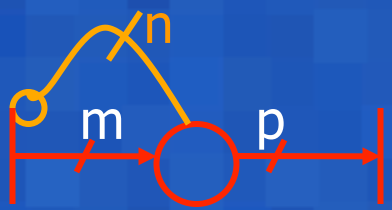
The transition can fire iff the inhibitor place does not contain tokens.

Petri Net : Inhibitor Arc



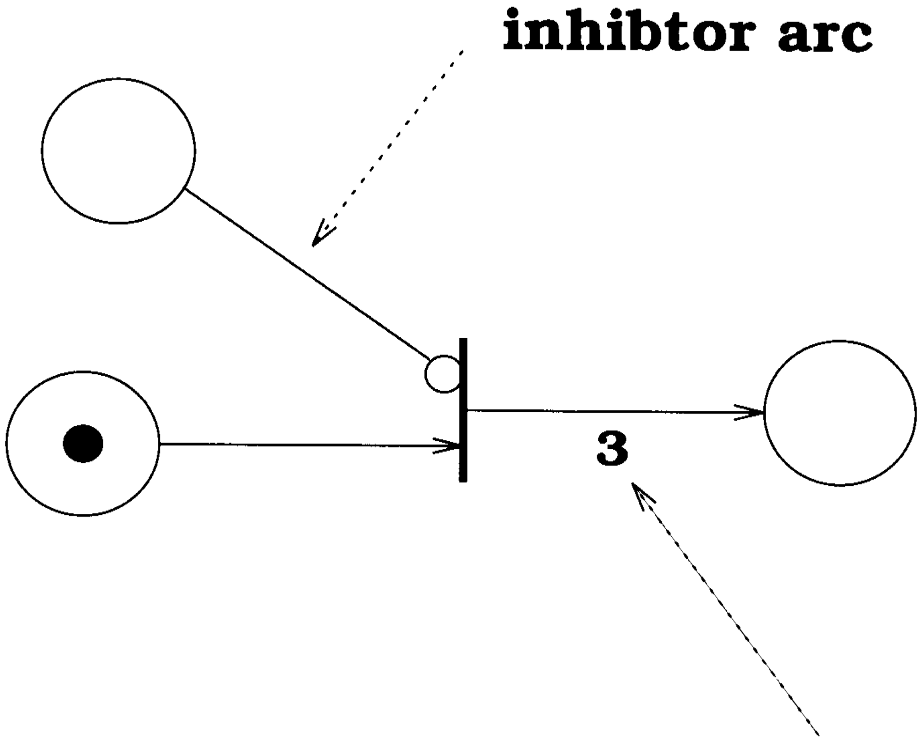
Petri Net : Multiple Inhibitor Arc

- An inhibitor arc drawn from place to a transition means that the transition cannot fire if the corresponding inhibitor place contains at least as many tokens as the cardinality of the corresponding inhibitor arc



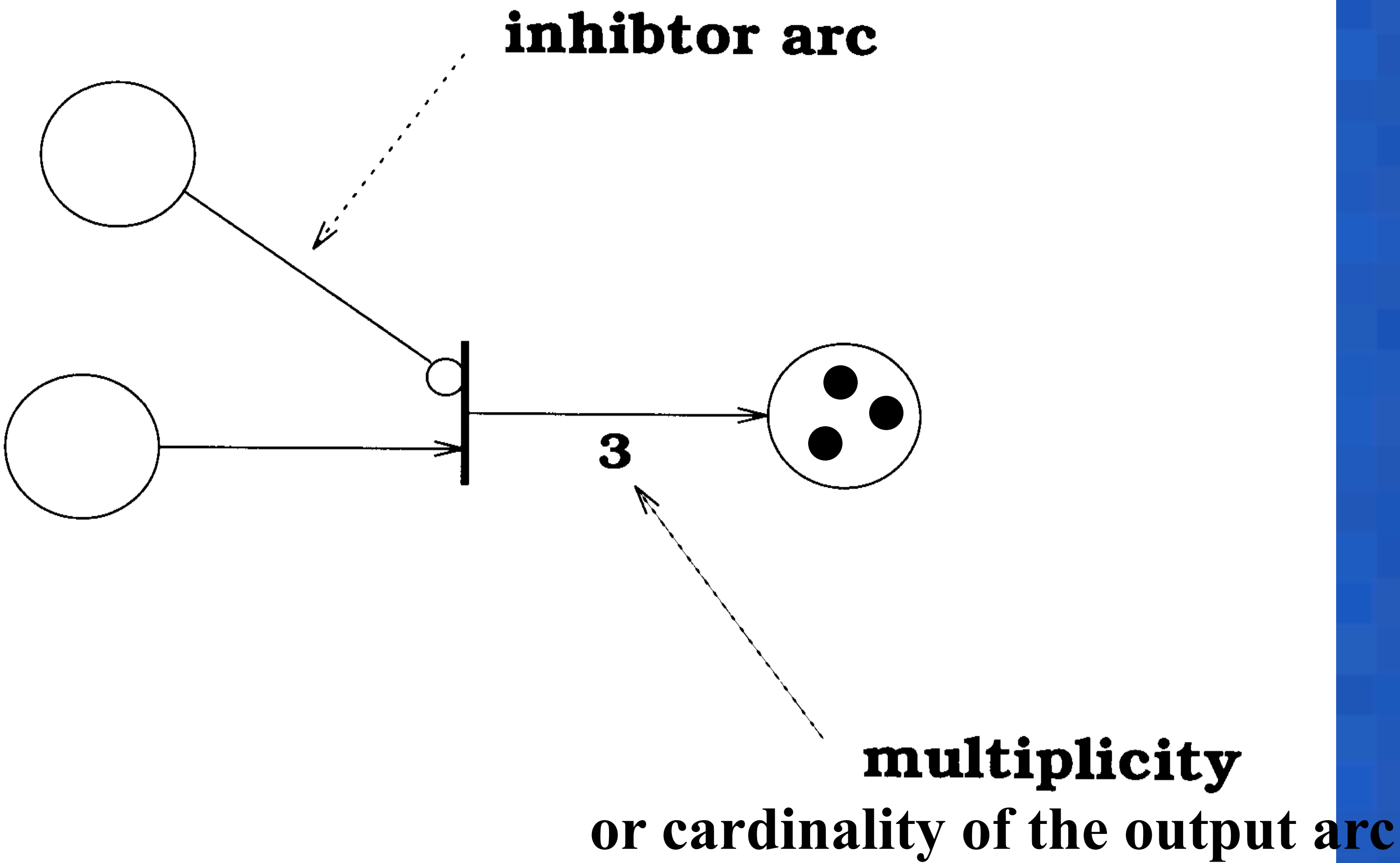
- Inhibitor arcs are represented graphically as an arc ending in a small circle at the transition instead of an arrowhead

An Example: Before



multiplicity
or cardinality of the output arc

An Example: After



Priority levels

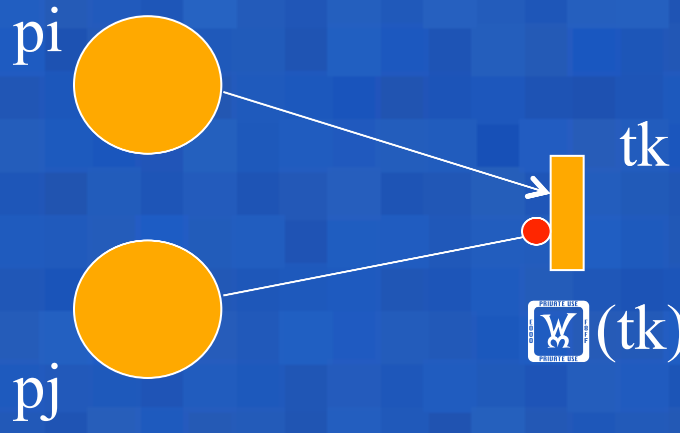
A priority level can be attached to each PN transition.

The standard execution rules are modified in the sense that, among all the transitions enabled in a given marking, only those with associated highest priority level are allowed to fire.

Enabling Functions

An enabling function (or guard) is a boolean expression composed with the PN primitives (places, trans, tokens).

The enabling rule is modified in the sense that beside the standard conditions, the enabling function must evaluate to true.



[illegible](tk) = #P1<2 & #P2=0

High Level (colored) Petri Nets

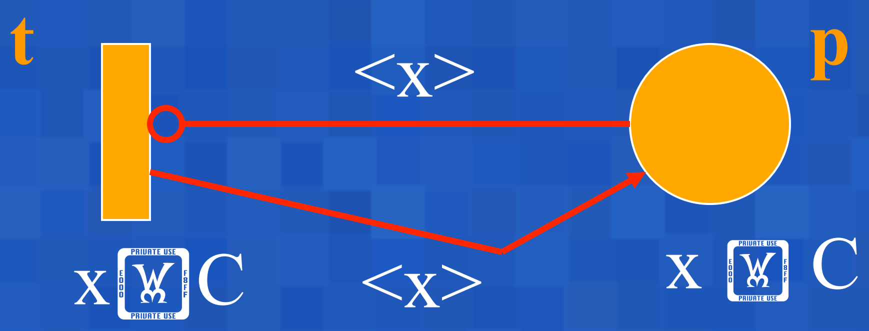
In standard PN tokens are indistinguishable entities.

The semantics of the model does not allow to follow the behavior of an individual token through the PN.

High Level PN overcome this limitation by assigning to each individual token an attribute (color).

Places, arcs and transitions can have functions and guards depending on the colors.

Colored Petri Nets



C is a set of colors of cardinality $|C|$ and x is an element of the set.

Place **p** can contain tokens of any color $x \in C$;

Transition **t** can fires tokens of any color $x \in C$.

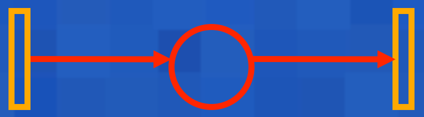
Stochastic Petri Nets (SPN)

- Petri nets are extended by associating time with the firing of transitions, resulting in timed Petri nets.

- A special case of timed Petri nets is stochastic Petri net (SPN) where the firing times are considered random variables.

Stochastic Petri Nets (SPN)

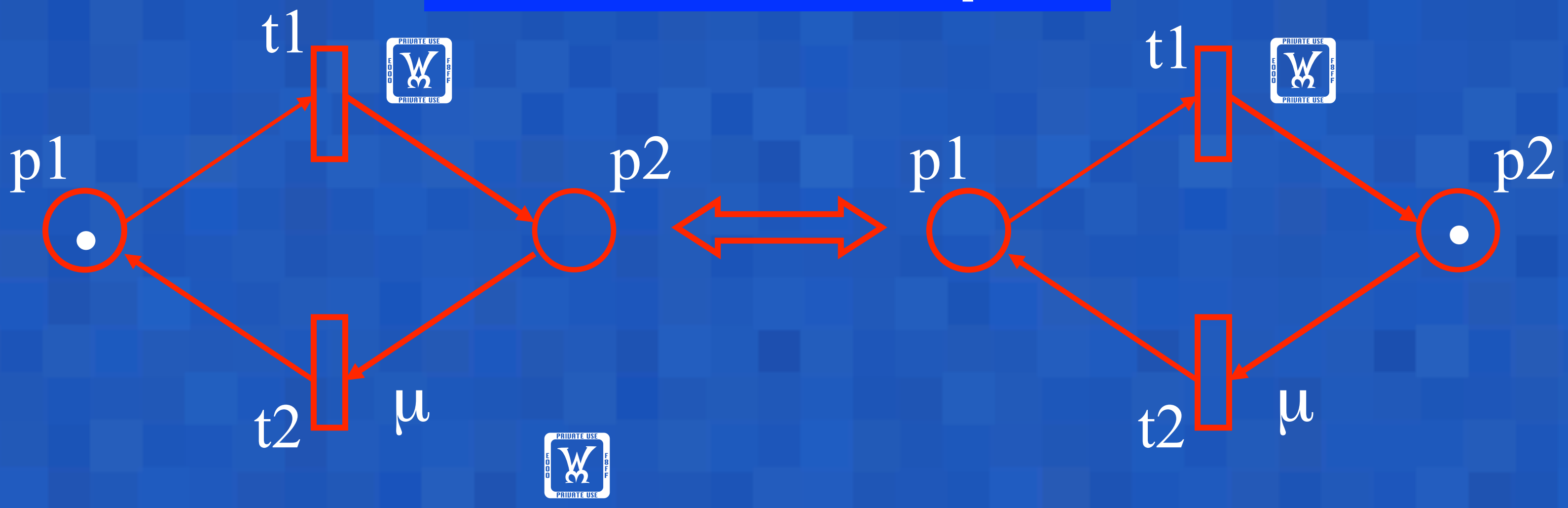
- A special case of stochastic Petri net (SPN) is where the firing times are exponentially distributed.



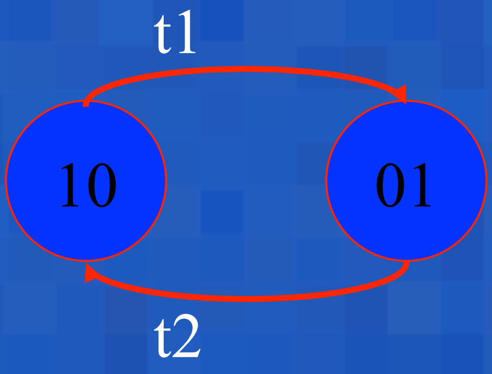
- The marking process is mapped into a continuous time Markov chain (CTMC) with state space isomorphic to the reachability graph of the PN.

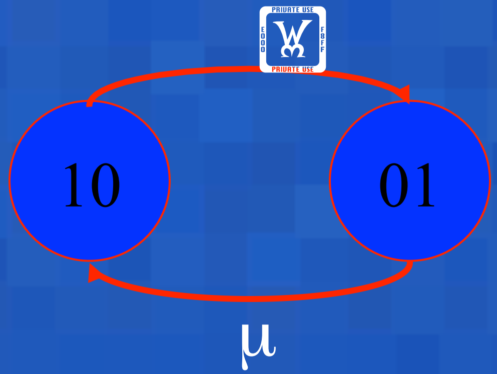
SPN: A Simple Example

Server Failure/Repair

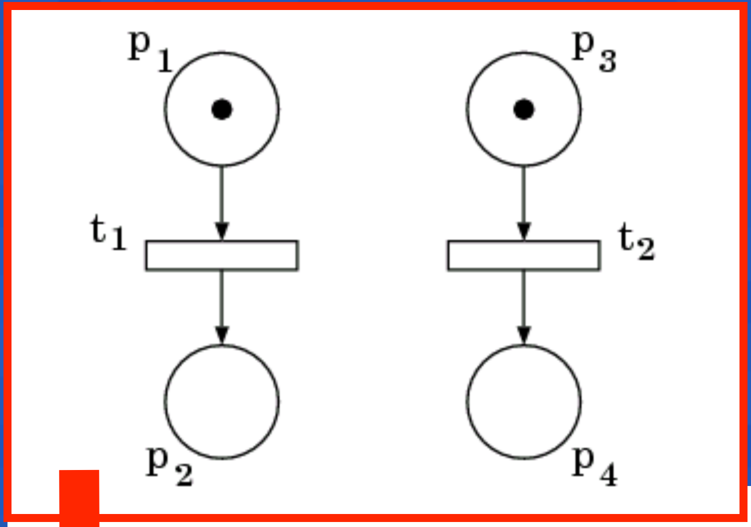


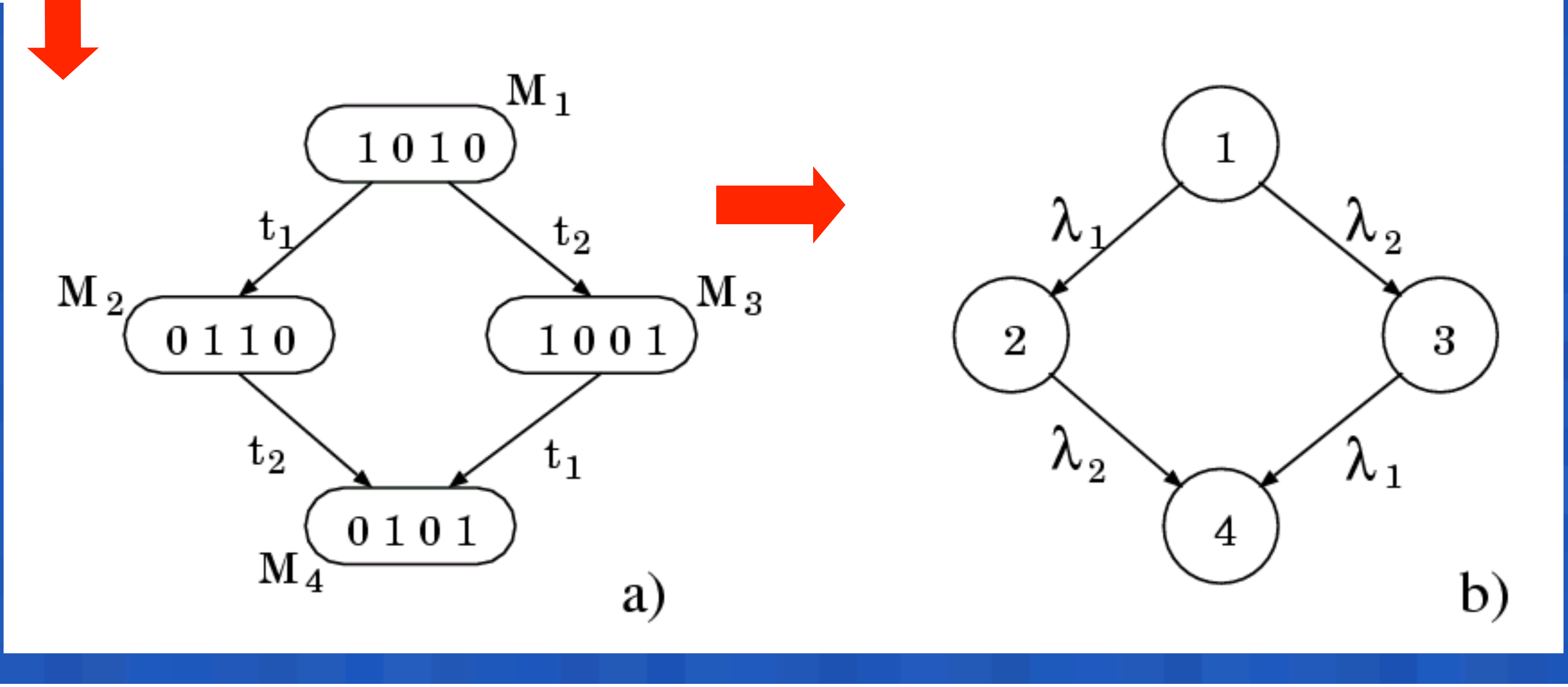
Reachability graph → **CTMC**



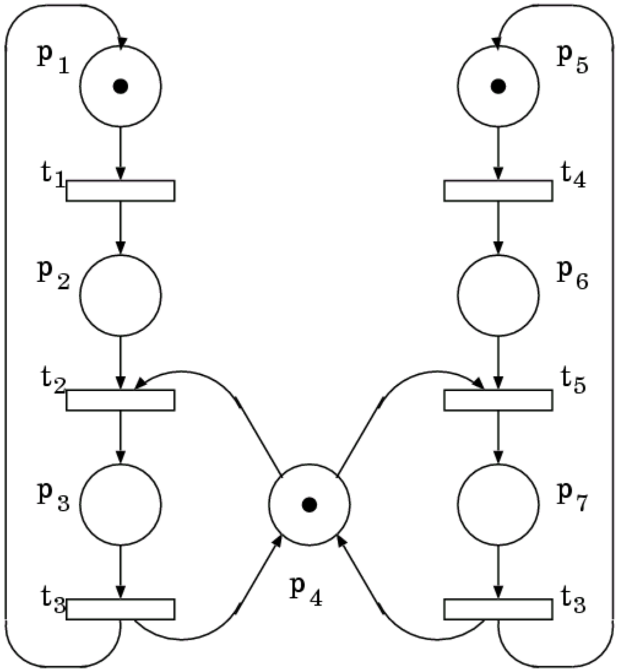


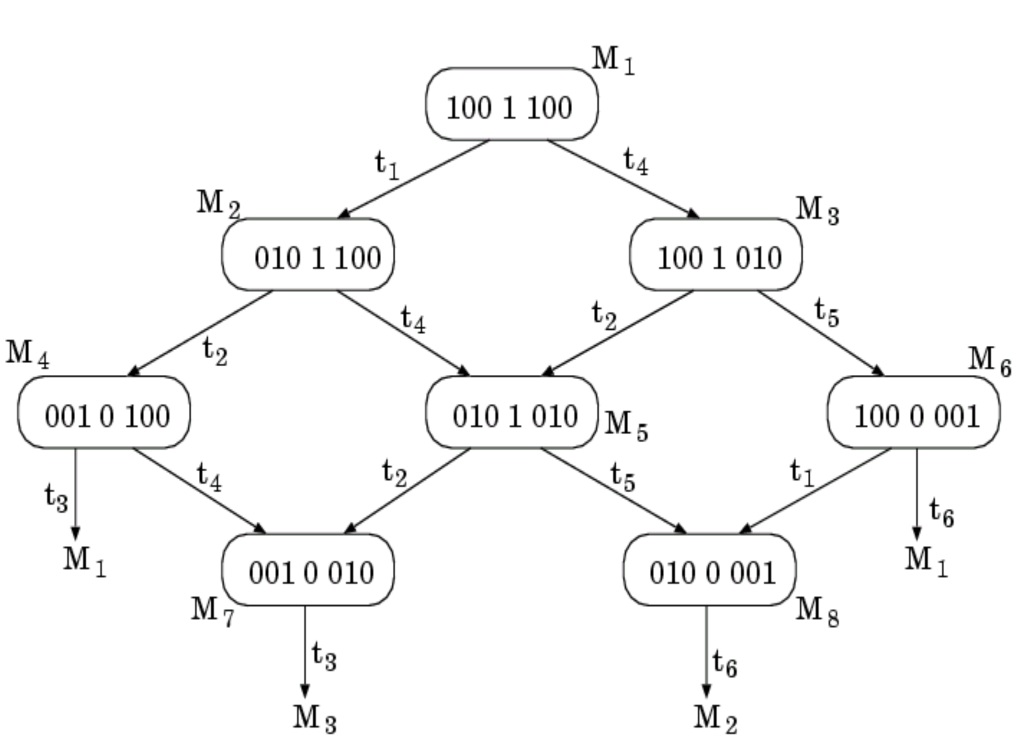
From SPN to CTMC: A Simple Example

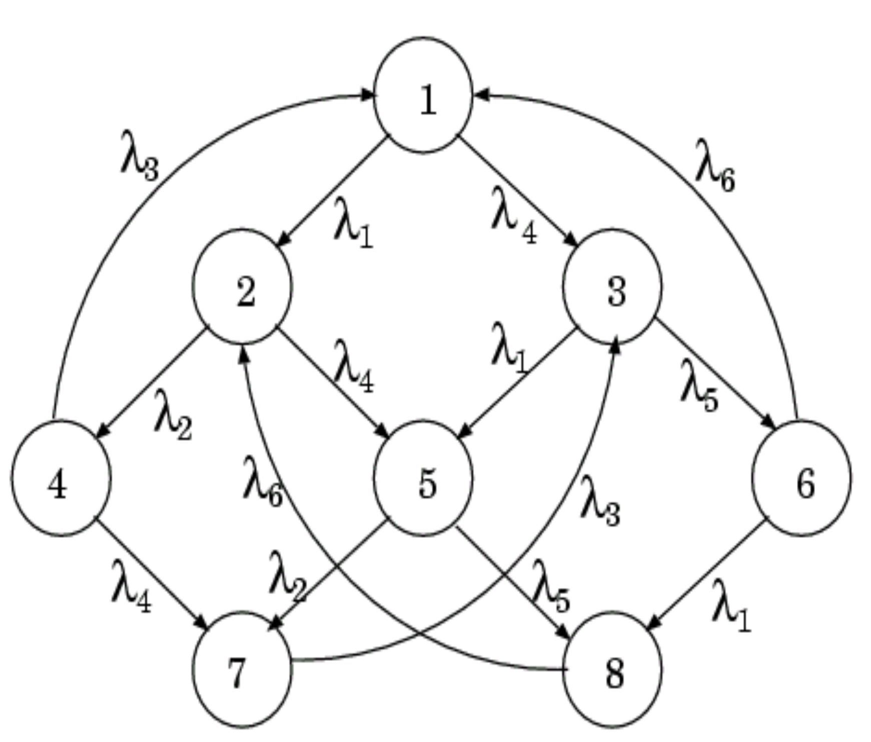




From SPN to CTMC: An Example

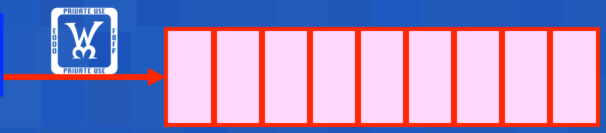




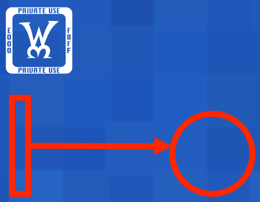


SPN: Poisson Process

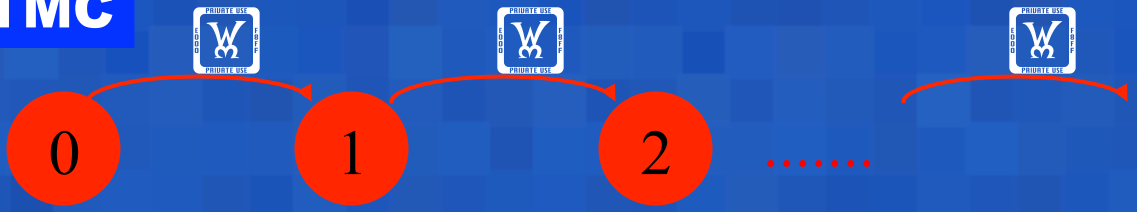
PP with rate



SPN model

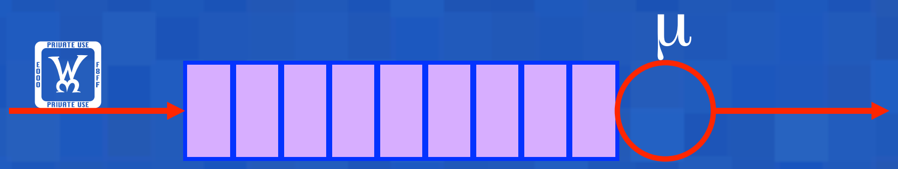


RG = CTMC

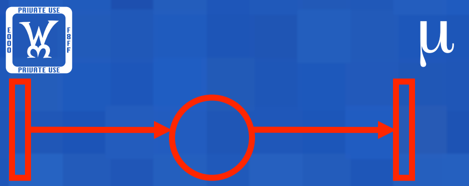


SPN: M/M/1 Queue

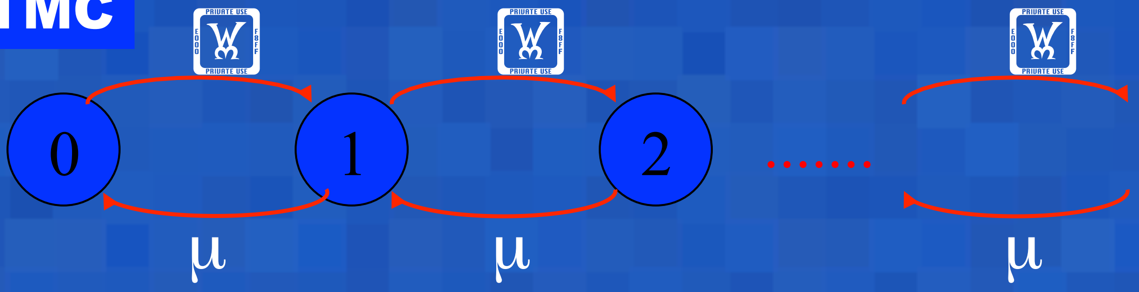
M/M/1



SPN model

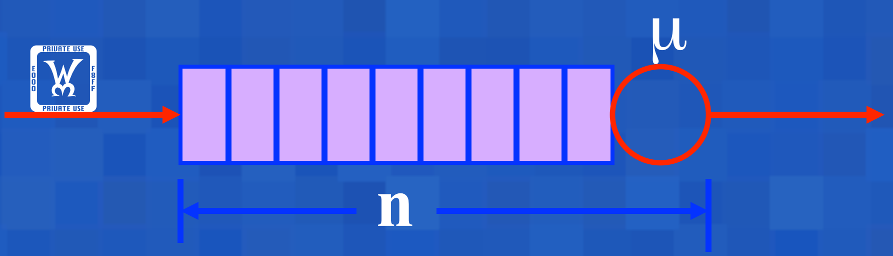


RG = CTMC

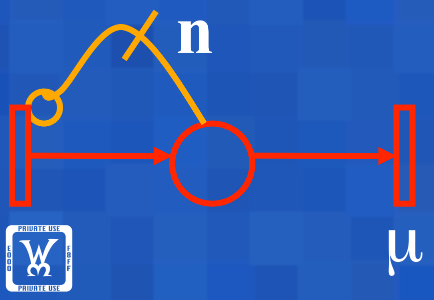


SPN: M/M/1/n Queue (1)

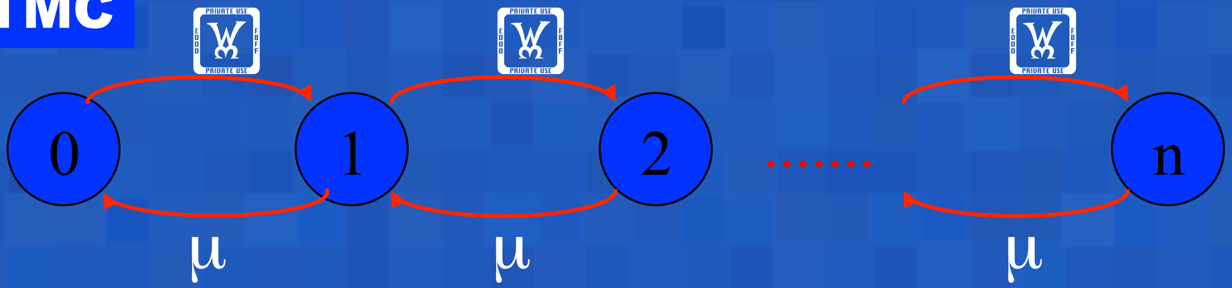
M/M/1/n



SPN model

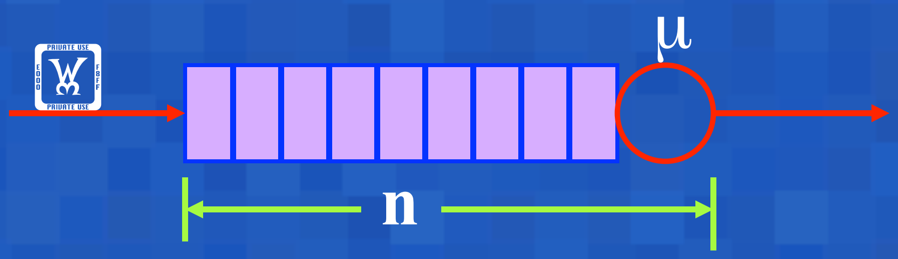


RG = CTMC

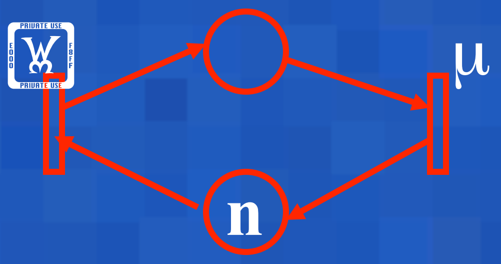


SPN: M/M/1/n Queue (2)

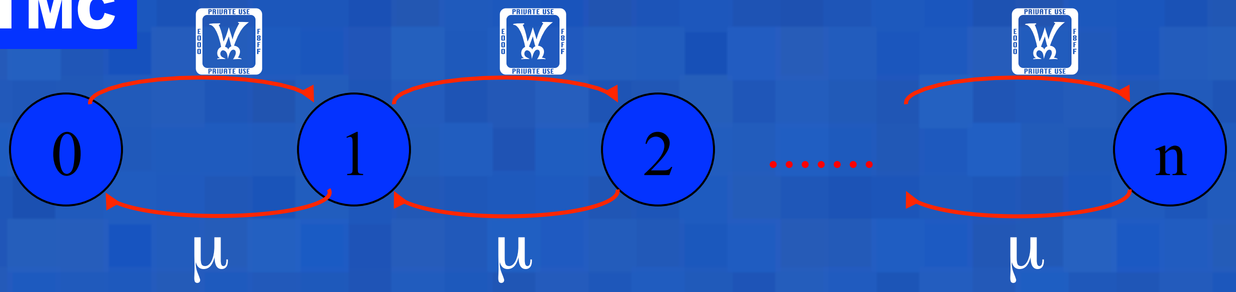
M/M/1/n



SPN model

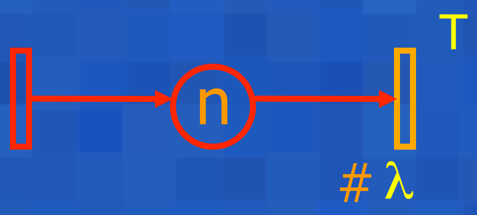


RG = CTMC



Marking dependent firing rate

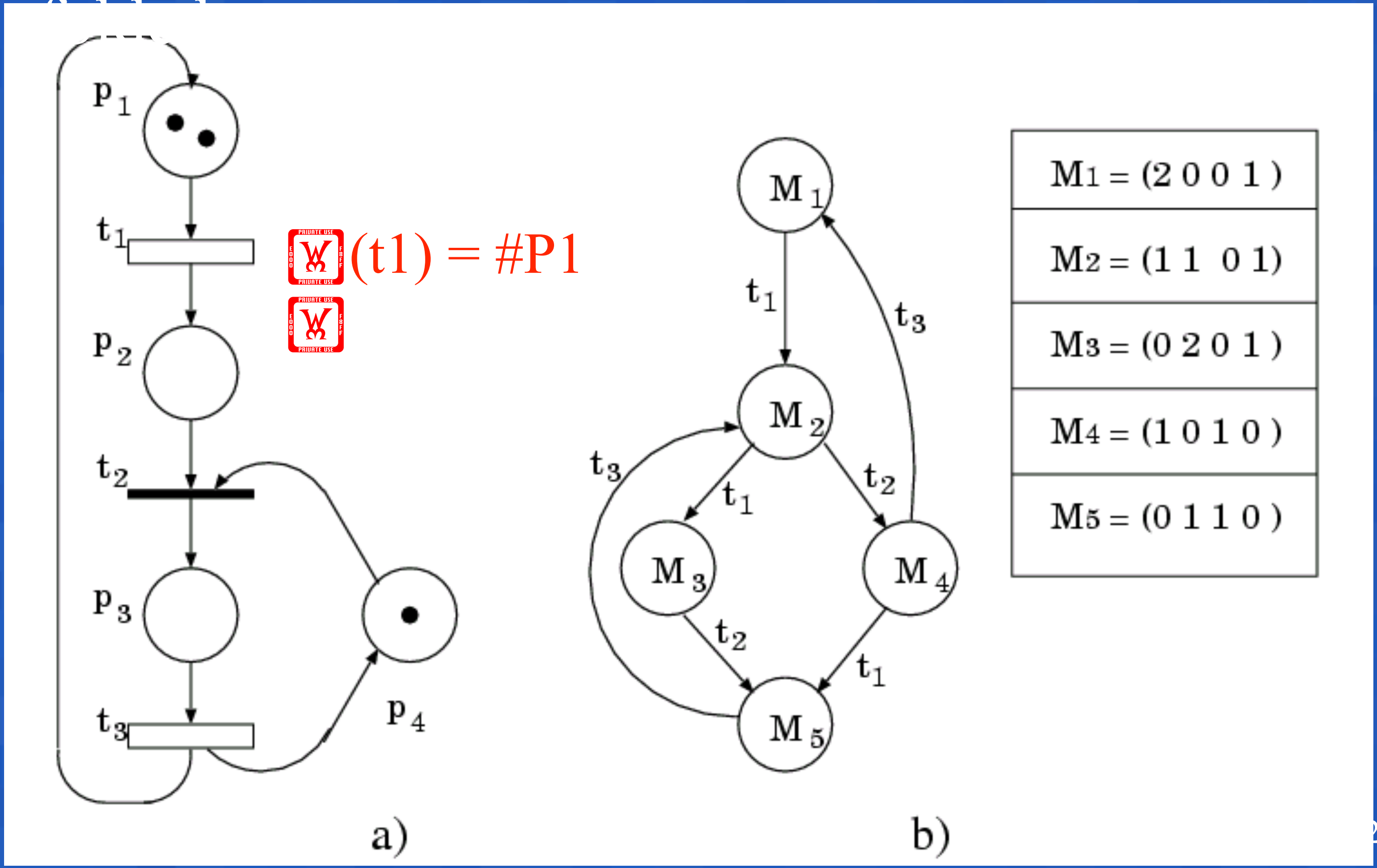
- A firing rate is associated with each timed transition.
- Firing rate of a transition may be marking dependent.



Rate of T = nλ

Marking dependent firing rate

The mutual exclusion problem can be folded

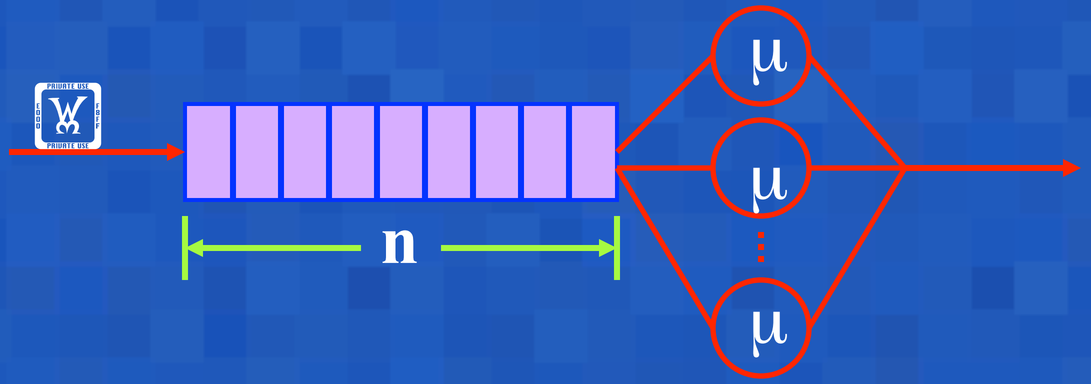


M1 = (2 0 0 1)
M2 = (1 1 0 1)
M3 = (0 2 0 1)
M4 = (1 0 1 0)
M5 = (0 1 1 0)

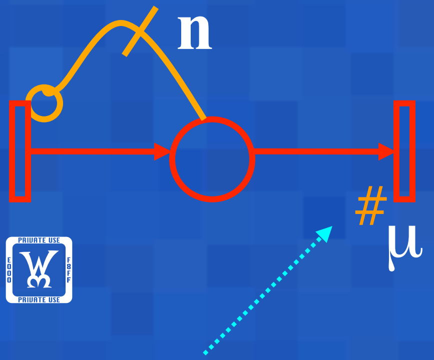
[illegible](t1) = #P1

SPN: M/M/n/n Queue

M/M/n/n

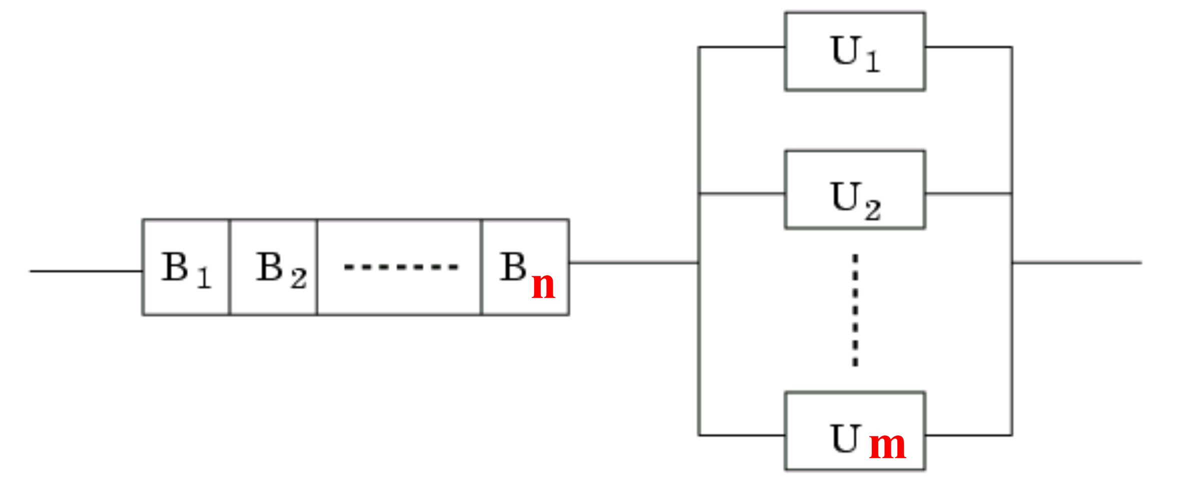


SPN model



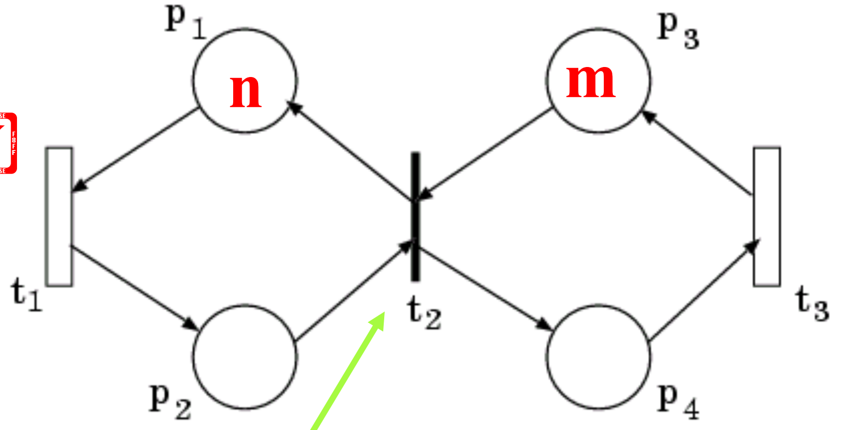
The use of marking-dependent rate

SPN: M/M/m/n Queue (1)



□(t3) = #P4 □

□(t1) = □

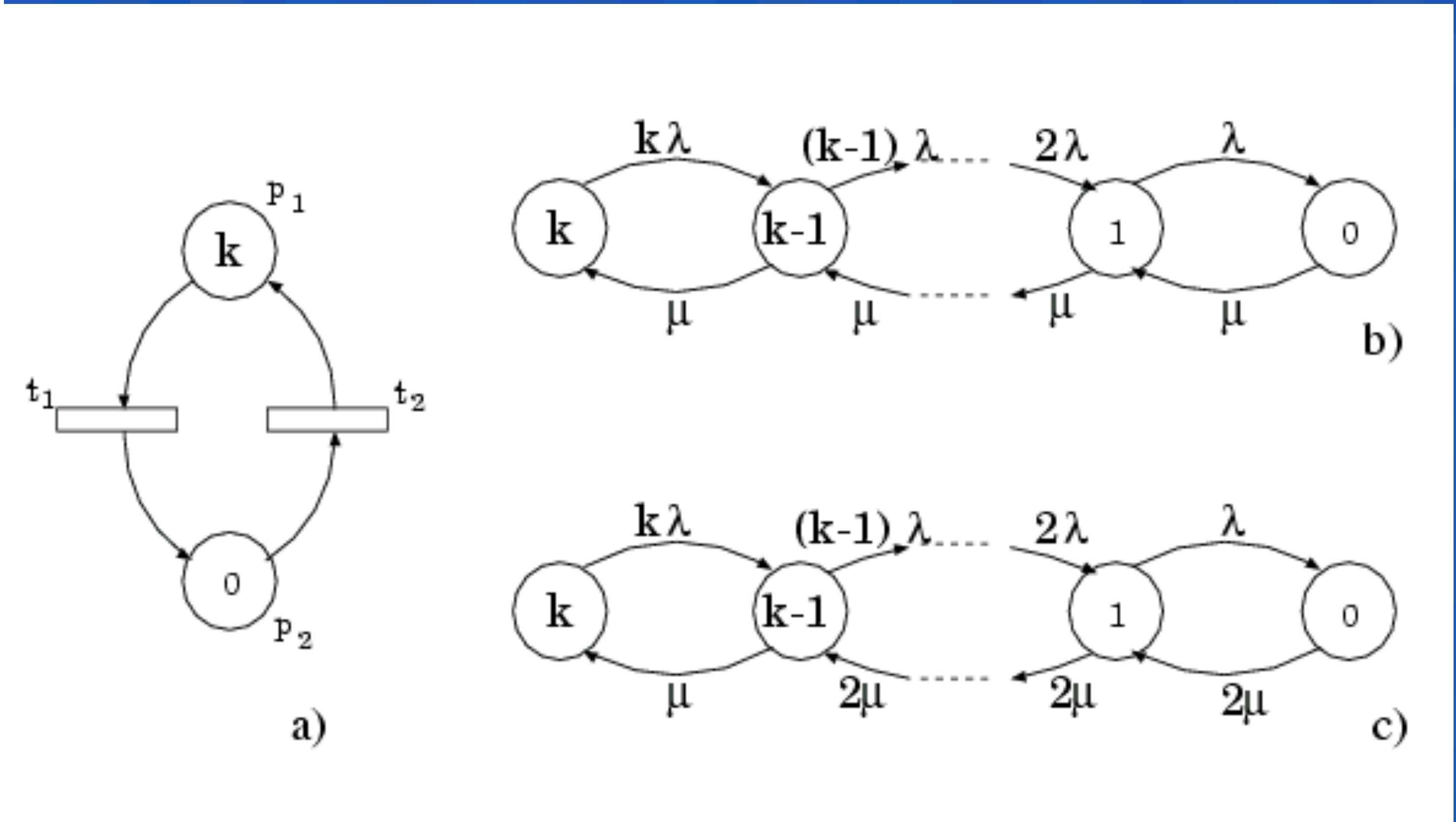


immediate trans.

t1 → arrival
t3 → service

SPN: M/M/m/n Queue (2)

K parallel repairable components

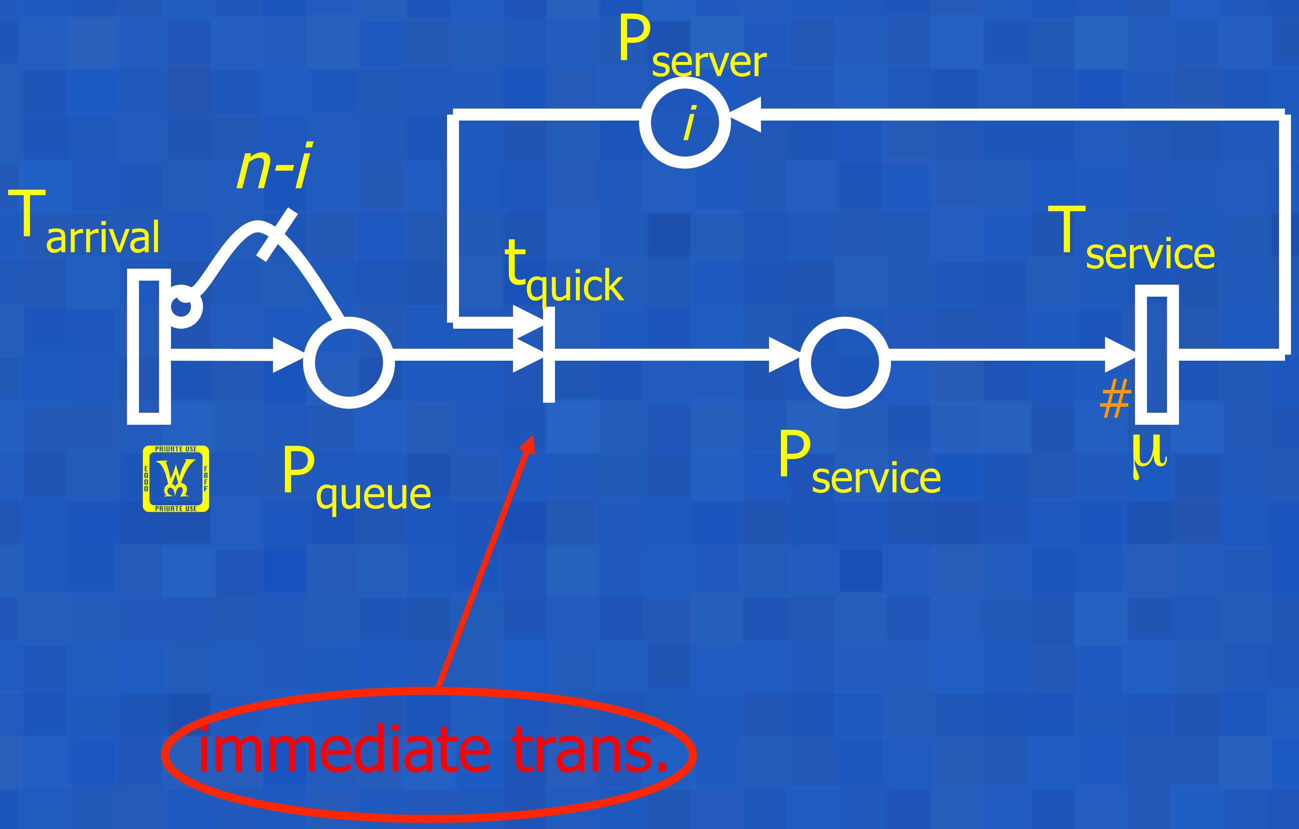


b) 1 repairman
M/M/1/n
□(t1) = #P1 □
□ (t2) = □

c) 2 repairmen
M/M/2/n
□(t1) = #P1 □

□ (t2) =
#P2 □ if #P2<2
2 □ otherwise

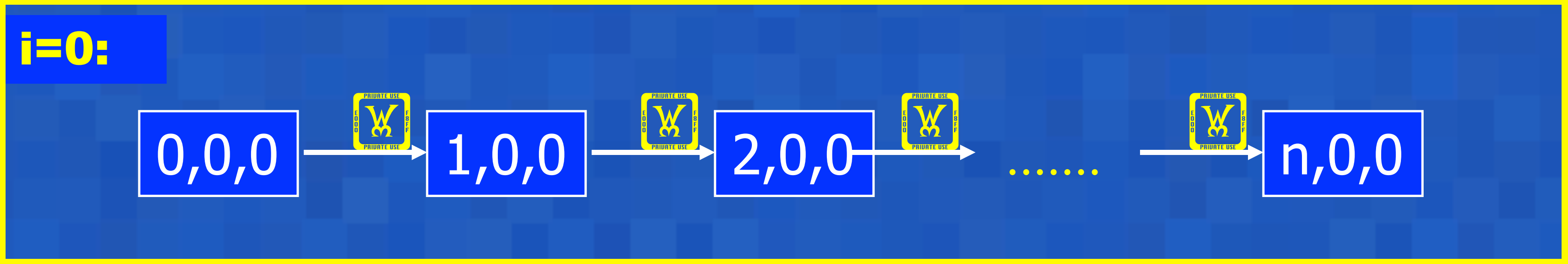
GSPN: M/M/i/n Queue



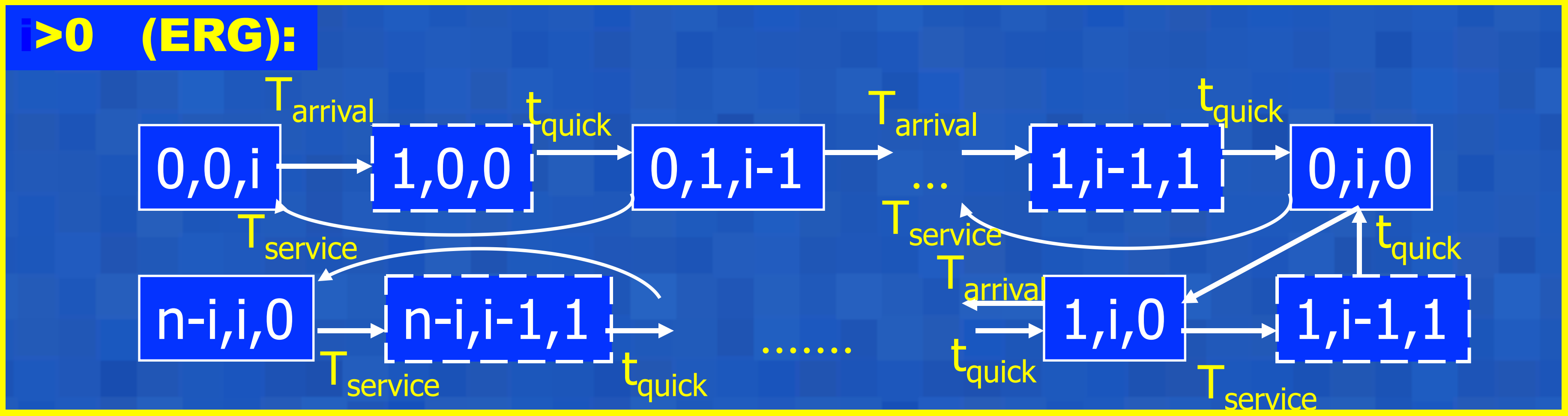
immediate trans.

ERG for M/M/i/n Queue

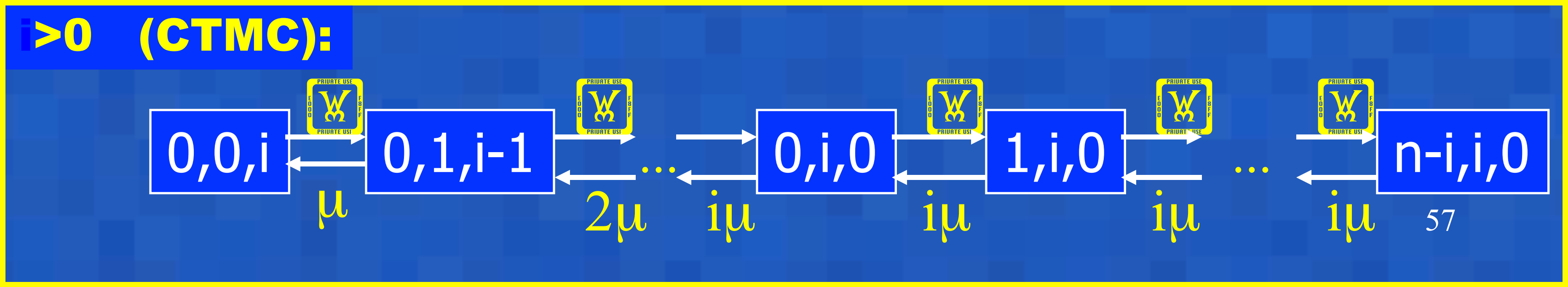
i=0:



i>0 (ERG):



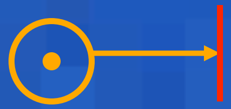
i>0 (CTMC):



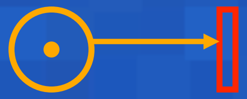
Generalized SPN

- Sometimes when some events take extremely small time to occur, it is useful to model them as instantaneous activities

- SPN models were extended to allow for such modeling by allowing some transitions, called immediate transitions, to have zero firing times

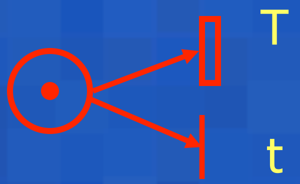


- The remaining transitions, called timed transitions, have exponentially distributed firing times



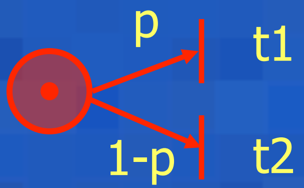
Generalized SPN

- The enabling rules are modified: if both an immediate and a timed transition are enabled in a marking, immediate transition has higher priority.



Immediate transition t is enabled!

- If more than one immediate transition is enabled in a marking, then the conflict is resolved by assigning firing probabilities to the immediate transitions.



Transition t1 & t2 will fire with p and (1-p).

GSPN Properties

- Markings (states) enabling immediate transitions are passed through in 0 time and are called vanishing.

- Markings (states) enabling timed transitions only, are called tangible.

- Since the process spends zero time in vanishing markings they do not contribute to the time behavior of the system and must be eliminated

GSPN Properties

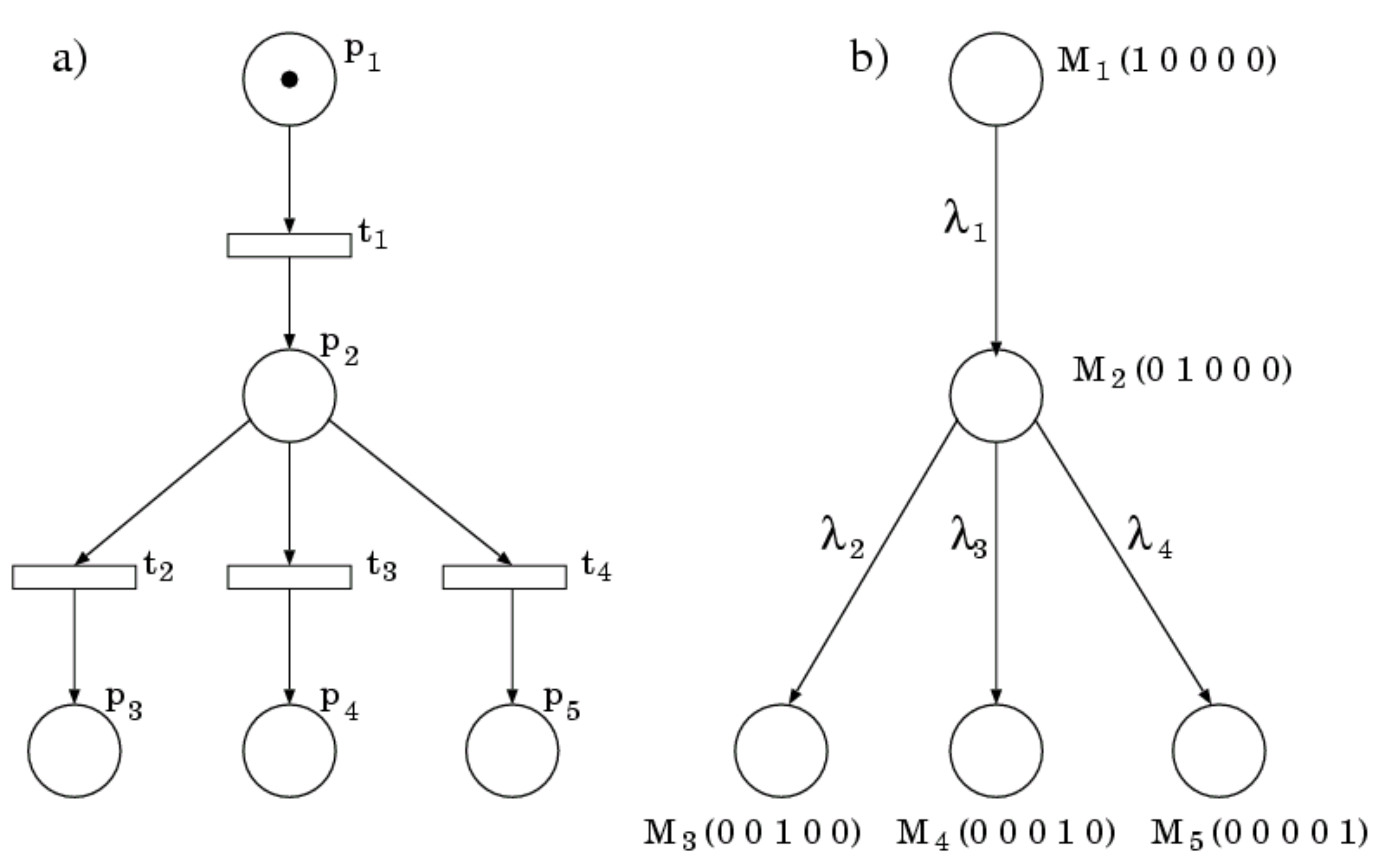
- The resulting reachability graph, referred to as the Extended Reachability Graph (ERG), contains vanishing marking, and is no longer a CTMC!

- Need to eliminate the vanishing markings to obtain the underlying CTMC.

Elimination of vanishing markings

Situation 1

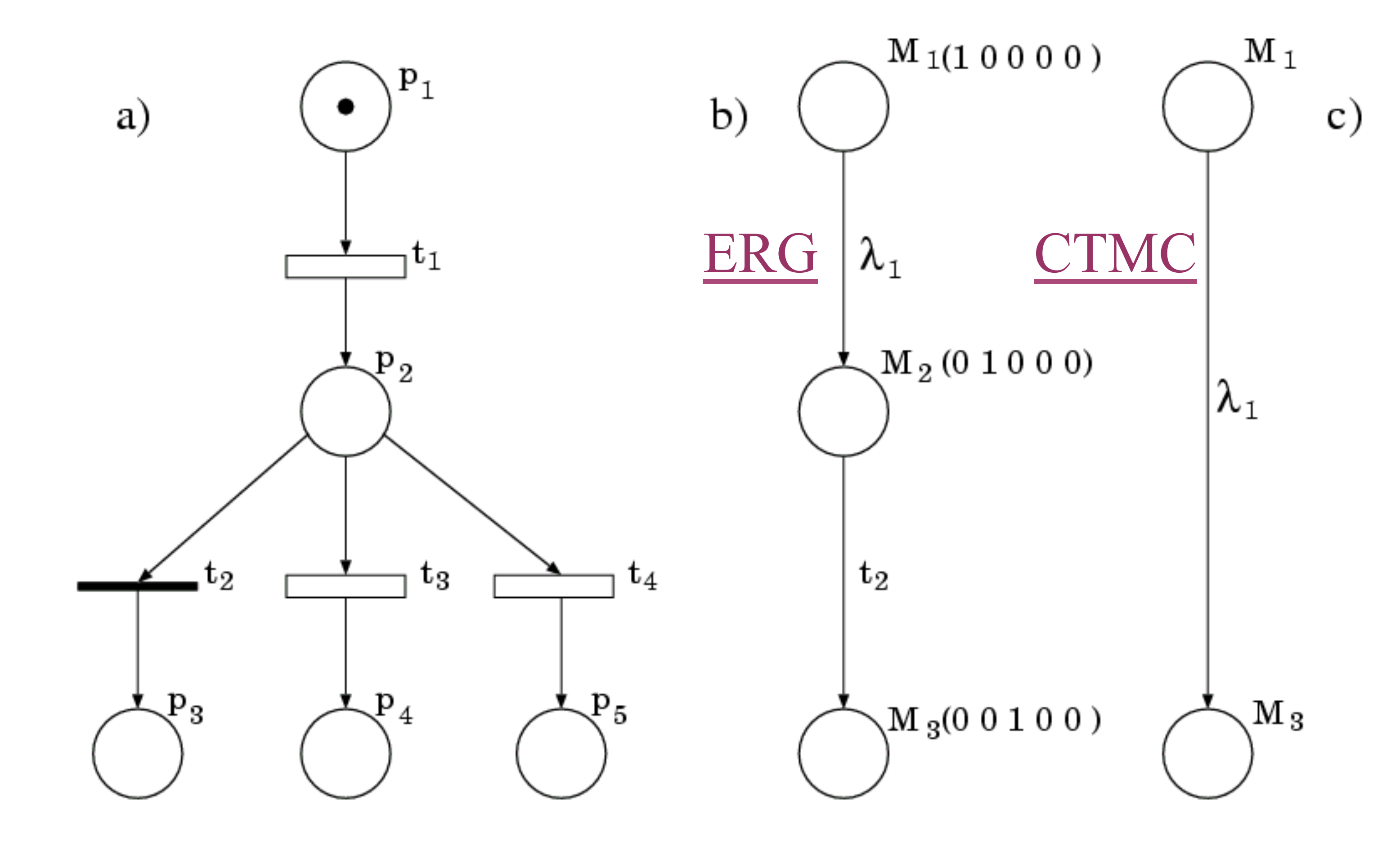
Only timed transitions are enabled.



Elimination of vanishing markings

Situation 2

One immediate and timed transitions are enabled.



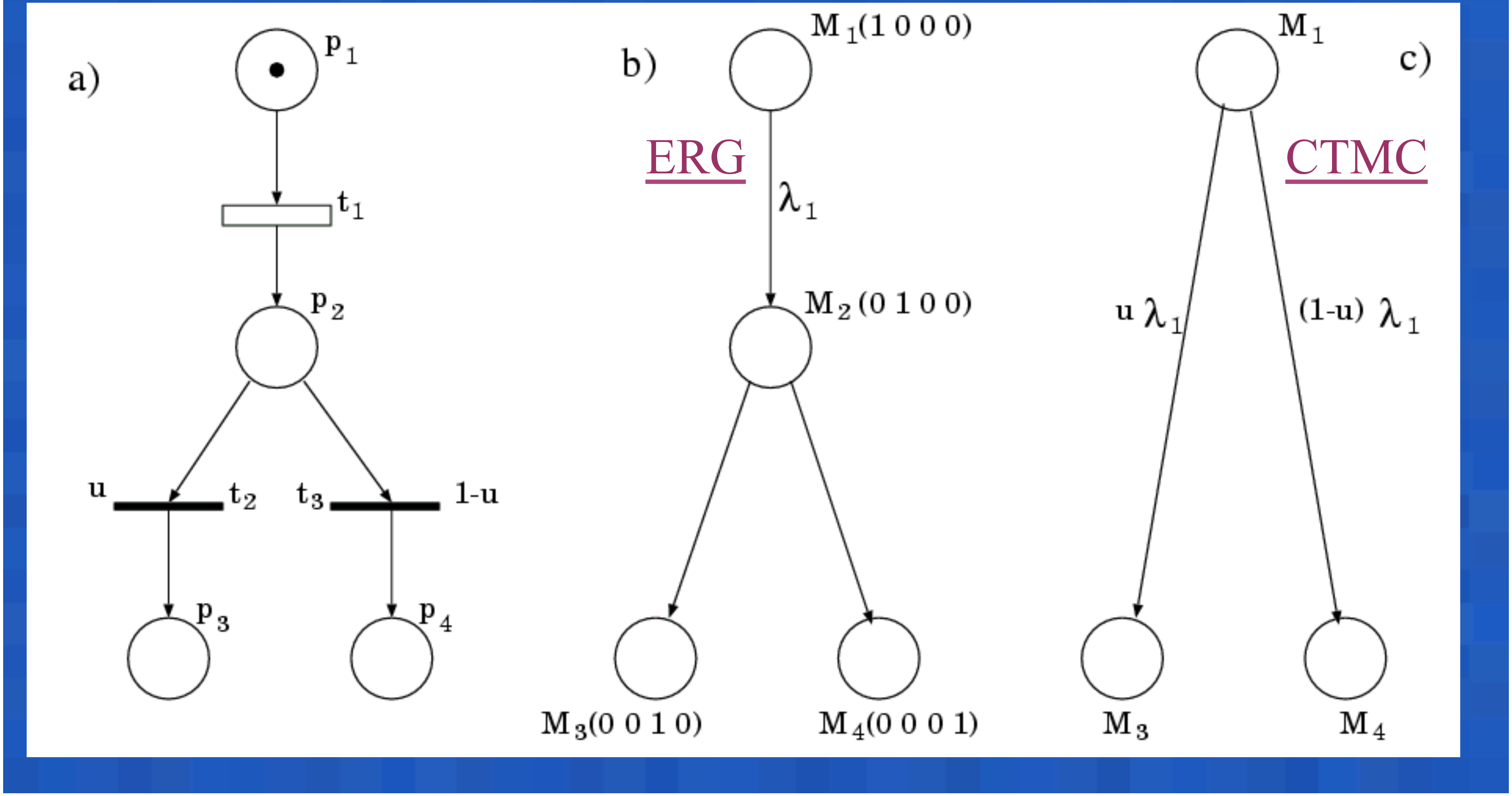
ERG

CTMC

Elimination of vanishing markings

Situation 3

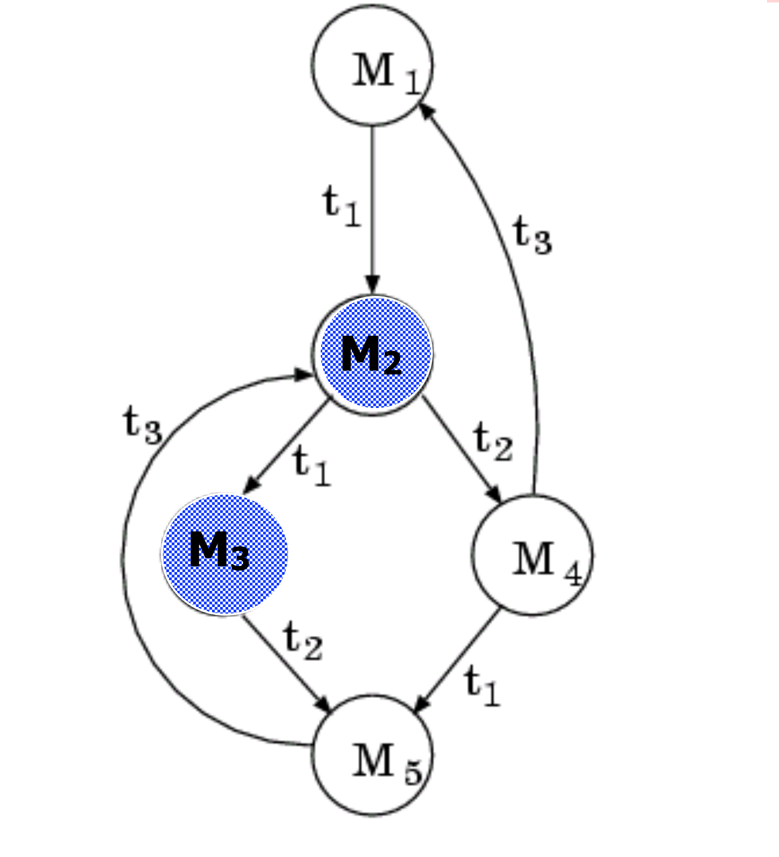
Several immediate transitions are enabled.

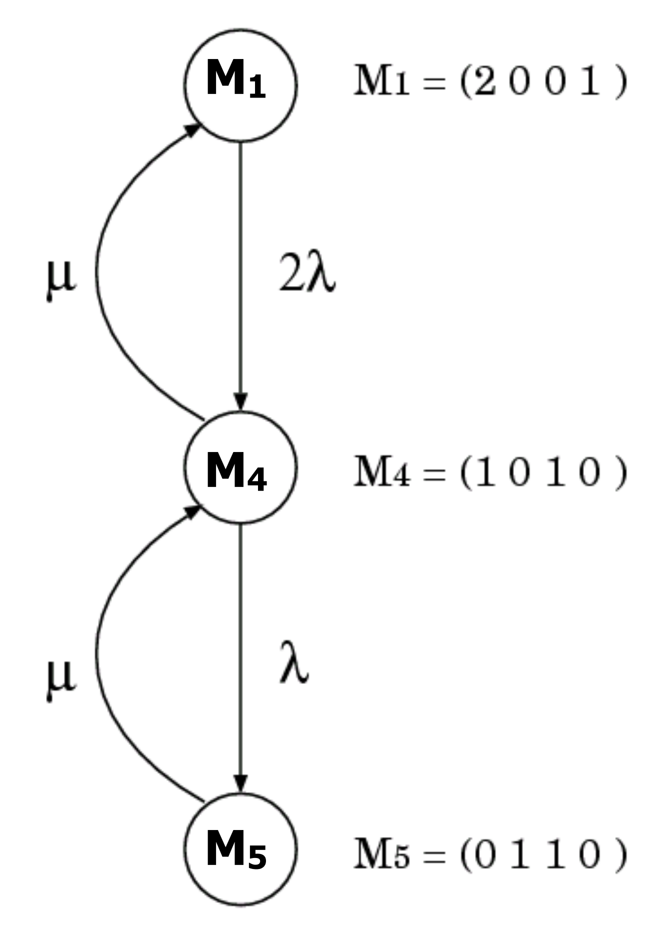


Elimination of vanishing markings

Example

t1 = #P1 □
t2 immed.
t3 = □





Traditional Methodology

- Step 3-b: Or, build a system of linear, first-order, ordinary differential equations

(Transient solution)

$$d\pi(t)/dt = \pi(t)\ Q$$

given $\pi(0) = \pi_0$

$\pi(t)$: state probability vector

Q : infinitesimal generator matrix

Traditional Methodology

- Step 3-a: Build a system of linear equations

(Steady-state solution)

$$\pi \mathbf{Q} = \mathbf{0}$$

$$\pi \mathbf{1} = \mathbf{1}$$

π **: steady-state probability vector**

$\mathbf{Q}$ **: infinitesimal generator matrix**

Measures of Reliability & Performance

Solving the model means evaluating the (transient / steady state) probability vector over the state space (markings).

However, the modeler wants to interact only at the PN: the analytical procedure must be completely transparent to the analyst.

There is a need to define the output measures at the PN level, in term of the PN primitives.

Measures of Reliability & Performance

Output measures defined at the PN level.

- Probability of a given condition on the PN;
- Time spent in a marking;
- Mean (first) passage time;
- Distribution of tokens in a place;
- Expected number of firing of a PN trans (throughput).

All these measures can be reformulated in terms of reward functions (MRM)

Solving models with SPN

The use of SPN requires only the topology of the PN, the firing rates of the transitions and the specification of the output measures.

All the subsequent steps, which consist in:

- generation of the reachability graph
- generation of the associated Markov chain;
- transient and s.s. solution of the Markov chain;
- evaluation of the relevant process measures.

must be completely automated by a computer program, thus making transparent to the user the associated mathematics.

Probability of a given condition on the PN

Define a condition by a logical function (e.g #Pf = 0) and find the subset of states S where the condition holds true.

$$Q_S(t) = Prob\,\{condition\ \ is\ \ true\ \ at\ \ time\ \ t\ \}$$

$$Q_S(t) = \sum_{s \in S} q_s(t)$$

In terms of reward rate $r_s = \begin{cases} 1 & s \in S \\ 0 & \text{otherwise} \end{cases}$

Expected time spent in a marking

Define a condition by a logical function (e.g #Pf = 0) and find the subset of states S where the condition holds true.

$$\psi_S(t) = \sum_{s \in S} \int_0^t q_s(z)\,dz$$

$$Q_S(\infty) = \sum_{s \in S} q_s(\infty)$$

In terms of reward rate $r_s = \begin{cases} 1 & s \in S \\ 0 & \text{otherwise} \end{cases}$

Mean first passage time

If the subset of states S is absorbing, Qs(t) is the probability of first visit to S.

The mean first passage time is:

$$\phi_S = \int_0^\infty [1 - Q_S(z)]\, dz$$

The above formula requires the transient analysis to be extended over long intervals (other more direct techniques are available).

Distribution of tokens in a place

The density mass of having k (k = 0, 1, 2, …) tokens in a place ***pi*** is ***f_i (k,t).***

f_i (k,t) can be evaluated by summing the probability of all the markings containing k (k = 0, 1, 2, …) tokens in ***pi***.

$$f_i\,(k,t) = \underset{\substack{s\ \square\ \#pi\ = \\ k}}{\square}\quad q_{\,s}\,(t)$$

Expected number of tokens in a place

Given the density mass of having k (k = 0, 1, 2, …) tokens in a place ***pi***, the expected number of tokens in place ***pi*** can be evaluated by:

$$E\left[m_i(t)\right] = \sum_{k=0}^{\infty} k\, f_i(k,t)$$

In terms of reward rate $r_s = k$

Expected number of firings

Given an interval (0,t) this quantity indicates how many times, on the average, an event modeled by a PN transition has occurred (throughput).

Let S be the subset of markings enabling tk.

$$\eta_k(t) = \sum_{s \in S} \int_0^t q_s(z)\, \lambda_k(s)\, dz$$

$$\nu_k = \sum_{s \in S} q_s(\infty)\, \lambda_k(s)$$

In terms of reward rate $r_s = \lambda_k(s)$

Example: Multiprocessor with failure

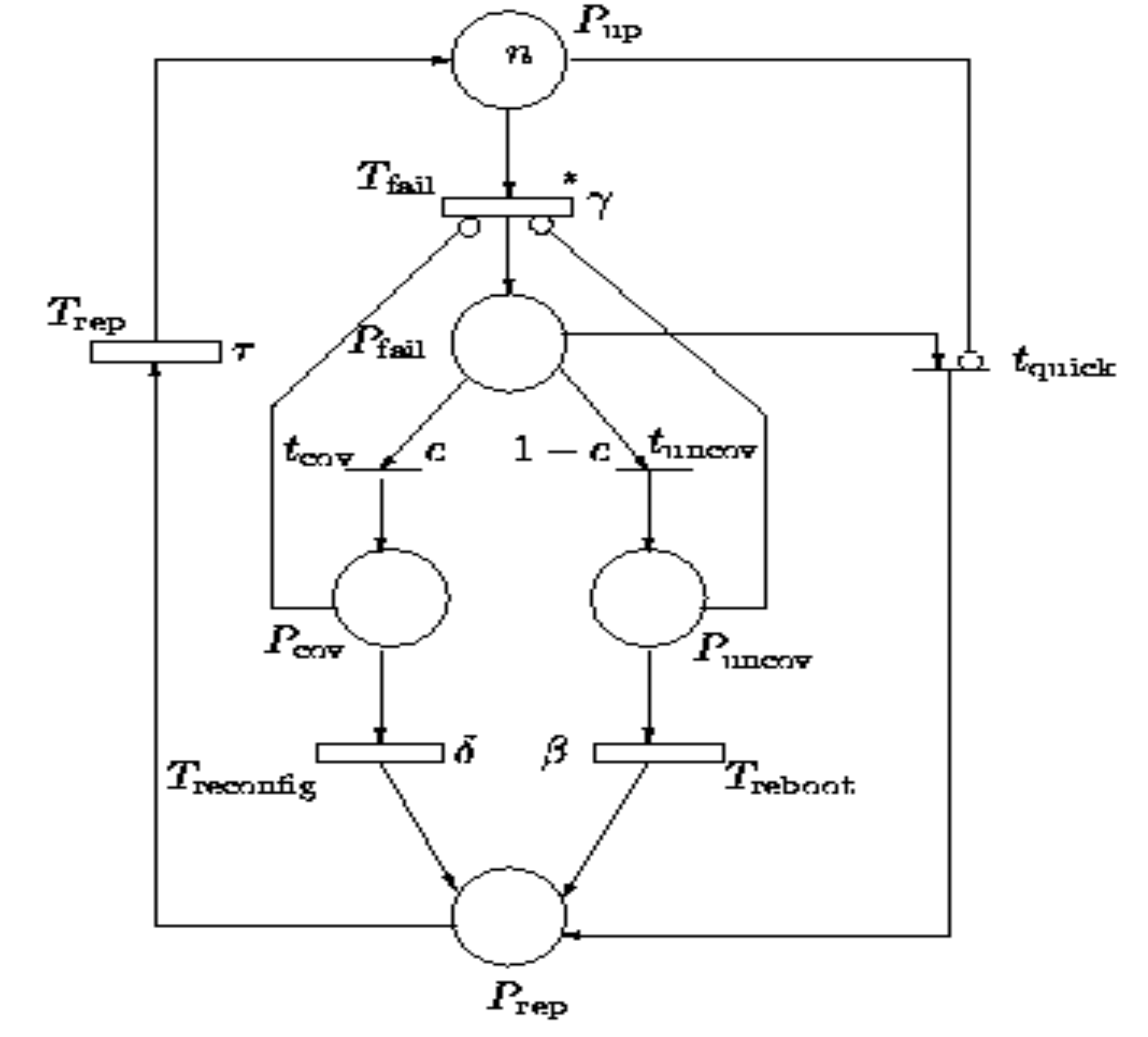
- Number of processors: *n*
- Single repair facility is shared by all processors
- A reconfiguration is needed after a covered fault
- A reboot is required after an uncovered fault

γ

Assumptions:

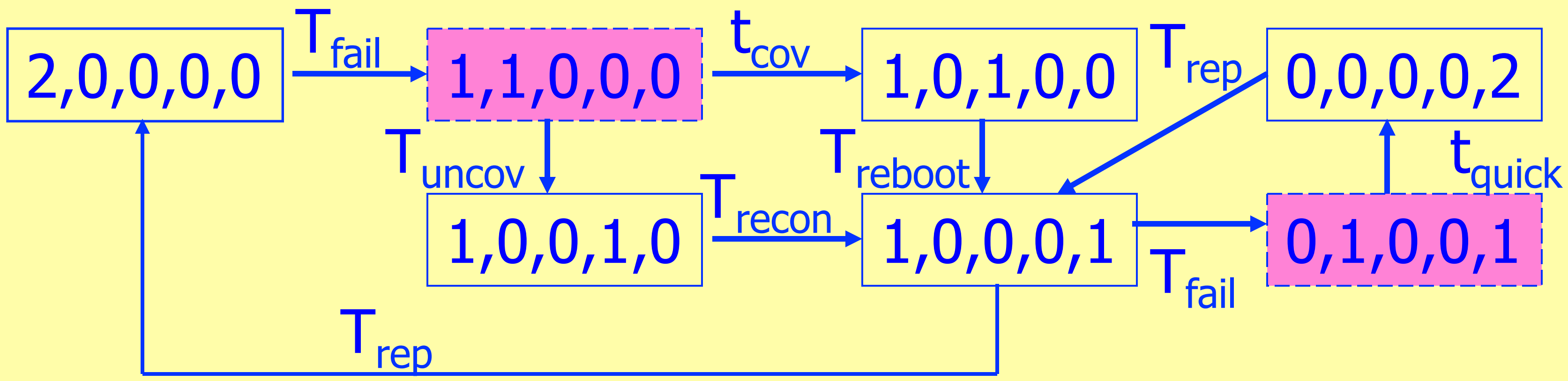
- The failure rate of each processor is γ
- The repair times are exponentially distributed with mean $1/\tau$
- A processor fault is covered with probability c
- The reconfiguration times and the reboot times are exponentially distributed with parameter δ and β, respectively

GSPN Model for Multiprocessor

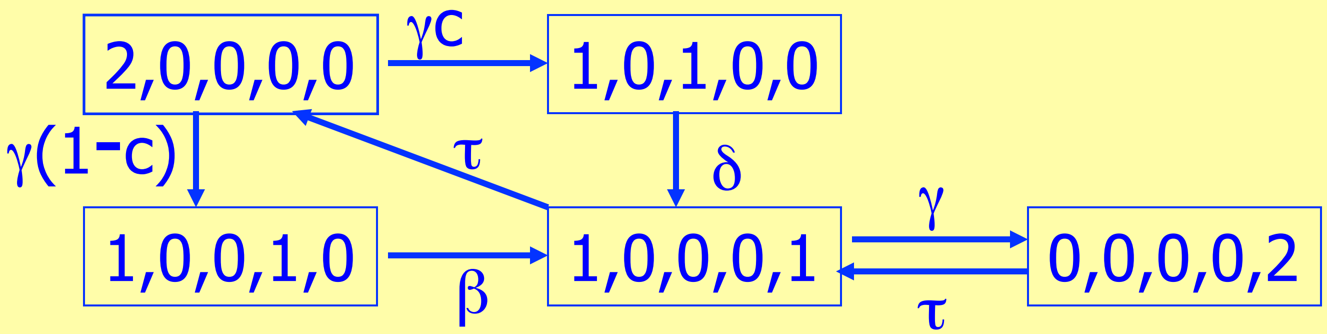


GSPN Model of a Multiprocessor

ERG for Multiprocessor Model (n=2)



Extended Reachability Graph for Multiprocessor model



Reduced ERG for Multiprocessor model

Example: Reward Rates for Multiprocessor Availability

- Reward rate at the net level for steady –state availability

$$r_i = \begin{cases} 1, & \#P_{up} \geq 1 \text{ and } (\#P_{\text{cov}} + \#P_{un\,\text{cov}}) = 0 \\ 0, & \text{otherwise} \end{cases}$$

- Reward rate at the CTMC level for steady-state availability (n=2)

$$r_i = \begin{cases} 1, & i = (2,0,0,0,0), (1,0,0,0,1) \\ 0, & \text{otherwise} \end{cases}$$

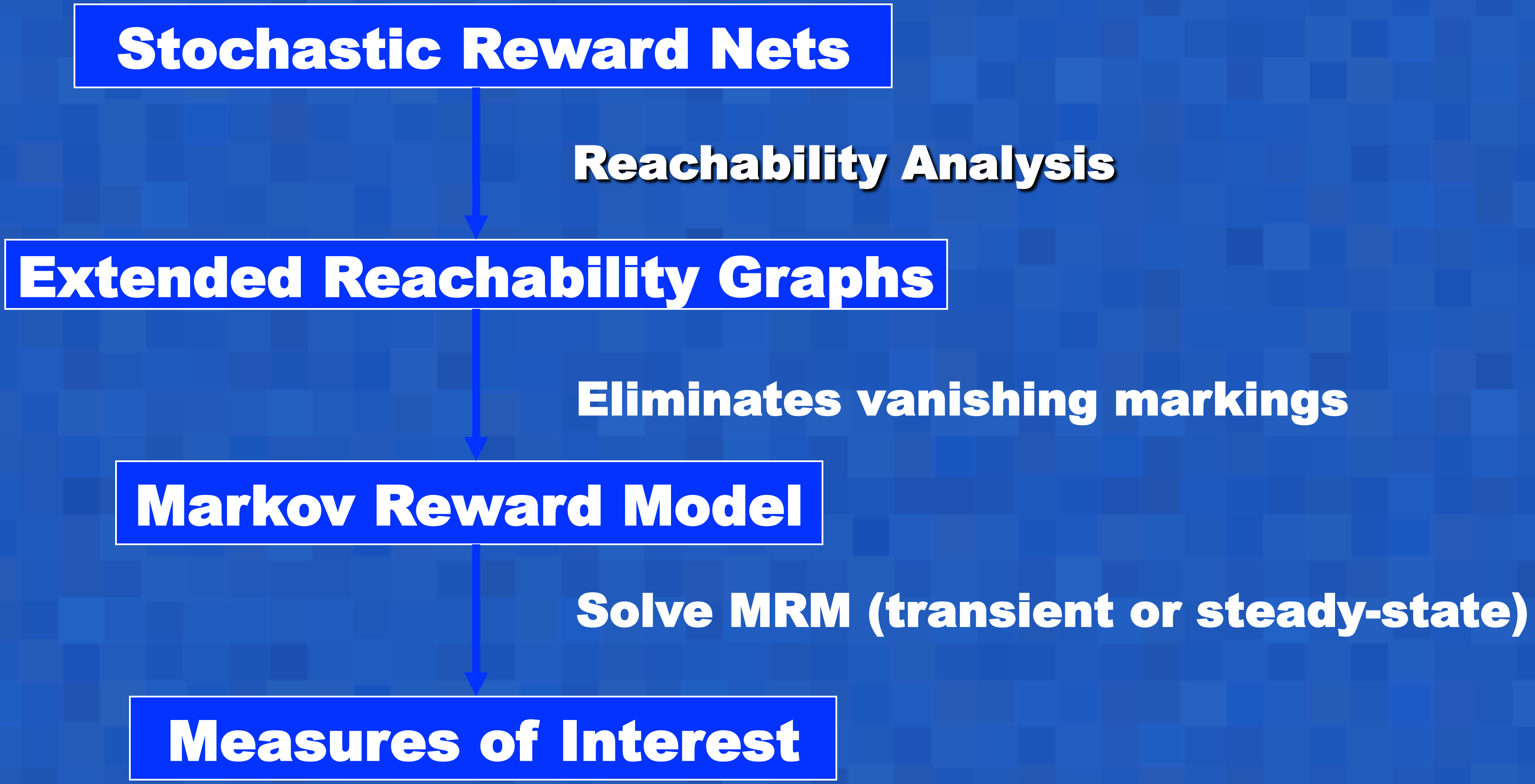
Stochastic Reward Net (SRN)

- Introduced by Ciardo, Muppala and Trivedi [1989]
- Structural characteristics
  - Extensive Marking dependency allowed for firing rates and firing probabilities
  - Transition Priorities
  - Guards (Enabling functions) for Transitions
  - Variable cardinality arcs

Stochastic Reward Net (SRN)

- Stochastic characteristics
  - Allow definition of reward rates in terms of net level entities
  - Automatically generate the reward rates for the markings
  - Enables computation of required measures of interest

Analysis Procedure of SRN



Stochastic Reward Nets

Reachability Analysis

Extended Reachability Graphs

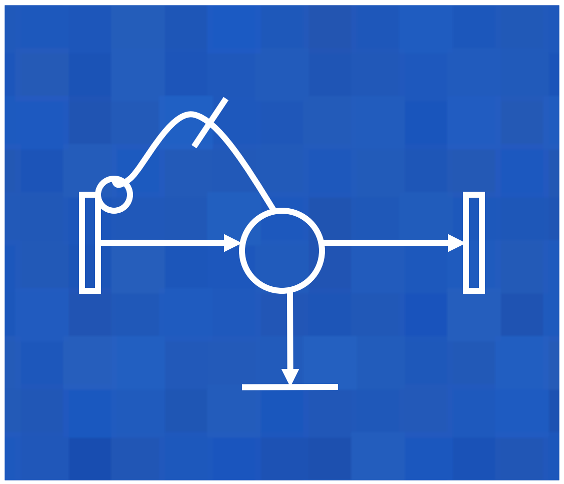
Eliminates vanishing markings

Markov Reward Model

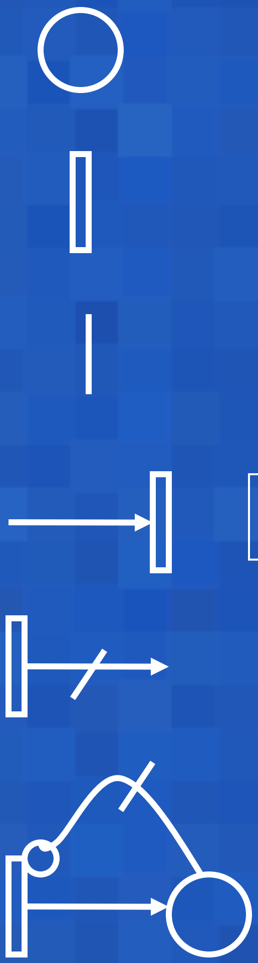
Solve MRM (transient or steady-state)

Measures of Interest

SRN Summary



An SRN



Place

Timed Transition

Immediate Transition

Input Arc

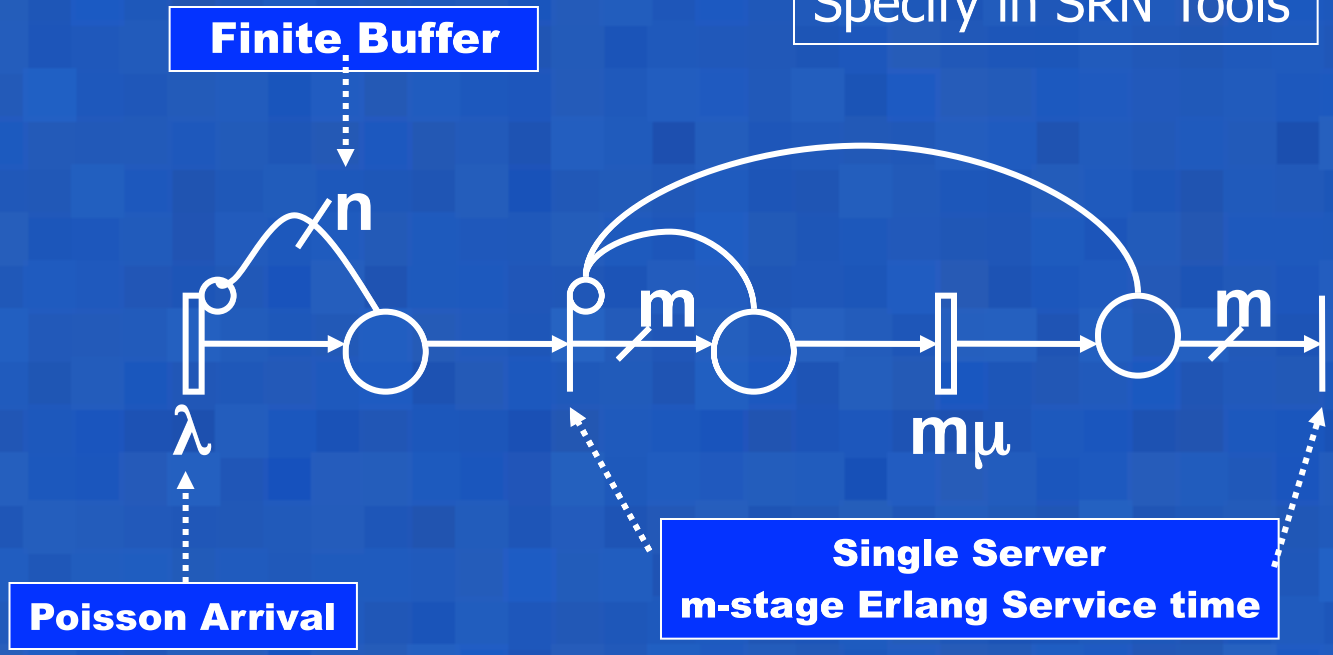
Output Arc

Inhibit Arc

SRN Analysis: Step-1

- Abstract the system -> SRN Model

Specify in SRN Tools



Finite Buffer

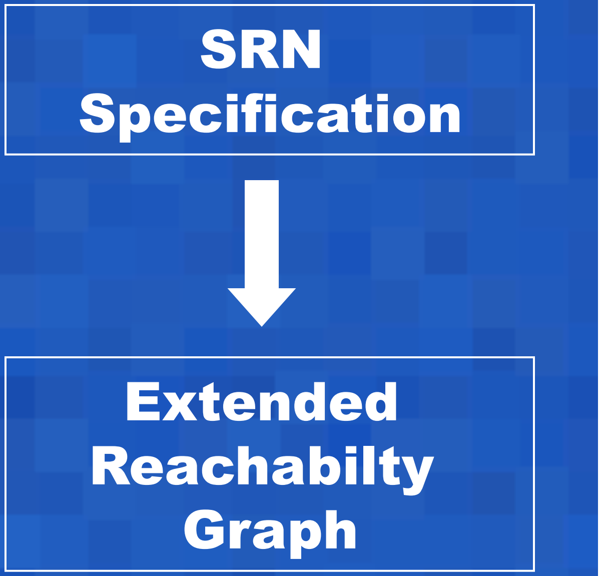
Poisson Arrival

Single Server
m-stage Erlang Service time

SRN of M/E$_m$/1/n Queue

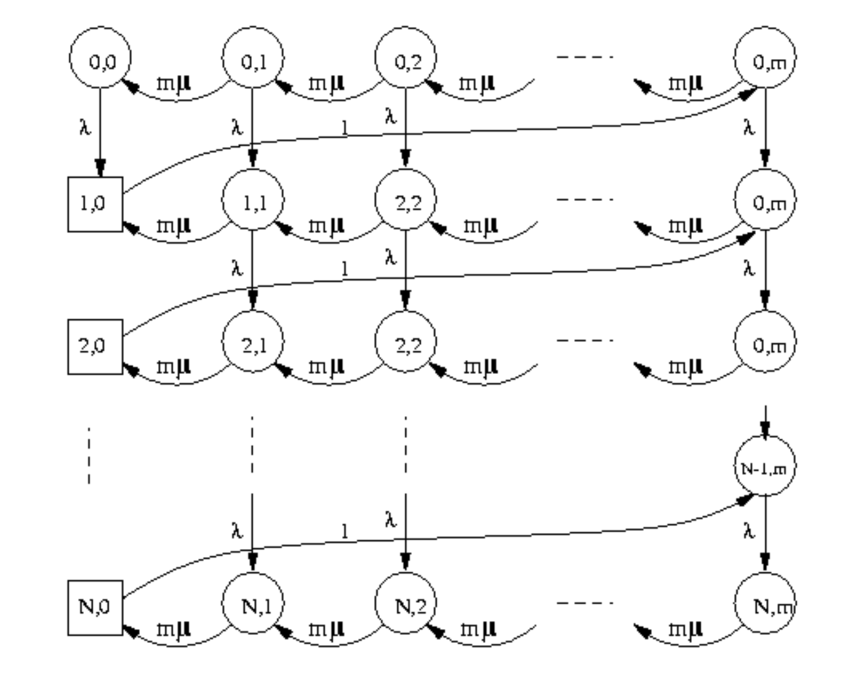
SRN Analysis: Step-2

- **Reachability Analysis: Automatically Generate ERG**



SRN Specification

Extended Reachabilty Graph

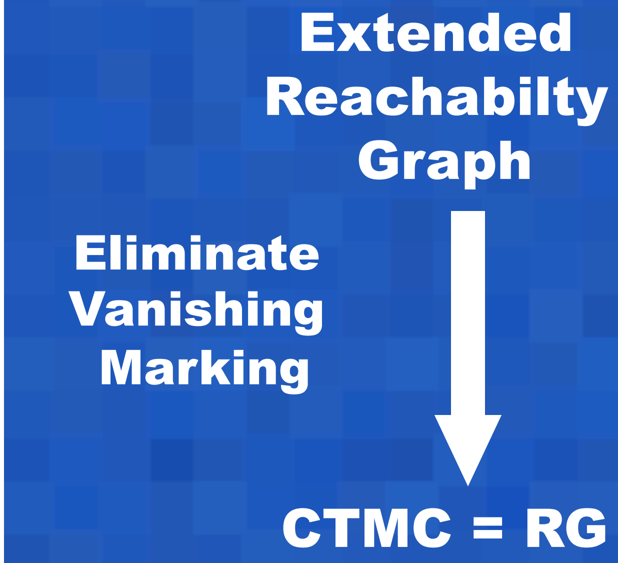


Extended Reachability Graph (ERG)

□ **Vanishing Marking** ○ **Tangible Marking**

SRN Analysis: Step-3

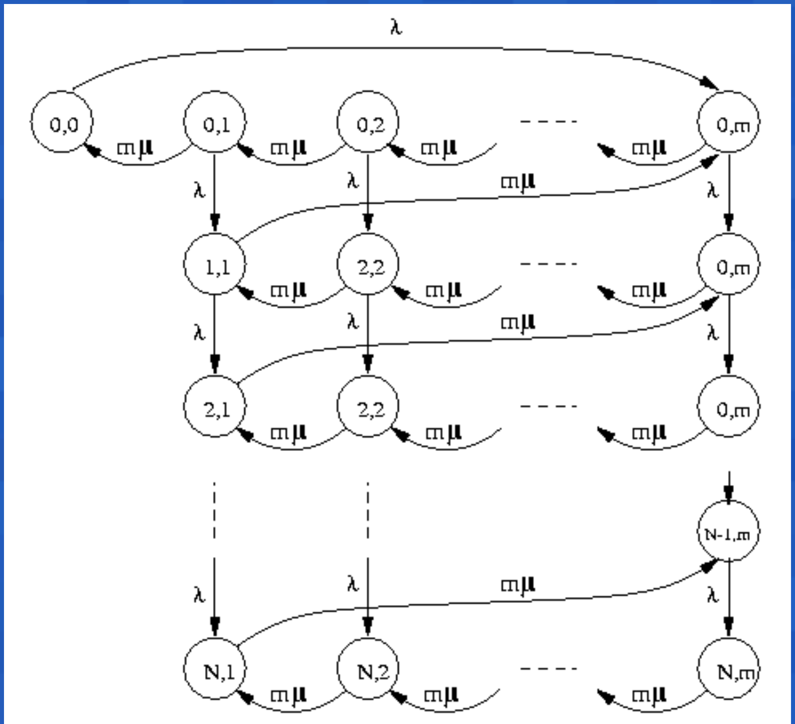
- **Reachability Analysis: Automatically Generate RG**



Extended Reachabilty Graph

Eliminate Vanishing Marking

CTMC = RG



□ ✗ ○ ✓

SRN Analysis: Step-4

- Solve CTMC
  - Steady-state Analysis: A System of Linear Equations
    - Gauss-Seidel, SOR (Successive over-relaxation)
    - Power method, etc.
  - Transient Analysis: A coupled system of ODE
    - Classical ODE Methods
    - Randomization (or Uniformization), etc.

SRN Analysis: Step-5

- Compute measures of interest
  - Measures of interests: Blocking/Dropping Probability, Throughput, Utilization, Delay etc.
  - Measures can be defined as reward functions which specify reward rates on net-level entities.

Steps 1-5: The SPN Tool does it all!

Non-Markovian SPN

Transition Firing Time: not exponentially distributed

H.Choi,V. Kulkarni, K. Trivedi
Markov regenerative stochastic Petri net (MRSPN)
Performance Evaluation, 20, 337-357, 1994
(A special case: At most one general transition can be enabled in any marking).

A. Bobbio and A. Puliafito and M. Telek and K. Trivedi.
Recent developments in non-Markovian stochastic Petri nets.
Journal of Systems Circuits and Computers, 8:1, 119-158, 1998.

Fluid Petri Net

- **Fluid stochastic Petri net (FSPN)**
  - **Introduced by K. Trivedi and V. Kulkarni (1993)**
  - **Allow both discrete and continuous places**
  - **Useful in fluid approximation of discrete queueing system**
  - **Powerful formalism of stochastic fluid queueing networks**
  - **Boundary conditions complicated. Solution techniques under investigation.**

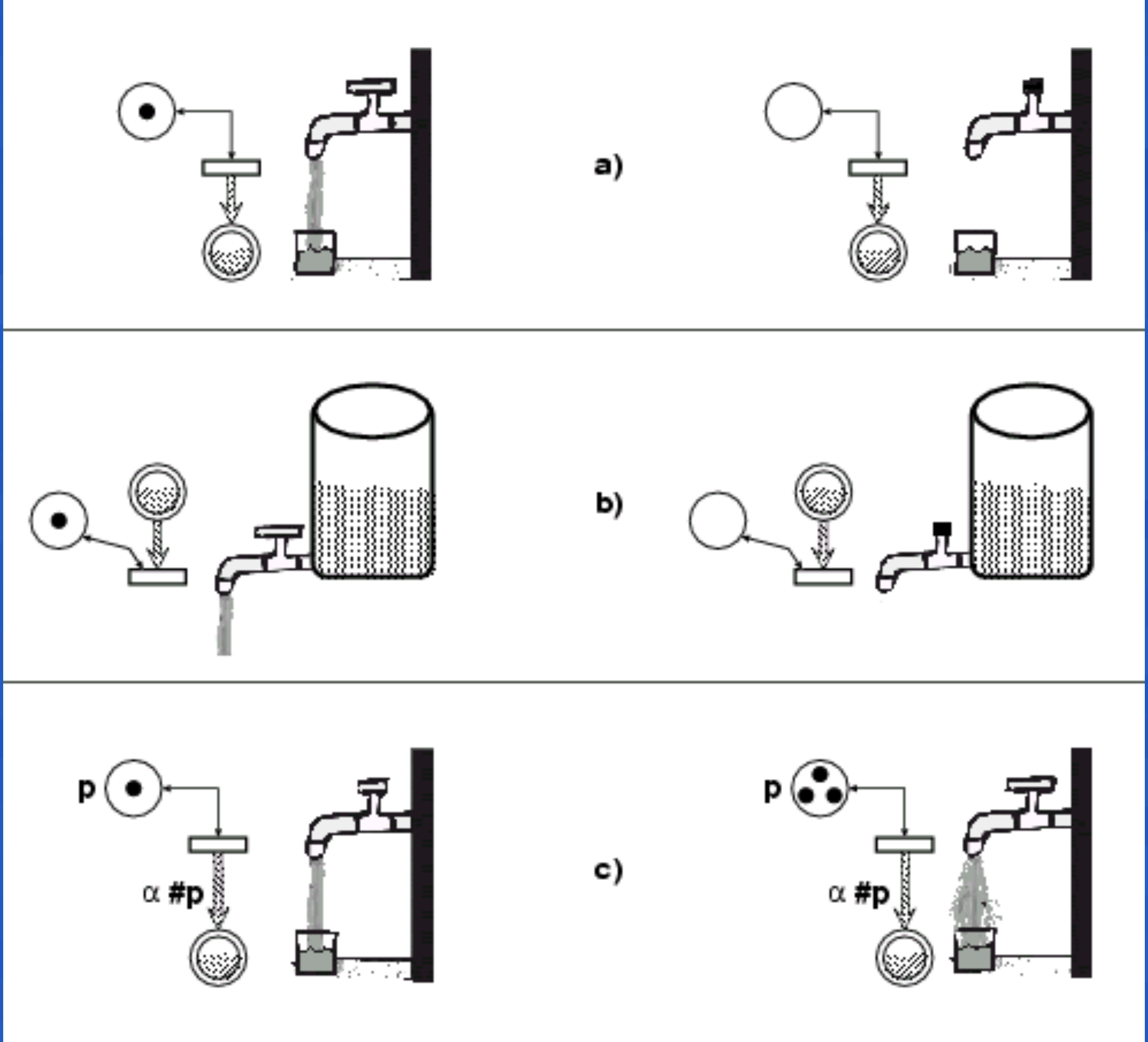
The Fluid Petri Net Model

FPN's are an extension of PN able to model the coexistence of discrete and continuous variables.

The primitives of FPN (places, transitions and arcs) are partitioned in two groups:

- discrete primitives that handle discrete tokens (as in standard PN);
- continuous (or fluid) primitives that handle continuous (fluid) quantities.
- fluid arcs are assigned instantaneous flow rates.

Fluid Petri Nets



References

- http://www.ee.duke.edu/~kst/

  then click on Stochastic Petri Nets

- K. Trivedi, Probability and Statistics with Reliability, Queuing, and Computer Science Applications, 2nd Ed., John Wiley and Sons, New York, 2001

Conclusion