SYSTEM MODELLING WITH PETRI NETS

Andrea BOBBIO

Istituto Elettrotecnico Nazionale Galileo Ferraris
Strada delle Cacce 91, 10135 Torino, Italy

Reprinted from:
A.G. Colombo and A. Saiz de Bustamante (eds.),
System Reliability Assessment,
Kluwer p.c.,
pp 102-143, (1990)

CONTENTS

SYSTEM MODELLING WITH PETRI NETS

Andrea BOBBIO

Istituto Elettrotecnico Nazionale Galileo Ferraris
Strada delle Cacce 91, 10135 Torino, Italy

ABSTRACT. Petri Nets (PN) are a graphical formalism which is gaining popularity in recent years as a tool for the representation of complex logical interactions (like synchronization, sequentiality, concurrency and conflict) among physical components or activities in a system. This notes are devoted to introduce the formalism of Petri nets with particular emphasis on the application of the methodology in the area of the performance and reliability modelling and analysis of systems. The quantitative analysis of the behaviour of systems in time requires the superposition of a stochastic timing mechanism to the classical representation of PN. Timed Petri nets and, in particular, Stochastic Petri nets (SPN) are the object of the second part of the notes. Finally, some fully developed examples enlighten peculiar aspects which differentiate PNs from other modelling techniques usual in reliability analysis. In few words, the goal of these notes is to show that the proposed methodology based on the PN formalism can be conveniently used as a user-friendly language to represent and evaluate complex stochastic systems.

1. Introduction

Petri Nets (PN) are a graphical tool for the formal description of the flow of activities in complex systems. With respect to other more popular techniques of graphical system representation (like block diagrams or logical trees), PN are particularly suited to represent in a natural way logical interactions among parts or activities in a system. Typical situations that can be modelled by PN are synchronization, sequentiality, concurrency and conflict.

The theory of PN originated from the doctoral thesis of C.A. Petri in 1962 [39]. Since then, the formal language of PN has been developed and used in many theoretical as well as applicative areas. Introductory survey papers can be found in [37, 3]. Several textbooks on the subject are also available: [38] (where an extended annotated bibliography is contained) [42, 13, 44]. An yearly Workshop on "Application and Theory of Petri Nets" is held in Europe (the IX edition of the workshop took place in Venice in June 1988).

The classical PNs do not convey any notion of time; in order to use the PN formalism for the quantitative analysis of the performance and reliability of system versus time, a class of Timed PN (TPN) has been introduced. The time variables associated to the PN can be either deterministic variables (leading to the class of models called deterministic PN), or random variables (leading to the class of models called Stochastic PN - SPN). The bibliography on TPN is not as wide as the one on classical PN, however, an extended

collection of papers and applications can be found in the proceedings of two international workshops specifically devoted to the use of TPN in performance and reliability evaluation [1, 2].

The first part (sections 3,4,5,6 and 7) of these lecture notes is aimed at introducing the classical theory of PN, while the second part (sections 8,9,10 and 11) discusses the stochastic timing of a PN with application in the field of reliability modelling and evaluation.

In particular, Section 2 contains the list of symbols. Section 3 defines the primitive elements of the PN and the execution rules by means of which the dynamic properties of the system are described. Section 4 illustrates typical examples of logical interactions among activities modelled by PN, while Section 5 introduces characteristic properties of PNs. Section 6 shows how a PN can be analyzed through the generation of the *reachability tree*, or by means of matrix techniques. Finally, some possible extensions of the modelling capabilities of classical PNs are considered in Section 7.

The second part of the lecture notes is devoted to illustrate how PNs can be conveniently used as a modelling language for the quantitative analysis of the performance and reliability of systems. The use of PN for this purpose requires that the duration of the activities representing the system operations can be specified and measured; therefore, the first step toward the definition of a suitable modelling framework is the insertion of the notion of time in classical PNs. This topic is addressed in Section 8.

Section 9 examines the class of Stochastic PN (SPN) in which the durations of the activities are exponentially distributed random variables. With this assumption, the dynamic behaviour of the PN can be mapped into a continuous-time homogeneous Markov chain. In this way we cast a natural bridge between SPN and Markov models in reliability analysis. With reference to the above models, we discuss the following topics: the generation of the Markov chain associated to the PN, the assignment of marking dependent transition rates and the partition of PN transitions into immediate and timed transitions. Section 10 shows how SPN models can be naturally used to define interesting measures for the characterization of the system behaviour versus time, and how these measures can be computed from the associated Markov chain. Section 11 illustrates some examples of application of the above methodology in reliability analysis. Section 12 briefly introduces the implementation of simulative techniques in the analysis of SPN.

2. List of Symbols

D^-, D^+, D	-	Input, Output and Incidence matrix
$\underline{e}_j$	-	0-vector whith entry j equal to 1
$\mathcal{E}$	-	Execution sequence
$\mathcal{G}_R$	-	Reachability graph
I	-	Input function
$HMSPN$	-	Homogeneous Markov Stochastic Petri Net
$L = \{\lambda_1, \lambda_2, \ldots, \lambda_{nt}\}$	-	Set of firing rates
$M = \{m_1, m_2, \ldots, m_{np}\}$	-	Marking
N	-	Cardinality of the reachability set (state space)
n_p	-	Number of places
n_t	-	Number of transitions

O	-	Output function
PN	-	Petri Net
$P = \{p_1, p_2, \ldots, p_{np}\}$	-	Set of places
$\underline{Q}(t)$	-	State probability vector of the associated Markov chain
$\overline{\mathcal{R}}$	-	Reachability set
SPN	-	Stochastic Petri Net
$T = \{t_1, t_2, \ldots, t_{nt}\}$	-	Set of transitions
$\mathcal{T}_E$	-	Timed execution sequence
TPN	-	Timed Petri Net
$\underline{U}_p$	-	Unitary vector
v	-	Branching probability in a random switch
$\underline{W}_p$	-	Vector of binary (0, 1) entries
$\underline{X}$	-	Integer vector
$\eta_j(t)$	-	Expected number of firings of t_j in $0 - t$
θ_j	-	Random firing time associated to t_j
λ, μ, γ, ρ	-	Firing rates
Λ	-	Transition rate matrix of the associated Markov chain
ϕ	-	Mean passage time
τ_j	-	Epoch of firing of $t_{(j)}$
$\psi(t)$	-	Expected time spent in a marking in $0 - t$
ω	-	Infinite reproducibility in the reachability tree

3. The Primitive Elements of a Petri Net

For definitions and notation we refer in general to [38]. A Marked PN is a quintuple (P, T, I, O, M), where:

- $P = \{p_1, p_2, \ldots, p_{np}\}$ is the set of n_p places (drawn as circles in the graphical representation);
- $T = \{t_1, t_2, \ldots, t_{nt}\}$ is the set of n_t transitions (drawn as bars);
- I is the transition input relation and is represented by means of arcs directed from places to transitions;
- O is the transition output relation and is represented by means of arcs directed from transitions to places;
- $M = \{m_1, m_2, \ldots, m_{np}\}$ is the marking. The generic entry m_i is the number of tokens (drawn as black dots) in place p_i in marking M.

The graphical structure of a PN is a bipartite directed graph: the nodes belong to two different classes (places and transitions) and the edges (arcs) are allowed to connect only nodes of different classes (multiple arcs are possible in the definition of the I and O relations [38]). Figure 1 is a PN [3].

The dynamics of a PN is obtained by moving the tokens in the places by means of the following execution rules:

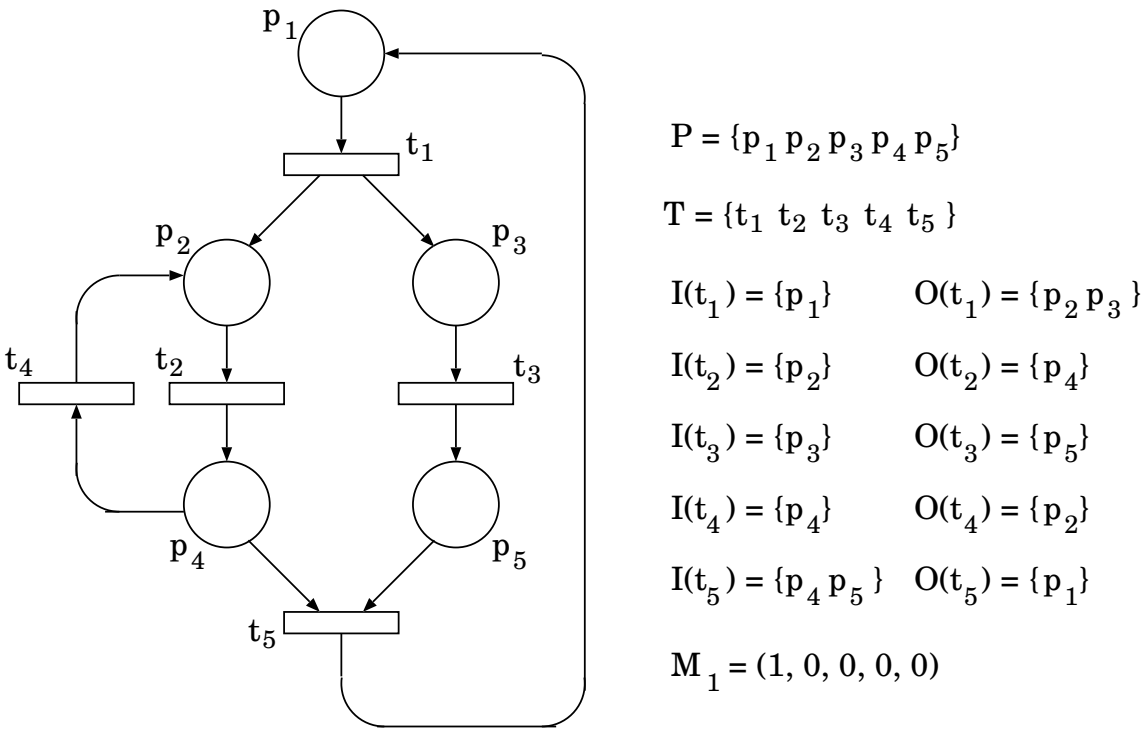

Figure 1: A PN graph with Input and Output functions.

- A transition is enabled in a marking M if all its input places carry at least one token;
- an enabled transition fires by removing one token per arc from each input place and adding one token per arc to each output place.

Given an initial marking M_1, the reachability set $\mathcal{R}(M_1)$ is the set of all the markings that can be obtained by repeated application of the above rules.

More formally we can say that t_k is enabled in marking M if:

$$\text{for any} \qquad p_i \in I(t_k) \qquad , \qquad m_i \geq 1$$

Marking M', obtained from M by firing t_k, is said to be *immediately reachable* from M, and the firing operation is denoted by the symbol $(M - t_k \rightarrow M')$. The token count in M' is pictorially represented in Figure 2, and is given by the following relationship:

$$M'(p_i) = \begin{cases} M(p_i) + 1 & \text{if} \quad p_i \in O(t_k)\ , \quad p_i \notin I(t_k) \\ M(p_i) - 1 & \text{if} \quad p_i \notin O(t_k)\ , \quad p_i \in I(t_k) \\ M(p_i) & \text{otherwise} \end{cases}$$

Let us examine the generation of the reachability set of the PN of Figure 1 given the initial marking $M_1 = (1, 0, 0, 0, 0)$. In M_1 the only enabled transition is t_1; firing of t_1 removes the token from p_1 and puts a token both in p_2 and p_3 producing the new

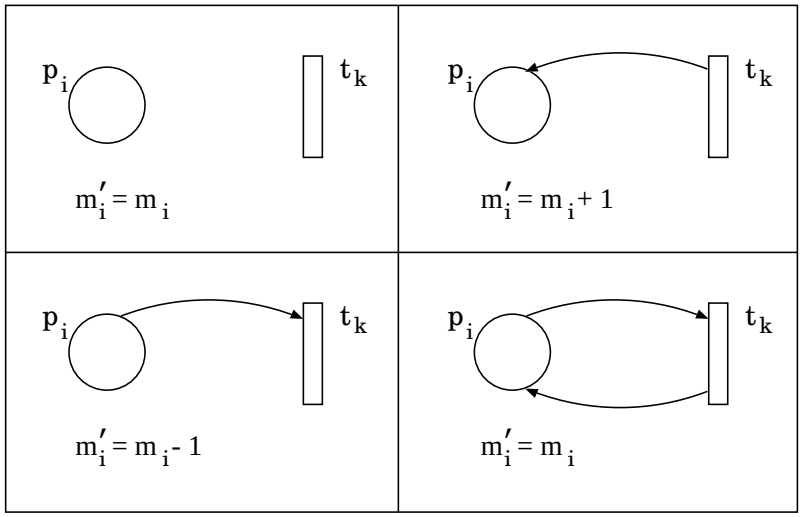

Figure 2: Token number modification in place p_i subsequent to the firing of transition t_k.

marking $M_2 = (0, 1, 1, 0, 0)$. In M_2 the transitions t_2 and t_3 are both enabled and can fire concurrently. Firing of t_3 leads to $M_3 = (0, 1, 0, 0, 1)$ and subsequent firing of t_2 leads to $M_4 = (0, 0, 0, 1, 1)$. In M_4 transitions t_4 and t_5 are both enabled, but the firing of either disables the other; the two transitions are in conflict. Firing of t_4 in M_4 produces marking M_3, while firing t_5 in M_4 produces the intial marking M_1. Note that a different firing sequence can be activated from marking M_2, by letting t_2 firing first and obtaining marking $M_5 = (0, 0, 1, 1, 0)$. With this, all the possible firing sequences have been examined, and the reachability set $\mathcal{R}(M_1)$ of the net of Figure 1 turns out to contain 5 elements M_1, M_2, M_3, M_4 and M_5.

4. Petri Nets and the Modelling of Systems

PN used for modelling real systems are sometimes referred to as *Condition/Events* nets. Places identify the conditions of the parts of the system (working, idle, queueing, failed), and transitions describe the passage from one condition to another (end of a task, failure, repair ...). An event occurs (a transition fires) when all the conditions are satisfied (input places are marked) and give concession to the event. Occurrence of the event modifies in whole or in part the status of the conditions (marking). The number of tokens in a place can be used to identify the number of resources lying in the condition denoted by that place. The following examples illustrate typical situations of interaction of activities arising in system modelling.

4.1. CONCURRENCY (OR PARALLELISM)

In the PN of Figure 3 transitions t_1 and t_2 are enabled simultaneously; the firing of one of them does not modify the state of the other. The activities modelled by the two transitions run concurrently. In reliability modelling, the PN of Figure 3 can represent two components $\mathbf{C}_1$ and $\mathbf{C}_2$ in parallel redundancy; in this case, places p_1 and p_3 represent the working

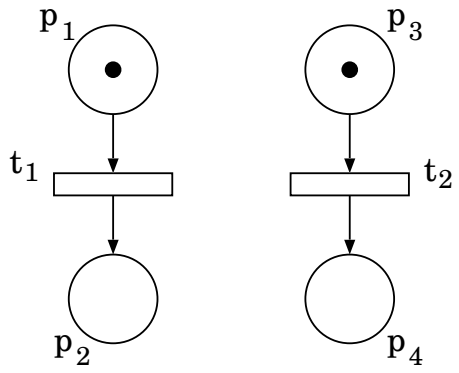

Figure 3: PN modelling two parallel activities.

condition, p_2 and p_4 the failed condition and t_1 and t_2 the event of failure of $\mathbf{C}_1$ and $\mathbf{C}_2$ respectively.

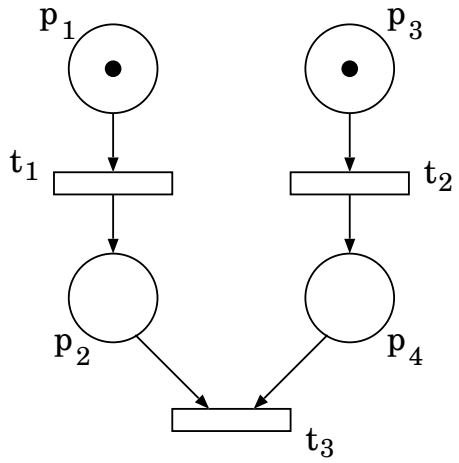

Figure 4: PN modelling two parallel activities with synchronization.

4.2. SYNCHRONIZATION

In Figure 3 the activities modelled by t_1 and t_2 run concurrently; however, if they represent routines of a parallel program, both should be terminated before the program execution can proceed. The synchronization activity is modelled in Figure 4 by means of transition t_3 whose firing requires a token both in p_2 and p_4.

4.3. LIMITED RESOURCES

A typical factor influencing the performance of distributed systems (multiprocessor sys-

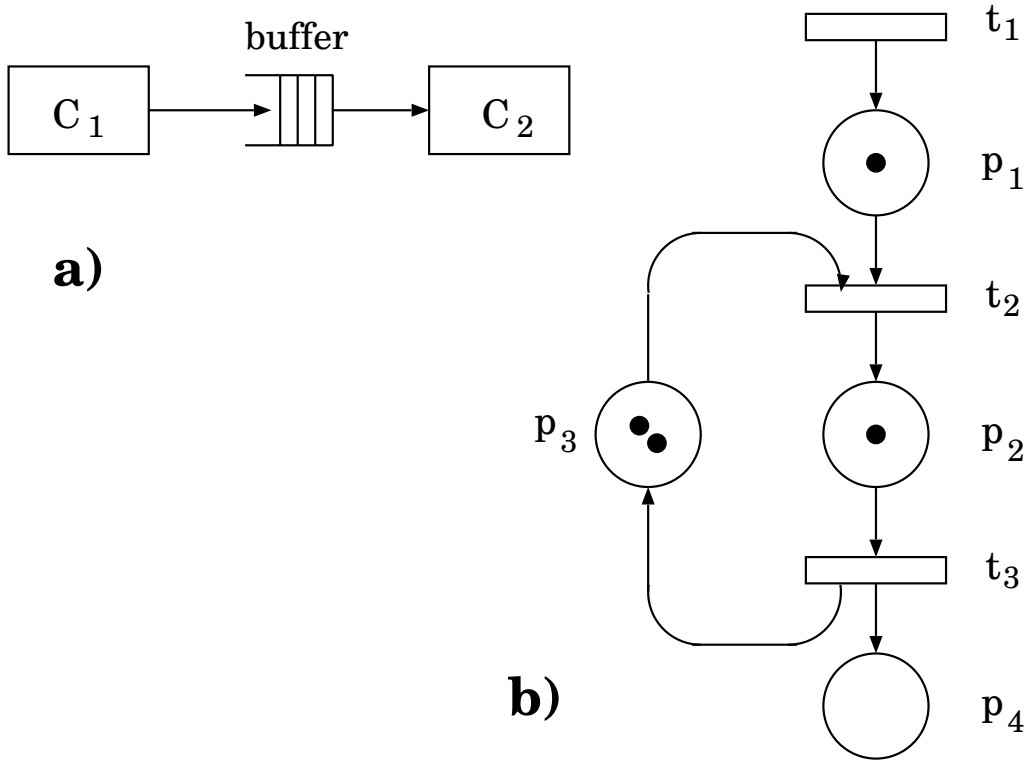

Figure 5: Block diagram and PN of a buffer with finite size.

tems, flexible manufacturing systems and so on) is the limited number of available resources. Exhaustion of the resources prevents the activities to proceed and blocks the system. Modelling and analysing systems with blocking is a difficult task in almost all modelling frameworks [15, 46]. A PN representation of a buffer with limited size is shown in Figure 5b (Figure 5a shows the corresponding block diagram representation). Place p_3 models the number of free buffer positions whereas p_2 the number of filled positions; note that the sum of tokens in p_2 and p_3 is constant and models the total number of available buffer positions (three positions in the figure). Transition t_2 models the filling of one buffer position and can fire if a position free (at least one token in p_3) exists and a task is available to be stored (a token in p_1). Transition t_3 is enabled when at least one buffer position is filled, and firing of t_3 moves one token from p_2 to p_3.

4.4. SEQUENTIALITY (THE PRODUCER/CONSUMER PROBLEM)

A producer produces objects that are put into a buffer from which can be removed and consumed by a consumer. The consuming process must be in sequence with respect to the production process. The PN solution to this problem is reported in Figure 6. A token in p_1 means that the producer is ready to produce. By firing t_1 and t_2 an object is produced (a token is put in the buffer p_5) and the producer is ready again. If the consumer is ready to consume (token in p_3) and an object is in the buffer, transition t_3 can fire removing one token from p_5.

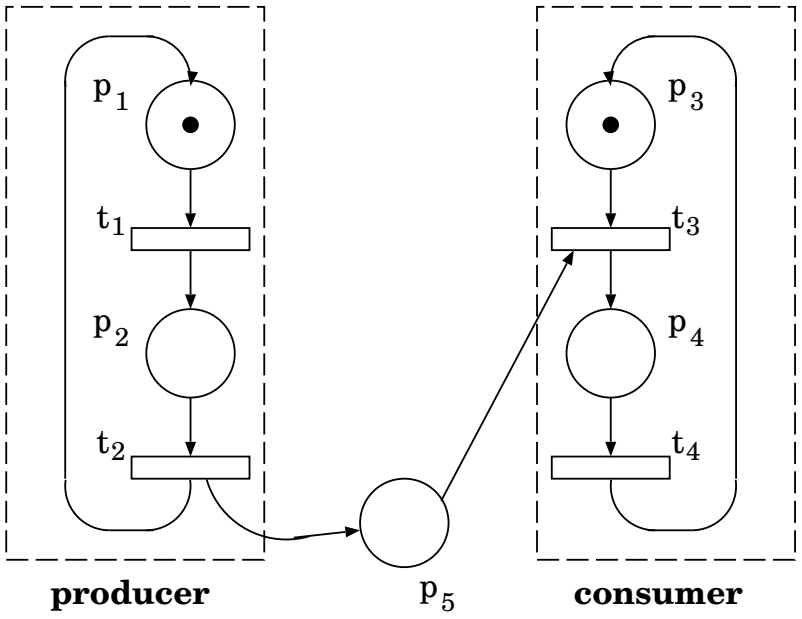

Figure 6: The producer/consumer problem with unbounded buffer.

In the PN of Figure 6 the production and the accumulation of objects in the buffer is unbounded. A more realistic situation is obtained by considering a buffer of limited capacity (as in 4.3). The corresponding PN is reported in Figure 7. Place p_6 models the free buffer positions and place p_5 the filled buffer positions; the number of tokens in p_5 and p_6 is constant and represents the total available buffer positions. If a single token is assigned to p_6 in the initial marking, we model the situation in which the producer cannot further produce until the consumer has consumed the object in the buffer (a strictly sequential ordering of activities).

4.5. MUTUAL EXCLUSION (CONFLICT)

Two resources $\mathbf{C}_1$ and $\mathbf{C}_2$ are allowed to work in parallel, but are connected to a shared resource $\mathbf{C}_s$ that cannot be accessed by $\mathbf{C}_1$ and $\mathbf{C}_2$ simultaneously (block diagram in Figure 8a). The corresponding PN is in Figure 8b. Places p_1 and p_5 represent $\mathbf{C}_1$ and $\mathbf{C}_2$ working independently; p_2 and p_6 represent $\mathbf{C}_1$ and $\mathbf{C}_2$ requesting access to $\mathbf{C}_s$; p_3 and p_7 represent $\mathbf{C}_s$ busy with $\mathbf{C}_1$ and $\mathbf{C}_2$ respectively. Place p_4 determines which resource can actually access $\mathbf{C}_s$, and prevents places p_3 and p_7 to be marked at the same time; in fact when p_2 and p_6 are both marked, transitions t_2 and t_5 are in conflict. Firing of one of them disables the other. Firing of t_3 or t_6 models the release of the common resource (token back in p_4) and the return to the working condition.

5. Properties of Petri Nets

We enumerate different properties which allow us to classify the primitive elements of a PN or the PN as a whole.

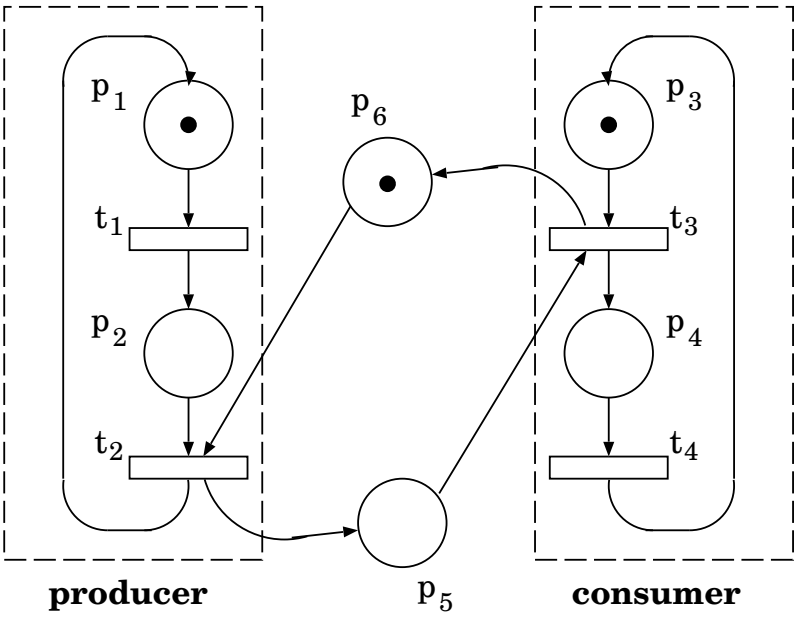

Figure 7: The producer/consumer problem with finite buffer.

5.1. LIVENESS

A transition is *potentially firable* in M if there exists a sequence of transition firings which leads to a marking in which the transition is enabled. A transition is *live* if it is potentially firable in any marking of $\mathcal{R}(M_1)$. A transition is *dead* in M if it is not potentially firable; if the PN enters marking M the dead transition cannot fire any more.

5.2. SAFENESS

A place is *safe* if the token count does not exceed 1 in any marking of $\mathcal{R}(M_1)$. A PN is safe if each place is safe. The PNs of Figures 1, 3 and 8b are safe.

5.3. BOUNDEDNESS

A simple generalization of safeness is the concept of boundedness. A place is bounded with bound k, if the token count does not exceed k in any marking of $\mathcal{R}(M_1)$. A PN is k-bounded if each place is k-bounded. The PN of Figure 7 is k-bounded where k is the number of buffer positions. On the contrary, the PN of Figure 6 is unbounded.

5.4. CONSERVATION

A PN is strictly conservative if the total number of tokens is constant in each marking of $\mathcal{R}(M_1)$. The PN of Figure 7 is k-bounded and strictly conservative, while the PN of Figure 8b is safe but not strictly conservative. A subset of places form a *place-invariant* [31] if it is strictly conservative. In the net of Figure 8b the subsets $\{p_1, p_2, p_3\}$, $\{p_5, p_6, p_7\}$ and $\{p_3, p_4, p_7\}$ are place-invariants.

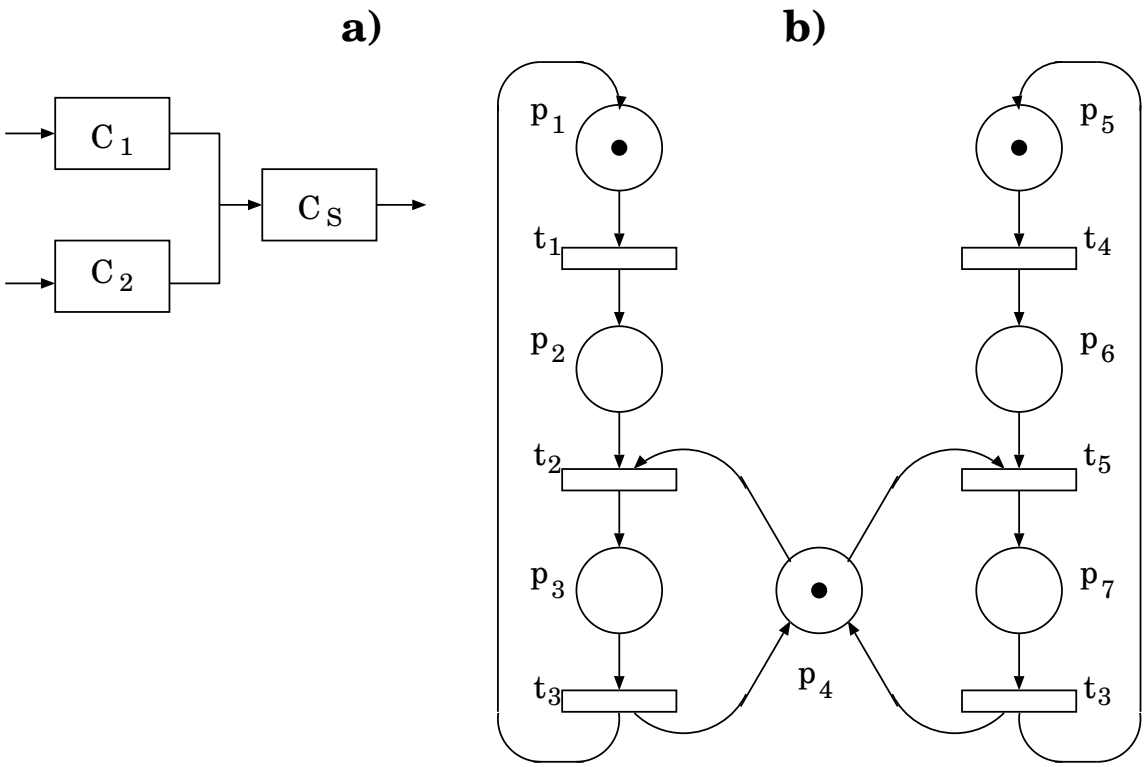

Figure 8: The mutual exclusion problem: two parallel tasks with a common resource.

6. Analysis Techniques

The success of any model depends on two factors: its modelling power and its decision power. Modelling power refers to the ability to correctly represent the system to be modelled; decision power refers to the ability to analyze the model and determine properties of the modelled system. The modelling power of PN has been examined in the previous sections, and in this section we take into consideration the analysis techniques of PNs.

6.1. THE REACHABILITY TREE AND REACHABILITY GRAPH

The reachability set $\mathcal{R}(M_1)$ of a PN is generated by means of the reachability tree. The initial marking M_1 is the root of the reachability tree. Starting from the root we search for all the enabled transitions; the firing of an enabled transition produces a new marking which is represented as a new leaf in the tree, from which the procedure is iterated.

By properly identifying the frontier nodes of the tree, the generation of the reachability tree involves a finite number of steps [38], even if the PN is unbounded. Let us introduce three kinds of frontier nodes:

- terminal (dead) nodes: nodes in which no transitions are enabled;
- duplicate nodes: nodes which have been already generated in the tree;

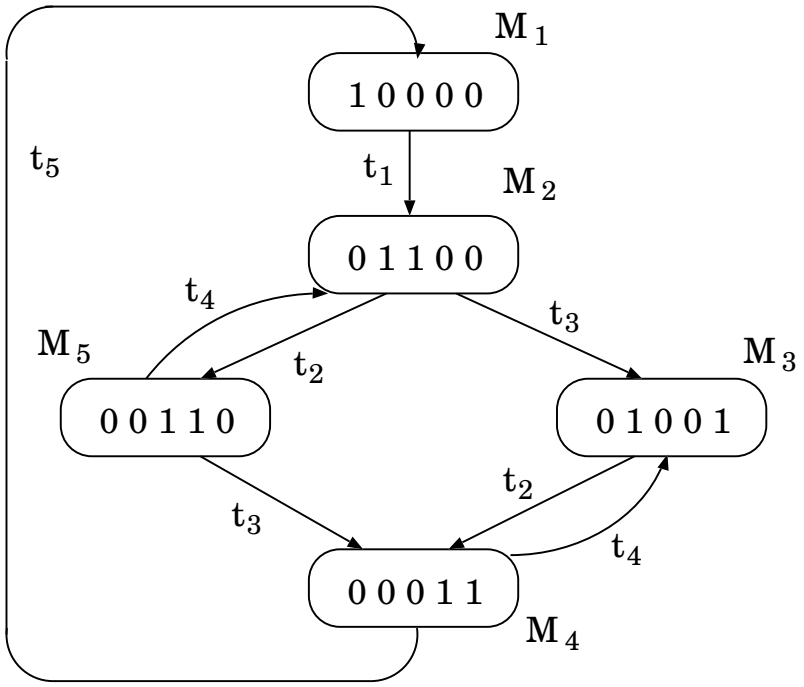

Figure 9: Reachability graph $\mathcal{G}_R(M_1)$ for the net of Figure 1.

- infinitely reproducible nodes. A marking M'' is an infinitely reproducible node if $M'' \geq M'$ ($m''_i \geq m'_i$, $i = 1, 2, \ldots, n_p$) for some M' already generated in the tree. Because of the stated relation, the transition sequence from which M'' has been generated starting from M' is surely firable in M''. Thus, the sequence $M' \rightarrow M''$ can be reproduced infinitely often, so that the token count in the places for which $m''_i \geq m'_i$ can increase indefinitely. We represent the arbitrarily large number of tokens which results from infinitely reproducible nodes by defining a special symbol ω with the following properties:

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

  for any positive constant a.

By allowing ω to be a legal symbol in the reachability tree specification, it can be shown that the generation of the reachability tree involves always a finite search algorithm [38]. If the generation of the reachability tree terminates without arriving to infinitely reproducible nodes, the PN is bounded. In this case the reachability set is finite and can be represented as a labelled directed graph whose vertices are the elements of $\mathcal{R}(M_1)$ and such that for each possible transition firing $M_i - t_k \rightarrow M_j$ there exists an arc (i, j) labelled k. The reachability graph associated to a reachability set $\mathcal{R}(M_1)$ will be denoted by $\mathcal{G}_R(M_1)$.

Figure 9 shows the reachability graph of the PN of Figure 1 with initial marking $M_1 = (1, 0, 0, 0, 0)$, as discussed in Section 3. Figure 10 shows the reachability graph for the mutual exclusion problem of Figure 8, with initial marking $M_1 = (1, 0, 0, 1, 1, 0, 0)$.

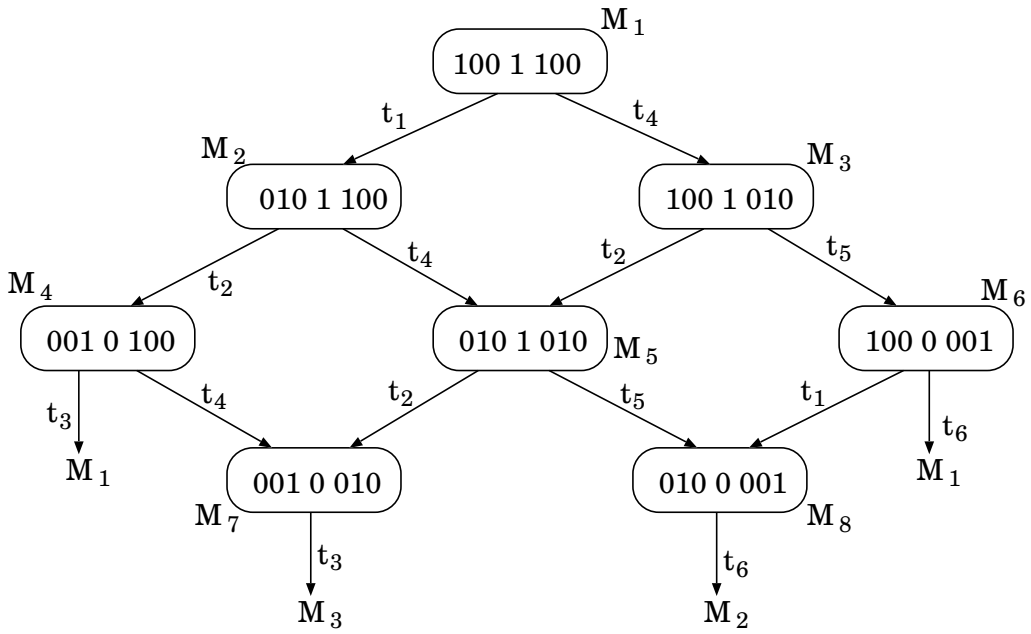

Figure 10: Reachability graph $\mathcal{G}_R(M_1)$ for the net of Figure 8.

In Figure 11 we have reported the reachability tree of the PN of Figure 6; since the net is unbounded, in order to keep the generation algorithm finite, the symbol ω has been introduced.

If a PN has a finite $\mathcal{R}(M_1)$ all the properties of the net (safeness, liveness, etc..) can be analyzed by inspection of the reachability graph. If the net is unbounded the finite reachability tree representation, by means of the symbol ω, can be an imperfect description of the net (it is possible to find PNs with different properties and behaviours that cannot be distinguished through the reachability tree, due to incomplete information carried by ω [38]).

6.2. MATRIX ANALYSIS

The input and output functions of a PN can be equivalently defined using a matrix notation. Let D^- denotes the input matrix. D^- is a $(n_t \times n_p)$ matrix, whose generic element d_{ij}^- is equal to the number of arcs connecting place p_j with transition t_i. Similarly we define the output matrix D^+ as a $(n_t \times n_p)$ matrix, whose generic element d_{ij}^+ is equal to the number of arcs connecting transition t_i with place p_j. The *incidence matrix* D is defined by the following relation:

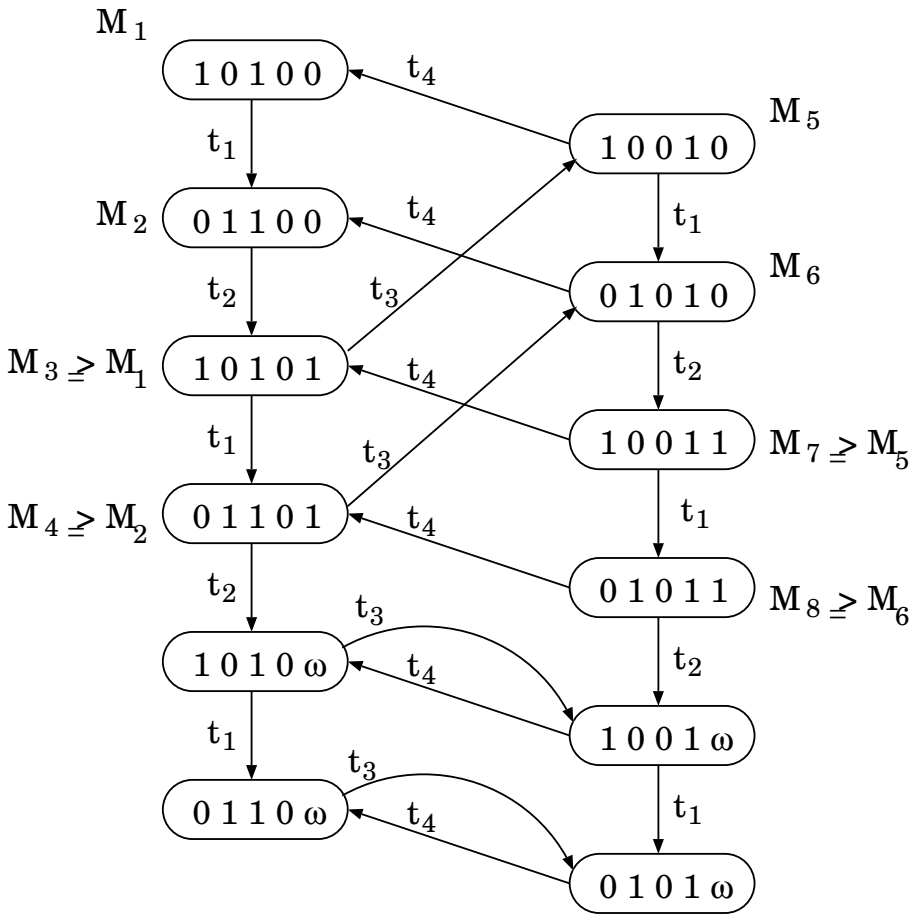

Figure 11: Generation of the reachability tree for the PN of Figure 6 with unbounded buffer.

$$D = D^+ - D^- \tag{1}$$

The matrices D^-, D^+ and D for the PN of Figure 8b are reported in the following:

$$D^- = \begin{array}{c|ccccccc} & p_1 & p_2 & p_3 & p_4 & p_5 & p_6 & p_7 \\ \hline t_1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ t_2 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ t_3 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ t_4 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ t_5 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ t_6 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}$$

$$D^+ = \begin{array}{c|ccccccc} & p_1 & p_2 & p_3 & p_4 & p_5 & p_6 & p_7 \\ \hline t_1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ t_2 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ t_3 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ t_4 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ t_5 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ t_6 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \end{array}$$

$$D = \begin{array}{c|ccccccc} & p_1 & p_2 & p_3 & p_4 & p_5 & p_6 & p_7 \\ \hline t_1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 \\ t_2 & 0 & -1 & 1 & -1 & 0 & 0 & 0 \\ t_3 & 1 & 0 & -1 & 1 & 0 & 0 & 0 \\ t_4 & 0 & 0 & 0 & 0 & -1 & 1 & 0 \\ t_5 & 0 & 0 & 0 & -1 & 0 & -1 & 1 \\ t_6 & 0 & 0 & 0 & 1 & 1 & 0 & -1 \end{array} \quad (2)$$

Let us further introduce the vector $\underline{e}_j$ which is a n_t-dimensional row vector with all the entries equal to 0 except entry j equal to 1. With this notation the execution rules of a PN becomes:

- a transition t_j is enabled in marking M iff $M \geq \underline{e}_j\, D^-$ (note that $\underline{e}_j\, D^-$ is the $j-th$ row of D^-);
- firing of t_j in M produces a marking M' given by:

$$M' = M - \underline{e}_j\, D^- + \underline{e}_j\, D^+ = M + \underline{e}_j\, D \quad (3)$$

From the previous definitions follows that, given a PN with initial marking M_1 and a firing sequence $t_i \rightarrow t_j \rightarrow t_k \rightarrow t_j \rightarrow t_i$, the marking obtained at the end of the sequence is given by the following matrix equation:

$$M_{fin} = M_1 + (\underline{e}_i + \underline{e}_j + \underline{e}_k + \underline{e}_j + \underline{e}_i\,)\, D \quad (4)$$

By means of the matrix representation, the following properties of PN can be inspected.

6.2.1. Reachability - A marking M' is reachable from M if an integer vector $\underline{X}$ exists such that (see equation 3):

$$M' = M + \underline{X}\, D \quad (5)$$

Equation (5) provides a necessary but not sufficient condition; all markings reachable from M are solution of equation (5) but not viceversa; for any integer vector $\underline{X}$ a solution to

equation (5) exists, but the transition firing sequence represented by $\underline{X}$ can be non-firable. Furthermore, note that the solution of (5) is not affected by the order of transition firings (but only by the number), while the semantics of the net is strongly affected by the order: changing the order a legal sequence can become non-firable.

6.2.2 Conservation - Given a conservative PN, and a n_p-dimensional column vector $\underline{U}_p^T$ with all the entries equal to one, for any marking $M' \in \mathcal{R}(M_1)$ the following relation should hold:

$$M_1 \, \underline{U}_p^T \; = \; M' \, \underline{U}_p^T \tag{6}$$

Thus, from equation (5):

$$M_1 \, \underline{U}_p^T \; = \; M_1 \, \underline{U}_p^T \; + \; \underline{X} \, D \, \underline{U}_p^T \tag{7}$$

since (5) must be satisfied for any $\underline{X}$, it follows:

$$D \, \underline{U}_p^T \; = \; 0 \tag{8}$$

Equation (8) is a necessary and sufficient condition for conservation.

6.2.3. Place Invariant - Let $\underline{W}_p$ be a vector of binary entries (either 0 or 1); we find all vectors $\underline{W}_p$ for which [31]:

$$D \, \underline{W}_p^T \; = \; 0 \tag{9}$$

the places p_i $(i = 1, 2, \ldots, n_p)$ for which $w_i = 1$, form a *place invariant* (a conservative subset of places). With reference to the incidence matrix D of Equation (2), it is easily verified that the following vectors are solution of Equation (9):

$$\begin{aligned} \underline{W}_p^{(1)} &= [\, 1\,1\,1\,0\,0\,0\,0 \,] \\ \underline{W}_p^{(2)} &= [\, 0\,0\,0\,0\,1\,1\,1 \,] \\ \underline{W}_p^{(3)} &= [\, 0\,0\,1\,1\,0\,0\,1 \,] \end{aligned}$$

and therefore, the subsets $\{p_1, p_2, p_3\}$ $\{p_5, p_6, p_7\}$ and $\{p_3, p_4, p_7\}$ are place invariant for the PN of Figure 8b.

7. Extensions

In the use of PN for modelling real systems several authors have found convenient to introduce special constructs either for making the model representation more compact in a given application or for extending the modelling power of the PN formalism. The extensions more often encountered in the literature (and that will be used in the sequel), have been proposed in response to difficulties in modelling priority disciplines by PN. All the extensions mentioned in the sequel are equivalent from the point of view of the modelling power, thus their use depends on the easiness or convenience of the implementation [16].

7.1. INHIBITOR ARCS

An inhibitor arc from place p_j to transition t_k modifies the enabling rules in the sense that the transition can fires only if place p_j does not contain tokens. The inhibition function is usually represented by circle-headed arcs, as in Figure 12 where transition t_k can fire iff p_i contains at least one tokens, but no tokens are present in p_j.

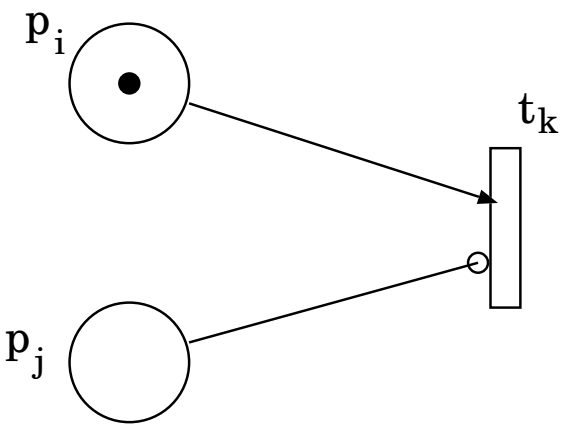

Figure 12: Inhibitor arc.

In the mutual exclusion problem of Figure 8, the standard PN language does not provide any means to establish precedence rules in the case both resources $\mathbf{C}_1$ and $\mathbf{C}_2$ are simultaneoulsy requesting access to the common resource $\mathbf{C}_s$ (places p_2 and p_5 simultaneously marked). With the insertion of an inhibitor arc from place p_2 to transition t_5 (Figure 13), we model the situation in which, as a conflict arises between $\mathbf{C}_1$ and $\mathbf{C}_2$, $\mathbf{C}_1$ has always the precedence, and blocks (inhibits) $\mathbf{C}_2$ until the common resource is released.

With respect to the reachability graph $\mathcal{G}_R(M_1)$ of the original PN of Figure 8 (reported in Figure 10), the reachability graph of the modified PN of Figure 13 is such that from marking M_5 only transition t_2 can fire while t_5 is inhibited.

7.2. PRIORITY LEVELS

An alternative, but equivalent way to model the same features considered with the introduction of inhibitor arcs, is obtained by attaching to each PN transition a priority level. 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. In Figure 13 exactly the same precedence policy can be modelled by attaching to transition t_2 a priority level greater than the one attached to t_5. In marking M_5 (see Figure 10) in which both transitions are enabled, only t_2 can fire.

7.3. CONDITIONING FUNCTIONS

More complex logical interactions between primitive elements of a PN can be considered by introducing logical *conditioning functions* [20]. Given a marking M, a PN transition is enabled if, beside the normal enabling requirements (including inhibitor arcs and priorities),

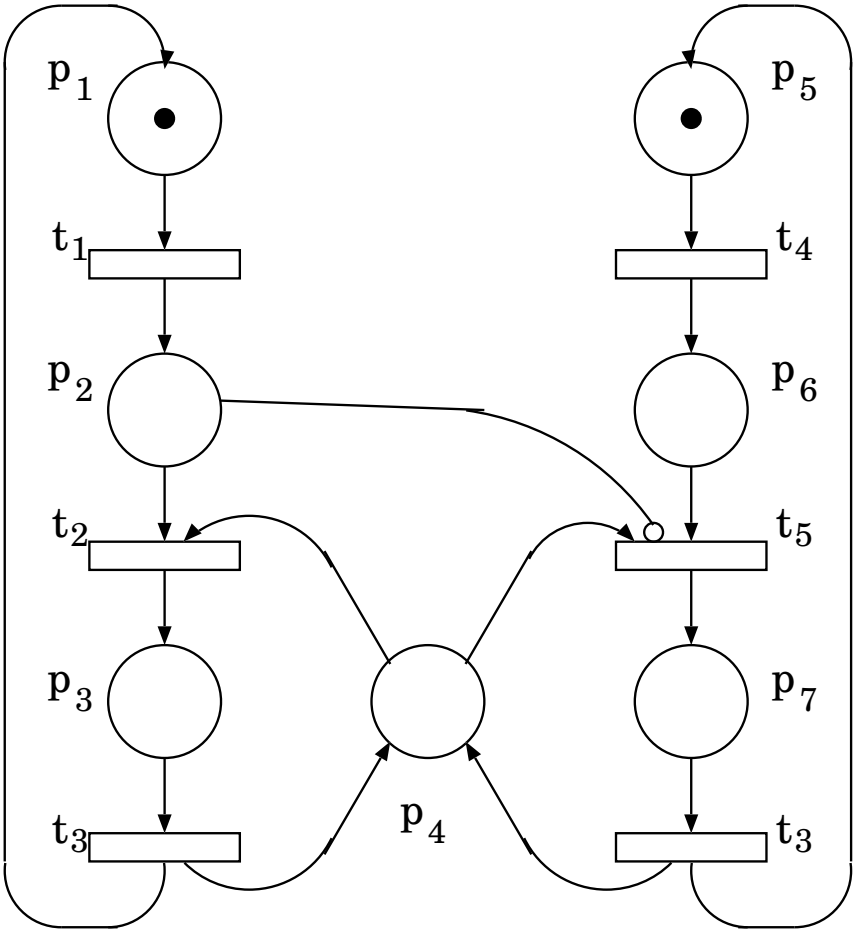

Figure 13: The mutual exclusion problem of Figure 8, with assigned priority.

the conditioning function is *true*. The conditioning functions can be very effective in reducing the graphical complexity of a PN, even if they do not extend the modelling power with respect to inhibitor arcs or priority levels.

7.4. HIGH LEVEL PETRI NETS

In the PN models discussed so far, the individual tokens are indistiguishable. The semantics of the model does not allow to follow the behaviour of an individual token through the net. To overcome this limitation a new class of models has been proposed and discussed. The common characteristic of these models, usually referred to as *high level* PN, is that the position of any single token can be tracked in the PN. Two labelling techniques have been originally proposed: the technique of colouring tokens (coloured PN introduced by Jensen [28]) and the technique of assigning to each token a predicate (*Predicate/Transition* net introduced by Genrich and Lautenbach [24]). However, this class is not further dealt with in the present notes.

8. Timed Petri Nets

An *execution sequence* $\mathcal{E}$ in a marked PN, is a sequence of legal markings obtained by firing a sequence of enabled transitions:

$$\mathcal{E} = \{ (M_{(1)}, t_{(1)}) ; (M_{(2)}, t_{(2)}) ; \ldots ; (M_{(j)}, t_{(j)}) ; \ldots \}$$

An execution sequence $\mathcal{E}$ can be viewed as a connected path in the reachability graph $\mathcal{G}_R(M_1)$ of the net.

A timed execution sequence $\mathcal{T}_E$ of a marked PN with intial marking $M_{(1)}$, is an execution sequence $\mathcal{E}$ augmented by a non-decreasing sequence of real values representing the epochs of firing of each transition, such that consecutive transitions $(t_{(j)} ; t_{(j+1)})$ in $\mathcal{E}$ correspond to ordered epochs $\tau_j \leq \tau_{j+1}$ in $\mathcal{T}_E$. Thus formally [23, 4]:

$$\mathcal{T}_E = \{ (M_{(1)}, t_{(1)}, \tau_1) ; (M_{(2)}, t_{(2)}, \tau_2) ; \ldots ; (M_{(j)}, t_{(j)}, \tau_j) ; \ldots \}$$

The time interval $\tau_j - \tau_{j+1}$ between consecutive epochs represents the period that the PN sojourns in marking $M_{(j)}$. In the sequel we always assume as initial epoch $\tau_1 = 0$.

> **Definition** - *A Timed PN (TPN) is a marked PN in which a set of specifications are provided and a set of rules are defined such that to each legal execution sequence $\mathcal{E}$ a timed execution sequence $\mathcal{T}_E$ can be univocally associated.*

A variety of timing mechanisms have been proposed in the literature. The distinguishing features of the timing mechanisms are whether the duration of the events is modelled by deterministic variables or random variables, and whether the time is associated to the PN places, transitions or tokens.

Earlier work in timed PN with deterministic timing can be found in [32, 43, 40, 47]. Application of deterministic-PN models are available in different areas, like: communication protocols, performance evaluation, manufacturing. However, in the reliability area stochastic modelling is more appropriate, and therefore we will consider in the sequel only TPN in which the timing mechanism is stochastic; we will refer to this class of models as Stochastic PN (SPN).

SPN were initially proposed in two doctoral thesis [36, 35]. In these models, interpreting PN as Condition/Event nets, time was naturally associated with activities that induce state changes, hence with the delay incurred before firing transitions. Although other possibilities have been explored, the choice of associating time with PN transitions is the most common in the literature, and is the only considered in the present notes.

When the random variables associated to PN transitions are exponentially distributed, the dynamic behaviour of the PN can be mapped into a time-continuous homogeneous Markov chain with state space isomorphic to the reachability graph of the PN. This case will be considered in details in the following sections.

Extensions to cover the case of generally distributed transition firing times have been considered in a number of papers [36, 21, 4, 23, 26, 5]. Releasing the memoryless property of the exponential distribution, in order to univocally associate to each execution sequence $\mathcal{E}$ a timed execution sequence $\mathcal{T}_E$ the concept of SPN *execution policy* needs to be introduced

[4, 5]. The execution policy consists of two parts: the way in which a transition is selected to fire among those enabled in a given marking, and the way in which the time spent is recovered after a transition firing. However, due to the complexity of the semantics of the SPN models, and of the associated stochastic process (both aspects are strictly dependent on the definition of the execution policy), this generalization is no further considered in this paper.

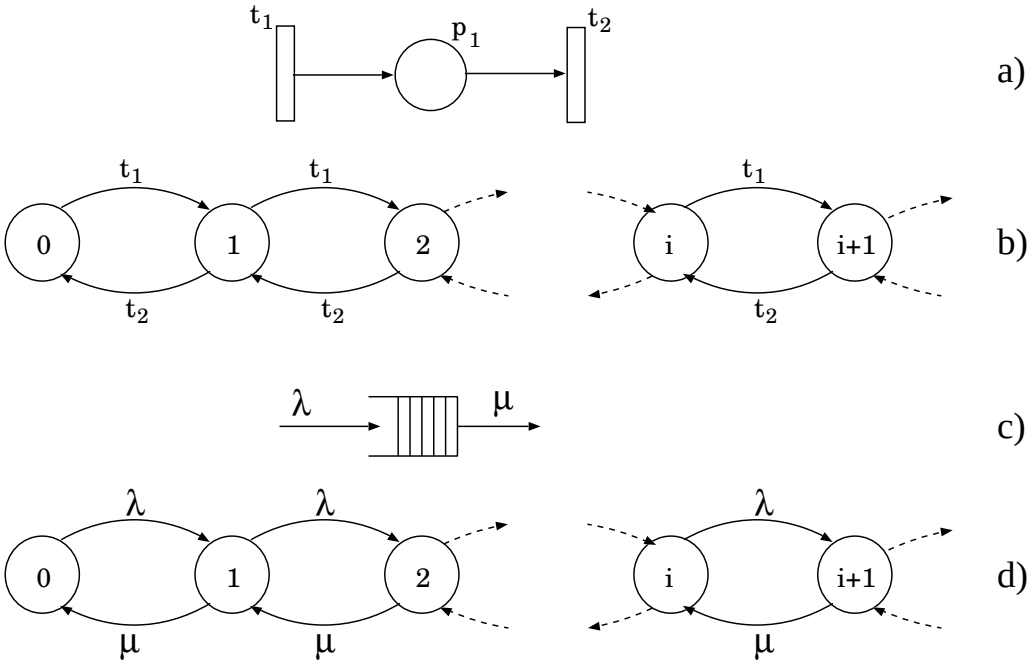

Figure 14: The $M/M/1$ queue: a) The PN representation; b) The reachability graph; c) The block diagram representation; d) The corresponding Markov transition graph.

9. Homogeneous Markov SPN (HMSPN)

Let us suppose that the activity modelled by a PN transition takes an exponentially distributed random amount of time to complete once initiated. This means that an exponentially distributed random variable θ_j with parameter $\lambda_j(M)$ is associated to each PN transition t_j. The firing of an enabled transition t_j in marking M becomes a random event which occurs with a time-independent (but possibly marking dependent) firing rate $\lambda_j(M)$. Therefore, knowing the transitions enabled in a given marking and the associated firing rates, we can univocally generate the stochastically timed sequence $\mathcal{T}_E$ from each execution sequence $\mathcal{E}$. In other words, the reachability graph $\mathcal{G}_R(M_1)$ of a marked PN can be univocally mapped into a discrete-state continuous time homogeneous Markov chain, by letting each marking of $\mathcal{G}_R(M_1)$ correspond to a state in the Markov chain, and by substituting the label of the PN transition in each edge of $\mathcal{G}_R(M_1)$ with the firing rate of the corresponding transition. With this definition we can speak indifferently of *marking*

M_i or *state i*.

Example 1 - The M/M/1 queue. Consider the PN of Figure 14a) where it is intended that a transition with no input places is always enabled. The corresponding reachability graph is reported in Figure 14b). The label inside each state is the marking, i.e. the number of tokens in place p_1. Firing of t_1 increases the token count by 1, while firing of t_2 decreases the token count by 1. By associating to transition t_1 the arrival rate λ and to t_2 the service rate μ, the PN of Figure 14a) models the *M/M/1* [29] queueing system. The usual block diagram representation is given in Figure 14c) and the corresponding Markov transition graph is given in Figure 14d). This example is also intended to show how the PN language is suitable to represent queueing systems or queueing networks.

Example 2 - Let the PN of Figure 3 denote the failure process of two components in parallel redundancy; t_1 is the event of failure of component 1 to which a failure rate λ_1 is assigned. Similarly we assigne to t_2 the failure rate λ_2 of component 2. Figure 15a) shows the reachability graph of the net and Figure 15b) the associated Markov chain representing the dynamic behaviour of the net in time.

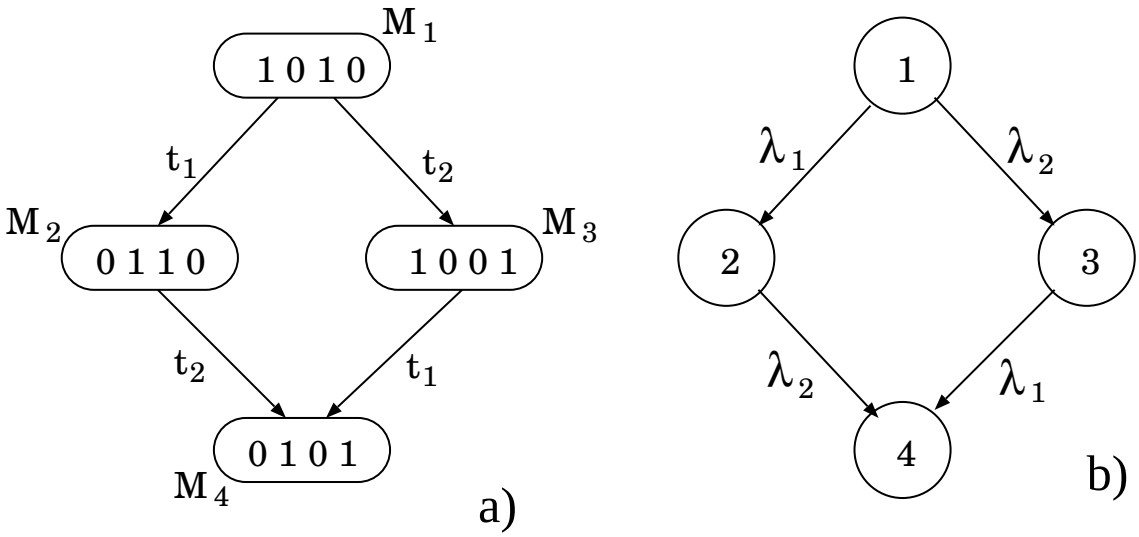

Figure 15: The reachability graph a) and the corresponding Markov chain b) of the SPN of Figure 3.

The probability of the original PN of being in marking M_4 at time t where both components are failed can be computed as the probability of being in state 4 at time t in the corresponding Markov chain.

Example 3 - The reachability graph of the PN of Figure 8 is reported in Figure 10. If all the PN transitions are assigned time-independent firing rates, the reachability graph of Figure 10 is mapped into the Markov chain of Figure 16.

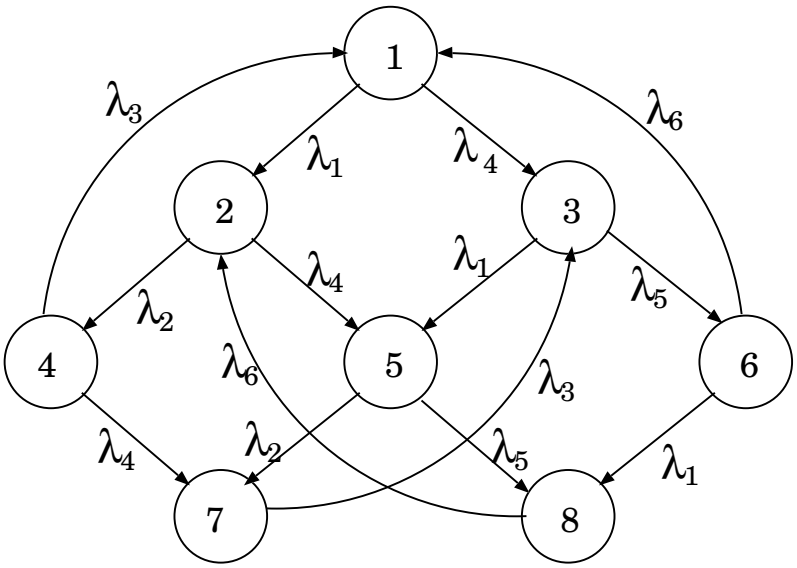

Figure 16: Markov chain corresponding to the reachability graph of Figure 10.

9.1. FORMAL DEFINITION OF THE MODEL

The Homogenous Markov SPN (HMSPN) is a six-tuple:

$$HMSPN = (P, T, I, O, M, L)$$

where P, T, I, O, M have the same meaning introduced in Section 3, and $L = \{\lambda_1(M), \lambda_2(M), \ldots, \lambda_{nt}(M)\}$ is a set of n_t non-negative real numbers representing the (marking dependent) firing rates of the exponential random variables associated to each PN-transition.

The knowledge of the reachability graph allows us to automatically generate the transition rate matrix Λ of the associated homogeneous Markov chain. Λ is a $N \times N$ matrix, where N is the cardinality of the reachability set $\mathcal{R}(M_1)$.

Let us define $\underline{Q}(t)$ a N-dimensional state probability vector, whose generic entry $q_i(t)$ is the probability of being in state i $(i = 1, 2, ..., N)$ at time t in the associated Markov chain. $\underline{Q}(t)$ is the solution of the standard Markov linear differential equation:

$$\frac{d\,\underline{Q}(t)}{d\,t} \;=\; \Lambda\,\underline{Q}(t) \tag{10}$$

with initial condition $\underline{Q}(0) = [1, 0, 0, ..., 0]^T$. If the steady state probability vector $\underline{Q}(\infty)$ of the Markov chain exists, it can be calculated from the equation:

$$\Lambda\,\underline{Q}(\infty) \;=\; \underline{0} \qquad \text{with} \qquad \sum_{i=1}^{N} q_i(\infty) \;=\; 1 \tag{11}$$

The numerical techniques for the solution of Equations (10) and (11) are outside the scope of the present notes. For a recent survey on methods and techniques for solving equation (10) see [41].

9.2. MARKING DEPENDENT FIRING RATES

In the formal definition of the HMSPN model the firing rates associated to each transition have been considered as marking dependent. This possibility increases the flexibility of the model and is often used to make the models more compact in the case of the presence of multiple identical resources.

Example 4 - The PN of Figure 17a) has the following physical meaning: place p_1 represents operation; place p_2 non operation; transition t_1 failure and transition t_2 repair. Suppose we have K identical components in parallel redundancy each one with failure rate λ. We can model the system operation by the PN of Figure 17a) with initial marking $M_1 = (K, 0)$ and associating to transition t_1 the marking dependent transition rate $\lambda_{t_1}(M_x) = m_{1x}\lambda$, where m_{1x} is the number of tokens in place p_1 in marking M_x.

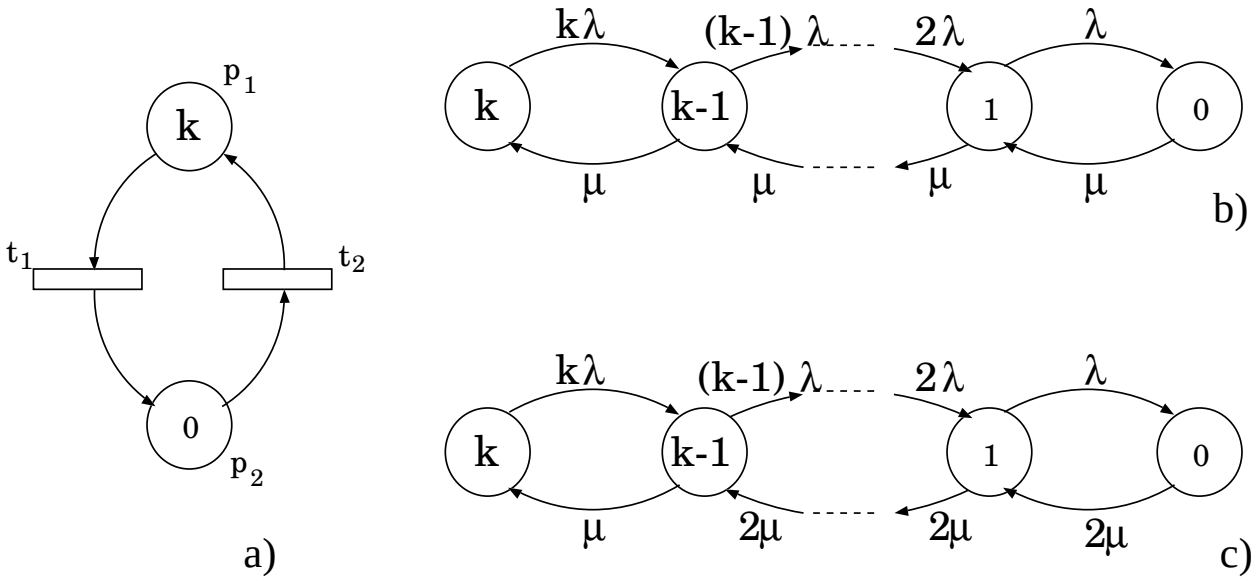

Figure 17: a) PN modelling K identical parallel components; b) The associated Markov chain with a single repairman; c) The associated Markov chain with 2 repairmen.

Moreover, we can easily model various repair policies: the single repairman policy is modelled by assigning to transition t_2 the repair rate μ. In this case, the Markov chain corresponding to the PN of Figure 17a) is reported in Figure 17b). The independent repair policy can be modelled by assigning to transition t_2 the marking dependent firing rate $\mu_{t_2}(M_x) = m_{2x}\mu$ (as many repairmen as failed components m_{2x}). The case of two repairmen can be modelled by means of more complex logical assignment to the firing rate $\mu_{t_2}(M_x)$ of transition t_2:

$$\mu_{t_2}(M_x) = \begin{cases} 2\,\mu & \text{if} \quad m_2 \geq 2 \\ \mu & \text{if} \quad m_2 = 1 \\ 0 & \text{if} \quad m_2 = 0 \end{cases}$$

In this last case the Markov chain generated by the PN of Figure 17a) is reported in Figure 17c).

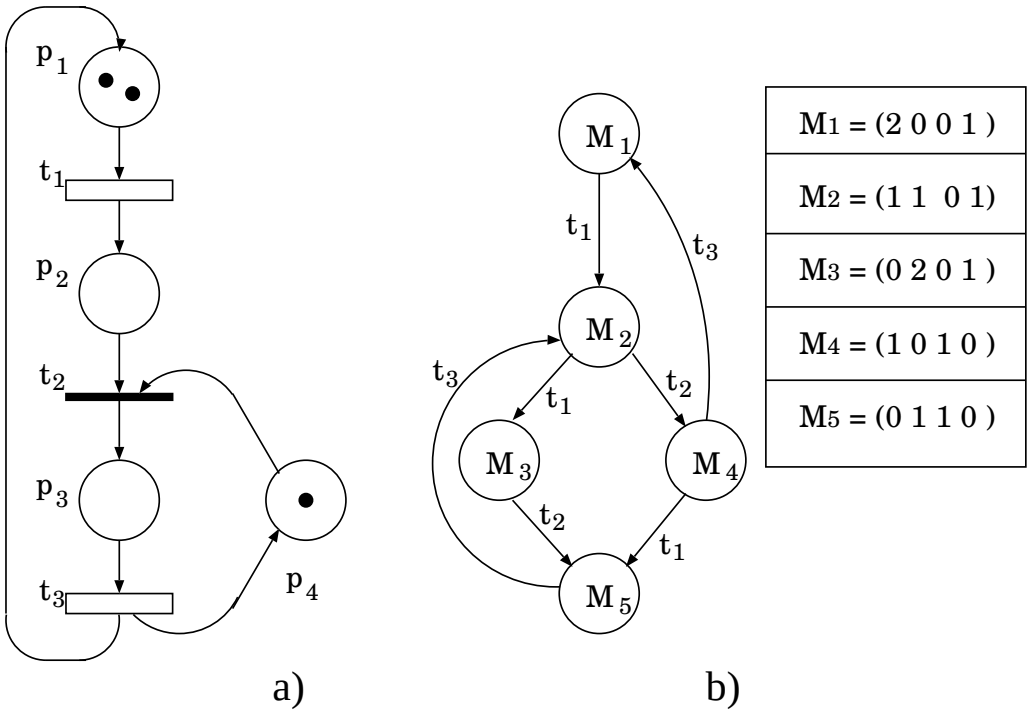

Figure 18: Folded PN modelling two identical resources; b) The associated Markov chain.

Example 5 - In the mutual exclusion problem of Figure 8, if the two resources $\mathbf{C}_1$ and $\mathbf{C}_2$ are identical, we can fold the two simmetric parts of the PN of Figure 8 in the PN of Figure 18a). The stochastic properties of the system are retained by assigning to transition t_1 a firing rate proportional to the number of tokens in p_1. The Markov chain associated to the PN of Figure 18a) is reported in Figure 18b). Note that folding the PN of Figure 18a) corresponds exactly to lumping the Markov chain of Figure 16 into the Markov chain of Figure 18b) (whenever lumpability conditions exist).

9.3. IMMEDIATE AND TIMED PN TRANSITIONS

Many authors [6, 11, 21] have recognized that the use of SPN for modelling real systems involves the presence of very brief or *fast* transitions, whose duration is short, or even negligible, with respect to the time scale of the problem. Different techniques have been proposed to tackle this problem.

The starting assumption in the GSPN model [6] is that it is desirable to associate a random time only to those transitions which are believed to have the largest impact on the system operation. Transitions are partitioned into two different classes: *immediate* transitions and *timed* transitions. Immediate transitions fire in zero time once they are enabled and have higher priority over timed transitions. Timed transitions fire after an exponentially distributed firing time. In the graphical representation of GSPN, immediate transitions are drawn as thin bars while timed transitions are drawn as thick bars.

Markings (states) enabling immediate transitions are passed through in zero time and are called *vanishing* states. Markings enabling only timed transitions are called *tangible*. Since the process spends zero time in the vanishing states, they do not contribute to the time behaviour of the system so that a procedure can be envisaged to eliminate them from the final Markov chain. With the partition of PN-transitions into a timed and an immediate class, we introduce a greater flexibility at the modelling level without increasing the dimensions of the final state space on which the set of equations (10) or (11) have to be computed.

Given a marking $M \in \mathcal{G}_R(M_1)$ of a GSPN, three different situations may arise:

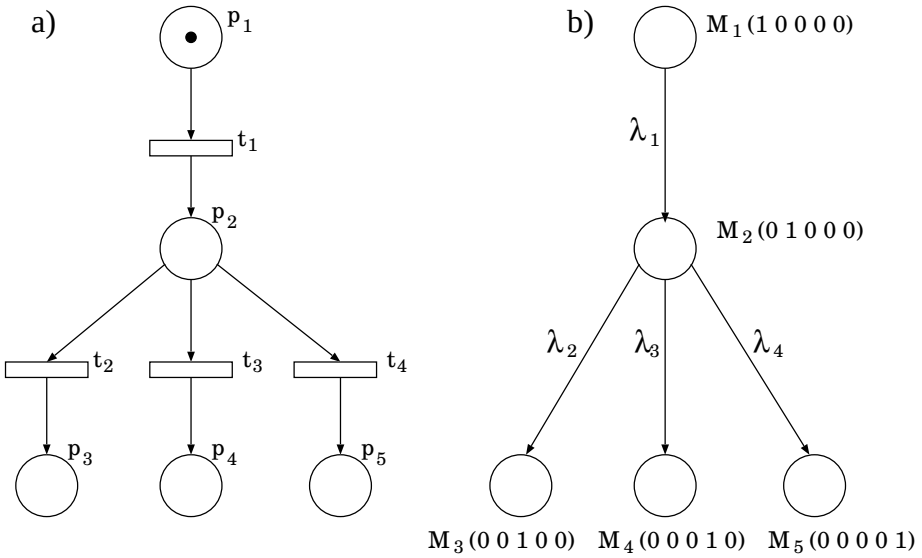

Figure 19: a) SPN with timed transitions only; b) The associated Markov chain.

Situation 1 (Figure 19)
Only timed transitions are enabled (Figure 19a) so that only tangible markings are generated (Figure 19b). The model, in this case, coincides with the HMSPN described in section 9.1.

Situation 2 (Figure 20)
Timed transitions are enabled simultaneously to one immediate transition (Figure 20a). Only the immediate transition is allowed to fire, generating the associate Markov chain of Figure 20b). However, marking M_2 is vanishing and can be eliminated from the chain producing the reduced Markov chain of Figure 20c), in which all the states are tangible.

Situation 3 (Figure 21)
Several immediate transitions are enabled in a marking. In this case in order to define which is the transition that fires first, a probability mass function need to be specified: in the language of GSPN this construct is called a *random switch*, and the probability mass function the *switching distribution*. In Figure 21a) the immediate transition t_2 fires with probability v and the immediate transition t_3 with the complementary probability $1 - v$. The equivalent Markov chain is reported in Figure 21b). State M_2 is vanishing and can be eliminated incorporating the switching distribution into the rates leading to state M_2. Elimination of the vanishing state leads to the Markov chain of Figure 21c), which contains only tangible states.

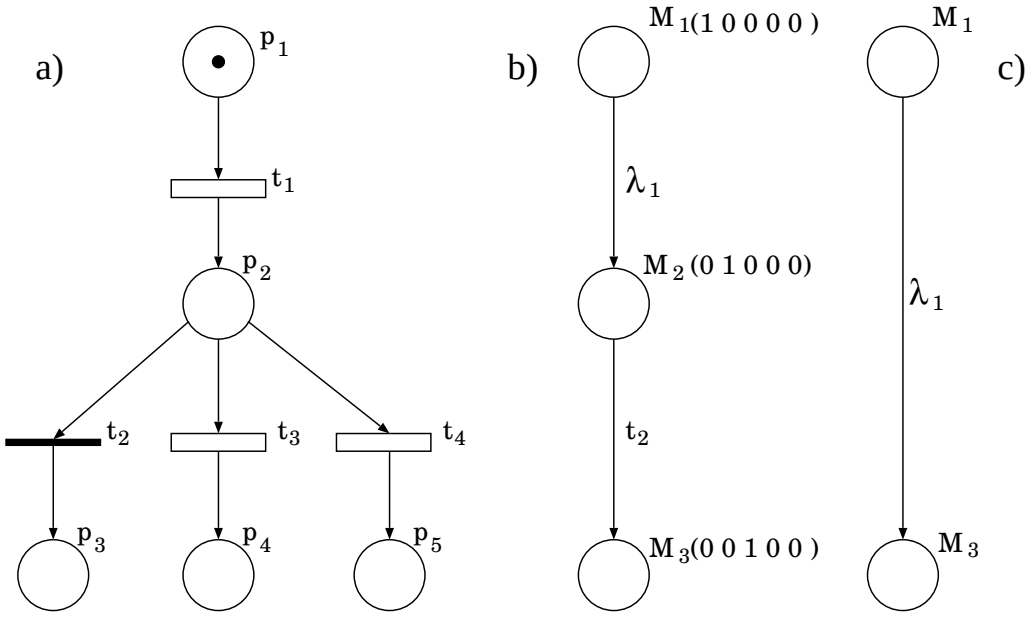

Figure 20: a) SPN with one immediate transition; b) The reachability graph; c) The reduced Markov chain defined over tangible markings.

An automatic algorithm can be implemented [6] which recognizes the three situations previously depicted and progressively eliminates vanishing states until a homogeneous Markov chain, defined over tangible states only, is obtained. In this way the reduction procedure becomes completely transparent to the analyst.

The problem of modelling a probabilistic decision, which does not consume time was also considered in [19], by introducing a different construct called *probabilistic arc*. For a comparison of probabilistic arcs with random swithches see [20].

In GSPN [6] only the steady state behaviour of the associated Markov chain is analysed. If the transient analysis is of interest, the use of immediate transitions does not allow to capture the true dynamics of the PN. In this case it is more appropriate to partition the PN-transition into *fast* transitions and *slow* transitions [11]. In this way the transition rate matrix Λ contains rates of very different orders of magnitude, so that the system of differential equations (10) becomes stiff [34]. The increase in the computational load

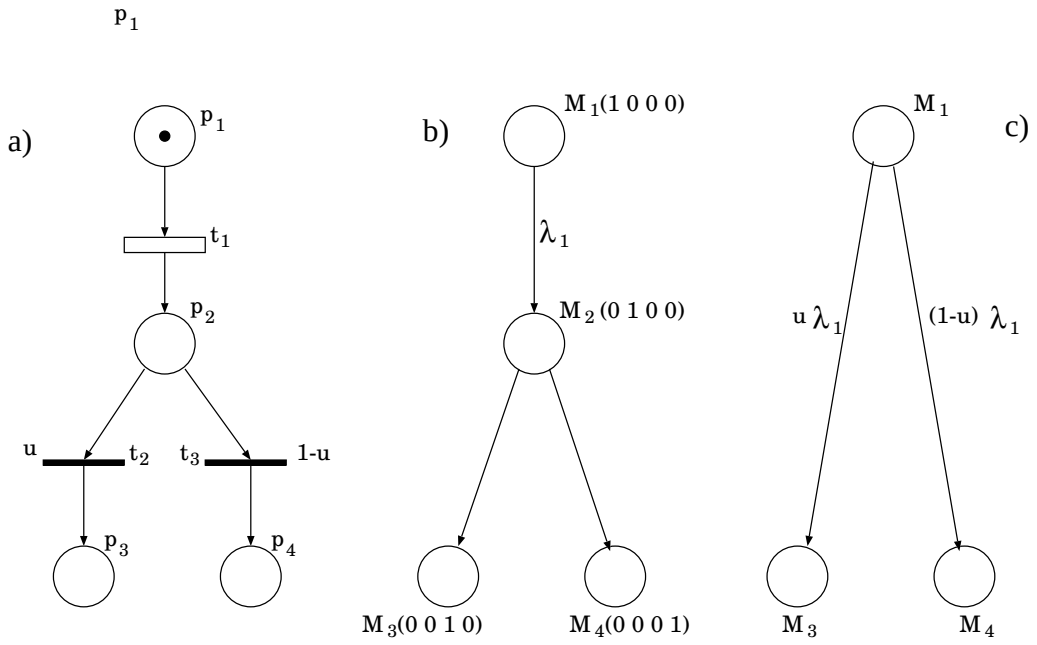

Figure 21: a) The random switch; b) The reachability graph; c) The reduced Markov chain defined over tangible markings.

due to stiffness can be overcome by resorting to a decomposition technique [11, 12]. This technique consists in decomposing the transition rate matrix Λ of the associated Markov chain in partitions, such that each partition contains rates of the same order of magnitude. An approximate solution to the original problem is obtained by solving each non-stiff partition in isolation [17]. This technique is incorporated in the package ESP [18].

Example 6 - In the folded PN of Figure 18 (the mutual exclusion problem with identical resources) transition t_2 has only the function of regulating the access to the shared resource $\mathbf{C}_s$ and thus can be modelled by an immediate transition (neglecting, in this case, the access time). The reachability set has 5 states (Figure 18a); markings M_2 and M_3 are vanishing since in these states the immediate transition t_2 is enabled. Eliminating the vanishing states by means of the previous rules leads to the reduced Markov chain of Figure 22 defined over tangible states only.

10. Computation of Measures of Reliability and Performance

A very important point of the time dependent representation of the system behaviour through SPN, is that they allow the user to define in a simple and natural way a large number of different measures related to the performance and reliability features of the system [7, 20]. In order to exploit this peculiarity, the input language must be structured for providing a friendly environment for the specification of the output measures. In the sequel we refer in particular to the language of the ESP package [18].

The stochastic behaviour of a SPN is determined by calculating the occurrence probabilities over the states of the reachability set $\mathcal{R}(M_1)$. Therefore, the output measures are

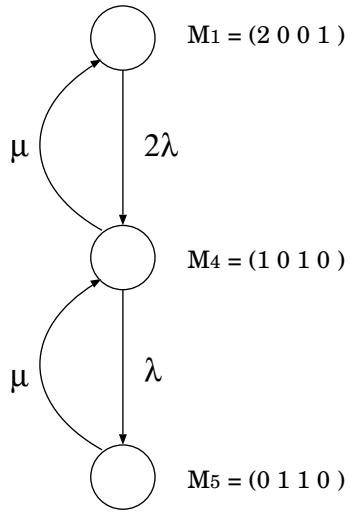

Figure 22: Markov chain defined over the tangible states of the PN of Figure 18.

defined at the net level, and the numerical computation is carried out automatically by solving the associated equation (10) and by scanning the states in $\mathcal{R}(M_1)$.

Since some of the output measures depend on the integral of the probabilities rather than on the probabilities themselves [25], it is necessary to provide the package with the appropriate computation of the integral of the state probabilities. In the following discussion it is implicitly intended that time t ranges from 0 to ∞, so that all definitions apply to the transient as well as to the steady state solution.

10.1. PROBABILITY OF A GIVEN CONDITION ON THE SPN

By means of logical or algebraic functions of the number of tokens in the PN places, we can specify an output condition (e.g. no tokens in the failed place). We identify in $\mathcal{R}(M_1)$ the subset of places S for which the output condition is true. The output measure

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

is given by:

$$Q_S(t) = \sum_{s \in S} q_s(t) \tag{12}$$

where $q_s(t)$ is the probability of being in state s at time t. For instance, if S is the set of operational states, $Q_S(t)$ in (12) is the usual definition of reliability (or availability).

A very useful case arises when we want to calculate the transient probability that the condition is satisfied for the first time. By using a standard device in the analysis of stochastic processes, we make the states $s \in S$ absorbing, and evaluate this quantity by stopping the process in S. An investigation of the application of SPN for computing the distribution of the completion time as a first marking problem is provided in [10].

10.2. TIME SPENT IN A MARKING

Let S be the subset of markings in which a particular condition is fulfilled. The expected time $\psi_S(t)$ spent in the markings $s \in S$ in the interval $0 - t$ is given by [8]:

$$\psi_S(t) = \sum_{s \in S} \int_0^t q_s(z)\, dz \tag{13}$$

Moreover, it is well known from the theory of Markov chains that as t approaches infinity the proportion of the time spent in states $s \in S$ equals the asymptotic probability:

$$Q_S(\infty) = \sum_{s \in S} q_s(\infty) \tag{14}$$

If S is the set of working states, $\psi_S(t)$ is the expected interval availability [25].

10.3. MEAN PASSAGE TIME

Given that $Q_S(t)$, as calculated in (12), is the probability of having entered subset S before t for the first time, the mean first passage time ϕ_S, has the usual expression:

$$\phi_S = \int_0^\infty [1 - Q_S(z)]\, dz \tag{15}$$

The above formula requires the transient analysis to be extended over long intervals. Of course, in this case, other well known direct techniques can be more effective [14].

10.4. DISTRIBUTION OF TOKENS IN A PLACE

Let p_i be a generic place of the PN. The cumulative distribution function (*Cdf*) of the number of tokens in p_i at time t is a staircase function in which the amplitude of the k-th step is obtained by summing up the probabilities of all the markings in $\mathcal{R}(M_1)$ containing k tokens ($k = 0, 1, 2, \ldots$) in p_i at time t. The density $f_i(k, t)$ is a mass function equal to the amplitude of the k-th step. The expected value of the number of tokens in place p_i at time t is:

$$E\,[m_i(t)\,] = \sum_{k=0}^{\infty} k\, f_i(k, t) \tag{16}$$

As an example, if place p_i represents identical units queueing up for a common resource the above quantities are the *Cdf* and the expected value of the number of units in the queue versus time. In reliability analysis a very interesting case arises when place p_i represents failed components. The above quantities provide the *Cdf* and the expected value of the number of failed components at time t.

10.5. EXPECTED NUMBER OF FIRINGS OF A PN-TRANSITION

Given an interval $(0, t)$ this quantity indicates how many times, on the average, an event modelled by a PN transition has occurred in that interval. Let t_k be a generic PN transition, and let S be the subset of $\mathcal{R}(M_1)$ which includes all the markings $s \in S$ enabling t_k. The expected number of firings of t_k in $(0, t)$ is given by:

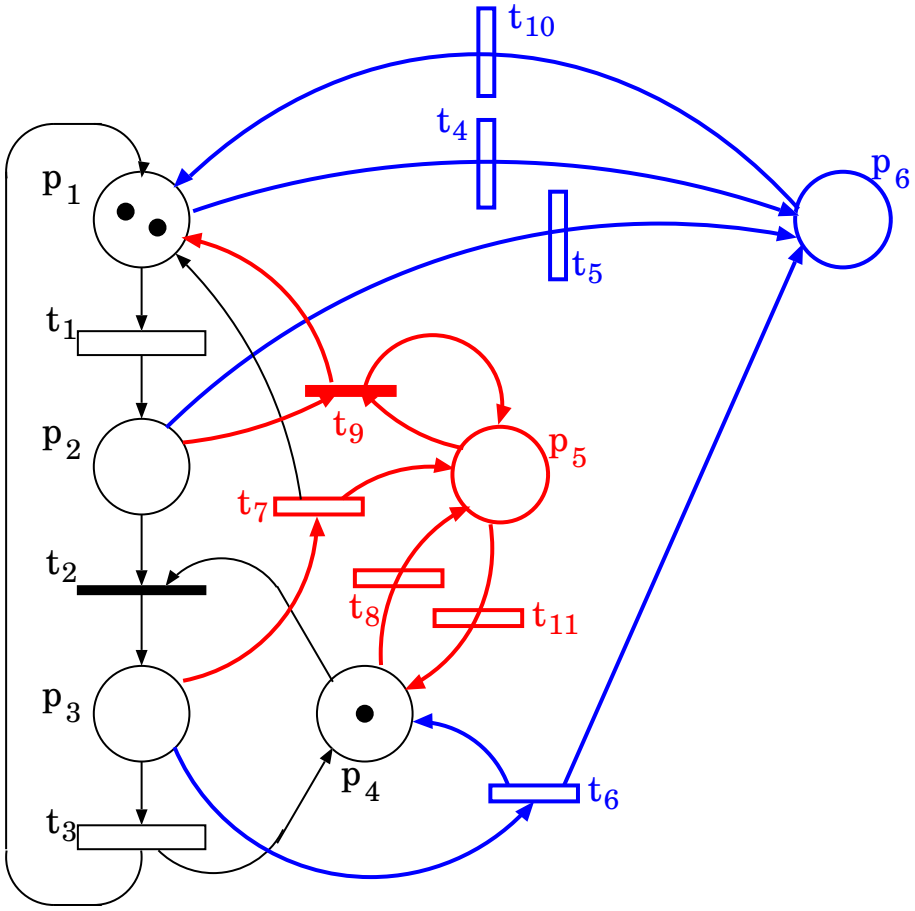

Figure 23: System of Figure 18 with failures and repairs.

$$\eta_k(t) = \sum_{s \in S} \int_0^t q_s(z)\, \lambda_k(s)\, dz \tag{17}$$

where $\lambda_k(s)$ is the firing rate of t_k in marking s.
In steady state, the expected number of firings per unit time becomes:

$$\nu_k = \sum_{s \in S} q_s(\infty)\, \lambda_k(s) \tag{18}$$

where $q_s(\infty)$ is the steady state probability of state s. As an example, if transition t_k indicates failure (repair) of a component, $\eta_k(t)$ in (17) provides the mean number of failures (repairs) of that component in $(0, t)$.

11. Performance/Reliability Modelling through SPN

Performance-oriented reliability analysis has been the subject of an extensive literature in recent years [9, 33, 27, 45]. We will show, by means of fully elaborated examples, that

the SPN language, described in the previous sections, is very suitable to model this class of problems.

11.1. PARALLEL UNITS WITH SHARED RESOURCE

This situation has been depicted in Figure 8, and arises very often in distributed systems. With reference to Figure 8 in a multiprocessor system $\mathbf{C}_1$ and $\mathbf{C}_2$ are independent processors working locally on their private memories and $\mathbf{C}_s$ is a shared global memory which contains common data for the two processors. In a manufacturing system $\mathbf{C}_1$ and $\mathbf{C}_2$ are two working cells connected to the same transportation system or to the same load/unload device $\mathbf{C}_s$.

Assuming $\mathbf{C}_1$ and $\mathbf{C}_2$ to be identical units, the SPN modelling the fault free system operation is reported in Figure 18. Taking into account the failure and repair of each unit the system operation is modelled by the SPN of Figure 23 [11].

TABLE I

Meaning of places and transitions in the SPN of Figure 23

p_1	Unit working independently	
p_2	Unit waiting for access to $\mathbf{C}_s$	
p_3	Unit operating with $\mathbf{C}_s$	
p_4	$\mathbf{C}_s$ free	
p_5	$\mathbf{C}_s$ failed	
p_6	Unit failed	
		firing rate
t_1	Unit requesting access to $\mathbf{C}_s$	1 m_1
t_2	Unit accessing $\mathbf{C}_s$	10^4
t_3	Unit releasing $\mathbf{C}_s$	5
t_4	Unit failure in local mode	$10^{-4}\, m_1$
t_5	Unit failure while waiting	$10^{-4}\, m_2$
t_6	Unit failure when working with $\mathbf{C}_s$	10^{-4}
t_7	$\mathbf{C}_s$ failure while working	10^{-4}
t_8	$\mathbf{C}_s$ failure while free	10^{-4}
t_9	Return to local mode when $\mathbf{C}_s$ failed	10_4
t_{10}	Unit repair	10^{-2}
t_{11}	$\mathbf{C}_s$ repair	10^{-2}

Table I reports the meaning of the places and transitions of Figure 23, and the numerical values assigned to the firing rates associated to each PN transition. With the initial marking M_1 shown in Figure 10, the reachability set $\mathcal{R}(M_1)$ consists in 15 states whose token distribution is reported in Table II. By inspection of Tables I and II the following subsets of states can be recognized:

- States 1,2,5,6,11: fault-free operation of the system.

- States 3,7,13: normal operation of one unit when the other one is in a failed condition.

- States 4,8,12: two units operating and the shared resource failed.

- States 10,14: one unit operating while the other one and the shared resource failed.

- State 9: two units failed.

- State 15: two units and the shared resource failed.

TABLE II

Reachability set and token distribution of the SPN of Figure 23

State	Marking					
	m_1	m_2	m_3	m_4	m_5	m_6
1	2	0	0	1	0	0
2	1	1	0	1	0	0
3	1	0	0	1	0	1
4	2	0	0	0	1	0
5	0	2	0	1	0	0
6	1	0	1	0	0	0
7	0	1	0	1	0	1
8	1	1	0	0	1	0
9	0	0	0	1	0	2
10	1	0	0	0	1	1
11	0	1	1	0	0	0
12	0	2	0	0	1	0
13	0	0	1	0	0	1
14	0	1	0	0	1	1
15	0	0	0	0	1	2

Table III is the literal description of the reachability graph $\mathcal{G}_R(M_1)$ of the SPN; for each state of $\mathcal{R}(M_1)$ on the first column, the enabled transitions and the immediately reachable states (in parentheses) are reported. Substituting the numerical values of the firing rates reported in Table I to the transition labels of Table III the transition rate matrix Λ of the associated Markov chain can be automatically generated.

By considering the access time to $\mathbf{C}_s$ as negligible with respect to the time constant of the system, t_2 and t_9 can be interpreted as immediate transitions. With this assumption it is seen from Table III that the states { 2,5,7,8,12,14 } become vanishing since in these states one of the immediate transitions is enabled. By reducing the state space with the rules of section 9.3, the final Markov chain is defined over a state space containing 9 tangible states.

An interesting performance/reliability measure for this system is the number of units doing useful work at time t, where by useful work we mean the work performed by each unit when operating independently. This measure takes into account the reduction in the system performance due to different effects: the congestion delays due to the sharing of the common resource, the transfer of data or pieces from each unit to $\mathbf{C}_s$ and the failure and repair cycles. By using the definitions of the previous section and looking at Table I, it is seen that this measure coincides with the expected number of tokens in place p_1 and thus can be easily defined at the PN level and computed by means of Equation (16).

TABLE III

Literal description of the Reachability Graph $\mathcal{G}_R(M_1)$

State	Enabled transition and immediately reachable state									
1	1	(2)	4	(3)	8	(4)				
2	1	(5)	2	(6)	4	(7)	5	(3)	8	(8)
3	1	(7)	4	(9)	8	(10)	10	(1)		
4	1	(8)	4	(10)	11	(1)				
5	2	(11)	5	(7)	8	(12)				
6	1	(11)	3	(1)	4	(13)	6	(3)	7	(4)
7	2	(13)	5	(9)	8	(14)	10	(2)		
8	1	(12)	4	(14)	5	(10)	9	(4)	11	(2)
9	8	(15)	10	(3)						
10	1	(14)	4	(15)	10	(4)	11	(3)		
11	3	(2)	5	(13)	6	(7)	7	(8)		
12	5	(14)	9	(8)	11	(5)				
13	3	(3)	6	(9)	7	(10)	10	(6)		
14	5	(15)	9	(10)	10	(8)	11	(7)		
15	10	(10)	11	(9)						

11.2. PARALLEL SYSTEM WITH FINITE INPUT BUFFER

The block diagram of the system is shown in Figure 24. It consists in u identical units $\mathbf{U}_1$, $\mathbf{U}_2, \ldots, \mathbf{U}_u$ and in an input buffer with b positions $\mathbf{B}_1$, $\mathbf{B}_2, \ldots, \mathbf{B}_b$ [33].

The GSPN model of the fault free system operation is shown in Figure 25 [7]; the sum of tokens in p_1 and p_2 is equal to b (number of buffer positions; see also Section 4.3), whereas the sum of tokens in p_3 and p_4 is equal to u (number of parallel units). In other words, $\{p_1, p_2\}$ and $\{p_3, p_4\}$ form place-invariants.

The firing rate associated to t_1 is the task arrival rate λ, while the firing rate associated to t_3 is the service rate proportional to the number of active units $m_4\,\mu$, being μ the service rate of a single unit and m_4 the number of tokens in p_4. t_2 is an immediate transition (we neglect the transfer time from the buffer to the service station).

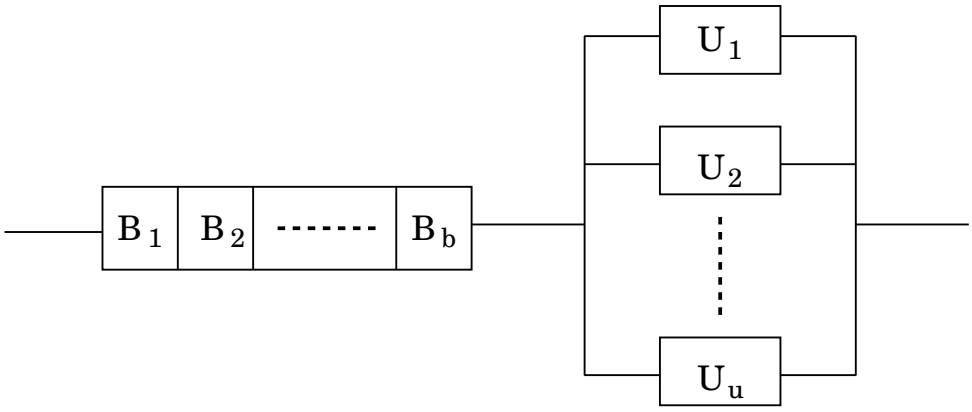

Figure 24: Block diagram of a parallel system with finite buffer.

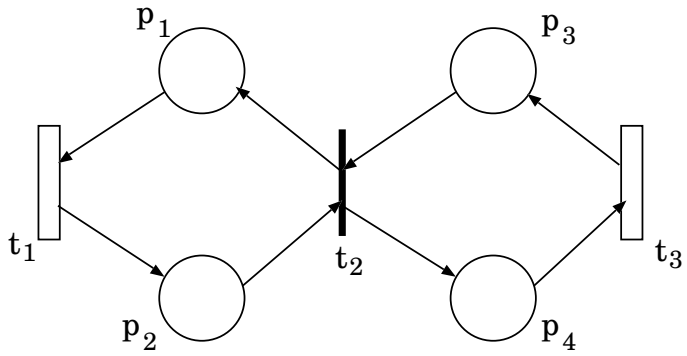

Figure 25: Fault-free SPN model of the system of Figure 24.

When failures and repairs are considered, the GSPN model becomes as in Figure 26. Heavy lines represent the fault-free operation, light lines failures and dotted lines repairs. Let us first focus our attention on the failure transitions; with reference to Figure 26 the following hypotheses have been considered:

- Buffer stages fail one at the time, either when free (t_4), or when occupied (t_5), with possibly different failure rates. t_6 and t_7 form a random switch modelling the fact that with probability v_B a buffer stage failure is recovered (the buffer continues to be operational with a storing capacity reduced by one stage), and with probability $1 - v_B$ the failure is not recovered and the buffer locks (inhibitor arc from p_7 to t_2).

- The units $\mathbf{U}_i$ ($i = 1, 2, \ldots, u$) fail either when idle (t_8) or when active (t_9), with possibly different failure rates. The failure of an idle unit is recovered with probability one, while the failure of an active unit is recovered with probability v_U (random switch t_{10} t_{11}). A task is lost only when an active unit fails.

By slightly modifying the GSPN of Figure 26, different design alternatives or recovery strategies could be accommodated. When repair is considered, t_{12} and t_{13} refer to buffer

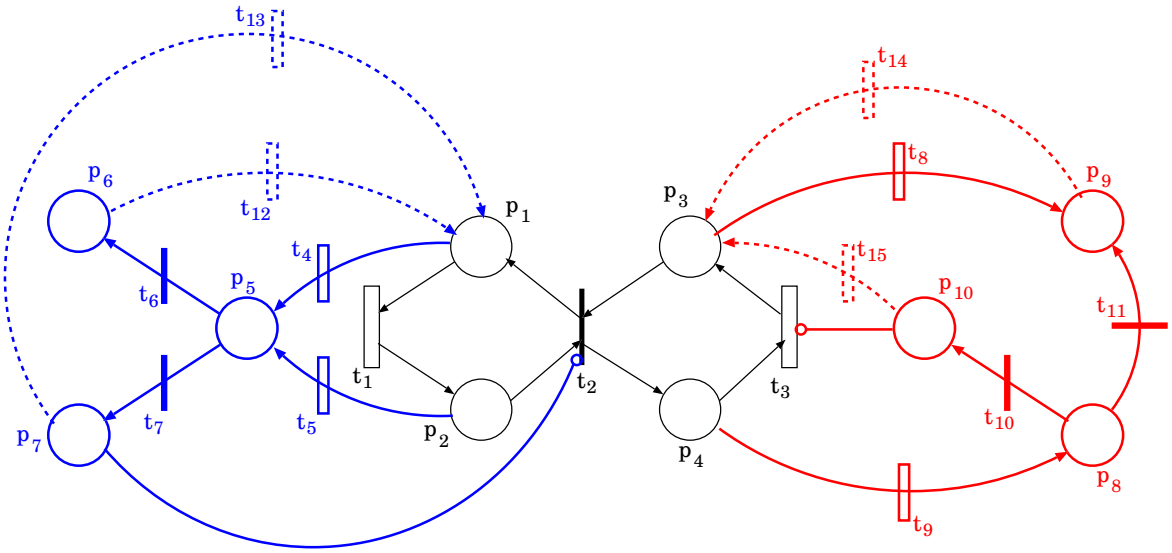

Figure 26: SPN model of the system of Figure 24 with failures and repairs.

stage repair, and t_{14} and t_{15} to processor repair; the considered model allows us to allocate different repair rates for recoverable and unrecoverable failures.

The meaning of places and transitions in Figure 26 is summarized in Table IV, where the expressions of the firing rates for the timed transitions, and of the switching probabilities for the immediate transitions, are also given.

The initial marking M_1 consists in b tokens in place p_1 and u tokens in p_3. As measures characterizing the system performance and reliability, we define the following.

. *Mean fraction of arrived tasks processed in 0-t.*
The mean number of processed tasks in $0-t$ is given by the mean number of firings of t_3 (Equation 17). The mean number of arrived tasks in $0-t$ is simply $\lambda \cdot t$, having assumed a Poisson arrival process with rate λ. Thus the performance/reliability index $Y(t)$, representing the mean fraction of arrived tasks processed in $0-t$ is calculated as:

$$Y(t) = \frac{\eta_3(t)}{\lambda\, t} \tag{19}$$

. *Mean number of failures (repairs) in 0-t.*
This quantity is given by $[\eta_4(t)+\eta_5(t)]$ (Equation 17) for buffer stage failure ($[\eta_{12}(t)+\eta_{13}(t)]$ for buffer stage repair), and by $[\eta_8(t)+\eta_9(t)]$ for unit failure ($[\eta_{14}(t)+\eta_{15}(t)]$ for unit repair).

. *Cdf and mean number of active, idle, failed, units or buffer stages.*
These quantities are obtained by applying the procedure of paragraph 10.4 to place p_4 for active units, to place p_3 for idle units, and to places $[p_9 + p_{10}]$ for failed units. Similarly, place p_1 indicates free buffer stages, p_2 filled buffer stages, and $[p_6 + p_7]$ failed buffer stages.

TABLE IV

Meaning of places and transitions in the SPN of Figure 26

p_1	Free buffer stage	
p_2	Occupied buffer stage	
p_3	Idle unit	
p_4	Active unit	
p_5	Failed buffer stage	
p_6	Recovered buffer stage failure	
p_7	Unrecovered buffer stage failure	
p_8	Failed active unit	
p_9	Recovered unit failure	
p_{10}	Unrecovered unit failure	
		firing rate
t_1	Buffer stage becomes occupied	λ
t_2	Transfer from buffer to unit	*immed.*
t_3	Unit ends a task	$m_4\,\mu$
t_4	Free buffer stage fails	$m_1\,\gamma_4$
t_5	Occupied buffer stage fails	$m_2\,\gamma_5$
t_6	Buffer stage failure is recovered	v_B
t_7	Buffer stage failure is not recovered	$(1 - v_B)$
t_8	Idle unit fails	$m_3\,\gamma_8$
t_9	Active unit fails	$m_4\,\gamma_9$
t_{10}	Unit failure is not recovered	$(1 - v_U)$
t_{11}	Unit failure is recovered	v_U
t_{12}	Repair of recovered buffer stage	ρ_{12}
t_{13}	Repair of unrecovered buffer stage	ρ_{13}
t_{14}	Repair of recovered unit	ρ_{14}
t_{15}	Repair of unrecovered unit	ρ_{15}

A numerical example has been run with $u = 2$ and $b = 2$. The reachability set, in this case, comprises 88 tangible states and 84 vanishing states. With reference to Table IV, we have assigned to the parameters the following numerical values (being $w = \lambda/\mu$ the load factor of the system):

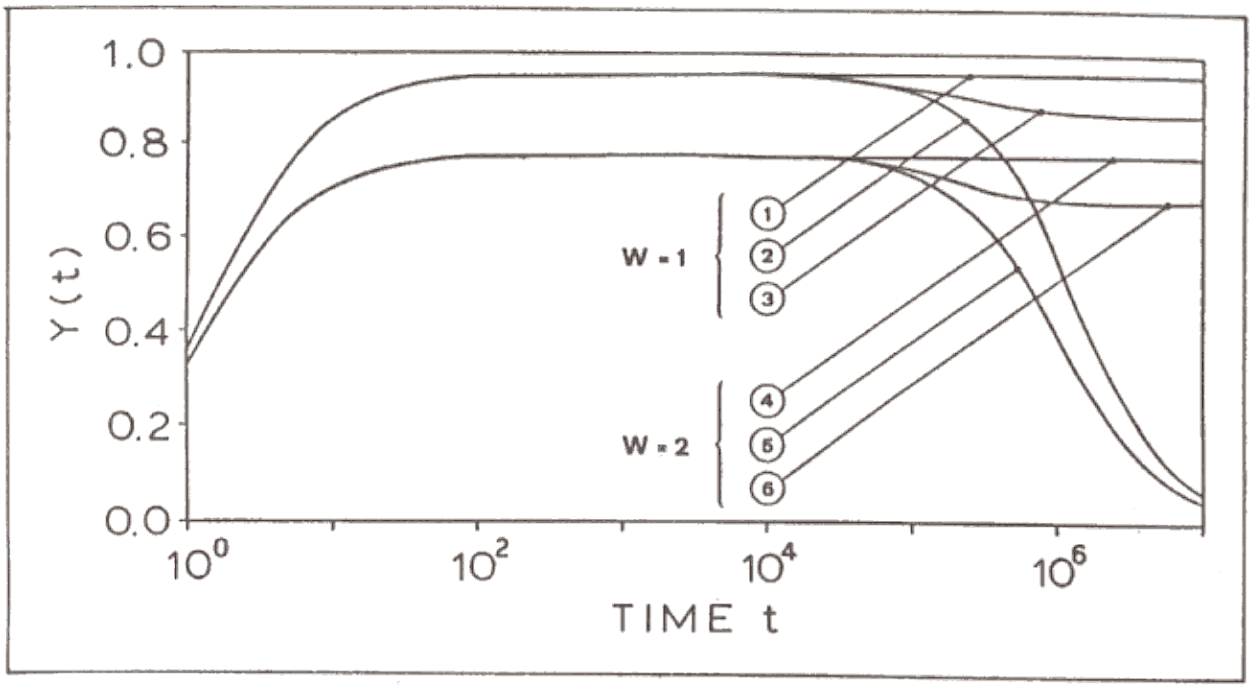

Figure 27: Mean fraction of arrived tasks processed in $0-t$ versus time.

$$
\begin{aligned}
\mu &= 1 \\
\lambda &= w\,\mu \\
\gamma_4 &= \gamma_5 = \gamma_8 = \gamma_9 = \gamma = 1.0\,10^{-6} \\
\rho_{12} &= \rho_{13} = \rho_{14} = \rho_{15} = 10\,\gamma \\
v_B &= v_U = 0.9
\end{aligned}
$$

Two cases have been examined with different load factors: $w = 1$ and $w = 2$. The last case represents the ideal load factor since the arrival rate is twice as large as the service rate, but there are two parallel service units. Numerical results have been obtained using the program ESP [18] and resorting to a decomposition technique due to the high spread in the firing rate values. Figure 27 shows $Y(t)$ (Equation 19) as a function of t for the two chosen values of w, and in three different conditions, namely: fault-free operation (curves 1 and 4); with failures (curves 2 and 5); with failures and repairs (curves 3 and 6). Figure 27 shows how the system performance (throughput) is degraded when considering failures and failures/repairs, and can be of valuable support at the design level.

12. Simulative Analysis of SPN

In the previous Sections the SPN was used as a language for generating an associated Markov chain whose transient and ergodic behaviour is obtained by solving Equations (10) and (11) respectively. However, a simulative approach is also possible.

The simulative approach is very simple from a logical point of view, and is easily implementable in a computer program, so that SPN can also be considered as a possible general simulative language. Due to these characteristics, it is conceivable to construct general SPN solvers [20] where both the analytic approach (when feasible and convenient) and the simulative approach are present.

The core of the simulator is that in each marking we need to choose the PN transition which actually fires, among those enabled. This choice is done by generating a random sample from the distribution function associated to each transition and selecting the transition with the minimum firing sample. The simulator clock is updated with the minimum sample and we move to the next marking where a new choice procedure is initiated. The basic algorithm for the generation of a timed execution sequence $\mathcal{T}_E$ (section 8) can be outlined as follows, when the firing times are exponentially distributed:

```
begin
marking ← initial marking
clock = 0
repeat
    for  j  := 1 to n_t do
        begin
        if { t_k is enabled } then
            generate a random sample θ_k
        end
    find minimum θ_k
    generate new marking
    clock=clock+min(θ_k)
until { terminating condition is fulfilled }
end
```

The termination criteria are driven by the type of simulation: transient simulation or ergodic simulation [19]. In the transient simulation the user defines exit places or absorbing places; the simulation trial is stopped once a token reaches an exit place. Statistics are gathered by generating random timed execution sequences $\mathcal{T}_E$ through the PN [22].

The ergodic simulation is a regenerative-type simulation [30], in which a return to the initial marking constitutes a regeneration point in the simulation. A trial is defined as the random timed execution sequence $\mathcal{T}_E$ starting and ending with the initial marking.

All the measures defined in Section 10 can be estimated as a result of the simulation approach. The definition of these measures in both transient and ergodic simulation is usually straightforward, and is outlined in [20]. Confidence intervals can be also calculated as a function of the number of trials [19].

The very important fact about the simulative approach, to be noted here, is that in each trial we only generate a single timed execution sequence $\mathcal{T}_E$, so that we do not need to generate and store all the reachable markings at the same time. Moreover, the extension of the simulative approach to the case in which the random variables associated to the PN transitions are generally distributed is, in principle, quite simple. In fact, once the execution policy is specified (i.e. the way in which the SPN keeps trace of the past history; see Section 8), the basic simulation algorithm must be modified by attaching a clock to

each PN-transitions. Each time a move is selected, the clocks are updated by recovering the elapsed time as specified by the execution policy, and the following selection is performed by comparing the values of all these clocks. This extension [21, 26] is not further considered in the present notes.

13. Conclusion

These lecture notes were intended as introductive material to the use of Petri nets as a general language for the modelling and analysis of the behaviour of complex systems versus time. In the first part, the aim was to show how the semantics of classical PN is suitable to model various kinds of logical as well as physical interactions among components in a system (interactions that are not easily reprensentable in other modelling frameworks).

The second part was more specifically devoted to define the Stochastic PN extension and to present examples taken from the reliability area. Only the case where the stochastic process associated to the SPN is a homogeneous Markov chain has been considered in details. This case arises when the firing times assigned to the PN transitions are exponentially distributed.

From the discussion contained in these lecture notes we can summarize some advantages and disadvantages of the SPN as a modelling tool. The main advantages include: the graphic nature, the conciseness in comparison with state graphs, the possibility of implementing analysis techniques. The graphic nature facilitates the use by non skilled users and allows to implement very friendly graphic editors for the specification of the input net. We finally stress that the use of SPN requires only the specification of the topology of the starting PN, the specification of the firing rates (or of the distribution functions in the general case) associated to the transitions and the specification of the output measures to be computed following the indications provided in Section 10. All the subsequent steps, which consist in:

- the generation of the reachability graph $\mathcal{G}_R(M_1)$;
- the generation of the associated Markov chain;
- the transient and ergodic solution of the Markov chain;
- the evaluation of the relevant process measures;

can be executed in a completely automated way by a computer program, thus making transparent to the user the associated mathematics.

The main disadvantages of SPN arise from the size of the net obtained in modelling very complex distributed systems. In this case the model is difficult to validate at the net level, and the number of reachable markings tends to explode, making analytically intractable the associated Markov chain. It should be recognized, however, that this drawback is common to almost all general purpose modelling techniques.

References

[1] *International Workshop Timed Petri Nets*, Torino (Italy), 1985. IEEE Computer Society Press No. 674.

[2] *International Workshop Petri Nets and Performance Models*, Madison, 1987. IEEE Computer Society Press No. 796.

[3] T. Agerwala. Putting Petri nets to work. *IEEE Computer*, pages 85–94, December 1979.

[4] M. Ajmone Marsan, G. Balbo, A. Bobbio, G. Chiola, G. Conte, and A. Cumani. On Petri nets with stochastic timing. In *Proceedings International Workshop on Timed Petri Nets*, pages 80–87, Torino (Italy), 1985. IEEE Computer Society Press no. 674.

[5] M. Ajmone Marsan, G. Balbo, A. Bobbio, G. Chiola, G. Conte, and A. Cumani. The effect of execution policies on the semantics and analysis of stochastic Petri nets. *IEEE Transactions on Software Engineering*, SE-15:832–846, 1989.

[6] M. Ajmone Marsan, G. Balbo, and G. Conte. A class of generalized stochastic Petri nets for the performance evaluation of multiprocessor systems. *ACM Transactions on Computer Systems*, 2:93–122, 1984.

[7] M. Ajmone Marsan, A. Bobbio, G. Conte, and A. Cumani. Performance analysis of degradable multiprocessor systems using generalized stochastic Petri nets. *IEEE Computer Society Newsletters*, 6, SI-1:47–54, 1984.

[8] R.E. Barlow and F. Proschan. *Statistical Theory of Reliability and Life Testing*. Holt, Rinehart and Winston, New York, 1975.

[9] M.D. Beaudry. Performance-related reliability measures for computing systems. *IEEE Transactions on Computers*, C-27:540–547, 1978.

[10] A. Bobbio. Petri nets generating Markov reward models for performance/reliability analysis of degradable systems. In R. Puigjaner and D. Poitier, editors, *Modeling Techniques and Tools for Computer Performance Evaluation*, pages 353–365. Plenum Press, 1989.

[11] A. Bobbio, A.Cumani, and R. Del Bello. Reduced markovian representation of stochastic Petri net models. *Systems Science*, 10:5–23, 1984.

[12] A. Bobbio and K.S. Trivedi. An aggregation technique for the transient analysis of stiff Markov chains. *IEEE Transactions on Computers*, C-35:803–814, 1986.

[13] G.W. Brams. *Réseaux de Petri: Théorie et pratique*. Masson, 1983. (in French).

[14] J.A. Buzacott. Markov approach to finding failure times of repairable systems. *IEEE Transactions on Reliability*, R-19:128–134, 1970.

[15] W.M. Chow, E.A. McNair, and C.H. Sauer. Analysis of manufacturing systems by Research Queueing Package. *IBM Journal of Research and Development*, 29:330–341, 1985.

[16] G. Ciardo. Toward a definition of modeling power for stochastic Petri net models. In *Proceedings International Workshop on Petri Nets and Performance Models*, pages 54–62, Madison, 1987. IEEE Computer Society Press no. 796.

[17] P.J. Courtois. *Decomposability: Queueing and Computer System Applications*. Academic Press, New York, 1977.

[18] A. Cumani. Esp - A package for the evaluation of stochastic Petri nets with phase-type distributed transition times. In *Proceedings International Workshop Timed Petri Nets*, pages 144–151, Torino (Italy), 1985. IEEE Computer Society Press no. 674.

[19] J. Bechta Dugan. Extended stochastic Petri nets: applications and analysis. Technical report, Phd Thesis, Department of Computer Science, Duke University, 1984.

[20] J. Bechta Dugan, A. Bobbio, G. Ciardo, and K. Trivedi. The design of a unified package for the solution of stochastic Petri net models. In *Proceedings International Workshop on Timed Petri Nets*, pages 6–13, Torino (Italy), 1985. IEEE Comp Soc Press no. 674.

[21] J. Bechta Dugan, K. Trivedi, R. Geist, and V.F. Nicola. Extended stochastic Petri nets: applications and analysis. In *Proceedings PERFORMANCE '84*, Paris, 1984.

[22] G.S. Fishman. *Concepts and methods in discrete event digital simulation*. Wiley, New York, 1973.

[23] G. Florin and S. Natkin. Les reseaux de Petri stochastiques. *Technique et Science Informatique*, 4:143–160, 1985.

[24] H.J. Genrich and K. Lautenbach. System modelling with high level Petri nets. *Theoretical Computer Science*, 13:109–136, 1981.

[25] A. Goyal, S. Lavenberg, and K.S. Trivedi. Probabilistic modeling of computer system availability. *Annals of Operations Research*, 8:285–306, 1987.

[26] P.J. Haas and G.S. Shedler. Regenerative stochastic Petri nets. *Performance Evaluation*, 6:189–204, 1986.

[27] B.R. Iyer, L. Donatiello, and P. Heidelberger. Analysis of performability for stochastic models of fault-tolerant systems. *IEEE Transactions on Computers*, C-35:902–907, 1986.

[28] K. Jensen. Coloured Petri nets and the invariant method. *Theoretical Computer Science*, 14:317–336, 1981.

[29] L. Kleinrock. *Queuing systems, Volume 1: Theory*. Wiley Interscience, New York, 1975.

[30] A.J. Lemoine M.A. Crane. An introduction to the regenerative method for simulation analysis. In A.V. Balakrishnan and M. Thorna, editors, *Lecture Notes in Control and Information Sciences*. Springer-Verlag, 1977.

[31] J. Martinez and M. Silva. A simple fast algorithm to obtain all invariants of a generalized Petri net. In *Proceedings 2-nd European Workshop on Application and Theory of Petri Nets*. Springer-Verlag, 1981.

[32] P.M. Merlin and D.J. Faber. Recoverability of communication protocols - Implication of a theoretical study. *IEEE Transactions on Communication*, COM-24:1036–1043, 1976.

[33] J.F. Meyer. Closed form solution of performability. *IEEE Transactions on Computers*, C-31:648–657, 1982.

[34] W.L. Miranker. *Numerical Methods for Stiff Equations*. Reidel, Dordrecht, 1981.

[35] M.K. Molloy. On the integration of delay and throughput measures in distributed processing models. Technical report, Phd Thesis, UCLA, 1981.

[36] S. Natkin. Les reseaux de Petri stochastiques et leur application a l'evaluation des systemes informatiques. Technical report, These de Docteur Ingegneur, CNAM, Paris, 1980.

[37] J.L. Peterson. Petri nets. *Computing Surveys*, 9:223–252, 1977.

[38] J.L. Peterson. *Petri net theory and the modeling of systems*. Prentice Hall, Englewood Cliffs, 1981.

[39] C.A. Petri. Kommunikation mit automaten. Technical report, Doctoral Thesis, University of Bonn, 1962. (Available in English as: *Communication with automata*, Technical Report RADC-TR-65-377, Rome Air Development Center, Griffiss NY, 1966).

[40] C.V. Ramamoorthy and G.S. Ho. Performance evaluation of asynchronous concurrent systems using Petri nets. *IEEE Transactions on Software Engineering*, SE-6:440–449, 1980.

[41] A. Reibman and K.S. Trivedi. Numerical transient analysis of Markov models. *Computers and Operations Research*, 15:19–36, 1988.

[42] W. Reisig. *Petri nets - An introduction*. Springer-Verlag, 1982.

[43] J. Sifakis. Use of Petri nets for performance evaluation. In H. Beilner and E. Gelenbe, editors, *Measuring, modelling and evaluating computer systems*, pages 75–93. North Holland, 1977.

[44] M. Silva. *Las Redes de Petri en la Automatica y la Informatica*. AC, Madrid, 1985.

[45] R. Smith, K. Trivedi, and A.V. Ramesh. Performability analysis: Measures, an algorithm and a case study. *IEEE Transactions on Computers*, C-37:406–417, 1988.

[46] W. Whitt. Blocking when service is required from several facilities simultaneously. *AT&T Technical Journal*, 64:1807–1856, 1985.

[47] W.M. Zuberek. Timed Petri nets and preliminary performance evaluation. In *Proceedings 7-th Annual Symposium on Computer Architecture*, pages 88–96, 1980.