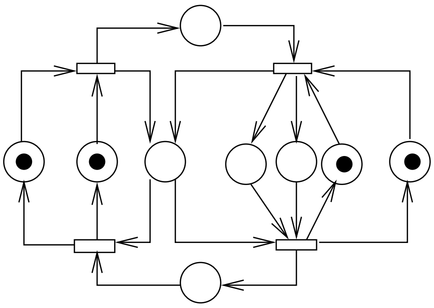
Stochastic Petri Nets

- An Introduction to the Theory -



Falko Bause

Informatik IV
Universität Dortmund
D-44221 DORTMUND
Germany
e-mail: bause@ls4.cs.uni-dortmund.de

Pieter S Kritzinger

Data Network Architectures Laboratory
Department of Computer Science
University of Cape Town
Private Bag, RONDEBOSCH
7700 South Africa

e-mail: psk@cs.uct.ac.za

Preface

Any developer of discrete event systems knows that the most important quality of the final system is that it be functionally correct by exhibiting certain functional, or *qualitative* properties decided upon as being important. Once assured that the system behaves correctly, it is also important that it is efficient in that its running cost is minimal or that it executes in optimum time or whatever performance measure is chosen. While functional correctness is taken for granted, the latter *quantitative* properties will often decide the success, or otherwise, of the system. Ideally the developer must be able to specify, design and implement his system and test it for both functional correctness and performance using only one formalism. No such formalism exists as yet. In recent years the graphical version of the *Specification and Description Language (SDL)* has become very popular for the specification, design and partial implementation of discrete systems. The ability to test for functional correctness of systems specified in SDL is, however, limited to time consuming simulative executions of the specification and performance analysis is not directly possible. Petri nets, although graphical in format are somewhat tedious for specifying large complex systems but, on the other hand were developed exactly to test discrete, distributed systems for functional correctness. With a Petri net specification one can test, e.g., for deadlock, liveness and boundedness of the specified system. Petri nets in their various formats, have been studied extensively since first proposed by Carl Adam Petri in 1962 [133] and several algorithms exist to determine the functional properties of nets. Another paradigm which is aimed at testing for functional correctness is that of process algebras or calculi for communicating systems.

The major drawback of Petri nets, as originally proposed and process algebras (amongst others) is that quantitative analyses are not catered for. As a consequence, the developer who needs to know about these properties in his system has to devise a different model of the system which, apart from the overhead concerned provides no guarantee of consistency across the different models. Because of the latter, computer scientists during the last decade added time, in various forms, to ordinary Petri nets to create Stochastic Petri nets (SPNs) and Generalized Stochastic Petri nets (GSPNs) for performance modelling and a great deal of theory has developed around Stochastic Petri nets as these are generically known.

Another aspect which also contributed significantly to the development of Stochastic Petri nets is the fact that their performance analysis is based upon Markov theory. Since the description of a Markov process is cumbersome, abstract models have been devised for their specification. Of these, queueing networks (QNs) was originally the most popular, especially since the analysis of a large class of QNs (product-form QNs) can be done very efficiently. QNs cannot, however, describe system behaviours like blocking and forking and with the growing importance of distributed systems this inability to describe synchronisation naturally turned the focus to Petri nets as well.

Stochastic Petri nets are therefore a natural development from the original Petri nets because of

- the advantage of their graphical format for system design and specification
- the possibility and existing rich theory for functional analysis with Petri nets
- the facility to describe synchronisation, and
- the natural way in which time can be added to determine quantitative properties of the specified system.

The disappointing thing about Stochastic Petri nets is that the integration of time changes the behaviour of the Petri net significantly. So properties proven for the Petri net might not hold for the corresponding time-augmented Petri net. E.g., a live Petri net might become deadlocked or a non-live Petri net might become live. We will see that the analysis techniques developed for Petri nets are not always applicable to SPNs. But there are ways around this, as we shall see in this book. Also, using Stochastic Petri nets to specify the sharing of resources controlled by specific scheduling strategies is very cumbersome. So the pendulum has swung back, in a sense, that we introduce certain concepts from queueing theory when presenting Queueing Petri nets (QPNs) which offer the benefits of both worlds, Petri nets and Queueing networks.

This book itself arose out of a desire by the authors to collect all one needs to understand Stochastic Petri net theory in one volume. It is in three parts. The first part is on stochastic theory leading to introductory queueing theory and simple queues. In Part I we emphasise Markovian theory, because where general queueing theory fails, Markovian analysis can often still be useful.

Part II is about Petri nets, starting with ordinary Petri nets and ending with Coloured Petri nets. Ordinary and Coloured Petri nets do not involve time and were developed to test the functionality of concurrent systems. In this part of the book we give an overview of the most important analysis techniques paying particular attention to the validation of those properties which are essential for Stochastic Petri nets.

Our emphasis in Part III is on those Stochastic Petri net models which can be analysed by Markovian techniques. The intention of this book is not to give an overview of several or all Stochastic Petri net models appearing in the literature, but to stress a combined view of functional and performance analysis in the context of some Stochastic Petri net models.

We hope that by reading this book, the reader will become as excited as we are about the subject of Stochastic Petri nets and the many unsolved problems arising from the increasing demands for correctness and performance when specifying discrete event systems.

Falko Bause and Pieter Kritzinger
Dortmund, Germany
Cape Town, South Africa
1995.

Preface to the Second Edition

A great deal of progress has been made in the analysis of Petri nets and Stochastic Petri nets since the first edition of this book appeared over 6 years ago. Amongst others, partial state space exploration methods have been proposed and state-based analysis techniques exploiting the structure of the system being modeled are now known. In the case of Queueing Petri nets and stochastic Petri nets in general, results and algorithms based on product-form solutions have been introduced.

The results are that nets with up to 50 million states can now be analysed on ordinary computing equipment. In order to guide the reader to these results we have added several links in this edition to the relevant literature and updated the *Further Reading* sections at the end of each chapter as starting points for more detailed information.

Naturally, we were tempted to include the new material mentioned in the book. That would have however, detracted from the focus and advantage of this text: A concise introduction to both the functional and performance aspects of Petri nets without emphasising the one or the other.

Falko Bause and Pieter Kritzinger
Dortmund, Germany
Cape Town, South Africa
2002.

For *Heinz Beilner*, our friend and mentor, without whom this book would never have been written.

Contents

Part I

STOCHASTIC THEORY

1 Random Variables

Much of the world around us is not very *deterministic* although it may not be apparent at first glance. Consider a computer, for instance, which given the same input values, will *always* give the *same* output. While a computer program is processing incoming data however, it is often not possible to predict from one moment to the next

– what input values will arrive for processing, or

– the time sequence in which they will arrive.

Think of the node of a computer network to understand this. Although the set of messages which may arrive at the node is finite and known, we cannot tell *for certain* from instant to instant which messages will arrive from where. Moreover, the network software is likely to be using the same processor(s) at the node as the operating system. When the process executing the network software will be interrupted and by which process cannot be said for certain. All of which makes it impossible to tell for certain what will happen next. We say the process just described is *stochastic*.

The term *stochastic* has an exact mathematical meaning and there is a vast theory developed to predict the behaviour of stochastic processes. This part of the book gives only a basic introduction to that theory. Goodman [86] provides a more thorough introduction to the subject while an extensive treatment can be found in Howard [90].

1.1 Probability Theory Refresher

In order to understand stochastic theory, one needs to know some fundamental concepts of probability theory. This section provides such a basic introduction. For students wishing a more fundamental introduction, there are many good books on probability theory, such as those of Feller [75] and Ross [152].

The first concept in probability theory that we need to know is that of an *exhaustive set of events* which is a set of events whose union forms the sample space $\mathcal{S}$ of all possible outcomes of an experiment. The sample space when we roll an ordinary die, consists of 6 events for instance.

If two events A and B are such that

$$A \cap B = \emptyset \text{ (the empty set)}$$

then the two events are said to be *mutually exclusive* or disjoint. This leads to the concept of *mutually exclusive exhaustive* events $\{A_1,A_2,\ldots,A_n\}$ which are events such that

$$\begin{aligned} A_iA_j = A_i \cap A_j &= \emptyset \text{ for all } i \neq j \\ A_1 \cup A_2 \cup \ldots \cup A_n &= \mathcal{S} \end{aligned} \tag{1.1}$$

The next important concept is that of *conditional probability*. The conditional probability of the event A, given that the event B occurred (denoted as $P[A|B]$) is defined as

$$P[A|B] := \frac{P[AB]}{P[B]}$$

whenever $P[B] \neq 0$.

The *statistical independence* of events can be defined as follows. Two events A and B are said to be statistically independent iff

$$P[AB] = P[A]P[B]. \tag{1.2}$$

For three statistically independent events A, B, C each pair of events must satisfy Eq. (1.2) as well as

$$P[ABC] = P[A]P[B]P[C]$$

and so on for n events requiring the n-fold factoring of the probability expression as well as the $(n-1)$-fold factorings all the way down to all the pairwise factorings. Moreover, for two independent events A and B

$$P[A|B] = P[A]$$

which merely states that the knowledge of the occurrence of an event B does not affect the probability of the occurrence of the independent event A in any way and vice-versa.

We also need to know the *theorem of total probability* for our basic understanding of probability theory.

Theorem 1.1 Theorem of Total Probability. *Consider an event B and a set of mutually exclusive exhaustive events $\{A_1,A_2,\ldots,A_n\}$. If the event B is to occur it must occur in conjunction with exactly one of the mutually exhaustive events A_i. That is*

$$P[B] = \sum_{i=1}^{n} P[A_iB]$$

From the definition of conditional probability we may always write

$$\begin{aligned} P[A_iB] &= P[A_i|B]P[B] \\ &= P[B|A_i]P[A_i] \end{aligned}$$

which leads to the second form of the theorem of total probability

$$P[B] = \sum_{i=1}^{n} P[B|A_i]P[A_i]$$

The last equation suggests that to find the probability of some complex event B, one simplifies the event by conditioning it on some event A_i in such a way that computing the probability of event B given event A_i is less complex and then to multiply by the probability of the conditional event A_i to yield the joint probability $P[A_iB]$. Having done this for a set of mutually exclusive exhaustive events $\{A_i\}$ we may then sum these probabilities to find the probability of the event B. If we need to simplify the analysis even further, we may condition event B on more than one event and then uncondition each of these events by multiplying by the probability of the appropriate condition and then sum all possible forms of all conditions.

The final bit of probability theory that we are certain to come across in our study of stochastic systems is *Bayes' theorem.*

Theorem 1.2 Bayes' theorem. *Let $\{A_i\}$ be a set of mutually exclusive and exhaustive events. Then*

$$P[A_i|B] = \frac{P[B|A_i]P[A_i]}{\sum_{j=1}^{n} P[B|A_j]P[A_j]} \tag{1.3}$$

Bayes' theorem allows us to compute the probability of one event conditioned on a second by calculating the probability of the second conditioned on the first and other terms.

Exercise 1.1 *If there are n people present in a room, what is the probability that at least two of them have the same birthday? How large may n be for this probability to be less than 0.5?*

Exercise 1.2 *A student writes a multiple-choice examination where each question has exactly m possible answers. Assume that a student knows the correct answer to a proportion p of all the questions; if he does not know the correct answer, he makes a random guess. Suppose that the student got the answer to a particular question wrong. What is the probability that he was guessing?*

1.2 Discrete Random Variables

We call a variable *random* and denote it χ if we cannot tell for certain what its value will be. Examples of such random variables are the temperature outside on any particular day, the number of customers in a supermarket checkout line or the number of messages arriving at a network node.

A random variable is said to be *discrete* if the set of possible values of χ is countable (but not necessarily finite). Since we do not know for certain what value it will have, we say that it will have value x with a probability $p_\chi(x)$. That is

$$p_\chi(x) = P[\chi = x]. \tag{1.4}$$

In this formula, x can be any real number and $0 \leq p_\chi(x) \leq 1$ for all values of x. $p_\chi(x)$ is called the *probability mass function* of χ.

Suppose that χ can take on the values x_1,x_2,x_3,x_4 or x_5 with probability p_1,p_2,p_3,p_4 and p_5 respectively. Clearly,

$$\sum_{i=1}^{5} p_i = 1$$

The following random variables are important for our studies.

Definition 1.1 Bernoulli variable. *A Bernoulli random variable χ takes on values of 0 and 1. The event $\{\chi = 1\}$ is called a* success *and occurs with probability p. The event $\{\chi = 0\}$ is called a* failure *and occurs with probability $q = 1 - p$.*

Suppose an experiment consists of spinning an unbiased coin. If we spin the coin n times we say that we have performed n *trials* and if the outcome is either 0 or 1, *true* or *false*, we refer to that as a *Bernoulli trial.*

Definition 1.2 Binomial variable. *The probability mass function of a binomial random variable χ that yields k successes in n independent Bernoulli trials is defined by*

$$p_\chi(k) = \left(\begin{array}{c} n \\ k \end{array} \right) p^k q^{n-k}$$

Definition 1.3 Geometric variable. *Suppose it took N Bernoulli trials to obtain the first success. The variable N is said to be geometric and its probability mass function is given by*

$$p_N(k) = q^{k-1} p \tag{1.5}$$

Another way of describing a random variable χ which takes values from an ordered set is to give a formula for the probability that it will take on values of x_i which are less than or equal to some value a. This leads to the following important definition.

Definition 1.4 *The* cumulative distribution function *of a random variable χ is the function*

$$F_\chi(a) = \sum_{x \leq a} p_\chi(x)$$

defined for all real variables a.

Another way of denoting the cumulative distribution often encountered in the literature is

$$F_\chi(x) = P[\chi \leq x]$$

Again, it should be evident that the values of the function F_χ are between 0 and 1. Using F_χ we can calculate

$$P[a < \chi \leq b] = F_\chi(b) - F_\chi(a) \tag{1.6}$$

This follows easily from the fact that

$$\{\chi \leq b\} = \{\chi \leq a\} \cup \{a < \chi \leq b\}$$

so that

$$F_\chi(b) = F_\chi(a) + P[a < \chi \leq b]$$

and the equation in (1.6) follows.

Exercise 1.3 *If the probabilities of a male or female offspring are both 0.5, find the probability of a family of five children being all male.*

Exercise 1.4 *A person has 18 bank-notes which includes 4 counterfeits in his purse. If he pays for an item with 2 bank-notes selected randomly from his purse, what are the probabilities that*

1. *both notes are genuine;*
2. *one of the notes is a counterfeit;*
3. *both notes are counterfeits?*

Exercise 1.5 *An* Ethernet *local network has k stations always ready to transmit. A station transmits successfully if no other station attempts to transmit at the same time as itself. If each station attempts to transmit with probability p, what is the probability that some station will be successful?*

1.3 Continuous Random Variables

In Sec.1.2 we obtained the distribution function of the discrete random variable by summing values of the mass function. When we now replace summation by integration, we obtain the notion of a *continuous random variable*.

A random variable χ is said to be continuous if there exists a nonnegative function $f_{\chi}(x)$ such that the cumulative distribution function $F_{\chi}(x)$ can be calculated from

$$F_{\chi}(a) = \int_{-\infty}^{a} f_{\chi}(x)dx \tag{1.7}$$

and frequently defined by the expression

$$F_{\chi}(a) = P[\chi \leq a]$$

The function $f_{\chi}(x)$ is called the *probability density function* of the random variable χ. Again, because we are concerned with probabilities, we must have the condition

$$\int_{-\infty}^{\infty} f_{\chi}(x)dx = 1 \tag{1.8}$$

Also, analogous to Eq. (1.6) we can calculate the probability that a random variable χ lies in the interval $(a{,}b)$ from

$$P[a < \chi < b] = \int_{a}^{b} f_{\chi}(x)dx \tag{1.9}$$

The density function $f_{\chi}(x)$ does not have to be continuous, but the distribution function $F_{\chi}(x)$ is automatically continuous. This implies

$$P[\chi = x] = 0 \tag{1.10}$$

for any value of x, so that the events

$$\{a \leq \chi < b\} \quad \{a < \chi \leq b\} \quad \{a < \chi < b\}$$

all have the same probability given by the integral in (1.9).

It should be clear that we can compute the density function from the distribution function from

$$f_{\chi}(x) = \frac{d}{dx}F_{\chi}(x) \tag{1.11}$$

The probability density function we will meet over and over again is the *negative exponential density function* given by

$$f_{\chi}(x) = \lambda e^{-\lambda x} \tag{1.12}$$

The constant λ is called the *parameter* of the distribution and the function is undefined for $x < 0$.
The corresponding cumulative distribution function is easily calculated to be

$$F\chi(a) = \int_0^a \lambda e^{-\lambda x} dx = 1 - e^{-\lambda a} \tag{1.13}$$

for $a \geq 0$, and $F(a) = 0$ if $a < 0$. Note also that $\lim_{a \to \infty} F(a) = 1$ as it should be, since it is certain that $0 \leq x < \infty$.

Exercise 1.6 *Find the probabilities that a random variable having an exponential distribution with parameter $\lambda = 10$ assumes a value between 0 and 3, a value greater than 5, and a value between 9 and 13.*

Exercise 1.7 *The life expectancy of a certain kind of lightbulb is a random variable with an exponential distribution and a mean life of 100 hours. Find the probability that the lightbulb will exceed its expected lifetime.*

1.4 Moments of a Random Variable

In most cases we are not interested in the specific distribution function of a random variable, but only in some characteristic values, the moments. The *mean* or *average value* of a real *positive* random variable $\chi(t)$ is used very often and it is more frequently referred to as the *expectation* of that variable. We write $E[\chi]$ or $\overline{\chi}$ for that value and it is defined by

$$E[\chi] = \int_0^\infty t f_\chi(t) dt$$

Note that we integrate only over the interval $t \in [0,\infty)$ since the independent variable will always be time in our discussions.
We will see later on that we will need to know the expectation of the *power* of a random variable as well. The expected value of the nth power of a random variable is referred to as its *nth moment*. Thus the more general nth moment (the *mean* is just the first moment) is given by

$$E[\chi^n] = \int_0^\infty t^n f_\chi(t) dt$$

Furthermore, the *nth central moment* of a random variable is defined to be

$$\overline{(\chi - \overline{\chi})^n} = \int_0^\infty (t - \overline{\chi})^n f_\chi(t) dt$$

The second central moment is used very often and is referred to as the *variance*, usually denoted by σ_χ^2 and defined as before by

$$
\begin{aligned}
\sigma_\chi^2 &= \overline{(\chi - \overline{\chi})^2} \\
&= \overline{\chi^2} - (\overline{\chi})^2
\end{aligned}
$$

The square root σ_χ of the variance is referred to as the *standard deviation*. The ratio of the standard deviation to the mean of a random variable is called the *coefficient of variation* denoted by

$$C_\chi = \frac{\sigma_\chi}{\overline{\chi}}$$

Exercise 1.8 *Referring to Ex. 1.5 (page 17), compute the mean number of collisions to be expected by any one of the k stations before a successful transmission.*

Exercise 1.9 *Compute the mean and coefficient of variation of a random variable which is exponentially distributed with parameter λ.*

1.5 Joint Distributions of Random Variables

Use the symbol $\mathbb{R}^n$ to denote the set of all n-tuples of real numbers. Let $\chi_1, \chi_2, \ldots, \chi_n$ be random variables. These random variables are said to have a joint *discrete* distribution if there exists a nonnegative function $p(x_1, x_2, \ldots, x_n)$ of n real variables that has the value 0 except at a countable set of points in $\mathbb{R}^n$, such that

$$P[\chi_1 = x_1, \chi_2 = x_2, \ldots, \chi_n = x_n] = p(x_1, x_2, \ldots, x_n)$$

for all points $(x_1, x_2, \ldots, x_n)$ in $\mathbb{R}^n$. Obviously we must have that

$$\sum_{x \in \mathbb{R}^n} p(x) = 1$$

Similarly we say that the collection $\chi_1, \chi_2, \ldots, \chi_n$ of random variables has a joint *continuous* distribution if there exists a nonnegative integrable function $f(x_1, x_2, \ldots, x_n)$ of n real variables that satisfies

$$P[\chi_1 \le a_1, \ldots, \chi_n \le a_n] = \int_{-\infty}^{a_1} \cdots \int_{-\infty}^{a_n} f(x_1, x_2, \ldots, x_n) dx_1 \ldots dx_n$$

for all choices of upper limits $a_1, \ldots, a_n$. The function f is called the *joint probability density function* of the random variables $\chi_1, \chi_2, \ldots, \chi_n$ and as in the discrete case, we must have

$$\int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} f(x_1, x_2, \ldots, x_n) dx_1 \ldots dx_n = 1$$

If we know the joint distribution f of $\chi_1,\chi_2,\ldots,\chi_n$, we can obtain the distribution of any one, say χ_m, of the random variables by simply integrating over all values of the remaining random variables. That is, $f_m(x) =$

$$\int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} f(x_1,\ldots,x_{m-1},x,x_{m+1},\ldots,x_n)dx_1 \ldots dx_{m-1}dx_{m+1} \ldots dx_n$$

is the probability density function of χ_m. The same holds true for the discrete case where we would sum, rather than integrate, over all possible values of the other variables.

1.6 Stochastic Processes

In the previous sections we frequently referred to a random variable χ taking on a value x. Nowhere did we make any mention of *time*, or, in other words, *when* χ took on *what* value or how that varied with time. Should we do that, we have what is known as a *stochastic process*. Mathematically then, a stochastic process is a family of random variables $\{\chi(t)\}$ defined over the same probability space. Put differently, the values (also called *states*) that members of the family $\chi(t)$ can take on all belong to the same set called the *state space* of $\chi(t)$.

Examples of stochastic processes are the number of persons on the beach as a function of the time of the day or the number of processes executing on a computer as a function of time. You will come to suspect already that if we can describe the latter mathematically we have made great progress at predicting the behaviour of the computer.

The classification of stochastic processes (some people also call them *random processes*) depends on three things: the *state space*; the nature of the *time parameter* and the *statistical dependencies* among the random variables $\chi(t)$ for different values of the time parameter.

Definition 1.5 *If the values* $\mathrm{x} = (x_1,x_2,\ldots)$ *in the state space of* $\chi(t)$ *are finite or countable, then we have a* discrete-state *process, also called a* chain. *The state space for a chain is usually the set of integers* $\{0,1,2,\ldots\}$. *If the permitted values in the state space may range over a finite or infinite* continuous interval, *then we say that we have a* continuous-state *process. The theory of continuous-state stochastic processes is not easy and we will only be considering discrete-state processes in this book.*

Definition 1.6 *If the times* $\mathrm{t} = (t_1,t_2,\ldots,t_n)$ *at which we observe the value of* $\chi(t)$ *are finite or countable, then we say that we have a* discrete-time *process; if these times may, however, occur anywhere within a set of finite intervals or an infinite interval of time, then we say that we have a* continuous-time *process. When time is discrete we write* χ_n *rather than* $\chi(t)$ *and refer to a stochastic* sequence *rather than a stochastic process.*

Definition 1.7 *Consider the joint distribution function (refer Sec. 1.5) of all the random variables* $X = \{\chi(t_1), \chi(t_2), \ldots\}$ *given by*

$$F_X(\mathrm{x};\mathrm{t}) = P[\chi(t_1) \leq x_1, \ldots, \chi(t_n) \leq x_n] \tag{1.14}$$

for all $\mathrm{x} = (x_1, x_2, \ldots, x_n)$, $\mathrm{t} = (t_1, t_2, \ldots, t_n)$ *and all* n. *Then the nature of* $F_X(\mathrm{x};\mathrm{t})$ *is the third quantity which determines the class of a stochastic process.*

In this book we will consider only the class of stochastic processes known as *Markov processes.*

2 Markov Processes

In 1907 a Russian mathematician, A.A. Markov, described a class of stochastic processes whose conditional probability density function is such that

$$\begin{aligned} P[\chi(t) = x|\chi(t_n) = x_n, \chi(t_{n-1}) = x_{n-1}, \ldots, \chi(t_0) = x_0] \\ = P[\chi(t) = x|\chi(t_n) = x_n], \quad t > t_n > t_{n-1} > \ldots > t_0 \end{aligned} \tag{2.1}$$

The above condition is known as the *Markov property*. A Markov process is a stochastic process $\{\chi(t), t \in T\}$ for which this property holds. We will assume that $T = [0,\infty)$ in our discussions and write $S = \{x_i = i;\ i \in \mathbb{N}_0\}$[1], the state space of the process.

The intuitive explanation of the Markov property is to say that the future of the process, from time t_n onwards, is determined only by the present state. However the process may have evolved to its present state $\chi(t_n)$, it does not influence the future. However, even if the history of the system up to the present state does influence the future behaviour, we may still be able to satisfy the Markov assumption by a change in the state structure. Let us assume, for example, that the next state depends on the last two states of our N state Markov chain (MC)[2]. Then we could define a new Markov process with N^2 states, where each state in the new process would consist of successive pairs of states in the old process. In this way the Markov property of Eq. (2.1) would still be satisfied, albeit at the expense of considerable increase in computational complexity. Any dependence of future behaviour on a finite number of historical steps can, at least in theory, be treated in the same way.

Definition 2.1 Homogeneous Markov Processes. *A Markov process* $\{\chi(t)\}$ *is said to be* homogeneous *or* stationary *if the following condition holds*

$$P[\chi(t+s) = x|\chi(t_n+s) = x_n] = P[\chi(t) = x|\chi(t_n) = x_n] \tag{2.2}$$

The equation expresses that a homogeneous Markov process is invariant to shifts in time.

Throughout our discussions we shall use as an example of a Markov process a surfer which goes from beach to beach in some random way as surfers tend to do. We shall describe the *state of this Markov process*, x_i, $i = 1,2,\ldots,N$ by the number i of the particular beach the surfer is on. In fact, for notational convenience, we shall use throughout the integer i to denote the state x_i of a Markov process.

[1] $\mathbb{N}$ denotes the set of positive integers and $\mathbb{N}_0$ additionally includes the 0.

[2] A Markov process with a discrete state space is also called a Markov chain.

In the case of a homogeneous Markov process, the particular instant t_n in Eq. (2.2) does not matter either so that the future of the process is completely determined by the knowledge of the present state. In other words,

$$p_{ij}(t-t_n) := P[\chi(t) = j|\chi(t_n) = i] \tag{2.3}$$

In fact, worse than that, an important implication is that the distribution of the *sojourn time* in any state must be memoryless. Our surfer does not know *how long* he has been at *this* beach! If you think about it, if the future evolution depends on the present state only, it cannot depend on the amount of time spend in the current state either.

When time is continuous, there is only one probability distribution $f_\chi(y)$ of the time y spent in a state which satisfies the property

$$P[\chi \geq y+s|\chi \geq s] = P[\chi \geq y]$$

and that is the negative exponential function

$$f_\chi(y) = \lambda e^{-\lambda y}, \;\; y \geq 0 \tag{2.4}$$

In other words, the sojourn times in a Continuous Time Markov Chain (CTMC) have an exponential probability distribution function. We will prove this fact in Sec. 2.3 on page 46. Not surprisingly, we will meet the exponential distribution many times in our discussions.

Similarly, for a Discrete Time Markov Chain (DTMC), the sojourn time η in a state must be a *geometrically distributed* random variable (cf. Eq. (1.5))

$$p_\eta(n) = P[\eta = n] = q^{n-1}(1-q), n = 1,2,3,\ldots; \;\; 0 \leq q < 1. \tag{2.5}$$

with cumulative distribution function $F_\eta(n)$

$$F_\eta(n) = \sum_{k=1}^{n} p_\eta(k)$$

Note that when a process has an interarrival time distribution given by $F_\eta(n)$ it is said to be a *Bernoulli arrival process*. Moreover, let $\eta = n\delta$ for n an integer and δ the basic unit of time. Then the mean time is given by

$$\delta \sum_{k=1}^{\infty} k p_\eta(k) = \frac{\delta}{(1-q)} \tag{2.6}$$

from which the mean arrival rate is $(1-q)/\delta$.

In order to decide whether a particular process is a Markov process, it suffices to check whether the distribution of sojourn times is either exponential or geometric and whether the probabilities of going from one state to another only depend on the state the process is leaving and on the destination state.

Exercise 2.1 *The weather bureau in an European country decided to improve its record for weather prediction. This is made a little easier by the fact there are never two sunny days in a row. If it is a sunny day however, the next day is just as likely to be rainy as it is likely to be just grey and dull. If it is not a sunny day, there is an even chance that the weather will be the same the next day. If there is a change from a rainy or dull day, there is only a 50 percent chance that the next day will be sunny.*

1. *Is the stochastic process we have just described Markovian?*
2. *If it is only approximately Markovian, what can one do to improve the approximation?*

2.1 Discrete Time Markov Chains

In this section we concern ourselves with the case where the time spent in a Markov state has a discrete distribution whence we have a Discrete Time Markov Chain (DTMC).

Definition 2.2 *The stochastic sequence* $\{\chi_n | n = 0,1,2,\ldots\}$ *is a* DTMC *provided that*

$$P[\chi_{n+1} = x_{n+1} | \chi_n = x_n, \chi_{n-1} = x_{n-1}, \ldots, \chi_0 = x_0] = \\ P[\chi_{n+1} = x_{n+1} | \chi_n = x_n] \tag{2.7}$$

for $n \in \mathbb{N}$.

The expression on the right-hand side of this equation is the *one-step transition probability* of the process and it denotes the probability that the process goes from state x_n to state x_{n+1} when the time (or index) parameter is increased from n to $n + 1$. That is, using the indices for notating the states,

$$p_{ij}(n,n+1) = P[\chi_{n+1} = j | \chi_n = i]$$

The more general form of the sth step transition probabilities is given by

$$p_{ij}(n,s) = P[\chi_s = j | \chi_n = i]$$

which gives the probability that the system will be in state j at step s, given that it was in state i at step n where $s \geq n$.
Note that the probabilities $p_{ij}(n,s)$ must satisfy the following requirements:

$$0 < p_{ij}(n,s) \leq 1, \;\; i,j = 1,2,\ldots,N; \;\; n,s = 0,1,2,\ldots$$

$$\sum_{j \in S} p_{ij}(n,s) = 1, \;\; i = 1,2,\ldots,N; \;\; n,s = 0,1,2,\ldots$$

The probability of going from state i to state j is the probability of somehow getting from i at time n to some intermediate state k at some time r and from there to state j. The events $\{\chi_r = k|\chi_n = i\}$ and $\{\chi_s = j|\chi_r = k\}$ are independent, so that using this and the fact that from the Markov property,

$$P[\chi_s = j|\chi_r = k,\chi_n = i] = P[\chi_s = j|\chi_r = k]$$

we can write recursively over all possible intermediate states k

$$\begin{aligned} p_{ij}(n,s) &= \sum_{k\in S} P[\chi_r = k|\chi_n = i]P[\chi_s = j|\chi_r = k] \\ &= \sum_k p_{ik}(n,r)p_{kj}(r,s) \end{aligned} \tag{2.8}$$

for $n \leq r \leq s$. Eq. (2.8) is known as the *Chapman-Kolmogorov equation* for DTMC.

If the DTMC is homogeneous (cf. Eq. (2.2)) which will be the case in all of our discussions, the probability of various states m steps into the future depends only upon m and not upon the current time n; so that we may simplify the notation and write

$$p_{ij}(m) = p_{ij}(n,n+m) = P[\chi_{n+m} = j|\chi_n = i]$$

for all $m \in \mathbb{N}$. From the Markov property we can establish the following recursive equation for calculating $p_{ij}(m)$

$$p_{ij}(m) = \sum_k p_{ik}(m-1)p_{kj}(1), \;\; m = 2,3,\ldots \tag{2.9}$$

We can write Eq. (2.9) in matrix form by defining matrix $P = [p_{ij}]$, where $p_{ij} := p_{ij}(1)$, so that

$$P^{(m)} = P^{(m-1)}P \tag{2.10}$$

where

$$P^{(0)} = I$$

the identity matrix. Note that

$$\begin{aligned} P^{(1)} &= P^{(0)}P = IP \\ P^{(2)} &= P^{(1)}P = P^2 \\ P^{(3)} &= P^{(2)}P = P^3 \end{aligned}$$

and in general

$$P^{(m)} = P^m, \;\; m = 0,1,2,\ldots \tag{2.11}$$

This equation enables us to compute the m-step transition probabilities from the one-step transition probabilities.

Next we consider a very important quantity, the probability $\pi_j^{(m)}$ of finding our DTMC in state j at the mth step:

$$\pi_j^{(m)} = P[\chi_m = j] \tag{2.12}$$

How can we calculate these probabilities?

If we write

$$p_{ij}^{(m)} = p_{ij}(m) = P[\chi_m = j | \chi_0 = i]$$

for the m-th step transition probability where we have assumed, without loss of generality, that we entered state i at time 0, then multiplying both sides of this equation by $\pi_i^{(0)} = P[\chi_0 = i]$ (cf. definition in Eq. (2.12)), summing over all states and applying theorem of Total Probability (cf. page 14), we obtain

$$\begin{aligned}
\sum_i P[\chi_0 = i] p_{ij}^{(m)} &= \sum_i P[\chi_0 = i] P[\chi_m = j | \chi_0 = i] \\
\sum_i \pi_i^{(0)} p_{ij}^{(m)} &= P[\chi_m = j] \\
&= \pi_j^{(m)}
\end{aligned}$$

or, alternatively

$$\pi_j^{(m)} = \sum_i \pi_i^{(0)} p_{ij}^{(m)} \tag{2.13}$$

That is, the state probabilities at time m can be determined by multiplying the multistep transition probabilities by the probability of starting in each of the states and summing over all states.

The row vector formed by the state probabilities at time m is called the state probability vector $\Pi^{(m)}$. That is,

$$\Pi^{(m)} = (\pi_0^{(m)}, \pi_1^{(m)}, \pi_2^{(m)}, \ldots)$$

With this definition, Eq. (2.13) can be written in matrix form as follows

$$\Pi^{(m)} = \Pi^{(0)} P^{(m)}, \ m = 0,1,2,\ldots$$

or from Eq. (2.11)

$$\Pi^{(m)} = \Pi^{(0)} P^m, \quad m = 0,1,2,\ldots \tag{2.14}$$

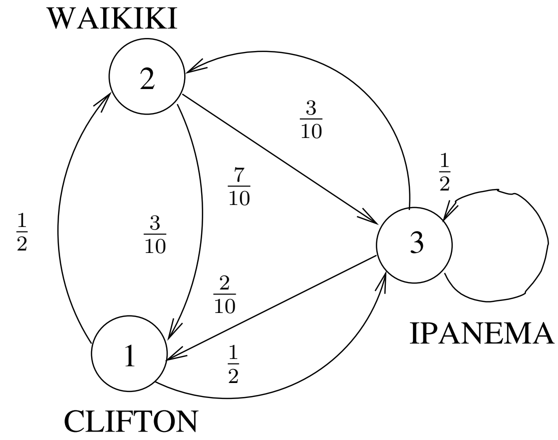


Figure 2.1 A Markov chain.

Example 2.1 *Consider the simple discrete time MC in Fig.2.1 which illustrates the behaviour of our surfer. This diagramme is also called the* state transition diagramme *of the DTMC. Every instant a unit of time elapses the surfer decides to do something. When at the Clifton, he decides to go to Waikiki with probability* $\frac{1}{2}$ *or may decide to go to Ipanema with the same probability (our surfer happens to be very affluent). When in Ipanema he may in fact decide to stay there with probability* $\frac{1}{2}$ *at the end of a time period. With our beaches numbered as shown, we have*

$$P = \begin{pmatrix} 0 & 0.5 & 0.5 \\ 0.3 & 0 & 0.7 \\ 0.2 & 0.3 & 0.5 \end{pmatrix}$$

Assume that our surfer starts off at Clifton (beach 1). In other words the initial distribution is $\Pi^{(0)} = (1{,}0{,}0)$*. From Clifton he can go to Ipanema or Waikiki with equal probability, i.e.,*

$$\Pi^{(1)} = (1{,}0{,}0)\begin{pmatrix} 0 & 0.5 & 0.5 \\ 0.3 & 0 & 0.7 \\ 0.2 & 0.3 & 0.5 \end{pmatrix} = (0{,}0.5{,}0.5)$$

from Eq. (2.14) and so on.

As we will see later, the vector $\Pi^{(m)}$ of state probabilities tends to a limit for $m \rightarrow \infty$. Even more, one can show that for specific DTMCs the effect of $\Pi^{(0)}$ on the vector $\Pi^{(m)}$ completely vanishes. For our surfer that means, e.g., even if we do not know at which beach he started the probability of finding him at a specific beach after a long time is nearly constant. This phenomenon does not hold for all DTMCs. Consider, e.g., the DTMC of Fig. 2.2. If the process starts in state 0 it stays there forever. But starting in state 3 there is a chance that the process gets absorbed in state 5. Clearly, the probability $\Pi^{(m)}$ is not independent of the initial

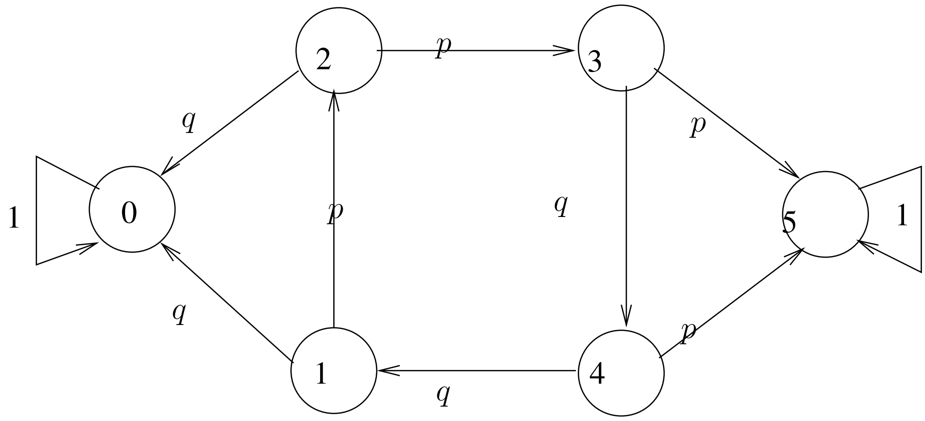


Figure 2.2 A simple Markov chain.

distribution. This effect or to be more precise the absence of such effects can be verified by investigating the structure of the state transition diagramme. E.g., from state 0 or 5 of the DTMC given in Fig. 2.2 no other state can be reached, thus intuitively explaining the described effect.
Next we consider a classification of Markov states based on the structure of the state transition diagramme.
Consider states $i,j \in S$. If there is a path from i to j, i.e., there exists an integer n (which may depend on i and j) such that

$$p_{ij}(n) > 0$$

then we write $i \rightharpoonup j$.
Two states i and j are said to *communicate*, written $i \rightleftharpoons j$, if there is a path from state i to state j and vice versa.
Let $C[i] = \{j | i \rightleftharpoons j;\;\; j \in S\}, \forall i \in S$. We call $C[i]$ the *class* of state i.

Example 2.2 *Consider the simple MC in Fig. 2.2. In that figure,* $C[0] = \{0\}, C[5] = \{5\}, C[1] = \{1,2,3,4\}$.

Definition 2.3 *A MC is said to be* irreducible *if every state communicates with every other state.*

An irreducible MC clearly has only one class of states, i.e. $C[i] = C[j]\ \forall i,j \in S$. The MC of Fig. 2.2 is *reducible* since $0 \rightleftharpoons 1$ is for instance not true.
Let C denote any class of state and $\overline{C}$ be the set of Markov states not in the class C.

Definition 2.4 *A class C is said to be* closed *if no single-step transition is possible from any state in C to any state in $\overline{C}$. If C consists of a single state, say i, then i is called an* absorbing state. *A necessary and sufficient condition for i to be an absorbing state is that* $p_{ii} = 1$.

Since the latter implies $p_{ij} = 0$ for $i \neq j$, an absorbing state does not communicate with any other state.
The MC of Fig. 2.2 has two absorbing states, 0 and 5.

Definition 2.5 *A class C is said to be* transient *if there is a path out of C. That is, if $\exists i \in C$ and $k \in \overline{C}$ such that $p_{ik} > 0$. The individual states in a transient class are themselves said to be* transient.

States 1, 2, 3 and 4 in the MC of Fig. 2.2 are all transient.

Definition 2.6 *A MC is said to be* absorbing *if every state in it is either absorbing or transient.*

Finally we define an ergodic class.

Definition 2.7 *A class C is said to be* ergodic *if every path which starts in C remains in C. That is*

$$\sum_{j \in C} p_{ij} = 1, \ \forall i \in C$$

The individual states in an ergodic class are called ergodic. An irreducible MC consists of a single ergodic class, i.e. $C[i] = S, \forall i \in S$.
Next write $f_j^{(m)}$ for the probability of a Markov process leaving a state j and *first* returning to the same state j in m steps. Clearly the probability of *ever returning* to state j is given by

$$f_j = \sum_{m=1}^{\infty} f_j^{(m)}$$

We now classify the states j of a MC depending on the value f_j of the state. Not surprisingly, if $f_j = 1$ we say the state is said to be *recurrent*; if a return is not certain, that is $f_j < 1$, then state j is said to be *transient*. Furthermore, if our MC can return to state j only at steps η,2η,$3\eta, \ldots$, where $\eta \geq 2$ is the largest such integer, then state j is said to be *periodic with period* η. If such an integer number η does not exist, then the state j is said to be *aperiodic*.
Knowing the probability $f_j^{(m)}$ of returning to state j in m steps, we can now define another interesting quantity, the *mean recurrence time* M_j of state j.

$$M_j = \sum_{m=1}^{\infty} m f_j^{(m)} \tag{2.15}$$

The mean recurrence time is thus the average number of steps needed to return to state j for the first time after leaving it.
We can further describe a state j to be *recurrent null* if $M_j = \infty$, whereas it is *recurrent nonnull* if $M_j < \infty$. Note that an irreducible MC can only have recurrent null states if the number of states is infinite.
With all this in mind, we can now state the following important result[108] without proof:

Theorem 2.1 *The states of an irreducible* DTMC *are all of the same type; thus they can be either*

- *all transient,*
- *all recurrent nonnull, or*
- *all recurrent null.*

Moreover, if periodic, then all states have the same period η.

Exercise 2.2 *Assume that we don't know for certain where our surfer has started. An oracle tells us that he might have started at Clifton with a chance of 19%, at Waikiki with 26% and at Ipanema, the beach he likes most, with a chance of 55%. What is our vector* $\pi^{(0)}$ *now? Calculate* $\pi^{(1)}$, $\pi^{(2)}$, $\pi^{(3)}$.

2.1.1 Steady State Distribution

The most interesting DTMCs for performance evaluation are those whose state probability distribution $\pi_j^{(m)}$ does not change when $m \to \infty$ or to put it differently, a probability distribution π_j defined on the DTMC states j is said to be *stationary* (or have reached a *steady state distribution)* if $\pi_j^{(m)} = \pi_j$ when $\pi_j^{(0)} = \pi_j$, that is, once a distribution π_j has been attained, it does not change in the future (with m).

Definition 2.8 *Define the* steady state probability distribution $\{\pi_j; j \in S\}$ *of a DTMC by*

$$\pi_j = \lim_{m \to \infty} \pi_j^{(m)}$$

We are after the steady state probability distribution $\{\pi_j\}$ of being in state j at some arbitrary point in the future. Clearly, if we know this, we can say a great deal about the system modelled by the MC. When the DTMC is irreducible, aperiodic and homogeneous the following theorem [108] helps us out.

Theorem 2.2 *In an irreducible and aperiodic homogeneous MC the limiting probabilities* π_j *always exist and are independent of the initial state probability distribution. Moreover, either*

1. *all states are transient or all states are recurrent null. In both cases* $\pi_j = 0 \; \forall \; j$ *and there exists* no *steady state distribution, or*
2. *all states are recurrent nonnull and then* $\pi_j > 0 \; \forall \; j$*, in which case the set* $\{\pi_j\}$ *is a steady state probability distribution and*

$$\pi_j = \frac{1}{M_j} \tag{2.16}$$

In this case the quantities π_j are uniquely *determined through the following equations*

$$\sum_i \pi_i = 1 \tag{2.17}$$

$$\sum_i \pi_i p_{ij} = \pi_j \tag{2.18}$$

where M_j is defined in Eq. (2.15).
A recurrent nonnull DTMC is also referred to as an *ergodic* MC and all the states in such a chain are ergodic.
From our definitions in the previous section, we know that a finite MC is ergodic if from any state it is possible to reach any other state and belong to a single class.
The limiting probabilities π_j of an ergodic DTMC are referred to as *equilibrium* or *steady state* probabilities. It should be clear that if we observe an ergodic MC for a fixed time T, that the average *sojourn time* $\overline{\tau}_i$ spent in state i by the DTMC during T can be computed from

$$\overline{\tau}_i = \pi_i T \tag{2.19}$$

Another quantity which will be useful to us is the average time υ_{ij} spent by the DTMC in state i between two successive visits to state j in steady state. This quantity is also known as the *visit ratio* or mean number of visits, and can be computed from

$$\upsilon_{ij} = \frac{\pi_i}{\pi_j} \tag{2.20}$$

Referring to Eq. (2.18), it is more convenient to find that equation expressed in matrix notation. In order to do this we define the probability vector Π as

$$\Pi = [\pi_0,\pi_1,\pi_2,\ldots]$$

so that we may now rewrite Eq. (2.18) as

$$\Pi = \Pi P \tag{2.21}$$

Note that Eq. (2.21) follows directly from the equation $\Pi^{(m)} = \Pi^{(m-1)}P$ by taking the limit as $m \to \infty$. The following example illustrates that no unique steady state distribution exists for a periodic MC.

Example 2.3 *Consider the periodic MC illustrated in Fig.2.3 and let* $\Pi^{(0)} = (1,0,0)$. *Then*

$$\begin{aligned}
\Pi^{(1)} &= \Pi^{(0)}P = (0,1,0)\\
\Pi^{(2)} &= \Pi^{(1)}P = (0,0,1)\\
\Pi^{(3)} &= \Pi^{(2)}P = (1,0,0)\\
\Pi^{(4)} &= \Pi^{(3)}P = (0,1,0)\\
&\ldots
\end{aligned}$$

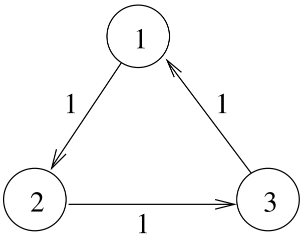


Figure 2.3 A simple periodic Markov chain.

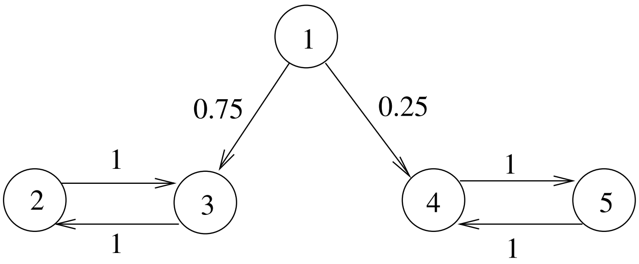


Figure 2.4 A simple reducible Markov chain.

Clearly the limit $\Pi = \lim_{m\to\infty} \Pi^{(m)}$ does not exist. Similarly the MC must be irreducible for a unique solution to exist as the following example illustrates.

Example 2.4 *Consider the reducible MC illustrated in Fig.2.4 and let* $\Pi^{(0)} = (1{,}0{,}0{,}0{,}0)$. *Then*

$$\Pi^{(1)} = \Pi^{(0)} \begin{pmatrix} 0 & 0 & 0.75 & 0.25 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix} = (0{,}0{,}0.75{,}0.25{,}0)$$

$$\begin{aligned} \Pi^{(2)} &= \Pi^{(1)}P = (0{,}0.75{,}0{,}0{,}0.25) \\ \Pi^{(3)} &= \Pi^{(2)}P = (0{,}0{,}0.75{,}0.25{,}0) \\ \Pi^{(4)} &= \Pi^{(3)}P = (0{,}0.75{,}0{,}0{,}0.25) \\ &\dots \end{aligned}$$

Again there is no limit $\Pi = \lim_{m\to\infty} \Pi^{(m)}$.
So far we have not said anything about the size of the state space of our DTMC. When the state space in our examples is finite we speak of a *finite* MC. The states of a finite aperiodic irreducible DTMC are always ergodic.
In the following example we determine the steady state distribution for our surfer example in Fig. 2.1.

Example 2.5 *Using Eq. (2.21) we can write the following set of linear equations:*

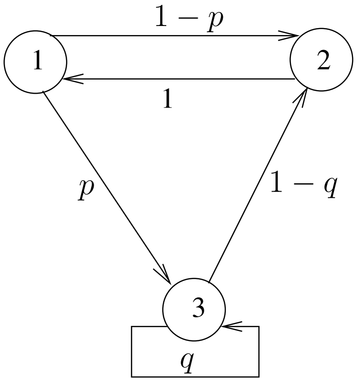


Figure 2.5 The homogeneous discrete time MC of the exercise

$$\begin{aligned} \pi_1 &= 0\pi_1 + 0.3\pi_2 + 0.2\pi_3 \\ \pi_2 &= 0.5\pi_1 + 0\pi_2 + 0.3\pi_3 \\ \pi_3 &= 0.5\pi_1 + 0.7\pi_2 + 0.5\pi_3 \end{aligned} \tag{2.22}$$

Note that in Eqs. (2.22) above, the last equation is a linear combination of the second and the first, indicating that there is a linear dependence among them. There will always be a linear dependence amongst the set of equations in Eq. (2.21) and it is the reason why we have to use the additional Eq. (2.17) to derive a solution. Using the latter equation and any two of the equations in (2.22) we obtain approximately

$$\begin{aligned} \pi_1 &= 0.188 \\ \pi_2 &= 0.260 \\ \pi_3 &= 0.552 \end{aligned}$$

So our surfer is most likely to be found on the beach at Ipanema (probability .552) and in fact he returns every $\frac{1}{0.552}$ or 1.81 days to that beach.

Exercise 2.3 *Consider the stochastic process described in Exercise 2.1. Let C,R and D represent a sunny, rainy and dull day respectively and in this way define a new stochastic process with 9 states. Determine the new transition probabilities. Consider this process to be a discrete time MC and find the probability of two dull days following upon one another.*

Exercise 2.4 *Consider the homogeneous MC illustrated in Fig. 2.5.*

1. *Give the probability matrix P for the chain.*
2. *Under what conditions will the chain be irreducible and aperiodic, if at all?*
3. *Solve for the steady state probability vector Π.*
4. *What is the mean recurrence time of state 3?*
5. *For what values of p and q will $\pi_1 = \pi_2 = \pi_3$?*

2.1.2 Absorbing Chains and Transient Behaviour

When using MCs to model real systems it is often very useful to know the number of steps (or, equivalently, the time) spent in the transient states before reaching an absorbing state. Think of executing a multi-layer network protocol: The time spent by processes executing the protocol in one layer (transient states) before going to the the next layer (absorbing state) is one example of such an application. The absorbing MC illustrated in Fig. 2.6 consisting of a set S_t of n_t transient states and a set S_a of n_a absorbing states, illustrates what we have in mind.

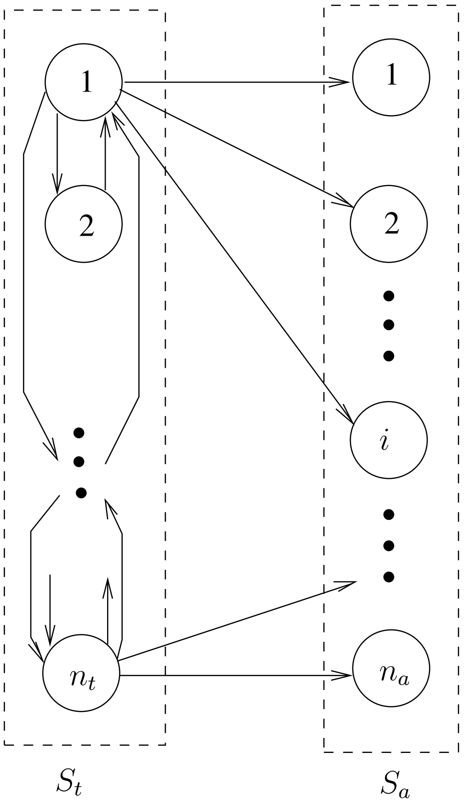


Figure 2.6 An absorbing MC.

We begin our analysis by numbering the states in the MC such that the n_a absorbing states occur first and writing the transition probability matrix P as

$$P = \begin{pmatrix} I & 0 \\ R & Q \end{pmatrix} \tag{2.23}$$

Once in an absorbing state the process remains there, so I is the identity matrix with all elements $p_{ii} = 1, \;\; 1 \le i \le n_a$. R is an $n_t \times n_a$ matrix describing the movement from the transient to the absorbing states, and Q is a $n_t \times n_t$

matrix describing the movement amongst transient states. Since it is not possible to move from the absorbing to the transient states 0 is the $n_a \times n_t$ zero matrix. Since the formula for matrix multiplication also applies to matrices written in block form, we can calculate the powers of P in terms of the matrices R and Q:

$$P^2 = \begin{pmatrix} I & 0 \\ R + QR & Q^2 \end{pmatrix}$$

and

$$P^3 = \begin{pmatrix} I & 0 \\ R + QR + Q^2R & Q^3 \end{pmatrix}$$

or in general

$$P^n = \begin{pmatrix} I & 0 \\ N_nR & Q^n \end{pmatrix}$$

where $N_n = I + Q + Q^2 + \ldots + Q^{n-1} = \sum_{i=1}^{n} Q^{i-1}$.
We can now state the following fundamental result for an *absorbing* MC:

Theorem 2.3 *When $n \to \infty$, then $Q^n \to 0$ and $N_n \to (I - Q)^{-1}$. In particular, the matrix I - Q is invertible.*

We will not prove the theorem (for a proof see [86]) but, from Eq. (2.14) above and the knowledge that for a transient state the steady state probability $\pi_j = 0$, the first part of the result is easy to accept intuitively.
Write

$$N = [n_{ij}] = (I - Q)^{-1}$$

N is called the fundamental matrix of the MC.
It follows from the last theorem that

$$\lim_{n \to \infty} P^n = \begin{pmatrix} I & 0 \\ NR & 0 \end{pmatrix}$$

For absorbing chains, the only interesting starting states are the transient ones. Assume that we start with an initial state $i \in S_t$. For each state $j \in S_t$, define the *random variable* υ_{ij} to be the number of visits to state j before an absorbing state is reached. Define $\upsilon_{ij} = 1$ when $i = j$.
We know from Th. 2.2 that $\upsilon_{ij} < \infty$ for any transient state j, and that υ_{ij} has finite expectation. Assuming these properties we can now prove the following theorem:

Theorem 2.4 *For every pair of transient states i,j*

$$E[\upsilon_{ij}] = n_{ij}$$

where $N = [n_{ij}]$ is the fundamental matrix as before.

Proof. Suppose that we move from starting state i to state k in the first step. If k is an absorbing state, we can never get to state j. If k is a transient state, we are in the same situation as before with starting state k instead. Using the Markov property,

$$E[\upsilon_{ij}] = \delta_{ij} + \sum_{k \in S_t} q_{ik} E[\upsilon_{kj}]$$

The term δ_{ij} is the Kronecker delta function with value 1 if $i = j$ and 0 otherwise and it counts the initial visit to state j in case the starting state is j. Denote by M the matrix whose i,j-entry is $E[\upsilon_{ij}]$ for all i,$j \in S_t$. Then the last equation can obviously be written

$$M = I + QM$$

so that $M = (I - Q)^{-1} = N$. □

Referring to Fig. 2.6, starting off in some state $i \in S_t$, the total number of steps (transitions) before reaching an absorbing state is clearly the sum of times we visit every state in S_t before absorption. Denote this random variable by υ_i and its expected value E$[\upsilon_i]$ by τ_i.

Theorem 2.5

$$\tau_i = \sum_{j \in S_t} n_{ij} \quad i \in S_t \tag{2.24}$$

and $\tau_i < \infty$.

Proof. Since the expectation of the sum is the sum of the expectations the latter result follows from the previous theorem. □

Write $\vec{\tau} = (\tau_1,\tau_2,\ldots,\tau_{n_t})$. Then

$$\vec{\tau} = N\, e^\top \tag{2.25}$$

where e is the row vector with 1's in every position.
What about the expected number of steps needed to reach an absorbing state, given that we will start in any of the transient states with probability distribution $\vec{r} = (r_1,r_2,\ldots,r_{n_t})$? This quantity we will denote by the scalar value τ and refer to it simply as *the* mean *time before absorption.*

Theorem 2.6 *Let* τ *be the mean time before absorption of a DTMC with transient state set* $S_t = \{1,2,\ldots,n_t\}$ *and initial probability distribution* $\vec{r} = (r_1,r_2,\ldots,r_{n_t})$. *Then*

$$\tau = \vec{r}\, N\, e^\top \tag{2.26}$$

Proof. Since r_i is the probability of a starting state $i \in S_t$, and τ_i the expected value of the number of steps to reach an absorbing state from that state i, the result follows. □

Furthermore, define σ_i^2 as the variance of the time υ_i before absorption starting in state $i \in S_t$ and $\vec{\sigma^2} = (\sigma_i^2, \ldots, \sigma_{n_t}^2)$ as the vector of the variance of these times. Let $\vec{\tau_2} = (\mathrm{E}[\upsilon_1^2], \ldots, \mathrm{E}[\upsilon_{n_t}^2])$. Then it can be shown (cf. [103], page 49) that

Theorem 2.7

$$\vec{\sigma^2} = (2N - I)\vec{\tau} - \vec{\tau_2}^2 \tag{2.27}$$

Proof. From Sec. 1.4 we know that

$$\sigma_i^2 = \mathrm{E}[\upsilon_i^2] - \tau_i^2 \tag{2.28}$$

Using the same argument as in Th. 2.4 we write for

$$\begin{aligned}
\mathrm{E}[\upsilon_i^2] &= \sum_{k \in S_a} q_{ik} \cdot 1 + \sum_{k \in S_t} q_{ik} \mathrm{E}[(\upsilon_i + 1)^2] \\
&= 1 + \sum_{k \in S_t} q_{ik} \mathrm{E}[\upsilon_i^2] + 2 \sum_{k \in S_t} q_{ik} \mathrm{E}[\upsilon_i] \\
&= 1 + \sum_{k \in S_t} q_{ik} \left(\mathrm{E}[\upsilon_i^2] + 2\mathrm{E}[\upsilon_i] \right)
\end{aligned}$$

Using vector and matrix notation, we can thus write

$$\mathrm{E}[\upsilon_i^2] = e^\top + Q \left(\mathrm{E}[\upsilon_i^2] + 2\vec{\tau} \right)$$

which reduces to

$$\begin{aligned}
\mathrm{E}[\upsilon_i^2] &= N(e^\top + 2Q\vec{\tau}) \\
&= \vec{\tau} + 2(N - I)\vec{\tau}
\end{aligned}$$

by substituting for NQ and $Ne^\top$ from Eq. (2.26). The result follows. □

In theory we therefore have the formulas in Eqs. (2.26) and (2.27) to compute the mean and variance respectively for the time before absorption. In practice, computing the matrix $N = (I - Q)^{-1}$ for a MC with a large state space is no mean task. Courtois and Semal[56] fortunately have devised a method of computing τ and σ^2 from P. We next describe their technique without proving the results. Proofs can be found in [103].

To start off, we define the augmented transition probability matrix

$$\begin{pmatrix} Q & (I - Q)e^\top \\ \vec{r} & 0 \end{pmatrix} \tag{2.29}$$

Note that $(I-Q)e^{\top}$ is the vector of transition probabilities from the states in S_t to a new state say, $a \in S_a$, and $\vec{r}$ and Q are the same as before.
The clever idea is that, assuming irreducibility and aperiodicity, the Markov process defined by the matrix in Eq. (2.29), has the same behaviour as a new process with the state a designated as an absorbing state provided one assumes that whenever the latter chain reaches the absorbing state a, it is restarted with the initial vector $\vec{r}$ and this is done infinitely many times. The new, absorbing MC is described by the matrix

$$\begin{pmatrix} Q & (I-Q)e^{\top} \\ 0 & 1 \end{pmatrix} \tag{2.30}$$

Again, the ergodic behaviour of the process described by Eq. (2.29) describes the behaviour of the absorbing chain of Eq. (2.30) over an infinite number of runs, each started with the initial distribution vector $\vec{r}$.

Theorem 2.8 *If τ is the mean time before absorption of the chain*

$$\begin{pmatrix} Q & (I-Q)e^{\top} \\ 0 & 1 \end{pmatrix}$$

when started with the initial distribution $\vec{r}$, then

$$\tau = \frac{1}{\pi_a} - 1,$$

where π_a is the last component of the steady state distribution of the DTMC described by the matrix

$$\begin{pmatrix} Q & (I-Q)e^{\top} \\ \vec{r} & 0 \end{pmatrix}$$

The proof of this theorem can be found in [56]. Intuitively, $1/\pi_a$ is the mean time between two visits to the last state a of the Markov process described by the matrix in Eq. (2.29) (cf. Th. 2.2) and each time the system is in this last state, one further step is needed to restart the absorbing chain with the initial distribution $\vec{r}$.
A similar result exists for the variance σ^2 of the time before absorption.

Theorem 2.9 *If σ^2 is the variance of the time before absorption of the chain*

$$\begin{pmatrix} Q & (I-Q)e^{\top} \\ 0 & 1 \end{pmatrix}$$

when started with the initial distribution $\vec{r}$, then

$$\sigma^2 = 2\tau\tau' - \tau - \tau^2$$

where τ is as before and τ' is given by

$$\tau' = \frac{1}{\pi'_a} - 1$$

and π'_a is the last component of the steady state distribution of the DTMC described by the matrix

$$\begin{pmatrix} Q & (I-Q)e^{\top} \\ \vec{r}' & 0 \end{pmatrix}$$

with

$$\vec{r}' = \frac{1}{1-\pi_a}(\pi_1,\pi_2,\ldots,\pi_{n_t})$$

where π_i is the steady state distribution of the MC given in Th. 2.8.
This concludes our study of discrete time MCs.

Exercise 2.5 *Two gamblers are betting on the outcome of an unlimited sequence of coin tosses. The first gambler always bets* heads, *which appears with probability p, $0 < p < 1$ on every toss. The second gambler always bets* tails, *which appears with probability $q = 1 - p$. They start with a total of C chips between them. Whenever one gambler wins he has to give the other one chip. The game stops when one gambler runs out of chips (is* ruined*). Assume the gamblers start with $C = 3$ chips between them.*

1. *Determine the probability, in terms of p and q, that a gambler is ruined?*
2. *How long will the game last if the first gambler starts with 1 coin?*

Exercise 2.6 *Write a program to simulate the game described in the previous exercise for a sufficient number of coin tosses. Use values of $p = 0.2,0.4,\ldots,0.8$. Compare your simulation results with the theoretical answers in the previous exercise. Assume $p = 0.6$.*

1. *Determine the theoretical mean time of a game. Compare your answer with your simulation results.*
2. *Determine the theoretical variance of the duration of a game. Compare your answer with your simulation results.*

2.2 Semi-Markov Processes

In our discussions thus far the Markov process had the property that a transition was made at every time instant. That transition may well return the process to the same state, but a transition occurred nevertheless.
We now turn our attention to a more general class of processes where the time *between transitions* may be several unit time intervals, and where this time can depend on the particular transition being made. This process is no longer strictly Markovian, however, it retains enough of the Markovian properties to deserve the name *semi-Markov* process[90].

2.2.1 Formal Model of a Semi-Markov Process

A semi-Markov process can be thought of as a process whose successive state occupancies are governed by the transition probabilities of a Markov process, but which spends time in any state described by an integer-valued random variable that depends on the state currently occupied and on the state to which the next transition will be made. At the transition instants, the semi-Markov process behaves just like a Markov process. We call this process the *embedded Markov process* and denote the single step state transitional probabilities by p_{ij}, $i,j = 1,2,\ldots,N$. For simplicity we assume that the process has a finite state space of size N.

The *times* at which transitions occur are now, however, governed by a different probability mechanism. Before making the transition from state i to the next state j, the process "sojourns"[3] for a time τ_{ij} in state i before proceeding to state j.

In our discrete time case, these sojourn times τ_{ij} with expected value $\overline{\tau_{ij}}$ are positive, integer-valued random variables, each governed by a probability mass function $s_{ij}(m),\ m = 1,2,\ldots$ called the sojourn time mass function for a transition from state i to state j. That is,

$$\begin{aligned} P[\tau_{ij} = m] &= s_{ij}(m),\ m = 1,2,3,\ldots;\ i,j = 1,2,\ldots,N \\ \overline{\tau_{ij}} &= \sum_{m=1}^{\infty} m s_{ij}(m) \end{aligned}$$

One distribution $s_{ij}(m)$ we are familiar with is the geometric distribution

$$(1-a)a^{m-1},\ m = 1,2,3,\ldots$$

We assume that the mean of the sojourn time is finite and that the sojourn times are at least one time unit in duration, i.e.,

$$s_{ij}(0) = 0$$

In other words, to fully describe a discrete-time semi-Markov process, we must specify N^2 holding time mass functions, in addition to the one step transition probabilities.

Observe that we can consider the discrete-time Markov process we discussed in the previous section to be the discrete-time semi-Markov process for which

$$s_{ij}(m) = \begin{cases} 1,\text{if m = 1,} \\ 0,\text{if m = 2,}\ldots \end{cases} \quad \forall\ i,j = 1,2,\ldots,N$$

that is, all sojourn times are exactly one time unit in length.

We next define the *waiting time* τ_i with expected value $\overline{\tau_i}$ as the time spent in state i, $i = 1,2,\ldots,N$ irrespective of the successor state and we define the probability mass function of this waiting time as

[3] Some authors refer to this time as the "holding time".

$$\begin{aligned} P[\tau_i = m] &= \sum_{j=1}^{N} p_{ij} s_{ij}(m) \\ \overline{\tau_i} &= \sum_{j=1}^{N} p_{ij} \overline{\tau_{ij}} \end{aligned}$$

That is, the probability that the system will spend m time units in state i if we do not know its successor state, is the probability that it will spend m time units in state i if its successor state is j, multiplied by by the probability that its successor state will indeed be j and summed over all possible successor states.

2.2.2 Interval Transition Probabilities

As in the DTMC case, we next set out to compute the $n-$step transition probabilities, which we denote $\phi_{ij}(n)$, for the semi-Markov case. That is, how can a process that started by entering state i at time 0 be in state j at time n?

One way this can happen is for i and j to be the same state and for the process never to have left state i throughout the period $(0,n)$. This requires that the process makes its first transition, to *any* other state, only *after* time n. That is

$$\delta_{ij} P[\tau_i > n]$$

where

$$\delta_{ij} = \begin{cases} 1, \text{if i = j}, \\ 0, \text{otherwise} \end{cases}$$

It is not difficult to see that the latter term can be written as

$$\delta_{ij} \left(1 - \sum_{m=0}^{n} \sum_{j=1}^{N} p_{ij} s_{ij}(m) \right) \tag{2.31}$$

Let $W(n) = \{\delta_{ij} \left(1 - \sum_{m=0}^{n} \sum_{j=1}^{N} p_{ij} s_{ij}(m)\right)\}$ be the matrix of these elements.

Every other way to get from state i to j in the interval $(0,n)$ would mean the process made its first transition from state i to some other state k at a time m, and then by a succession of such transitions to state j at time n. Note that we have to consider all intermediate times $m,\ 0 < m \leq n$ and intermediate states $k \in S$, S the Markov state space. In other words

$$\sum_{m=0}^{n} \sum_{k=1}^{N} p_{ik} P[\tau_{ik} = m] \phi_{kj}(n-m) \tag{2.32}$$

where $i,j = 1,2,\ldots,N;\ \ n = 0,1,2,\ldots$ or

$$\sum_{m=0}^{n}\sum_{k=1}^{N} p_{ik}s_{ik}(m)\phi_{kj}(n-m) \tag{2.33}$$

Define the congruent matrix multiplication $C = A \circ B$ of two matrices $A = \{a_{ij}\}$ and $B = \{b_{ij}\}$ by $C = \{a_{ij}b_{ij}\}$, for all i,j. Furthermore, write

$$\begin{aligned}\Phi(n) &= \{\phi_{ij}(n)\}\\ S(n) &= \{1 - \sum_{m=0}^{n} s_{ij}(m)\}\end{aligned}$$

With these definitions, Eq. (2.33) becomes

$$\sum_{m=0}^{n}(P \circ S(m))\Phi(n-m) \quad n = 0,1,2,\ldots$$

and

$$\Phi(n) = W(n) + \sum_{m=0}^{n}(P \circ S(m))\Phi(n-m) \quad n = 0,1,2,\ldots \tag{2.34}$$

$\Phi(n)$ is called the interval transition probability matrix for the semi-Markov process in the interval $(0,n)$ and clearly

$$\Phi(0) = I$$

Eq. (2.34) provides a convenient recursive basis for computing $\Phi(n)$ for any semi-Markov process. The quantities P and $S(m)$ come directly from the definition of the process.

Example 2.6 *Consider again our typical surfer of Fig. 2.1 on page 28, but let us now give the example a semi-Markovian flavour. The nature of the surfer now dictates that the length of time he will stay on a particular beach will depend on both which beach he is on and where he intends to go next. The (sojourn) time τ_{ij} is thus the length of time spent surfing on beach i with the intention of going to beach j (where $j = i$ is certainly possible). The lifeguards on each beach have been keeping record of our surfer and have come up with the following probability mass functions describing the surfer's behaviour:*

$$\begin{aligned}
s_{11}(m) &= \left(\frac{1}{3}\right)\left(\frac{2}{3}\right)^{m-1} \quad s_{12}(m) = \left(\frac{1}{4}\right)\left(\frac{3}{4}\right)^{m-1} \quad s_{13}(m) = \left(\frac{1}{3}\right)\left(\frac{2}{3}\right)^{m-1}\\
s_{21}(m) &= \left(\frac{1}{2}\right)\left(\frac{1}{2}\right)^{m-1} \quad s_{22}(m) = \left(\frac{1}{8}\right)\left(\frac{7}{8}\right)^{m-1} \quad s_{23}(m) = \left(\frac{2}{5}\right)\left(\frac{3}{5}\right)^{m-1}\\
s_{31}(m) &= \left(\frac{1}{4}\right)\left(\frac{3}{4}\right)^{m-1} \quad s_{23}(m) = \left(\frac{1}{3}\right)\left(\frac{2}{3}\right)^{m-1} \quad s_{33}(m) = \left(\frac{1}{2}\right)\left(\frac{1}{2}\right)^{m-1}
\end{aligned}$$

for $m = 1,2,3,\ldots$

Solution. *Consider the general geometric distribution function (cf. Sec. 1.2) $p(n) = (1-a)a^{(n-1)}$. The first moment or mean $\overline{n}$ can be calculated in the following way:*

$$\overline{n} = \sum_{n=0}^{\infty} n(1-a)a^{(n-1)}$$

Using the property that $\sum \frac{d}{da} = \frac{d}{da} \sum$ we write

$$\begin{aligned}
\overline{n} &= (1-a)\sum_{n=0}^{\infty} \frac{d}{da} a^n \\
&= (1-a)\frac{d}{da}\frac{1}{1-a} \\
&= \frac{1}{1-a}
\end{aligned}$$

so that the first moment $\overline{\tau}_{ij}$ in the example is given by

$$\begin{aligned}
\overline{\tau}_{11} &= 3; \quad \overline{\tau}_{12} = 4; \quad \overline{\tau}_{13} = 3; \\
\overline{\tau}_{21} &= 2; \quad \overline{\tau}_{22} = 8; \quad \overline{\tau}_{23} = 2.5; \\
\overline{\tau}_{31} &= 4; \quad \overline{\tau}_{32} = 3; \quad \overline{\tau}_{33} = 2;
\end{aligned}$$

Clearly beach 2 (Waikiki) is the most popular with our surfer, since he spends 8 units of time on the average surfing there and then immediately returning to it. The mean time he spends on that beach, irrespective of where he might go next is given by

$$\begin{aligned}
\overline{\tau}_2 &= p_{21}\overline{\tau}_{21} + p_{22}\overline{\tau}_{22} + p_{23}\overline{\tau}_{23} \\
&= 0.3 \times 2 + 0 \times 8 + 0.7 \times 2.5 \\
&= 2.35
\end{aligned}$$

Exercise 2.7 *In Ex. 2.6, compute the mean time the surfer spends on Ipanema, assuming that he has arrived there from anywhere else but Waikiki.*

2.2.3 Steady State Behaviour

We now set out to find the limiting behaviour of the interval transition probabilities over long intervals. It is important to note that the MC structure of a semi-Markov process is the same as that of its embedded Markov process. Therefore the interval transition probabilities of a semi-Markov process can exhibit a unique limiting behaviour only within the chain of the embedded Markov process.

We begin by defining a limiting interval transition probability matrix Φ for our process by

$$\Phi = \lim_{n\to\infty} \Phi(n)$$

with elements ϕ_{ij}. However, in steady state, the limiting interval transition probabilities ϕ_{ij} do not depend on the starting state i and we therefore write $\phi_{ij} = \phi_j$. Define a vector $\varphi = (\phi_0, \phi_1, \ldots, \phi_N)$ as the vector of probabilities ϕ_j that the semi-Markov process is in state j as time $n \to \infty$ and let $\Pi = \{\pi_0, \pi_1, \ldots, \pi_N\}$ be the steady state probability vector of the equivalent embedded MC [cf. Eq. (2.21)]. One can prove (see e.g., Howard[90]) that

$$\phi_j = \frac{\pi_j \overline{\tau}_j}{\sum_{i=1}^N \pi_i \overline{\tau}_i} \tag{2.35}$$

or

$$\varphi = \frac{1}{\overline{\tau}} \Pi M$$

where we have written

$$\overline{\tau} = \sum_{j=1}^N \pi_j \overline{\tau}_j$$

and M is the diagonal matrix $[\overline{\tau}_j]$ of mean waiting times.
Although the proof requires transform analyses which are beyond the scope of this book, Eq. (2.35) is intuitively somewhat obvious in the following way: The probability ϕ_j of finding the semi-Markov process in state j is the product of the average time $\overline{\tau}_j$ spent in that state, multiplied by the probability π_j of being in that state in the embedded MC and normalised to the mean total time $\overline{\tau}$ spent in all of the N possible states.
Note that Eq. (2.35) can also be written as

$$\begin{aligned}\phi_j &= \frac{\overline{\tau}_j}{\sum_{i=1}^N \frac{\pi_i}{\pi_j} \overline{\tau}_i} \\ &= \frac{\overline{\tau}_j}{\sum_{i=1}^N \upsilon_{ij} \overline{\tau}_i}\end{aligned}$$

where υ_{ij}, given by Eq. (2.20), is the visit ratio defined on page 32.

Example 2.7 *Suppose we want to know the steady state probability of finding our surfer on Waikiki beach. This we can do by applying Eq. (2.35). In Ex. 2.5 we determined that*

$$\Pi = (0.188, 0.260, 0.552)$$

so that

$$\begin{aligned}\phi_2 &= \frac{\pi_2 \overline{\tau}_2}{\pi_1 \overline{\tau}_1 + \pi_2 \overline{\tau}_2 + \pi_3 \overline{\tau}_3} \\ &= \frac{0.260 \times 2.35}{0.188 \times 3.5 + 0.260 \times 2.35 + 0.552 \times 2.7} \\ &= 0.22\end{aligned}$$

There is thus a 22 percent chance of finding him on Waikiki or beach number 2.

Exercise 2.8 *A car rental company has determined that when a car is rented in* Town 1 *there is a 0.8 probability that it will be returned to the same town and a 0.2 probability that it will be returned to* Town 2. *When rented in* Town 2, *there is a 0.7 probability that the car will be returned to* Town 2, *otherwise it is returned to* Town 1. *From its records, the company determined that the rental period probability mass functions are:*

$$\begin{aligned} s_{11}(m) &= \left(\frac{1}{3}\right)\left(\frac{2}{3}\right)^{m-1} \quad s_{12}(m) = \left(\frac{1}{6}\right)\left(\frac{5}{6}\right)^{m-1} \quad m = 1,2,3,\ldots \\ s_{21}(m) &= \left(\frac{1}{4}\right)\left(\frac{3}{4}\right)^{m-1} \quad s_{22}(m) = \left(\frac{1}{12}\right)\left(\frac{11}{12}\right)^{m-1} \quad m = 1,2,3,\ldots \end{aligned}$$

What percentage of the time does a car spend in Town 2?

2.3 Continuous Time Markov Chains

In Sec. 2.1 we described how our surfer had to decide at regular, equal intervals of time whether to leave or whether to stay on the beach where he is. If we now allow him to decide at *an arbitrary time* which beach to go to next, we have the continuous-time version of that example.
The Continuous Time Markov Chain (CTMC) version of the Markov property, Eq. (2.1), is given by

Definition 2.9 *The stochastic process* $\{\chi(t),t \geq 0\}$ *is a CTMC provided*

$$\begin{aligned} P[\chi(t) = x|\chi(t_n) = x_n,\chi(t_{n-1}) = x_{n-1},\ldots,\chi(t_0) = x_0] \\ = P[\chi(t) = x|\chi(t_n) = x_n] \end{aligned} \tag{2.36}$$

for any sequence $t_0,t_1,\ldots,t_n$ *such that* $t_0 < t_1 < \ldots < t_n$ *and* $x_k \in S$ *where* S *is the (discrete)* state space *of the process.*

The right-hand side of the above equation is the transition probability of the CTMC and we write

$$p_{ij}(t,s) = P[\chi(s) = x_j|\chi(t) = x_i]$$

to identify the probability that the process will be in state x_j at time s, given that it is in state x_i at time $t \leq s$. Since we are still considering discrete state Markov processes (chains) we will continue to use $i \in \mathbb{N}$ rather than x_i to denote a state of our Markov processes.
Note that we need to *define*

$$p_{ij}(t,t) = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{otherwise} \end{cases} \tag{2.37}$$

to establish the fact that the process may not leave immediately to another state. We already mentioned in Sec. 2 on page 24 that the time a Markov process spends in any state has to be memoryless. In the case of a DTMC this means that the chain must have geometrically distributed state sojourn times while a CTMC must have exponentially distributed sojourn times. This is such an important property that we include a proof from Kleinrock[108] of it here. The proof is also instructive in itself.
Let y_i be a random variable which describes the time spent in state i. The Markov property specifies that we may not remember how long we have been in state i which means that the *remaining sojourn time* in i may only depend upon i. Assume that this remaining time t has distribution $h(t)$. Then our Markov property insists that

$$P[y_i > s + t | y_i > s] = h(t)$$

i.e. $h(t)$ is independent of s. Using conditional probabilities, we may write

$$\begin{aligned} P[y_i > s + t | y_i > s] &= \frac{P[y_i > s + t \cap y_i > s]}{P[y_i > s]} \\ &= \frac{P[y_i > s + t]}{[y_i > s]} \end{aligned}$$

where the last step follows from the fact that the event $y_i > s + t$ implies that $y_i > s$. Rewriting the last equation we obtain

$$P[y_i > s + t] = P[y_i > s]h(t) \tag{2.38}$$

Setting $s = 0$ and since we know $P[y_i > 0] = 1$ we have

$$P[y_i > t] = h(t)$$

which we then substitute in Eq. (2.38) to obtain the relation

$$P[y_i > s + t] = P[y_i > s]P[y_i > t] \tag{2.39}$$

for all s,$t \geq 0$. All that remains is to show that the *only* continuous distribution function which satisfies Eq. (2.39) is the negative exponential distribution. Write $f_i(t)$ for the corresponding density function. Then we can write

$$\begin{aligned} \frac{d}{dt} P[y_i > t] &= \frac{d}{dt}(1 - P[y_i \leq t]) \\ &= -f_i(t) \end{aligned} \tag{2.40}$$

Using the latter result and differentiating Eq. (2.39) with respect to s we obtain

$$\frac{dP[y_i > s + t]}{ds} = -f_i(s)P[y_i > t]$$

Dividing both sides by $P[y_i > t]$ and setting $s = 0$ we obtain the differential equation

$$\frac{dP[y_i > t]}{P[y_i > t]} = -f_i(0)ds$$

which by using (2.40) gives the following density function

$$f_i(t) = f_i(0)e^{-f_i(0)t}$$

for all $t \geq 0$. Setting $\lambda = f_i(0)$ we are back to $f_i(x) = \lambda e^{-\lambda x}$ which we had before in Eq. (2.4) on page 24. □

In other words, if a stochastic process has the Markovian property, the time spent in a state will have a negative exponential distribution. This may seem rather restrictive since many real systems do not have exponential time distributions.

In 1955 Cox [57] showed however, that a random variable χ whose distribution $F_\chi(x)$ has a rational Laplace transform can be represented as a series-parallel network of stages as illustrated in Fig. 2.7 where the time distribution of each stage has a different negative exponential distribution. In that diagramme stage j, $j = 2,\ldots,k$ follows stage $j-1$ with probability α_j or does not follow with probability $1-\alpha_j$.

The first and second moments of this distribution are given by [102]

$$E[\chi] = \sum_{i=1}^{k} \delta_i \sum_{j=1}^{i} \frac{1}{\mu_j}$$

$$E[\chi^2] = \sum_{i=1}^{k} \delta_i \left(\sum_{j=1}^{i} \frac{1}{\mu_j^2} + \left(\sum_{j=1}^{i} \frac{1}{\mu_j} \right)^2 \right)$$

where

$$\delta_i = (1-\alpha_i) \prod_{j=1}^{i-1} \alpha_j$$

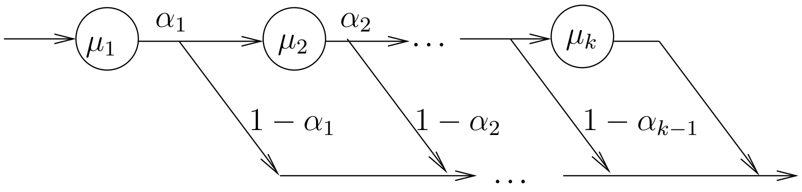


Figure 2.7 The Coxian stage-type distribution.

If we now encode the stage number in which the process finds itself in the state descriptor of our stochastic process, it is clear that the process will again be Markovian.

In fact one can approximate almost any distribution as closely as one wishes. The drawback is that the size of the corresponding state space increases by a factor k.

The obvious goal of CTMC analysis is to find the probability that the process will be in state i at a given time t or alternatively, its steady state distribution. In the continuous time case the solution is different from that for discrete time in Th. 2.2, although the arguments are analogous. Let us now find that solution. First of all define the following time-dependent transition probability

$$p_{ij}(s,t) = P[\chi(t) = j|\chi(s) = i]$$

where the process is in state j at time t given that it was in state i at time s; $s \leq t$. Using the same arguments as in Sec. 2.1 on page 25 that the process must pass at some intermediate time u through some intermediate state k in order to get from state i to state j we can arrive at the Chapman-Kolmogorov equation for *continuous-time* MCs:

$$p_{ij}(s,t) = \sum_k p_{ik}(s,u)p_{kj}(u,t) \tag{2.41}$$

where $i,j = 0,1,2,\ldots$ Using the notation

$$H(s,t) = (p_{ij}(s,t))$$

we can write Eq. (2.41) as

$$H(s,t) = H(s,u)H(u,t), \quad s \leq u \leq t$$

where from Eq. (2.37)

$$H(t,t) = I$$

for I the identity matrix.

In Sec. 2.1 the solution Π we wanted was given by $\Pi = \Pi P$ with P the one-step transition probability matrix. For the CTMC the one-step transition probabilities are replaced by the *infinitesimal rates* which are in turn given in terms of the time derivative of $p_{ij}(s,t)$ as $t \rightarrow s$.

In order to determine how this comes about we first of all write from Eq. (2.41)

$$p_{ij}(s,t+\Delta t) = \sum_k p_{ik}(s,t)p_{kj}(t,t+\Delta t)$$

Subtracting $p_{ij}(s,t)$ from both sides and dividing by Δt we obtain

$$\frac{p_{ij}(s,t+\Delta t) - p_{ij}(s,t)}{\Delta t} = \sum_k p_{ik}(s,t)\left(\frac{p_{kj}(t,t+\Delta t) - p_{kj}(t,t)}{\Delta t}\right)$$

since $p_{ij}(s,t) = \sum_k p_{ik}(s,t)p_{kj}(t,t)$. $\Delta t \rightarrow 0$ yields the following alternate form of the *forward* Chapman-Kolmogorov equations for continuous-time MCs

$$\frac{\partial H(s,t)}{\partial t} = H(s,t)Q(t), \quad s \leq t \tag{2.42}$$

where we have written the matrix $Q(t)$ for

$$Q(t) = \lim_{\Delta t \to 0} \frac{H(t,t+\Delta t) - I}{\Delta t}$$

whose elements are defined as follows

$$\begin{aligned} q_{ii}(t) &= \lim_{\Delta t \to 0} \frac{p_{ii}(t,t+\Delta t) - 1}{\Delta t} \\ q_{ij}(t) &= \lim_{\Delta t \to 0} \frac{p_{ij}(t,t+\Delta t)}{\Delta t}, \quad i \neq j \end{aligned}$$

The intuitive interpretation of the quantities $q_{ii}(t)$ and $q_{ij}(t)$ is that when our process is in state i at time t, then the probability that a transition occurs to any state other than i is given by $-q_{ii}(t)\Delta t + o(\Delta t)$. Thus, $-q_{ii}(t)$ is the *rate* at which the process leaves state i and similarly the probability of a transition to state j in time interval Δt is $q_{ij}(t)\Delta t + o(\Delta t)$ or rate $q_{ij}(t)$. Since we know that $\sum_j p_{ij}(s,t) = 1$ it follows that

$$\sum_j q_{ij}(t) = 0, \quad \forall\, i \tag{2.43}$$

Using the same arguments as above we could also derive the *backward* Chapman-Kolmogorov equations

$$\frac{\partial H(s,t)}{\partial s} = -Q(s)H(s,t), \quad s \leq t$$

Note that in terms of the individual matrix elements we can write the *forward* Chapman-Kolmogorov equations as

$$\frac{\partial p_{ij}(s,t)}{\partial t} = q_{jj}(t)p_{ij}(s,t) + \sum_{k \neq j} q_{kj}(t)p_{ik}(s,t) \tag{2.44}$$

The *backward* Chapman-Kolmogorov equations can be written as

$$\frac{\partial p_{ij}(s,t)}{\partial s} = -q_{ii}(s)p_{ij}(s,t) - \sum_{k \neq i} q_{ik}(s)p_{kj}(s,t) \tag{2.45}$$

From these equations we can solve for $H(s,t)$, but what we really need to know is the probability $\pi_j(t)$ that the Markov process will be in state j at time t. Define the vector of state probabilities

$$\Pi(t) = (\pi_0(t),\pi_1(t),\pi_2(t),\ldots)$$

If we are given the initial state distribution $\Pi(0)$ then we can solve for the time-dependent state probabilities in a way similar to that in Sec. 2.1.1 from

$$\Pi(t) = \Pi(0)H(0,t)$$

Differentiating both sides of the last equation we obtain after applying Eq. (2.42)

$$\begin{aligned} \frac{d\Pi(t)}{dt} &= \Pi(0)\frac{dH(0,t)}{dt} \\ &= \Pi(0)H(0,t)Q(t) \\ &= \Pi(t)Q(t) \qquad (2.46) \end{aligned}$$

Writing the latter matrix equation in terms of its elements, we obtain

$$\frac{d\pi_j(t)}{dt} = q_{jj}(t)\pi_j(t) + \sum_{k\neq j} q_{kj}(t)\pi_k(t) \qquad (2.47)$$

Note the similarity between Eqs. (2.44) and (2.47). The former equation describes the probability that the process is in state j at time t given that it was in state i at time s.

We now know that we can solve for the *time-dependent* probability distributions $\Pi(t)$ from Eq. (2.47). The explicit solution of this system of differential equations is given by

$$\Pi(t) = \Pi(0)e^{\int_0^t Q(u)du}$$

which is difficult to solve in most interesting cases and numerical methods have to be used. As in the case of the DTMC we therefore fall back on the steady state distribution when $t \rightarrow \infty$ while remembering that there may indeed be cases where the time-dependent solution is very useful.

Before concluding this section we would like to note that concepts such as reducibility, transient and absorbing states and chains which we discussed for DTMCs apply to CTMCs as well and we will not repeat those discussions here.

2.3.1 Steady State Distribution

Consider the case where our CTMC is homogeneous. If so, we can drop the dependence on time and adopt the following notation

$$\begin{aligned} p_{ij}(t) &= p_{ij}(s,s+t) \\ q_{ij} &= q_{ij}(t), \quad i,j = 1,2,\ldots \\ H(t) &= H(s,s+t) = (p_{ij}(t)) \\ Q &= Q(t) = (q_{ij}) \end{aligned}$$

The forward and backward Chapman-Kolmogorov equations now become, respectively,

$$\frac{dp_{ij}(t)}{dt} = q_{jj}p_{ij}(t) + \sum_{k \neq j} q_{kj}p_{ik}(t) \tag{2.48}$$

and

$$-\frac{dp_{ij}(t)}{dt} = -q_{ii}p_{ij}(t) - \sum_{k \neq i} q_{ik}p_{kj}(t)$$

or for the state probabilities themselves

$$\frac{d\pi_j(t)}{dt} = q_{jj}\pi_j(t) + \sum_{k \neq j} q_{kj}\pi_k(t) \tag{2.49}$$

When our CTMC chain is irreducible, homogeneous and all states are recurrent nonnull, it can be shown (cf. [108]) that the following limit exists and is independent of the initial state distribution. That is,

$$\lim_{t \to \infty} \pi_j(t) = \pi_j$$

The vector $\Pi = (\pi_0, \pi_1, \pi_2, \ldots)$ is the steady state probability distribution we are after. As was the case for a DTMC, the MC is said to be *ergodic*.

This limiting distribution is given uniquely by the solution of the following system of linear equations

$$\begin{aligned} -q_{jj}\pi_j + \sum_{k \neq j} q_{kj}\pi_k &= 0 \\ \sum_j \pi_j &= 1 \end{aligned} \tag{2.50}$$

where the latter equation again follows from the rules for probability conservation. In matrix form Eq. (2.50) may be expressed as

$$\Pi Q = 0 \tag{2.51}$$

This latter equation is the CTMC equivalent of the very important Eq. (2.21) on page 32 for the DTMC case. In the latter case P was the matrix of *transition probabilities*, whereas the so-called *infinitesimal generator* Q is a matrix of *transition rates*.

Eq. (2.50) are also known as the *global balance* equations of the MC. If in addition the following property holds, namely

$$q_{ij}\pi_i = q_{ji}\pi_j \ \ \forall i,j \tag{2.52}$$

then the MC is said to satisfy *local balance* and the solution to Eq. (2.52) is also a solution to Eq. (2.50). Note that the reverse is not necessarily true.

Exercise 2.9 *A computing center has only one technician and 3 computers. Each computer randomly breaks down at an exponential rate of twice a day, and the technician fixes them at an exponential rate of 3 computers per day. Whenever all three computers are broken, the users help the technician and 4 computers are fixed per day. Answer the following:*

1. *On the average how many computers are out of order?*
2. *At what mean rate are the computers breaking down?*

2.4 Embedded Markov Chains

In the case of a continuous time Markov process it is again possible to have an embedded DTMC and arrive at a solution for the steady state probability distribution in the way described in Sec. 2.2. The steady state transition probabilities

$$p_{ij} = P[\chi_{n+1} = j|\chi_n = i]$$

can be calculated as a function of the transition rates q_{ij}. In the case of a CTMC the sojourn times $\tau_{ij};\ i,j = 1,2,\dots,N$ (cf. page 48) are all exponentially distributed with parameter q_{ij}.
That is,

$$\begin{aligned} P[\tau_{ij} \le x] &= 1 - e^{-q_{ij}x} && (2.53) \\ \text{or}\quad P[\tau_{ij} > x] &= e^{-q_{ij}x} && (2.54) \end{aligned}$$

The probability p_{ij} is the same as the probability that τ_{ij} is the smallest of the sojourn times $\tau_{ik},\ k \neq i$. Denote this greatest lower bound by τ_{glb}. Then

$$\begin{aligned} P[\tau_{glb} \le x] &= P[\bigcup_{\substack{k\neq i\\k\neq j}} \{\tau_{ik} \le x\}] \\ &= 1 - P[\bigcap_{\substack{k\neq i\\k\neq j}} \{\tau_{ik} > x\}] \\ \text{or}\ \ F_{\tau_{glb}}(x) &= 1 - e^{-\sum_{\substack{k\neq i\\k\neq j}} q_{ik}x} \end{aligned}$$

from Eq. (2.54). The corresponding density function $f_{\tau_{glb}}(x)$ is thus

$$f_{\tau_{glb}}(x) = -\sum_{\substack{k\neq i\\k\neq j}} q_{ik} e^{-\sum_{\substack{k\neq i\\k\neq j}} q_{ik}x}$$

However,

$$\begin{aligned} -q_{ii} &= \sum_{k\neq i} q_{ik} \\ \text{or}\quad -(q_{ii} + q_{ij}) &= -\sum_{\substack{k\neq i\\k\neq j}} q_{ik} \end{aligned}$$

so that

$$f_{\tau_{glb}}(x) = -(q_{ii} + q_{ij})e^{(q_{ii}+q_{ij})x}$$

It follows that

$$\begin{aligned} p_{ij} &= \int_0^\infty f_{\tau_{glb}}(x) P[\tau_{ij} \le x] dx \\ &= -\int_0^\infty (1 - e^{-q_{ij}x})(q_{ii} + q_{ij}) e^{(q_{ii}+q_{ij})x} dx \\ &= \frac{q_{ij}}{-q_{ii}} \end{aligned}$$

The steady state probabilities π are then computed in a manner similar to that described in Sec. 2.2.
This completes our discussion of discrete-state Markov processes.

Exercise 2.10 *Consider the car rental exercise of Ex. 2.8 except that now you are given that cars with destination* Town 2 *are rented at the average rate of 2 per day in* Town 1 *and 14 per day in* Town 2*. Cars destined for* Town 1 *are rented at the average rate of 6 per day in* Town 2 *and 8 per day in* Town 1*. Again calculate the percentage of the time a car spends in* Town 2?

3 General Queueing Systems

With the basic theory of stochastic processes behind us we can now proceed to apply that theory to model the performance of systems. Think about a typical computer system, be it a multiprocessor, a computer network or a database system. In all cases we have entities (processes, packets, transactions) requiring service from some resource or set of resources (central processors, nodal processor, file server). In doing so, these entities usually have to wait their turn before being served in some order. Even humans are regrettably only too familiar with the world we have just described: it is known as queueing and a vast body of theory known as *queueing theory* has developed around the subject over the years.

What then is the link between queueing theory and stochastic processes? As you know from experience it is not always possible to predict the length of the waiting line at the post office or bank. We know that the length of that line is a stochastic process as is, for instance, the number of packets in the input buffer at a network node at any time. It would thus make sense to have an understanding of basic queueing theory in our study of performance modelling.

We focus our attention on the general queueing system illustrated in Fig. 3.1. In queueing theory terminology the entities requiring service are called *customers* and they are said to wait in a *waiting line* to be served by one or more *servers* using some *scheduling strategy* such as First-Come-First-Served (FCFS). The queue may have a finite capacity while the customers in turn may come from a finite population.

In order to describe and analyse a queue we tag each customer with a subscript n and denote the customer itself by C_n. So C_n is the nth customer to enter the (queueing) system. As suggested already, we define the random process $N(t)$ as

$$N(t) := \text{number of customers in the system at time t.}$$

Another stochastic process is the *unfinished work* $U(t)$ that exists in the system at time t. When $U(t) > 0$, then the system is said to be *busy*, and when $U(t) = 0$

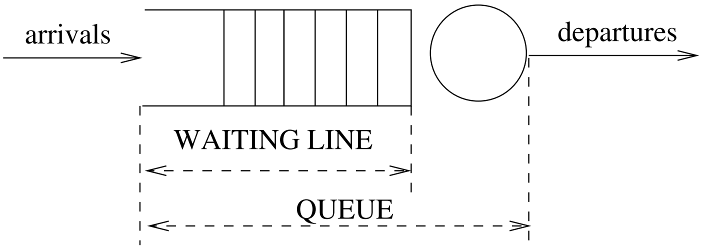


Figure 3.1 A queue.

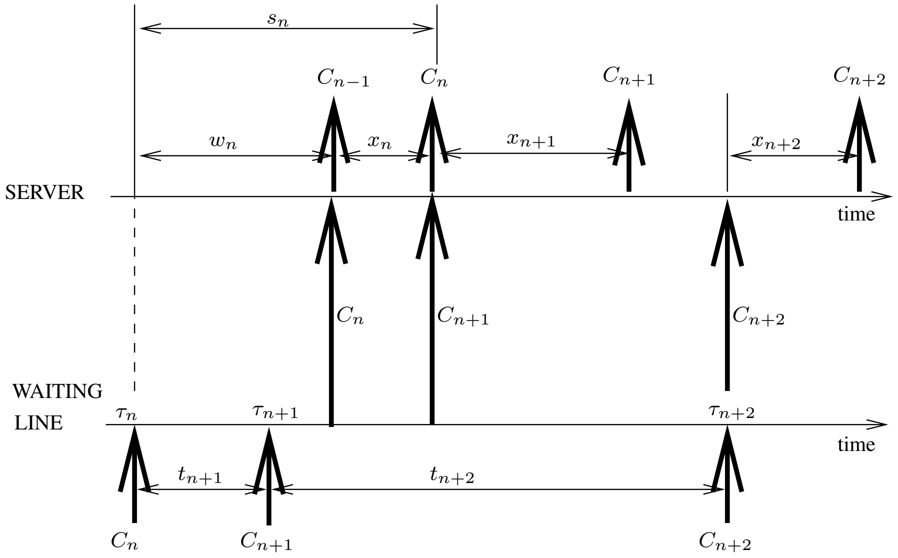


Figure 3.2 Time sequence diagramme describing the queueing process.

it is said to be *idle*. These stochastic variables are displayed in the time sequence diagramme in Fig. 3.2 where a customer C_n,$C_{n+1}, \ldots$ arrives at the queue, waits for a time w_n before proceeding to the server as soon as the previous customer C_{n-1},$C_n, \ldots$ has completed his service (FCFS service) and eventually leaves after a service time of duration x_n.
More formally then, we define the arrival time of customer C_n to the queueing system as the instant

$$\tau_n := \text{arrival time for } C_n$$

and define the time between the arrivals of C_{n-1} and C_n (the *interarrival time*) as

$$\begin{aligned} t_n &:= \text{interarrival time between } C_{n-1} \text{ and } C_n \\ &= \tau_n - \tau_{n-1} \end{aligned}$$

We assume moreover that the interarrival times are drawn from a continuous distribution $A(t)$, so that

$$P[t_n \leq t] = A(t)$$

independent of n. The average interarrival time is a quantity of special significance in queueing theory and it is given the symbol $1/\lambda$. That is

$$\frac{1}{\lambda} = \int_0^\infty t dA(t).$$

Note that the average *arrival rate* is given by λ. Similarly, we define the service time for C_n as

$$x_n \quad := \quad \text{service time for } C_n$$

and write $B(x)$ for the distribution of that time. That is

$$P[x_n \leq x] = B(x)$$

and write μ for the average *service rate* with

$$\frac{1}{\mu} = \int_0^\infty x dB(x)$$

for the average service time which are also quantities we will come across very frequently.

The sequences $\{t_n\}$ and $\{x_n\}$ are the independent variables of the queueing system. The quantity of most interest to us is the *waiting time* which we define as

$$w_n \quad := \quad \text{time taken by } C_n \text{ waiting in line.}$$

The total time spent in the system by C_n, which we will call the *queueing time*[1] is thus given by

$$\begin{aligned} s_n \quad &:= \quad \text{waiting time plus service time for } C_n \\ &= \quad w_n + x_n \qquad (3.1) \end{aligned}$$

Thus we have defined the interarrival time, the waiting time, the service time and the queueing time for our customer C_n. Before concluding this section we would like to introduce a very convenient shorthand notation that is in common use for describing a queue. The reader will suspect already that the mathematical notion of a queue is described by the arrival distribution $A(t)$ and the service distribution $B(x)$. Other determining attributes are the *number m of servers* and the system capacity K, the total number of customers, including those in service, the queue can accommodate. Another attribute is the scheduling strategy at the server while it may also be necessary to specify the size of the customer population if this is not infinite.

The shorthand way of describing the abovementioned attributes of a queue, known as Kendall's notation, is to write *A/B/m/K/Z/Sched* to identify the arrival process/service process/number of servers/ maximum capacity/user population/scheduling strategy of a queue. In the description of the arrival or service process the following conventions are used:

[1] In this book, the terms "waiting time" and "queueing time" will refer to the *time waiting in line* and the *total time spent waiting in line* **and** *being served* respectively. The meaning of these is not consistent throughout the queueing theory literature and the wise reader will make sure he knows which meaning the particular author uses.

G: general distribution

M: exponential distribution

C_k: k-phase Coxian distribution

D: deterministic distribution

Thus the queueing system *M/G/1///FCFS* has exponentially distributed interarrival times, a general service time distribution, a single server, infinite capacity, infinite user population and uses the FCFS scheduling strategy. Usually if a field is empty the extra slashes are left out, e.g. *M/G/1/FCFS*.

Exercise 3.1 *Think of any freeway in your city. Can that be considered as a queueing system? What would the customers be? And the server(s)?*

Exercise 3.2 *Give the shorthand notation for a queue with fixed interarrival times and a 4-phase Coxian type service time distribution where the capacity of the queue is K.*

3.1 Little's Law

Most of us would suspect that the faster customers arrive at a queue and the heavier their service requirements, the longer they will spend in the queueing system. This is indeed the case and the rule which describes this relationship is known as *Little's Law* (or theorem). A theoretical proof of this law can be found in the original work of Little [115] and several other papers [68, 100, 170, 180]. We will use operational (i.e., "how it works") arguments to derive the same result.
Suppose we observe the queueing process described by Fig. 3.2 for an interval of time $(0,t)$, at the beginning of which there are no customers and count the number $a(t)$ of customers which arrive during this time. That is

$$a(t) := \text{number of arrivals in } (0,t)$$

Simultaneously we count the number $d(t)$ of departures during the same interval. Thus

$$d(t) := \text{number of departures in } (0,t)$$

A typical behaviour for each of these two functions is illustrated in Fig. 3.3.
The number $N(t)$ of customers in the system at time t is then clearly

$$N(t) = a(t) - d(t)$$

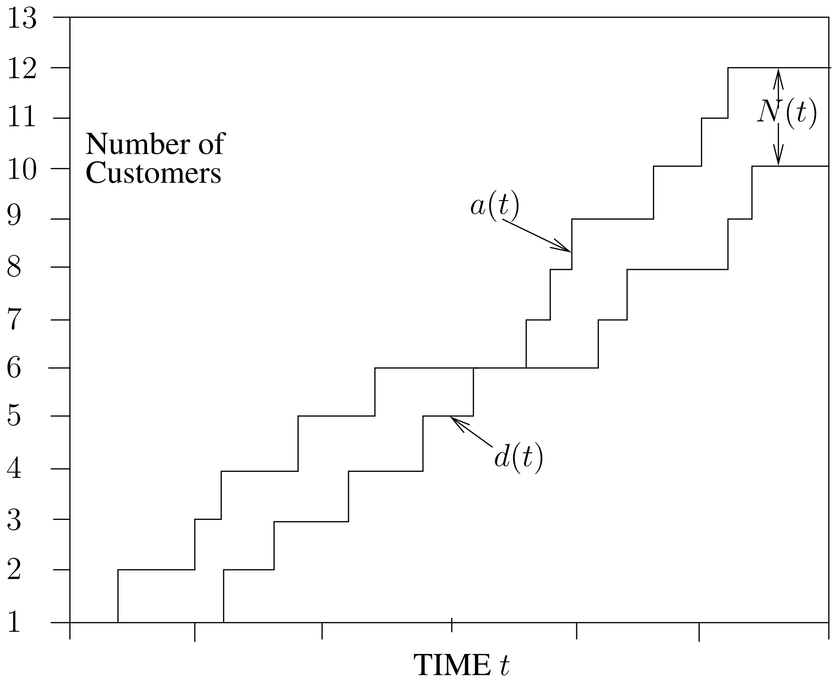


Figure 3.3 Arrival and departure functions at a typical queue.

The area from 0 to time t between the curves $a(t)$ and $d(t)$ in Fig. 3.3 is the integral over time of the quantity $N(t)$ and obviously represents the time spent by all the customers in the system (measured in units of customer-seconds) up to point t. Denote this area by $I(t)$. Moreover let λ_t be defined as the average arrival rate of customers during the interval $(0,t)$; that is,

$$\lambda_t := \frac{a(t)}{t}$$

If we define T_t as the average queueing time (remember, waiting time plus service time) per customer during the interval $(0,t)$, then clearly

$$T_t = \frac{I(t)}{a(t)}$$

Finally define N_t as the average number of customers in the system during the interval $(0,t)$, which can obviously be computed from

$$\begin{aligned} N_t &= \frac{I(t)}{t} \\ &= \frac{a(t)}{t}\frac{I(t)}{a(t)} \\ &= \lambda_t T_t \end{aligned}$$

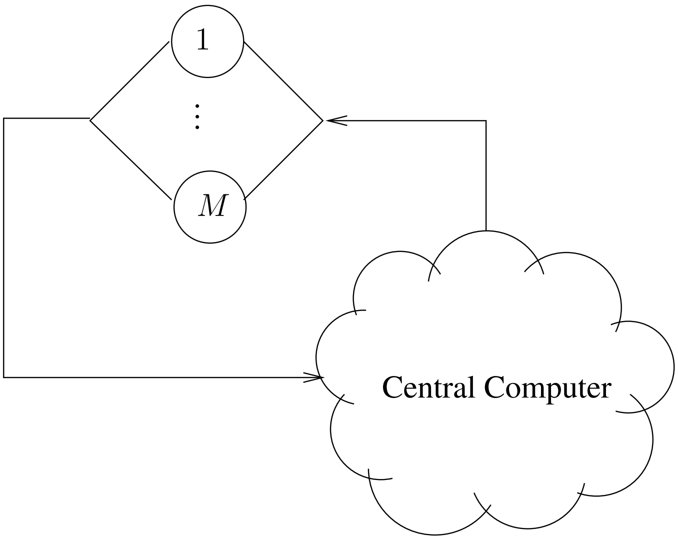


Figure 3.4 Simplified computer system used in the exercise

Assuming that the limits

$$\begin{aligned}\lambda &= \lim_{t\to\infty} \lambda_t \\ T &= \lim_{t\to\infty} T_t\end{aligned}$$

exist, then so will the limit N for N_t the average number of customers in the system. The mean arrival rate λ we have seen before and T is merely the average value of the time s_n in the system defined in Eq. (3.1) on page 57. This leads us to the following important result

Theorem 3.1 Little's Law. *The average number N of customers in a queueing system with arbitrary service and arrival distribution, with average arrival rate λ and average time in the system T, is given by*

$$N = \lambda T \tag{3.2}$$

Note that in the derivation described earlier in this section we made no assumptions about the distributions of the service and interarrival times nor did we specify the service discipline or number of servers. *Little's law thus applies regardless* and can be applied to *any* system with average arrival rate λ and average time T in the system. Moreover, if we know any two of the three unknowns in the equation we can always compute the third.
There are also extensions to Little's law dealing with the calculation of moments of the random variables [46, 116, 120].

Exercise 3.3 *Consider the computer system in Fig. 3.4. Assume that the think time at each of the M terminals has a mean of 15 seconds and that, on the average, half of them are busy "thinking". What is the delay in the central computer when there are 10 terminals in use?*

3.2 Birth-Death Processes

A very useful class of Markov chain is the *birth-death* process. Birth-death processes may be either discrete- or continuous-time processes in which the defining condition is that state transitions take place between state k and its *nearest neighbours* $k-1$,k and $k+1$ only. In fact, as the name indicates, the birth-death process is appropriate for modelling changes in the size of a population. When the population is of size k we will say that the system is in state k. Moreover a transition from state k to $k+1$ will signify a "birth" within the population, whereas a transition from k to $k-1$ will denote a "death" in the population.[2]
Although the time parameter may be discrete as well (as the state space) for the birth-death process, the continuous-time case is much more interesting and the one we will analyse. We begin by introducing the notion of a birth rate λ_k, which describes the rate at which births occur when the population is of size k, and similarly we define a death rate μ_k which is the rate at which deaths occur when the population is of size k. Note that these rates depend only on k and are independent of time so that our Markov chain is thus homogeneous. In terms of our notation in the previous chapter, we have

$$q_{kj} = \begin{cases} \lambda_k, & \text{if } j = k+1 \\ \mu_k, & \text{if } j = k-1 \\ -(\lambda_k + \mu_k), & \text{if } j = k \\ 0, & \text{otherwise} \end{cases}$$

Note that the fact that $q_{kk} = -(\lambda_k + \mu_k)$ above, follows directly from Eq. (2.43) on page 50. Note the assumptions that $\lambda_k > 0$ for all k, that $\mu_0 = 0$ (no deaths are possible in a population of size 0), and that $\mu_k > 0$ for all $k > 0$.
Using the notation developed in the previous chapter we can write the infinitesimal generator Q for the general homogeneous birth-death process as

$$Q = \begin{pmatrix} -\lambda_0 & \lambda_0 & 0 & 0 & 0 & \\ \mu_1 & -(\lambda_1+\mu_1) & \lambda_1 & 0 & 0 & \\ 0 & \mu_2 & -(\lambda_2+\mu_2) & \lambda_2 & 0 & \\ 0 & 0 & \mu_3 & -(\lambda_3+\mu_3) & \lambda_3 & \dots \\ & & & \vdots & & \end{pmatrix}$$

Note that the sum of all elements in each row of the matrix Q is zero as it always will be. Compare this with the sum of all row elements of the transition probability matrix P in Sec. 2.1 which will always be 1.
The state probabilities $\pi_k(t),\ k = 0,1,\dots$ at time t can thus be found by solving the set of differential equations equivalent to Eq. (2.47)

[2] The astute reader will already suspect that this process resembles a queue where a "birth" signifies an arrival to and a "death" a departure from the queue. That this is indeed the case for a queue with very specific interarrival time and service time distributions (the M/M/1 queue) will be discussed at the end of this chapter.

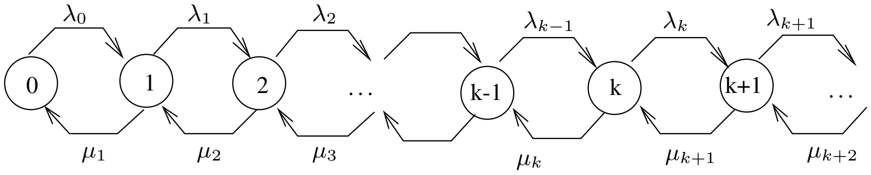


Figure 3.5 State-transition diagramme of the birth-death process.

$$\frac{d\pi_k(t)}{dt} = -(\lambda_k+\mu_k)\pi_k(t) + \lambda_{k-1}\pi_{k-1}(t) + \mu_{k+1}\pi_{k+1}(t), k \geq 1 \quad (3.3)$$

$$\frac{d\pi_0(t)}{dt} = -\lambda_0\pi_0(t) + \mu_1\pi_1(t) \quad (3.4)$$

These equations can be solved for the general transient case, but that is not easy. Instead, we consider the steady state probabilities $\pi_k,\ k = 0,1,\ldots$. In that case the above equations are a particular case of Eq. (2.51) and can be written

$$-(\lambda_k+\mu_k)\pi_k + \lambda_{k-1}\pi_{k-1} + \mu_{k+1}\pi_{k+1} = 0, \quad k \geq 1 \quad (3.5)$$

$$-\lambda_0\pi_0 + \mu_1\pi_1 = 0 \quad (3.6)$$

Eqs. (3.5) and (3.6) are known as the *global balance equations.* Before proceeding to solve these equations we digress to Fig. 3.5 which illustrates the state transition diagramme of the birth-death process. Concentrating on state k we observe that one may enter it only from the state $k-1$ or from the state $k+1$ and similarly one leaves state k only to enter state $k-1$ or state $k+1$. From that diagramme it is clear why we refer to the process we described as the nearest-neighbour, birth-death process.

The clever thing is to note that we can obtain Eqs. (3.5) – (3.6) directly from the state transition diagramme in Fig. 3.5 by equating the *rates of flow* into and out of each state $k,\ k = 0,1,\ldots$ Indeed, the flow into state k comes from state $k-1$ (with probability π_{k-1}) at rate λ_{k-1}, and from state $k+1$ at rate μ_{k+1}. The total flow into state k is the sum of the rates weighted by the probabilities of the origin states, or

$$\text{flow into state } k = \lambda_{k-1}\pi_{k-1} + \mu_{k+1}\pi_{k+1}$$

The flow out of state k is the product of the probability of state k and the sum of the rates out of state k. That is,

$$\text{flow out of state } k = (\lambda_k+\mu_k)\pi_k$$

Using the fact that under steady state conditions the flow out of state k equals the flow into state k and Eqs. (3.5) – (3.6) follow. We can solve the latter equations from the fact that

$$\pi_1 = \frac{\lambda_0}{\mu_1}\pi_0$$

and recursively from there on

$$\pi_k = \pi_0 \prod_{j=0}^{k-1} \frac{\lambda_j}{\mu_{j+1}} \tag{3.7}$$

for all $k = 1,2,\ldots$ Imposing the normalising condition $\sum_k \pi_k = 1$ we finally obtain

$$\pi_0 = \left(1 + \sum_{k\neq 0} \prod_{j=0}^{k-1} \frac{\lambda_j}{\mu_{j+1}}\right)^{-1}$$

Substituting into Eq. (3.7) we obtain

$$\pi_k = \left(\prod_{i=0}^{k-1} \frac{\lambda_i}{\mu_{i+1}}\right) \left(1 + \sum_{k\neq 0} \prod_{j=0}^{k-1} \frac{\lambda_j}{\mu_{j+1}}\right)^{-1} \tag{3.8}$$

Note that for this limiting distribution to exist we must have

$$\sum_{k\neq 0} \prod_{j=0}^{k-1} \frac{\lambda_j}{\mu_{j+1}} < \infty$$

and we call the latter the *condition of ergodicity* (i.e., that a steady state solution exists) for the birth-death process.

Exercise 3.4 *Consider a birth-death system with the following birth and death coefficients*

$$\begin{aligned} \lambda_k &= (k+2)\lambda \quad k = 0,1,2,\ldots \\ \mu_k &= k\mu \quad k = 1,2,3,\ldots \end{aligned}$$

1. *Find π_k and give the condition of ergodicity.*
2. *Determine the average number of customers in the system.*

3.3 Poisson Process

We mentioned in the last section that to find the transient solution for Eqs. (3.3) – (3.4) is not very easy in general. In fact, more than that, the birth-death process in the general case is not all that interesting either. A special case which is, however, of central importance in queueing theory, is that for a *pure birth* process in which we assume that $\mu_k = 0$ for all k. Moreover, to simplify the problem even further, we will assume that $\lambda_k = \lambda$ for all $k = 0,1,2,\ldots$ Substituting this into Eqs. (3.3) – (3.4) we have

$$\frac{d\pi_k(t)}{dt} = -\lambda\pi_k(t) + \lambda\pi_{k-1}(t), \quad k \geq 1 \tag{3.9}$$

$$\frac{d\pi_0(t)}{dt} = -\lambda\pi_0(t) \tag{3.10}$$

Assuming that the process starts out from state 0 at time $t = 0$, or

$$\pi_k(0) = \begin{cases} 1, & k = 0 \\ 0, & k \neq 0 \end{cases}$$

then solving for $\pi_0(t)$ from Eq. (3.10) we have

$$\pi_0(t) = e^{-\lambda t}$$

Inserting this result into Eq. (3.9) for $k = 1$ results in

$$\frac{d\pi_1(t)}{dt} = -\lambda\pi_1(t) + \lambda e^{-\lambda t}$$

with solution

$$\pi_1(t) = \lambda t e^{-\lambda t}$$

Solving recursively, we finally obtain as a solution to Eqs. (3.9)-(3.10):

$$\pi_k(t) = \frac{(\lambda t)^k}{k!} e^{-\lambda t}; \quad k \geq 0, t \geq 0 \tag{3.11}$$

The latter is the well-known *Poisson* distribution which is central to our studies of queueing theory. Eq. (3.11) gives the probability of k arrivals (births) in the interval $(0,t)$ when the average arrival rate is λ. The Poisson process is important because it resembles or models many real-life processes very well. So, for instance, it was observed that the number of telephone calls arriving from subscribers at a central exchange is modelled very well by the Poisson process. The same is true for packets arriving at a node in a packet switching computer network.

There is another equally important reason, however, why the Poisson process is so popular: Consider the random variable $\tilde{t}$ which represents the time between the arrivals of a process which has a Poisson arrival distribution. As we did before, we denote the probability distribution and probability density functions of these times by $A(t)$ and $a(t)$ respectively. From the definition $a(t)\Delta t + o(\Delta t)$ is the probability that the next arrival occurs at least t and at most $(t + \Delta t)$ seconds after the event of the last arrival. Furthermore,

$$\begin{aligned} A(t) &= P[\tilde{t} \leq t] \\ &= 1 - P[\tilde{t} > t] \end{aligned}$$

But $P[\tilde{t} > t]$ is the probability that there was no arrival in the interval $(0,t)$, that is $\pi_0(t)$ for the Poisson distribution. Therefore, we have

$$A(t) \quad = \quad 1 - \pi_0(t)$$

and so from Eq. (3.11) we obtain, for the Poisson distribution, that the interarrival time distribution is described by

$$A(t) \quad = \quad 1 - e^{-\lambda t}, \quad t \geq 0$$

which is of course the negative *exponential distribution*!
What we have just shown is that when the arrival process has a Poisson distribution, the interarrival times are exponentially distributed, and we have proven already that the exponential distribution has the remarkable *memoryless property* which was necessary for our Markov condition.

Exercise 3.5 *Let X be the time interval required to observe k Poisson arrivals. Consider the two events: exactly $k-1$ arrivals occur in the interval $(0,t-\Delta t)$ and the event that exactly one arrival occurs in the interval $(t-\Delta t,t)$. By letting $\Delta t \rightarrow 0$ show that*

$$f_X(x) = \frac{\lambda(\lambda x)^{k-1}}{(k-1)!} e^{-\lambda x} \quad x \geq 0$$

3.4 M/M/1 Queue

The last special case of Eqs. (3.3) – (3.4) we will consider is a birth-death process in which all the birth rates are the same and equal to λ for $k \geq 0$ and all death coefficients are equal to μ for $k \geq 1$. This birth-death process with constant coefficients is the simplest model of a queue, namely the well-known M/M/1 queue. The arrival process we know now from the previous section is Poisson with exponential interarrival times. The service time distribution we define to be exponential as well, or

$$B(x) \quad = \quad 1 - e^{-\mu x}, \quad x \geq 0$$

Clearly, if we consider the number $N(t)$ of customers in the queue at time t as the stochastic variable in this case and assume furthermore that the interarrival times and service times are mutually independent as well as independent from the number of customers in the system, our process is Markovian. The steady state distribution is given by Eq. (3.7) or, writing ρ for λ/μ, by

$$\pi_k \quad = \quad (1-\rho)\rho^k \tag{3.12}$$

since

$$\begin{aligned} \pi_0 \quad &= \quad \left(1 + \sum_{k=1}^{\infty} \rho^k\right)^{-1} \\ &= \quad 1 - \rho \end{aligned} \tag{3.13}$$

where we obviously require $0 \leq \rho < 1$ for this solution to exist (cf. condition of ergodicity on page 63). The ratio ρ is known as the *utilisation* of the system. Note for instance that π_0 or the probability that the system will be *idle* is given by

$$\pi_0 = 1 - \rho$$

or, conversely the probability that the system will be busy is given by ρ so that "utilisation" seems an appropriate name for this quantity.
Another important measure of any queueing system is the average number N of customers in the system. Since we know π_k we can now easily compute that quantity from

$$\begin{aligned} N &= \sum_{k=0}^{\infty} k\pi_k \\ &= (1-\rho)\sum_{k=0}^{\infty} k\rho^k \end{aligned} \tag{3.14}$$

There is a very useful trick one uses to evaluate Eq. (3.14), namely the fact that for an absolutely convergent series of functions, the derivative of the sum of terms is the sum of the derivative of the individual terms. Applying this trick we obtain from Eq. (3.14)

$$\begin{aligned} N &= (1-\rho)\rho\sum_{k=0}^{\infty}\frac{\partial}{\partial\rho}\rho^k \\ &= (1-\rho)\rho\frac{\partial}{\partial\rho}\sum_{k=0}^{\infty}\rho^k \\ &= (1-\rho)\rho\frac{\partial}{\partial\rho}\frac{1}{1-\rho} \\ &= \frac{\rho}{1-\rho} \end{aligned}$$

Applying Little's law, Eq. (3.2), we can write for T the average time in the M/M/1 queueing system

$$\begin{aligned} T &= \frac{N}{\lambda} \\ &= \frac{1}{\mu-\lambda} \end{aligned}$$

This dependence of average time in the M/M/1 queue on the utilisation ρ is illustrated in Fig. 3.6. Note that when $\rho = 0$ the average time in the system is just $1/\mu$, the average service time expected by a customer as we would suspect.

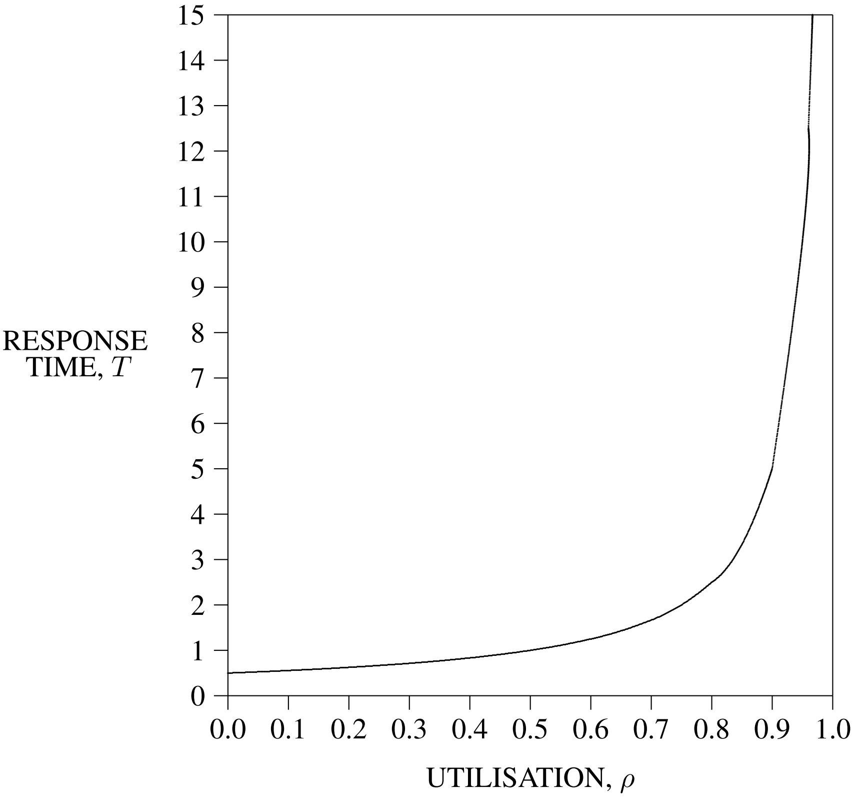


Figure 3.6 Delay time as a function of utilisation in an M/M/1 queue.

The behaviour illustrated in Fig. 3.6 drawn for $\mu = 2.0$, is rather dramatic. As $\rho \rightarrow 1$ the average delay $T \rightarrow \infty$. This type of behaviour with respect to ρ as it approaches 1 is characteristic of almost any queueing system one encounters. It is also the explanation of why many a computer system manager has lost his job: The system works well with increasing load in that response times increase only slowly until, all of a sudden, the knee in Fig. 3.6 is reached and response times become unacceptably long almost overnight!

Exercise 3.6 *Consider an M/M/1 queue where there is no room for waiting customers. We can define this as a birth-death process with coefficients*

$$\lambda_k = \begin{cases} \lambda, & k = 0 \\ 0, & k \neq 0 \end{cases} \qquad \mu_k = \begin{cases} \mu, & k = 1 \\ 0, & k \neq 1 \end{cases}$$

1. *Give the differential equations for* $\pi_k(t)$ $(k = 0{,}1)$.
2. *Solve these equations in terms of* π_0 *and* π_1.

Exercise 3.7 *At an automatic carwash there is room for only* C *waiting cars. Dirty cars arrive at the rate of 6 per hour and are washed at an exponential rate of 3 per hour. If there are more than two cars* waiting *the chances are 50 percent that the arriving car will try to find another carwash.*

1. *Set up the global balance equations for the system.*

2. *Calculate the average number of cars at the carwash at any moment.*

3. *If a car wash costs* ECU*10, what is the income of the carwash per hour?*

3.5 M/M/m Queue

An interesting generalisation of the M/M/1 queue is the case of m servers rather than a single one. This queue is illustrated in Fig. 3.7 and the CTMC with the number of customers in the system as the state variable is illustrated in Fig. 3.8.

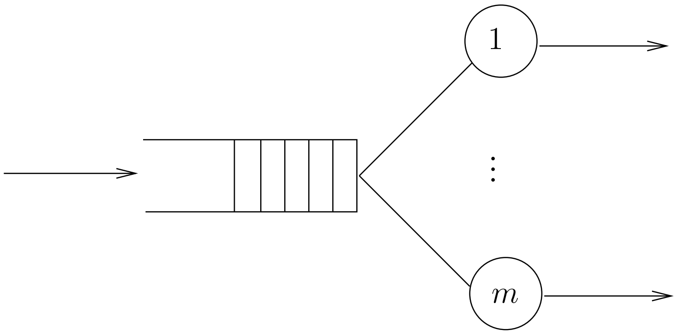


Figure 3.7 The M/M/m queueing system.

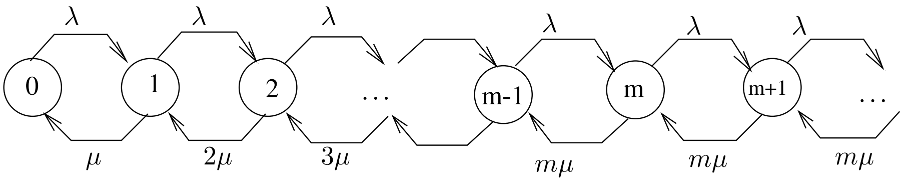


Figure 3.8 State-transition diagramme of the M/M/m queueing process.

From Fig. 3.8 we can easily set up the global balance equations:

$$\begin{aligned} \mu\pi_1 &= \lambda\pi_0 \\ (i+1)\mu\pi_{(i+1)} + \lambda\pi_{(i-1)} &= (\lambda + i\mu)\pi_i, \quad i < m \\ m\mu\pi_{(i+1)} + \lambda\pi_{(i-1)} &= (\lambda + m\mu)\pi_i, \quad i \geq m \end{aligned}$$

So that

$$\pi_k = \begin{cases} \pi_0 \left(\frac{\lambda}{m\mu}\right)^k \frac{m^k}{k!}, & k < m \\ \pi_0 \left(\frac{\lambda}{m\mu}\right)^k \frac{m^m}{m!}, & k \geq m \end{cases}$$

with

$$\pi_0 = \left(\sum_{k=0}^{m-1} \frac{(m\rho)^k}{k!} + \frac{(m\rho)^m}{m!} \frac{1}{1-\rho} \right)^{-1}$$

where $\rho = \frac{\lambda}{m\mu}$. Solving we obtain for the average number N of customers in the system

$$N = \sum_{k=0}^{\infty} k\pi_k = m\rho + \rho \frac{(m\rho)^m}{m!} \frac{\pi_0}{(1-\rho)^2}$$

Exercise 3.8 *Show that the average time in an M/M/m system is less than in m, M/M/1 queues.*

Exercise 3.9 *Consider the freeway mentioned in Ex. 3.1. Assume that cars have an interarrival rate of 3 per second and that the 4 lane freeway can accommodate 6 cars passing a particular point per second.*

1. *What is the utilisation of the freeway?*
2. *Assuming the same arrival rate, determine what happens if one of the lanes is suddenly closed.*

3.6 Queues with Processor Sharing Scheduling Strategy

In practice the processor scheduler of a computer system often uses a principle known as time-slicing, whereby all processes requiring service are served for a fixed length of time (the time-slice) in a round-robin fashion. A customer whose service is completed leaves immediately. This scheduling strategy is approximated in theory by a Processor Sharing (or PS for short) server.
All customers have an identical mean service requirement $\frac{1}{\mu}$ and all n customers in the queue are served simultaneously. Each customer is served at a rate $\frac{1}{n}\mu$. It is not difficult to see that, if we represent the state of the process by the number of customers present, that the corresponding CTMC is similar to that for the M/M/1 queue illustrated in Fig. 3.5. The rate of a service completion is given by $n\frac{1}{n}\mu$. Note that the minimum of a set of exponentially distributed random variables is again exponentially distributed (cf. Sec. 2.4). The steady state probability distribution is thus given by the identical Eqs. (3.13) and (3.12).

3.7 Queues with Infinite Servers

A variation of the PS scheduling strategy is that called an Infinite Server (IS) where the processor rate is independent of the number of customers present, yet all are served simultaneously with the same rate μ. Thus $\mu_k = k\mu$ for an IS queue. As for the PS scheduling strategy there is no waiting time in the queue and each customer is merely delayed for a random length of time with distribution that of the service time. An Infinite Server can also be viewed as a queue with an infinite number of servers so that a free server can be dedicated to each newly arriving customer. An example of such a queue is, e.g., a terminal pool.

Queues with the PS and IS service disciplines belong to a larger family of queues know as being of the BCMP-type[15]. Queueing networks using these servers can be analysed very efficiently, since the steady state probability distribution of the entire network can be represented by a (possibly normalised) product of the steady state probability distributions of the queues considered in isolation. Such queueing networks are also known as having *product-form*.

3.8 Queues with Priority Service

Up to this point in our discussions the stochastic behaviour of all our customers was identical and independent. There are practical situations however where we may want to distinguish between different *classes* of customers[102]. This may be the case because different classes of customers may have different service requirements or because the server may wish to give different priorities to different classes of customers or both. In this last section on queueing we will investigate priority service.

When one has priority service it is obviously necessary to define exactly what the priority rules for the different classes of customers are. One may decide for instance that an arriving priority customer will pre-empt a lower priority customer in service. In the latter case it is then necessary to make a distinction between the case where the service of the lower priority customer is *resumed* where it had left off on pre-emption, or whether the service will be *repeated* from scratch.

Example 3.1 *Consider an M/M/1 queue with two classes of customers whose service follow a negative exponential distribution with parameters μ_1 and μ_2, and with arrival rates λ_1 and λ_2 respectively. We assume the n_1 customers of class 1 have priority and that the n_2 customers of class 2, whose service is interrupted, will resume their service. Resuming service or restarting the service is equivalent in this case, however, since the service times are exponentially distributed (the memoryless property, cf., page 24).*

Since it is clear that priority customers are not affected by customers of class 2, we have

$$\pi_1(n_1) = \rho_1^{n_1}(1-\rho_1) \ \ \textit{with} \ \ \rho_1 = \frac{\lambda_1}{\mu_1}$$

as was the case for the M/M/1 queue.

Write $\pi(n_1,n_2)$ for the steady state probability distribution of the number of class 1 and class 2 customers in the system. In this case the global balance equations are

$$\begin{aligned}
(\lambda_1+\lambda_2+\mu_1)\pi(n_1,n_2) &= \lambda_1\pi(n_1-1,n_2)+\mu_1\pi(n_1+1,n_2)\\
&\quad +\lambda_2\pi(n_1,n_2-1),\ n_1>0,n_2>0,\\
(\lambda_1+\lambda_2+\mu_1)\pi(n_1,0) &= \lambda_1\pi(n_1-1,0)\\
&\quad +\mu_1\pi(n_1+1,n_2),\ n_1>0,n_2=0,\\
(\lambda_1+\lambda_2+\mu_2)\pi(0,n_2) &= \mu_1\pi(1,n_2)+\lambda_2\pi(0,n_2-1)\\
&\quad +\mu_2\pi(0,n_2+1),\ n_1=0,n_2>0,\\
(\lambda_1+\lambda_2)\pi(0,0) &= \mu_1\pi(1,0)+\mu_2\pi(0,1)
\end{aligned}$$

It is not easy to solve these balance equations and we will not describe how to do so. Instead we simply quote the results (see, e.g., [77]). The mean number N_1 and N_2 of class 1 and 2 customers, respectively, are given by

$$\begin{aligned}
N_1 &= \frac{\rho_1}{1-\rho_1}\\
N_2 &= \frac{\rho_2+N_1\rho_2}{1-\rho_1-\rho_2}
\end{aligned}$$

As one might expect $\pi(0,0)=1-\rho_1-\rho_2$ as in the case of the M/M/1 queue.

Exercise 3.10 *Explain how one can solve a queue using semi-Markov analysis?*

Exercise 3.11

(a) *Consider a queue with K different classes of customers arriving. The arrival process for each customer class $i,i=1,\ldots,K$ is specified by a Poisson process with parameter λ_i. The service distribution of customers of class i is Coxian with mean value $\frac{1}{\mu_i}$.*

Calculate the steady state probability distribution of the number of customers in the system. I.e., compute $P[(n_1,\ldots,n_K)]$, the steady state probability that n_i customers of class i ($i=1,\ldots,K$) are in the queue. Assume a queue with a PS service scheduling strategy.

(b) *Consider the same queue as in part (a), but now with an Infinite Server strategy. Determine $P[(n_1,\ldots,n_K)]$ again. Also compute the mean number of customers in the queue.*

4 Further Reading

There is a huge selection of books on the subjects of with stochastic theory and especially Markov processes and queueing theory. The following subjective selection of books is thus not an exhaustive list. With the basics already covered in this book, the reader can choose the books with the style and notation (s)he likes. E.g., an introduction to stochastic processes is given in [53] and an old, but still excellent introduction to the theory of single queues can be found in the book by Leonard Kleinrock [108]. Results on priority queues is given in [92]. All these references also discuss queues with general arrival and service time distributions. In this book we only touched on the theory of queues and focused on models where single customers arrive at and leave a queue. Queues with batch (or bulk) arrival and service rates are found in [49].

The analysis of networks of queues is also a vast subject, in particular the so-called product-form networks, which can be analysed very efficiently. The first product form networks were analysed by Jackson [91] in 1957 and Gordon/Newell [87] in 1967. Before that the output process of a single queue, for example the M/M/m queue described above, was found to have the identical Poisson departure process as the input process. In other words, the inter-departure times are exponentially distributed with the same parameter λ as the input Poisson process.[48, 102, 151].

Similar results hold for all M/M-queues with work-conserving[1] scheduling disciplines which are independent of the service time requirements [102]. A *feed-forward* network of such queues is easy to analyse since the output process is Poisson with the same parameter as the input process. Each queue can be considered in isolation and the steady-state distribution of the overall network is given by the product of the steady-state distributions of each individual queue.

Unfortunately the above is not true for a network of queues with feedback since the output process is no longer Poisson, as successive inter-departure times are no longer identically distributed because of the (partial) dependencies of the input and output processes. In open queueing networks one can show that the output from the network is still Poisson, but that in closed networks none of the flows between queues is Poisson [102].

Surprisingly, one can nevertheless solve such networks of queues by analysing each queue in isolation. In a seminal paper published in 1975 by F. Baskett, K. M. Chandy, R. R. Muntz and F. G. Palacios, established a product form solution for queueing networks, called BCMP-networks, which have a mixture of four different type of stations [15]. Based on the product-form property very efficient

[1] A scheduling discipline is work-conserving when no overhead or loss of work is incurred by scheduling.

algorithms can be designed to compute the steady state probability distribution and performance metrics of these BCMP-networks. The best known of such algorithms are the convolution algorithm and the mean value analysis for closed nets [102].

Mean Value Analysis (MVA) is based on Little's theorem and the arrival theorem for closed product-form queueing networks. The arrival theorem states that a job entering a queue, observes the steady-state distribution for the network with one customer removed. In subsequent years significant research went into extending the class of queueing networks for which a product-form solution of the steady-state distribution can be found, e.g. [14]. For further reading we recommend [42].

Further information on queues and queueing networks can be found in the World Wide Web at URL http://liinwww.ira.uka.de/bibliography/Distributed/QLD/index.html. Here the reader finds a bibliography on queueing systems and telecommunication, which is part of the collection of computer science bibliographies.

Additional addresses offering search facilities (also for information on other scientific computer relevant subjects) are http://liinwww.ira.uka.de/bibliography/waisbib.htmland http://citeseer.ist.psu.edu/.

This concludes our introduction to stochastic theory.

Part II

PETRI NETS

5 Place-Transition Nets

Petri nets were invented in 1962 by Carl Adam Petri [133] and are a formalism for the description of concurrency and synchronisation inherent in modern distributed systems. Since first described by Petri, many variations of his original nets have been defined such as Coloured Petri nets, which we shall discuss in Ch. 6. In this chapter we shall define Place-Transition nets, also referred to as "ordinary" Petri nets. Throughout this book we shall use the term "Petri nets" to refer to any type of net which is derived from that originally defined by Petri.
Apart from modelling and formally analysing modern distributed systems, Petri nets also provide a convenient graphical representation of the system being modelled. The graphical representation of a Place-Transition net comprises the following components:

places, drawn by circles. Places model conditions or objects, e.g., a program variable. Places might contain

tokens, drawn by black dots. Tokens represent the specific value of the condition or object, e.g., the value of a program variable.

transitions, drawn by rectangles. Transitions model activities which change the values of conditions and objects.

arcs, specifying the interconnection of places and transitions thus indicating which objects are changed by a certain activity.

Place-Transition nets are bipartite graphs, i.e. we may only connect a place to a transition or vice versa, but we must not connect two places or transitions. This also would not make sense, if we keep the interpretation of the components of a Place-Transition net in mind.
Consider Fig. 5.1. p_1 is a place marked with one token and is connected to transition t_1. Because of the direction of the arc connecting p_1 and t_1 we will call p_1 an input place of t_1 and t_1 accordingly an output transition of p_1. Places and transitions might have several input/output elements. E.g., place p_2 has three input transitions: t_1, t_3, t_5, while t_4 has two output places: p_4 and p_5. The latter is marked with two tokens.
p_6 and t_7 are not interconnected to any other element. Such elements will be called isolated places or transitions respectively. As expected, isolated elements of a Place-Transition net do not influence the rest of the net and therefore we can neglect them. Until now we have only described the static structure of a Petri net. Its dynamic behaviour or the modification of the condition/object values, can be expressed via the following rules:

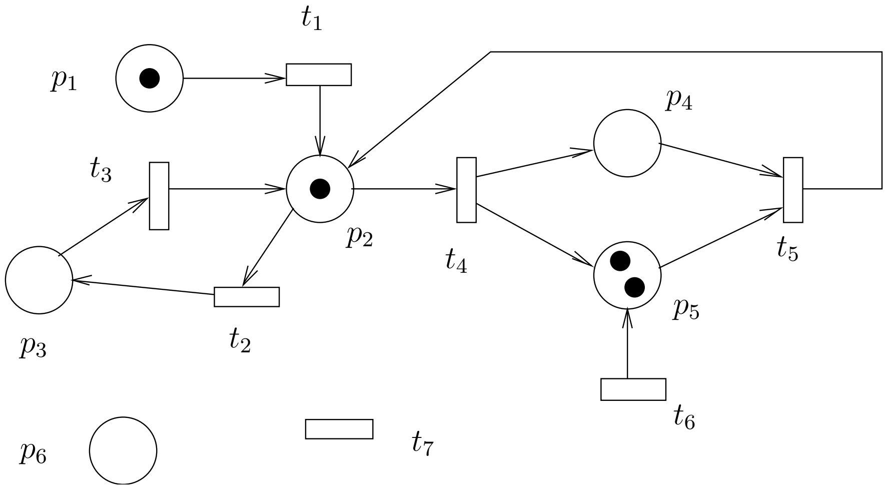


Figure 5.1 A Place-Transition net

Enabling of a transition:

A transition is enabled if all its input places are marked at least with one token. For example transitions t_1 and t_4 in the Place-Transition net of Fig. 5.1 are enabled.

Firing of an enabled transition:

An enabled transition may fire. If a transition fires, it destroys one token on each of its input places and creates one token on each of its output places. E.g., if the enabled transition t_1 fires, it destroys the token on p_1 and creates one token on p_2 yielding the Place-Transition net given in Fig. 5.2. Note that a transition need not fire.

Apart from modeling the behaviour of complex systems, Place-Transition nets were originally designed to assist the analysis of such systems. In order to understand how this might be done, consider the following example.

Our example system consists of a sender and a receiver communicating with each other via two unidirectional channels. Since we are interested in an implementation of this system, we do not wish to waste hardware resources. So we want to implement channels which need to hold at most one message each. With that restriction, sender and receiver have to obey a certain communication protocol. We first of all write down an implementation of the sender and receiver in a Pascal-like notation.

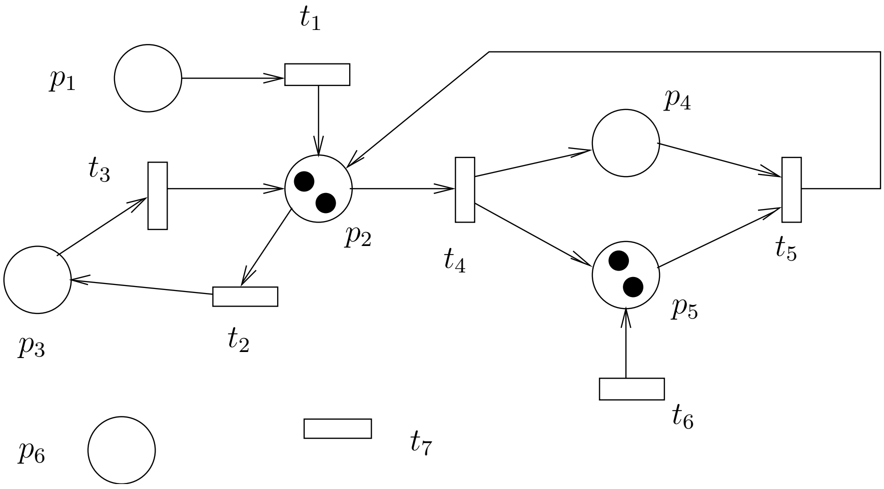


Figure 5.2 Marking of the Place-Transition net after firing of t_1

Sender:

```
    while true do
    begin
     prepare message
     send message to receiver via channel1
     read acknowledgement of receiver on channel2
     interpret acknowledgement
    end
```

Receiver:

```
    while true do
    begin
     read message from sender on channel1
     interpret message
     prepare acknowledgement
     send acknowledgement to sender via channel2
    end
```

Given this description of our system, we have to prove that it will operate as intended. Particularly, this means that neither sender nor receiver send a message/acknowledgement via the corresponding channel before a previous message/acknowledgement has been read by the communication partner.

How can a Place-Transition net help us to prove that our system is correct? All the above statements describe activities of the sender and receiver. Activities can be modelled by transitions. So with each statement we associate a transition as follows:

Sender:

```
      while true do
      begin
$t_1$   prepare message
$t_2$   send message to receiver via channel1
$t_3$   read acknowledgement of receiver on channel2
$t_4$   interpret acknowledgement
      end
```

Receiver:

```
      while true do
      begin
$t_5$   read message from sender on channel1
$t_6$   interpret message
$t_7$   prepare acknowledgement
$t_8$   send acknowledgement to sender via channel2
      end
```

Which objects do these transitions modify? There are several objects involved, e.g., the message, the acknowledgement, the state of a channel, the states of sender and receiver. Since we are not concerned with the content of the messages, we will neglect such information, thus, e.g., the internal state of a channel is given by the number of messages/acknowledgements currently present. The Place-Transition net illustrated in Fig. 5.3 represents our system, where the places p_1 to p_8 represent the internal state of sender and receiver respectively. p_9 represents the channel from sender to receiver and p_{10} the channel in reverse direction. Using this Place-Transition net model, we can get a better insight into the functionality of our system.

So far we presented an informal introduction of Petri nets. In the following sections we will introduce Place-Transition nets formally.

Exercise 5.1 *Consider the Place-Transition net of Fig. 5.1.*

1. *Name the input and output places of t_5, t_6, t_7 and the input and output transitions of p_3, p_4, p_5, p_6.*

2. *How many tokens are on p_4?*

3. *Transitions t_3 and t_5 are enabled. Is that correct?*

4. *What happens after firing t_2. Is t_4 still enabled?*
   Such a situation is called a conflict, since firing one transition disables another transition.

5. *Count the number of all tokens in the net before and after firing of t_4. This illustrates, that a transition does not merely move tokens around, but really changes the value of objects by destroying and creating tokens.*

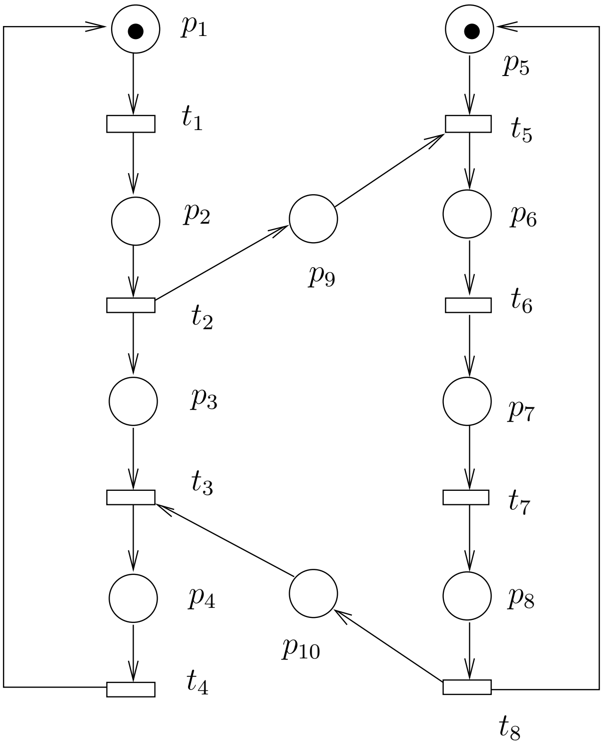


Figure 5.3 Place-Transition net model of the sender/receiver system

Exercise 5.2 *Play the token game for the Place-Transition net of Fig. 5.3 by putting tokens onto the places and "move" them around according to the enabling and firing rules given before. Is our implementation correct?*

5.1 Structure of Place-Transition Nets

A Place-Transition net is a bipartite directed graph composed of places, drawn as circles, and transitions, drawn as rectangles. As usual, a directed graph is formally given by the description of its elements (here separated into two sets, since we have got two kinds of nodes) and functions or matrices specifying their interconnection. Here we employ a functional notation.

Definition 5.1 (Place-Transition net) *A* Place-Transition net *(P-T net) is a 5-tuple* $PN = (P,T,I^-,I^+,M_0)$ *where*

- $P = \{p_1,\ldots,p_n\}$ *is a finite and non-empty set of places,*
- $T = \{t_1,\ldots,t_m\}$ *is a finite and non-empty set of transitions,*

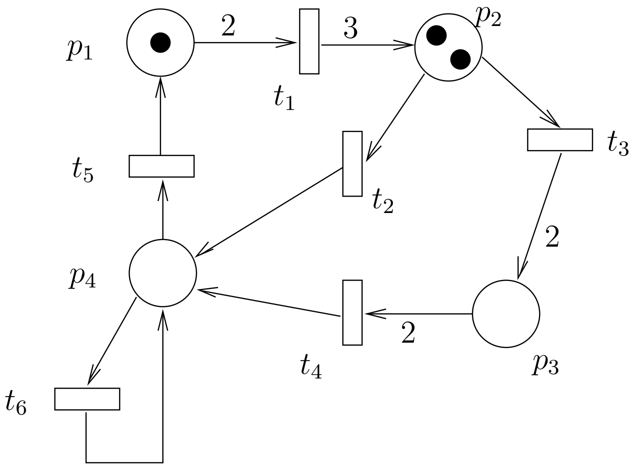


Figure 5.4 Place-Transition net

- *$P \cap T = \emptyset$,*
- *$I^-,I^+ : P \times T \rightarrow \mathbb{N}_0$ are the backward and forward incidence functions, respectively*
- *$M_0 : P \rightarrow \mathbb{N}_0$ is the initial marking.*

If we refer to the structure only, a Place-Transition net can also be viewed as a 4-tuple $PN = (P,T,I^-,I^+)$ omitting the initial marking M_0. Some authors refer to the unmarked net $PN = (P,T,I^-,I^+)$ as the Place-Transition net and call a marked net $PN = (P,T,I^-,I^+,M_0)$ a *system*.
The functions I^- and I^+ specify the connection between places and transitions. If $I^-(p,t) > 0$, an arc leads from place p to transition t; therefore I^- is called the backward incidence function of transition t. I^- maps into the natural numbers thus attaching a weight to the arc leading from p to t. In the graphical representation of a Place-Transition net this arc weight is written near the corresponding arc. It is convention that an arc weight of 1 is not shown explicitly. Arc weights specify that a transition is enabled only when at least as many tokens as given by the arc weight are located on that place. Firing will destroy exactly this number of tokens from p. Similarly $I^+(p,t)$ specifies the number of tokens created on place p in case of firing t.

Example 5.1 *The formal definition of the Place-Transition net depicted in Fig. 5.4 is given as follows:*
$PN = (P,T,I^-,I^+,M_0)$ where

- *$P = \{p_1,p_2,p_3,p_4\}$*
- *$T = \{t_1,t_2,t_3,t_4,t_5,t_6\}$*

- $I^-(p_1,t_1) = 2$, $I^-(p_2,t_2) = 1$, $I^-(p_2,t_3) = 1$, $I^-(p_3,t_4) = 2$, $I^-(p_4,t_5) = 1$, $I^-(p_4,t_6) = 1$. *All other values of function I^- are zero.*
- $I^+(p_2,t_1) = 3$, $I^+(p_4,t_2) = 1$, $I^+(p_3,t_3) = 2$, $I^+(p_4,t_4) = 1$, $I^+(p_1,t_5) = 1$, $I^+(p_4,t_6) = 1$. *All other values of function I^+ are zero.*
- $M_0(p) = \begin{cases} 1 & \textit{if } p = p_1 \\ 2 & \textit{if } p = p_2 \\ 0 & \textit{otherwise} \end{cases} \quad \forall p \in P.$

The input and output elements of a place and a transition can be defined formally as follows.

Definition 5.2 *Let $PN = (P,T,I^-,I^+,M_0)$ be a Place-Transition net.*

- *Input places of transition t: $\bullet t := \{p \in P | I^-(p,t) > 0\}$,*
- *Output places of transition t: $t\bullet := \{p \in P | I^+(p,t) > 0\}$,*
- *Input transitions of place p: $\bullet p := \{t \in T | I^+(p,t) > 0\}$ and*
- *Output transitions of place p: $p\bullet := \{t \in T | I^-(p,t) > 0\}$,*

the usual extension to sets $X \subseteq P \cup T$ is defined as $\bullet X = \bigcup_{x \in X} \bullet x$, $X\bullet = \bigcup_{x \in X} x\bullet$.

The sets $\bullet p$, $\bullet\, t$ and $p\, \bullet$,$t\bullet$ are also respectively referred to as the *preset* and *postset* of the place p and transition t respectively.
For the Place-Transition net of Fig. 5.4 we get, e.g., $\bullet p_1 = \{t_5\}$, $p_2\bullet = \{t_2,t_3\}$, $\bullet\,\{t_2,t_3\} = \{p_2\}$,$\{p_2,t_2,p_4\}\bullet = \{t_2,t_3,t_5,t_6,p_4\}$.
A Place-Transition net can also be viewed as a usual directed graph and a flow relation defined as follows:

Definition 5.3 *Let $PN = (P,T,I^-,I^+,M_0)$ be a Place-Transition net.*

1. *$F \subseteq (P \times T) \cup (T \times P)$ given by $F := \{(x,y) | x,y \in P \cup T : x \in \bullet y\}$ is called the* flow relation *of PN.*

   Let F^* denote the reflexive and transitive closure of F, i.e. $\forall x,y,z \in P \cup T$:

   (a) $(x,x) \in F^*$
   (b) $(x,y) \in F \Longrightarrow (x,y) \in F^*$
   (c) $(x,y) \in F^*$ *and* $(y,z) \in F^* \Longrightarrow (x,z) \in F^*$

2. *PN is* weakly connected *iff*[1] $\forall x,y \in P \cup T : xF^*y$ *or* yF^*x

[1] *iff* stands for "if and only if".

3. *PN is* strongly connected *iff* $\forall x,y \in P \cup T : xF^*y$ *and* yF^*x

For the Place-Transition net of Fig. 5.4 the following, e.g., holds:

$$p_1Ft_1,p_2F^*p_2,t_1F^*t_3,t_3F^*p_1,t_1F^*p_1$$

Exercise 5.3 *Give the formal definition of the Place-Transition net given in Fig. 5.1.*

Exercise 5.4 *Determine the following quantities for the Place-Transition net of Fig. 5.1.*

1. $\bullet t_1$, $\bullet t_5$, $t_2\bullet$, $\bullet t_6$, $\bullet p_3$, $p_4\bullet$, $\bullet p_2$, $p_2\bullet$.
2. $\bullet\{t_1,t_2,t_3\}$, $\bullet\{t_1,t_2,t_6\}$, $\bullet\{t_1,t_2,t_7\}$, $\bullet\{t_1,t_2,t_4\}$, $\bullet\{p_2,p_3\}$, $\{p_2,p_3\}\bullet$, $\{p_2,p_3,p_5\}\bullet$, $\{p_2,p_6\}\bullet$, $\{t_2,t_3\}\bullet$, $\{t_4,t_3\}\bullet$, $\bullet\{t_1,p_2\}$.
3. $\bullet T$, $T\bullet$, $\bullet P$, $P\bullet$, $\bullet(T \cup P)$, $(T \cup P)\bullet$.

Exercise 5.5 *Prove that the Place-Transition nets of Fig. 5.1 and 5.4 are strongly or weakly connected.*

5.2 Dynamic Behaviour of Place-Transition Nets

In the last section we defined the structure of a Place-Transition net. The dynamic behaviour is determined by enabling and firing of transitions as given as follows.

Definition 5.4 *Let $PN = (P,T,I^-,I^+,M_0)$ be a Place-Transition net.*

1. *A* marking *of a Place-Transition net is a function $M : P \mapsto \mathbb{N}_0$, where $M(p)$ denotes the number of tokens in p.*
2. *A set $\tilde{P} \subseteq P$ is* marked *at a marking M, iff $\exists p \in \tilde{P} : M(p) > 0$; otherwise $\tilde{P}$ is* unmarked *or* empty *at M.*
3. *A transition $t \in T$ is* enabled *at M, denoted by $M[t >$, iff $M(p) \geq I^-(p,t), \forall p \in P$.*
4. *A transition $t \in T$, enabled at marking M,* may fire *yielding a new marking M' where*

$$M'(p) = M(p) - I^-(p,t) + I^+(p,t), \forall p \in P,$$

denoted by $M[t > M'$. We say M' is directly reachable *from M and write $M \rightarrow M'$. Let $\rightarrow^*$ be the reflexive and transitive closure of $\rightarrow$. A marking M' is* reachable *from M, iff $M \rightarrow^* M'$.*

5. *A* firing sequence *(occurrence sequence) of PN is a finite sequence of transitions $\sigma = t_1 \dots t_n, n \geq 0$ such that there are markings $M_1, \dots, M_{n+1}$ satisfying $M_i[t_i > M_{i+1}, \forall i = 1, \dots, n$. A shorthand notation for this case is $M_1[\sigma >$ and $M_1[\sigma > M_{n+1}$ respectively. The empty firing sequence is denoted by ε and $M[\varepsilon > M$ always holds.*

The initial marking is the starting point of the dynamic behaviour of the Place-Transition net and is therefore singled out and denoted M_0. Since transitions are only enabled if their input places are marked with at least as many tokens as specified by the backward incidence function (cf. Def. 5.4(3)), we are especially interested in marked (sub)sets of places.
The expression $M'(p) = M(p) - I^-(p,t) + I^+(p,t)$ in Def. 5.4(4) corresponds to our intuitive understanding of the firing process. If transition t fires, it destroys $I^-(p,t)$ tokens at place p and creates additional $I^+(p,t)$ tokens on p. Note that the place p is the same in both expressions $I^-(p,t)$ and $I^+(p,t)$. In most cases one of these expressions is 0, but there might exist some cases in which a transition can destroy and create tokens at the same place. For instance, if transition t_6 in Fig. 5.4 fires at some marking in which it is enabled, the number of tokens on place p_4 remains unchanged. Such connection structures are called self-loops.

Example 5.2 *Consider PN, the Place-Transition net illustrated in Fig 5.4.
A marking of PN is, e.g., $M(p_1) = 100, M(p_2) = 1, M(p_3) = 0$, $M(p_4) = 10^9$. Note that a marking is only a mapping and need not be reachable from the initial marking. Of course, in analysis of Place-Transition nets we are particularly interested in reachable markings.
$\tilde{P}_1 := \{p_2\}$ and $\tilde{P}_2 := \{p_1, p_2\}$ are marked at M_0. t_2 is enabled at M_0, so it is correct to write $M_0[t_2 >$. After firing t_2 a new marking M' is reached, denoted by $M_0[t_2 > M'$ where $M'(p_1) = M_0(p_1) = 1$, $M'(p_2) = M_0(p_2) - 1 = 1$, $M'(p_3) = M_0(p_3) = 0$, $M'(p_4) = M_0(p_4) + 1 = 1$. Furthermore $\sigma = t_2 t_5 t_1$ is a firing sequence of PN, since $M_0[\sigma >$.*

Exercise 5.6 *Use the Place-Transition net in Fig 5.4 for the following exercises.*

1. *Determine all sets of places marked at M_0.*
2. *Prove or disprove:*
   $M_0[t_1 >, M_0[t_2 >, M_0[t_4 >, M_0[t_6 >, M_0[t_2 t_5 t_3 t_4 >$,
   $M_0[t_3 t_4 t_5 t_1 >, M_0[\varepsilon >$.
3. *Calculate the following markings M:*
   - $M_0[t_2 > M$,
   - $M_0[t_3 t_4 > M$,
   - $M_0[t_2 t_5 t_1 > M$,
   - $M_0[t_2 t_6 t_6 t_6 t_5 t_1 > M$,
   - $M_0[\varepsilon > M$.

4. *Prove or disprove:*
   - $M_0 \rightarrow^* M$*, where* M *is given by* $M_0[t_3t_2 > M$.
   - $M_0 \rightarrow^* M$*, where* M *is given by* $M_0[t_3t_2t_4t_5 > M$.
   - $M_0 \rightarrow^* M$*, where* M *is given by* $M(p_1) = 0, M(p_2) = 2, M(p_3) = 2, M(p_4) = 0$.
   - $M_0 \rightarrow^* M$*, where* M *is given by* $M(p_1) = 1, M(p_2) = 3, M(p_3) = 0, M(p_4) = 0$.
   - $M_0 \rightarrow^* M$*, where* M *is given by* $M(p) = 0, \forall p \in P$.

5.3 Properties of Place-Transition Nets

Now that we have described the dynamic behaviour of Place-Transition nets we turn our interest to the properties of such nets and how one might verify them. Obviously we are only interested in properties concerning reachable markings.
One property to verify for our sender-receiver system in Chapter 5 was to ensure that the places representing the channels contain at most one token in every reachable marking.
Another property concerns the firing of transitions. Since transitions model activities, it is often desirable that no marking can be reached, such that a transition is never enabled again. If such situations are excluded the Place-Transition net is called "live".
Last but not least, it is also desirable that when we view the set of reachable markings as a graph, that it contains exactly one strongly connected subset of markings. We know from our Markovian theory, that this is necessary for the existence of a steady state distribution. Fig. 5.5 depicts two possible structures of the reachability set drawn as a graph, where the subsets I_j are assumed to be strongly connected. The left part of Fig. 5.5 contains two strongly connected subsets. If we assume that this graph represents the state space of a Markov process, we would have no steady state distribution (cf. Ex. 2.4 on page 33). Concerning the right-hand part of Fig. 5.5, a steady state distribution exists assuming a finite reachability set. Note that this situation is characterised by some marking M which is reachable from all other markings.
All these remarks give rise to the following definition.

Definition 5.5 *Let* $PN = (P,T,I^-,I^+,M_0)$ *be a Place-Transition net.*

1. *The* reachability set *of PN is defined by* $R(PN) := \{M | M_0 \rightarrow^* M\}$. *If* PN *denotes an unmarked Place-Transition net or if we want to consider parts of the reachability set, the set of reachable markings for a given marking* $\tilde{M}$ *will be denoted by* $R(PN,\tilde{M}) := \{M | \tilde{M} \rightarrow^* M\}$. *Thus for a marked Place-Transition net we have* $R(PN) = R(PN,M_0)$.

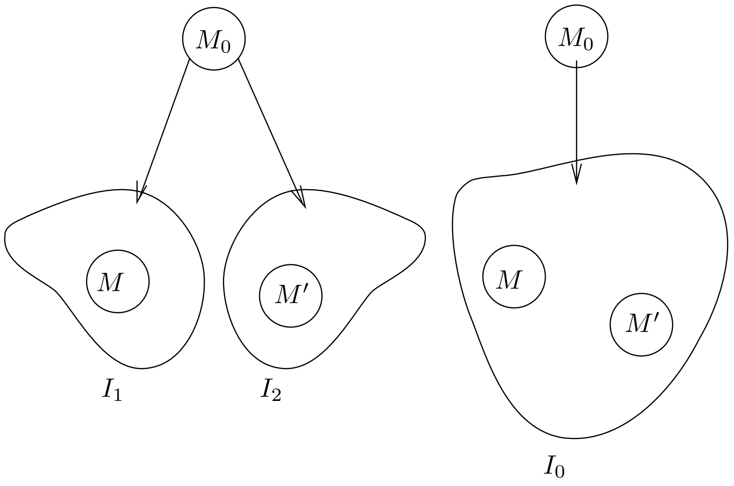


Figure 5.5 Two possible structures of the reachability set

2. *PN is a* bounded Place-Transition net, *iff* $\forall p \in P : \exists k \in \mathbb{N}_0 : \forall M \in R(PN) : M(p) \leq k$.
   PN *is* safe, *iff* $\forall p \in P : \forall M \in R(PN) : M(p) \leq 1$.
3. *A transition $t \in T$ is live, iff* $\forall M \in R(PN) : \exists M' \in R(PN) : M \to^* M'$ *and* $M'[t >$.
   PN *is* live, *iff all transitions are live, i.e.* $\forall t \in T, M \in R(PN) : \exists M' \in R(PN) : M \to^* M'$ *and* $M'[t >$.
4. *A marking $M \in R(PN)$ is a* home state, *iff* $\forall M' \in R(PN) : M' \to^* M$.

Safeness is of interest if the places of a Place-Transition net represent conditions. In that case the presence or absence of a token models that the condition is or is not satisfied. When modelling conditions, we are only interested in safe nets, since two or more tokens on a place make no sense. Historically, one of the first Petri nets, Condition-Event nets, were especially designed for this purpose. Condition-Event nets do not allow multiple arc weights and transitions are only enabled if their output places are empty, thus avoiding markings with more than one token on a place. The situation that output places of an enabled transition (with respect to Def. 5.4) are marked, is called *contact*.

We will use a vector notation to describe markings in order to simplify our notation. If the set of places P is given by $\{p_1, \ldots, p_n\}$ the function $M : P \mapsto \mathbb{N}_0$ can also be viewed as a vector $M := (M(p_1), \ldots, M(p_n))^T$. For notational convenience we will represent markings also by row vectors.

Example 5.3 *Consider the net in Fig. 5.3 with the following reachability set*

$$\begin{aligned} R(PN) = \; & \{(1,0,0,0,1,0,0,0,0,0),(0,1,0,0,1,0,0,0,0,0),\\ & (0,0,1,0,1,0,0,0,1,0),(0,0,1,0,0,1,0,0,0,0),\\ & (0,0,1,0,0,0,1,0,0,0),(0,0,1,0,0,0,0,1,0,0),\\ & (0,0,1,0,1,0,0,0,0,1),(0,0,0,1,0,1,0,0,0,0)\} \end{aligned}$$

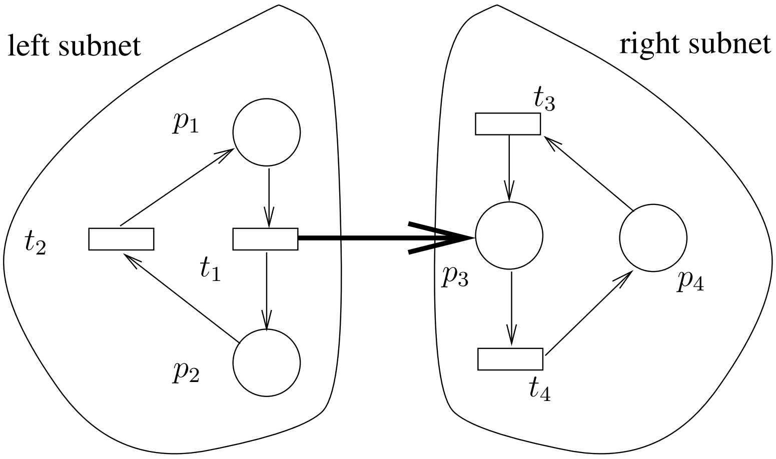


Figure 5.6 Place-Transition net not being live **and** bounded

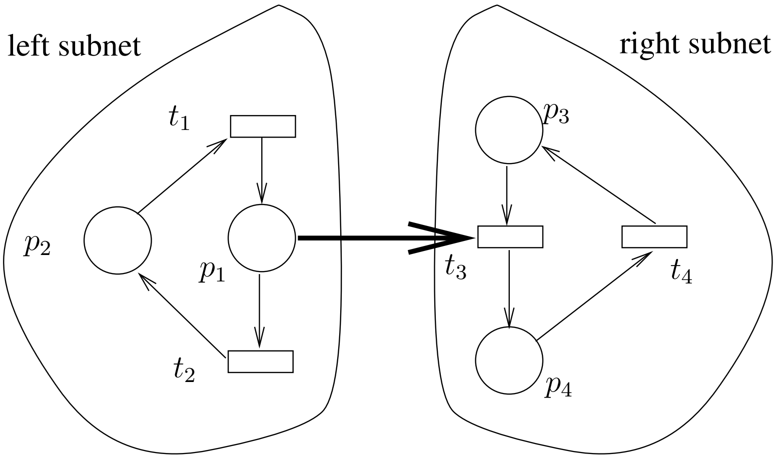


Figure 5.7 Further Place-Transition net not being live **and** bounded

From this reachability set it is easy to derive that the Place-Transition net is bounded and in fact safe $(M(p) \leq 1, \forall p \in P, M \in R(PN))$ *and that* M *is a home state* $\forall M \in R(PN)$*. Furthermore the Place-Transition net is live.*

Liveness and boundedness impose a restriction on the structure of a Place-Transition net. Suppose transition t_1 of the Place-Transition net in Fig. 5.1 fires emptying place p_1. Since place p_1 has no input transitions it will remain empty in all further reachable markings, implying that t_1 is not live. In the same Place-Transition net, transition t_6 has no input places. Thus it is always enabled, so that no upper bound for the number of tokens can be found for place p_5. This example shows that if the Petri net has source (or sink) elements, it can not be both live and bounded. The reader should convince himself that also sink elements prevent live- and boundedness.

Figures 5.6 and 5.7 illustrate that liveness and boundedness can not hold both even in weakly, but not strongly, connected Place-Transition nets. Considering Fig. 5.6 we can find two strongly connected subnets which are connected via the arc from t_1 to p_3. If we assume that the left subnet is live, then the right subnet is obviously not bounded. On the other hand, if we assume that the right subnet is bounded, the left subnet can not be live. So in summary the net of Fig. 5.6 can not be both live and bounded for any initial marking. The same can be concluded for the Place-Transition net of Fig. 5.7 following similar arguments. As the reader might expect the following holds.

Theorem 5.1 ([34]) *Let PN be weakly connected.*
PN is live and bounded $\Longrightarrow$ PN is strongly connected.

In other words strong connectedness is a necessary condition for a PN to be live and bounded. Since we are only interested in Place-Transition nets for which these properties hold, we will only consider strongly connected Place-Transition nets in the following.
Note that if a Place-Transition net is not weakly connected it may still be live and bounded. In that case the Place-Transition net consists of more than one isolated strongly connected Place-Transition nets which can be analysed separately.

Exercise 5.7 1. *Determine $R(PN)$ for the Place-Transition net displayed in Figures 5.1 and 5.4. Prove or disprove the following conjectures for both Place-Transition nets:*

- *PN is bounded.*
- *PN is live.*

2. *Consider the Place-Transition net in Fig. 5.4 and change the arc weight of the arc leading from t_1 to p_2 to 2, i.e. define $I^+(p_2,t_1) = 2$.*
Prove or disprove the following conjectures:

- *PN is bounded.*
- *PN is live.*
- *M_0 is a home state of PN.*

Exercise 5.8

- *Find further examples of weakly connected Place-Transition nets which are live, but not bounded.*
- *Find further examples of weakly connected Place-Transition nets which are not live, but bounded.*
- *Disprove:*
  PN strongly connected $\Longrightarrow$ PN is live and bounded.

5.4 Analysis of Place-Transition Nets

Verifying that a given Place-Transition net satisfies certain properties is typical of the type of analysis we may wish to do. E.g., we want to show that the Place-Transition net is live.
The most common way to analyse a Place-Transition net is to analyse its reachability set, since all properties are defined from this reachability set.

5.4.1 Analysis of the Reachability Set

The reachability set of a Petri net, cf. Ex. 5.3 on page 87, is often drawn as a tree, where the nodes of the tree are markings of the Place-Transition net. Two nodes M and M' are connected with an directed arc iff $M[t > M'$ for some $t \in T$. This arc is also labelled with $t \in T$.
The reachability tree can be generated starting with the initial marking of the Place-Transition net and adding directly reachable markings as leaves. Next we proceed with these new markings and determine their directly reachable markings. These markings now become the new leaves of the already generated part of the reachability tree etc. If we reach a marking previously explored we need not continue building the tree any further at that node.

Example 5.4 *The reachability tree of the net in Fig. 5.8 is shown in Fig. 5.9.*

Reachability trees can be transformed directly into graphs by removing multiple nodes and connecting the nodes appropriately. Such a graph is called a reachability graph. The corresponding reachability graph of the reachability tree displayed in Fig 5.9 is shown in Fig. 5.10.
Unfortunately the method of creating the reachability tree by simulating the token game fails in the case of unbounded nets like the one depicted in Fig. 5.11 (see Exercise 5.9). In order to cope with infinite reachability sets a special symbol

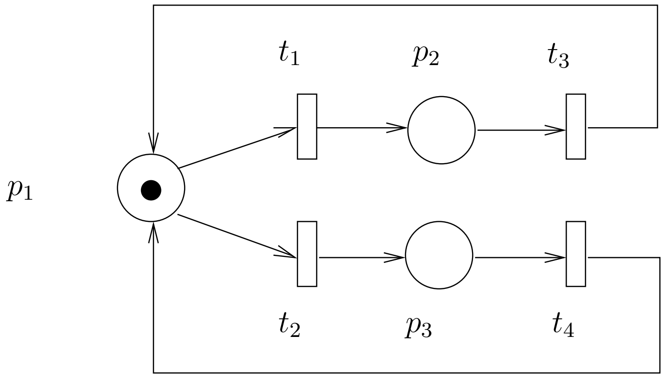


Figure 5.8 A Place-Transition net

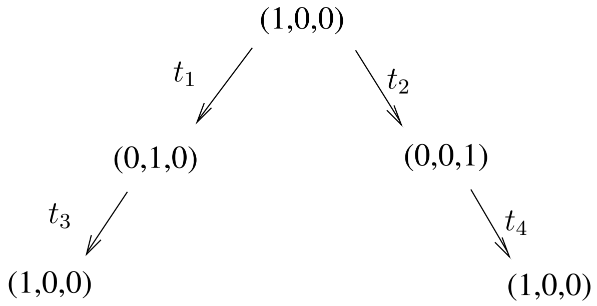


Figure 5.9 Its reachability tree

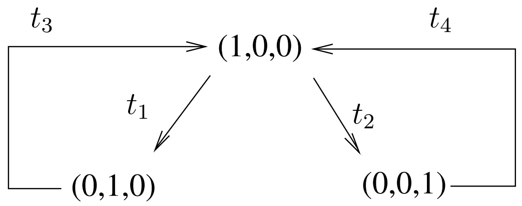


Figure 5.10 The corresponding reachability graph

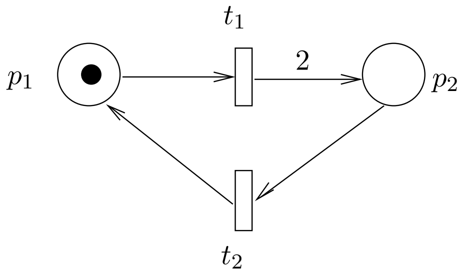


Figure 5.11 Place-Transition net with an infinite reachability set

ω is introduced for unbounded nets to represent the marking of an unbounded place of the Place-Transition net. ω can be referred to as infinity.
The arithmetic rules for ω are: $\forall a \in \mathbb{N}_0 : \omega + a = \omega, \omega - a = \omega,$
$a < \omega, \omega = \omega.$

Definition 5.6 *A marking M covers a marking M', denoted by $M \geq M'$, iff $M(p) \geq M'(p), \forall p \in P$.*

The following algorithm leads to a finite representation of the reachability tree even for unbounded nets. Since the marking of unbounded places is represented by ω, which is larger than every natural number, this representation of the reachability tree is also called a *coverability tree*. The terminal nodes of the tree are called leaves. A node is called an inner node, if it is not a leaf of the tree.

Algorithm 5.1 (cf. [132])

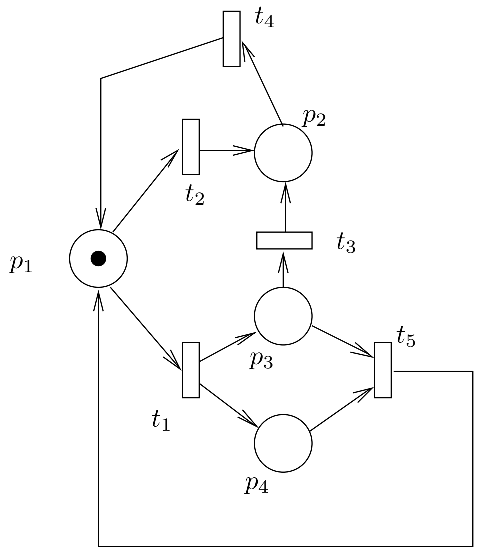


Figure 5.12 Unbounded Place-Transition net

```
$X := \{M_0\}$; /* $M_0$ is the root of the coverability tree */
while $X \neq \emptyset$ do
begin
    Choose $x \in X$.
    $\forall t \in T : x[t >$ do
        create a new node $x'$ given by $x[t > x'$ and connect $x$ and $x'$
        by a directed arc labelled with $t$.
        Check $\forall p \in P$ :
            If there exists a node $y$ on the path from $M_0$ to $x'$ with
            $y \leq x'$ and $y(p) < x'(p)$ then set $x'(p) := \omega$.
    $X := \{x | x$ is a leaf of the coverability tree generated so far,
        in $x$ at least one transition is enabled
        and there is no inner node $y$ with $y = x$ $\}$
end
```

The reason for setting $x'(p) = \omega$ is that x' can be reached from y by a firing sequence $\sigma \in T^*$, i.e. $y[\sigma > x'$. Because $x' \geq y$ $x'[\sigma >$ also holds, leading to a marking x'' with $x'[\sigma > x''$ where $x''(p) > x'(p) > y(p)$. This shows that place p is unbounded, which justifies the use of ω. The coverability tree coincides with the "normal" reachability tree if the net is bounded.

Example 5.5 *The coverability tree of the Place-Transition net of Fig. 5.12 is given in Fig. 5.13. Note that the algorithm does not necessarily lead to a unique coverability tree for a given Place-Transition net, because of the arbitrary choice of $x \in X$.*

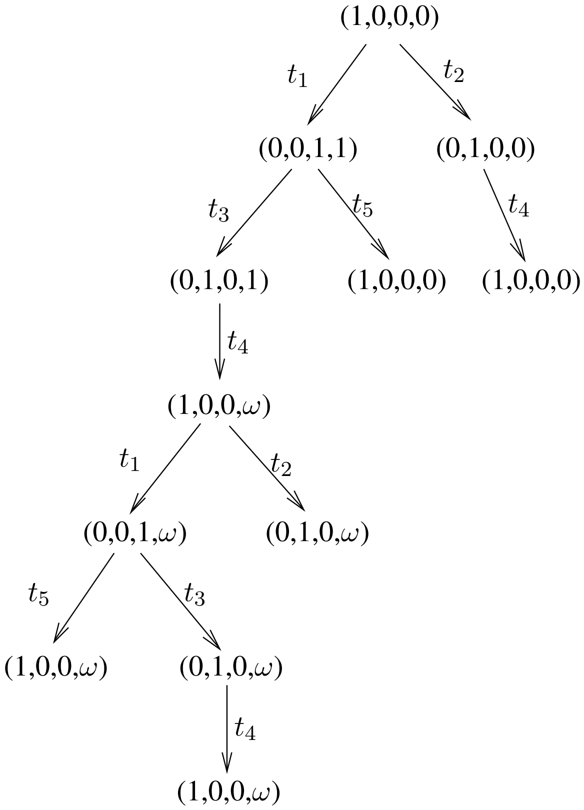


Figure 5.13 Coverability tree of the unbounded Place-Transition net

The coverability tree can be used for the analysis of Place-Transition nets. The following theorem is obviously true.

Theorem 5.2 *PN is bounded iff no node in its coverability tree contains the symbol ω.*

In the case of bounded Place-Transition nets, liveness can also be checked by investigating the coverability tree. The idea is to transform the coverability tree into the corresponding coverability graph, similar to the transformation of the reachability tree into a reachability graph.

Theorem 5.3 *Let PN be bounded. PN is live iff all transitions appear as a label in all terminal strongly connected components of the coverability graph.*

A strongly connected component is terminal if no arcs leave that component. In terms of Markov chains such a subset of markings (states) is called closed (cf. Def. 2.4).

Definition 5.7 $\tilde{R} \subseteq R(PN)$ *is a terminal strongly connected component of* $R(PN,M_0)$ *iff* $\forall M \in \tilde{R}, M' \in R(PN)$:
$(M \to^* M' \Longrightarrow M' \to^* M$ *and* $M' \in \tilde{R})$.

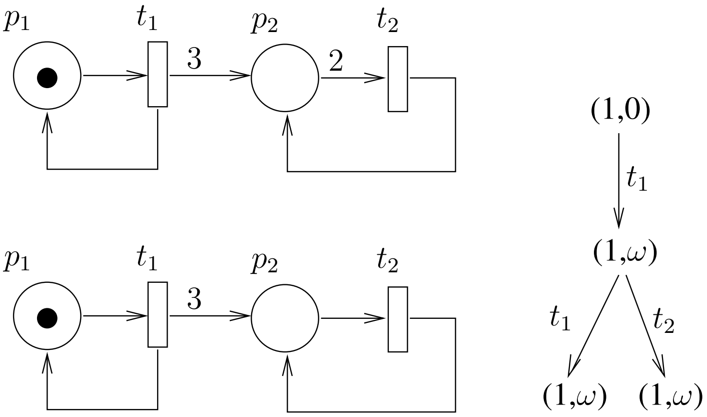


Figure 5.14 Two Place-Transition nets and their coverability trees

Note that, checking the existence of home states is also straightforward for bounded nets.

Theorem 5.4 *Let PN be bounded. A home state exists in a Place-Transition net iff its coverability graph contains exactly one terminal strongly connected component.*

For unbounded Place-Transition nets the information given by their coverability trees is not always sufficient to decide whether home states do exist. Consider the Place-Transition nets of Fig. 5.14, which both have the same coverability tree. In the first net the marking $(1{,}1)$ is a home state, since by firing t_2 the token count on place p_2 can always be reduced to one. This is not possible in the second Place-Transition net, which therefore has no home states.
As you may have noticed this kind of Place-Transition net analysis will lead to large coverability trees/graphs even for fairly simple Place-Transition nets. Even modern computers can not cope with the problem of generating the coverability tree of the Place-Transition net depicted in Fig. 5.15. This problem is also known as the “state space explosion problem”, since the state space or the coverability tree becomes intractably large.
In the following subsections we will learn about analysis techniques which address this problem.

Exercise 5.9 *Determine the reachability set of the unbounded Place-Transition net given in Fig. 5.11.*

Exercise 5.10

1. *Use algorithm 5.1 to determine the coverability tree of the Place-Transition nets of Fig. 5.1 and 5.4.*
2. *Determine the coverability trees of the Place-Transition nets given in Fig. 5.16 ([16]).*

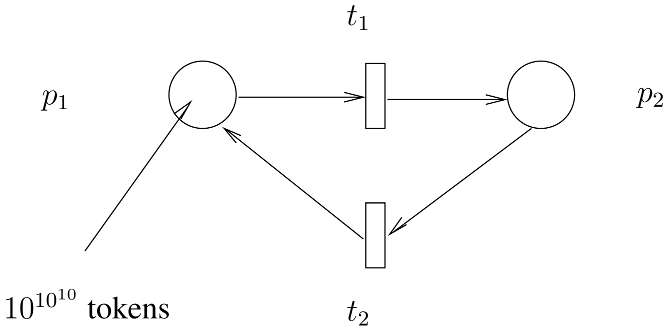


Figure 5.15 Place-Transition net with a huge reachability set

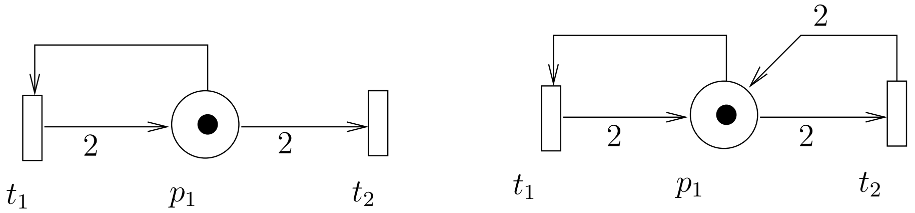


Figure 5.16 Two Place-Transition nets with the same coverability tree

Exercise 5.11 *Show that theorem 5.3 does not hold for unbounded nets?*

5.4.2 Invariant Analysis

We have seen before that a marking can be represented in either functional or in vector notation. Analogously the incidence functions I^- and I^+ can be notated as matrices. These matrices (C^- and C^+) are called the incidence matrices and are defined as follows.

Definition 5.8 $PN = (P,T,I^-,I^+,M_0)$.
The backward incidence matrix $C^- = (c^-_{ij}) \in \mathbb{N}_0^{n\times m}$ *is defined by*
$c^-_{ij} := I^-(p_i,t_j), \forall p_i \in P, t_j \in T,$
the forward incidence matrix $C^+ = (c^+_{ij}) \in \mathbb{N}_0^{n\times m}$ *is defined by*
$c^+_{ij} := I^+(p_i,t_j), \forall p_i \in P, t_j \in T,$
and the incidence matrix *of PN is defined as $C := C^+ - C^-$.*

Enabling and firing of a transition can now be expressed in terms of these incidence matrices. A transition $t_i \in T$ is enabled in a marking M, iff $M \geq C^- e_i$, where e_i is the i-th unit vector $(0,\ldots,0,\underbrace{1}_{i},0,\ldots,0)^T$.

If an enabled transition $t_i \in T$ fires in marking M the successor marking M' is given by

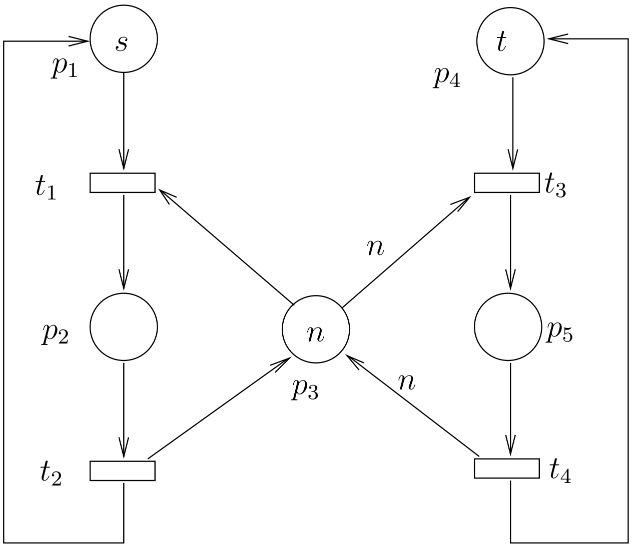


Figure 5.17 $M_0 = (s,0,n,t,0)^T$ and $I^-(p_3,t_3) = I^+(p_3,t_4) = n$

$$M' = M + Ce_i \tag{5.1}$$

Eq. (5.1) states that the firing process can be described by adding the vector Ce_i to the given marking M yielding the successor marking M'. Ce_i is the i-th column of the incidence matrix C specifying the influence of transition $t_i \in T$ on its input and output places. Since markings can be calculated in that way by the addition of vectors, Place-Transition nets can be viewed as vector addition systems with the restriction that only enabled transitions may fire.
Consider $\sigma = t_{k_1} \dots t_{k_j}, j \in \mathbb{N}_0$ a firing sequence with
$M_0[t_{k_1} > \dots M_{j-1}[t_{k_j} > M_j$. Marking M_j can be calculated as follows. For each reachable marking the following equation holds:

$$M_i \quad = \quad M_{i-1} + Ce_{k_i}, \;\; i = 1, \dots, j$$

Substituting the right hand side of the equation for $i = r$ into the equation for $i = r+1$ yields

$$M_j \quad = \quad M_0 + C\sum_{i=1}^{j} e_{k_i}$$

The vector $f := \sum_{i=1}^{j} e_{k_i}$ is called a *firing vector* and is of dimension $|T|$. A component f_i of f characterises the number of occurrences of transition t_i in σ.

Example 5.6 *The Petri net of Fig. 5.17 has the following incidence matrices:*

$$C^- = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & n & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad C^+ = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & n \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

$$C = \begin{pmatrix} -1 & +1 & 0 & 0 \\ +1 & -1 & 0 & 0 \\ -1 & +1 & -n & +n \\ 0 & 0 & -1 & +1 \\ 0 & 0 & +1 & -1 \end{pmatrix}$$

At the initial marking t_1 is enabled and the resultant marking after firing t_1 is given by

$$M' = M_0 + C \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} s \\ 0 \\ n \\ t \\ 0 \end{pmatrix} + \begin{pmatrix} -1 \\ 1 \\ -1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} s-1 \\ 1 \\ n-1 \\ t \\ 0 \end{pmatrix}$$

Assume $s \geq 3$ and $n \geq 3$, then $\sigma = t_1t_1t_1t_2$ is a firing sequence with $f = (3{,}1{,}0{,}0)^T$ its corresponding firing vector. The marking M'' reached after the occurrence of σ is given by

$$M'' = M_0 + C \begin{pmatrix} 3 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} s \\ 0 \\ n \\ t \\ 0 \end{pmatrix} + 3 \begin{pmatrix} -1 \\ 1 \\ -1 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 1 \\ -1 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} s-2 \\ 2 \\ n-2 \\ t \\ 0 \end{pmatrix}$$

Since reaching a marking implies the existence of an appropriate firing sequence the following theorem obviously holds.

Theorem 5.5

$$\forall M \in R(PN,M_0) : \exists f \in \mathbb{N}_0^m : \quad M = M_0 + Cf \tag{5.2}$$

For verifying invariant properties of all reachable markings we can exploit Th. 5.5 as follows. If $v \in \mathbb{Z}^n$ is[2] multiplied with equation 5.2 we get

$$v^TM = v^TM_0 + v^TCf,\ \forall M \in R(PN,M_0). \tag{5.3}$$

Of special interest are those vectors $v \in \mathbb{Z}^n$ satisfying $v^TC = 0$, since Eq. (5.3) becomes $v^TM = v^TM_0$ in those cases. Since v^T and M_0 are known, (5.3) then establishes a condition on all reachable markings M.

Definition 5.9 *$v \in \mathbb{Z}^n$,$v \neq 0$, is a P-invariant iff $v^TC = 0$.*

Theorem 5.6 *If $v \in \mathbb{Z}^n$ is a P-invariant then*
$\forall M \in R(PN,M_0) : v^TM = v^TM_0$.

[2] $\mathbb{Z}$ denotes the set of integers and $\mathbb{N}$ the set of positive integers.

Example 5.7 (Readers/Writers-Problem) *Consider a system with several processes reading or writing a common file. Readers never modify the file, while writers do modify it. So it is obvious that several readers might simultaneously access the file, but for writing operations all other writers and readers must be excluded from accessing the file.*

A possible solution might be to introduce an integer semaphore S, initialised with n, with parameterised P- and V-operations, specifying the value to be subtracted. So our solution might look in pseudo-code:

```
Readers:                          Writers:
    ... (before reading)              ... (before writing)
    P(S,1);                           P(S,n);
    read file;                        write file;
    V(S,1);                           V(S,n);
    ... (after reading)               ... (after writing)
```

The problem is now to verify that this solution is correct, e.g., respects the mutual exclusion condition. Let s denote the number of readers and t the number of writers.

A possible Petri net model of this system is the one of Fig. 5.17, where the places p_1 and p_2 specify the number of readers currently not reading and reading respectively. The places p_4 and p_5 have a similar function for modeling the state of the writers, and place p_3 represents the semaphore.

Solving the system of linear equations $v^T C = 0$ (cf. Ex. 5.6) yields, e.g., the following P-invariants:

$$v_1 = (1,1,0,0,0)^T, v_2 = (0,0,0,1,1)^T, v_3 = (0,1,1,0,n)^T$$

Substituting these solutions for $v^T M = v^T M_0$ we get

$$\begin{array}{rcl} M(p_1) + M(p_2) & = & s \\ M(p_4) + M(p_5) & = & t \\ M(p_2) + M(p_3) + nM(p_5) & = & n, \forall M \in R(PN). \end{array}$$

The first two equations simply express that the number of readers and writers is constant. The last equation allows the following conclusions:

$\forall M \in R(PN)$ we have

a) $M(p_2) \geq 1 \Longrightarrow M(p_5) = 0$

b) $M(p_5) \geq 1 \Longrightarrow M(p_2) = 0$

c) $M(p_5) \leq 1$

In terms of the readers/writers system this means:

- *if a reader is reading, no writer is writing (a);*
- *if a writer is writing, no reader is reading (b);*

- *and finally that at most one writer is writing (c).*

Furthermore we can conclude that $M(p_2) \leq n$ showing that we should choose $n \geq s$ to allow all readers to operate in parallel. Since we have proved that our Petri net model has the desired properties, we know the same for our system provided the Petri net reflects the modelled system behaviour correctly.

We have seen that Th. 5.6 assists us in the analysis of Petri nets. For live Place-Transition nets the converse of Th. 5.6 also holds.

Theorem 5.7 *Let PN be a live Place-Transition net and $v \in \mathbb{Z}^n, v \neq 0$. If $v^T M = v^T M_0, \forall M \in R(PN, M_0)$ then v is a P-invariant.*

Proof. Since PN is live we have $\forall t_i \in T : \exists M, M' \in R(PN, M_0) : M[t_i > M'$ and $M' = M + Ce_i$ giving $v^T M' = v^T M + v^T Ce_i$. Since $v^T M = v^T M_0$ this yields $v^T Ce_i = 0$. Since this holds for all transitions we get $\forall t_i \in T : v^T Ce_i = 0$ and thus $v^T C = 0$, showing that v is a P-invariant. □

P-invariants are sometimes called S-invariants, because some authors denote the set of places by S after the German word "Stellen" ($\equiv$ "Places").
P-invariants can be employed for checking the boundedness property.

Definition 5.10 *PN is covered by positive P-invariants iff*
$\forall p_i \in P : \exists$ P-invariant $v \in \mathbb{Z}^n$ with $v \geq 0$ and $v_i > 0$.

Theorem 5.8 *PN is covered by positive P-invariants $\Longrightarrow$ PN is bounded.*

Proof. PN is covered by positive P-invariants $\Longrightarrow \exists$ P-invariant $v \in \mathbb{N}_0^n : v_i > 0, \forall i \in \{1, \ldots, n\}$. With $v^T M = v^T M_0 = const., \forall M \in R(PN, M_0)$, we get $v_i M(p_i) \leq v^T M_0, \forall p_i \in P$, since $M(p) \geq 0, \forall M \in R(PN, M_0)$.
$v_i > 0$ yields $M(p_i) \leq \frac{v^T M_0}{v_i}, \forall p_i \in P$ proving that the net is bounded. □

If the net is not covered by positive P-invariants, but there are some positive P-invariants, then at least these covered places are bounded.

Corollary 5.1 *If there exists a P-invariant $v \in \mathbb{Z}^n : v \geq 0, v_i > 0$ then p_i is bounded, i.e. $\exists k \in \mathbb{N} : \forall M \in R(PN, M_0) : M(p_i) \leq k$.*

Proof. The proof follows that of Th. 5.8. □

Another invariant which is useful in Place-Transition net analysis is called the T-invariant. Suppose we have a marking M and after firing of some transitions we want to reach M again. So let $f \in \mathbb{N}_0^m$ be the corresponding firing vector. Since we started in M and reached M again the following equation holds (cf. Eq. (5.2))

$$M = M + Cf$$

which implies $Cf = 0$. In order to return to a marking of the Place-Transition net it is necessary that the firing vector satisfies $Cf = 0$. This leads to the following definition.

Definition 5.11 $w \in \mathbb{Z}^m, w \neq 0$, *is a T-invariant iff* $Cw = 0$.

Definition 5.12 *PN is covered by positive T-invariants iff*
$\forall t_i \in T : \exists$ *T-invariant* $w \in \mathbb{Z}^m$ *with* $w \geq 0$ *and* $w_i > 0$.

Covering by T-invariants is a necessary condition for a Place-Transition net to be bounded and live.

Theorem 5.9 *PN is bounded and live* $\Longrightarrow$ *PN is covered by positive T-invariants.*

Proof. Since PN is bounded its reachability set is finite.
Let $\tilde{R} \subseteq R(PN, M_0)$ be a terminal strongly connected subset of $R(PN, M_0)$.
Choose an arbitrary marking $M \in \tilde{R}$. M can always be reached after some transition firings, since $\tilde{R}$ is strongly connected.
Since PN is live there $\exists f \in \mathbb{N}_0^m : f_i > 0, \forall i \in \{1, \ldots, m\} : M = M + Cf$, which implies $Cf = 0$.
Thus f is a positive T-invariant covering all transitions in the Place-Transition net. □

Corollary 5.2 *Let PN be bounded.*
If $t_i \in T$ *is live then* $\exists$ *T-invariant* $w \in \mathbb{Z}^m : w \geq 0, w_i > 0$.

Proof. The proof is similar to that of Th. 5.9. □

Analysing a Place-Transition net by calculating its invariants is in most cases more time and space efficient than inspecting the reachability set, since the complexity of this kind of analysis depends only on the number of places and transitions and not on the size of the reachability set. In [117] an algorithm is described for determining the invariants of a Place-Transition net. This algorithm is e.g. part of the GreatSPN-tool [50] and the QPN-tool (cf. [29]) as well.
A Place-Transition net can also be analysed very efficiently if its structure is restricted. Imposing restrictions on the structure of Place-Transition nets leads to several classes of Place-Transition nets, which will be our next topic.

Exercise 5.12 *Determine the incidence matrices of the Place-Transition net given in Fig. 5.4.*
From these prove if $M_0 \geq C^- e_1$ *and* $M_0 \geq C^- e_2$ *holds. What exactly does that mean?*
Calculate: $M_1 = M_0 + Ce_2$ *and* $M_2 = M_1 + Ce_5$

Exercise 5.13 *Prove, that if* v_1 *and* v_2 *are P-invariants then*

- $v_1 + v_2$
- $rv_1, r \in \mathbb{Z}$

are also P-invariants.

Exercise 5.14 *Disprove the converse of Theorem 5.8.*

5.4.3 Analysis of Net Classes

The analysis of net classes has been a topic for Petri net researchers since the ′70s. In this section we give an overview on the most important results, paying particular attention to Extended Free Choice (EFC)-nets.

Definition 5.13 *Let* $PN = (P,T,I^-,I^+,M_0)$ *be a Place-Transition net with* $I^-(p,t),I^+(p,t) \in \{0,1\},\forall p \in P,t \in T$. *PN is called a*

1. State Machine (SM) *iff* $\forall t \in T : |\bullet t| = |t\bullet| \leq 1$.[3]

2. Marked Graph (MG) *iff* $\forall p \in P : |\bullet p| = |p\bullet| \leq 1$

3. Free Choice net (FC-net), *iff*
$\forall p,p' \in P : p \neq p' \Longrightarrow (p\bullet \cap p'\bullet = \emptyset$ *or* $|p\bullet| = |p'\bullet| \leq 1)$.

4. Extended Free Choice net (EFC-net), *iff*
$\forall p,p' \in P : p\bullet \cap p'\bullet = \emptyset$ *or* $p\bullet = p'\bullet$.

5. Simple net (SPL-net), *iff*
$\forall p,p' \in P : p \neq p' \Longrightarrow (p\bullet \cap p'\bullet = \emptyset$ *or* $|p\bullet| \leq 1$ *or* $|p'\bullet| \leq 1)$.

6. Extended Simple net (ESPL-net), *iff*
$\forall p,p' \in P : p\bullet \cap p'\bullet = \emptyset$ *or* $p\bullet \subseteq p'\bullet$ *or* $p'\bullet \subseteq p\bullet$.

Fig. 5.19 provides a description of these different classes of nets. For these net classes the following relations hold (cf. Fig. 5.18):
Marked Graphs ⊂ FC-nets, State Machines ⊂ FC-nets ⊂ EFC-nets ⊂ ESPL-nets and FC-nets ⊂ SPL-nets ⊂ ESPL-nets.
We first of all investigate State Machines, which can be analysed very easily.

Analysis of State Machines

All transitions of a State Machine have at most one input and one output place. Since we are interested in live and bounded nets, Th. 5.1 tells us that we only have to consider strongly connected State Machines. Fig. 5.20 depicts such a State Machine.
If a State Machine is strongly connected every transition has exactly one input and one output place. A token in a strongly connected State Machine can be "moved" from one place to every other place of the net, since this token directly "enables" all output transitions of the place it is currently located. This observation yields the following characterisation of liveness.

Theorem 5.10 (Liveness in State Machines [38]) *Let PN be a State Machine with* $|\bullet t| = |t\bullet| = 1, \forall t \in T$.
PN is live iff PN is strongly connected and $M_0 \neq 0$.

[3] $|x|$ denotes the number of elements in the set x.

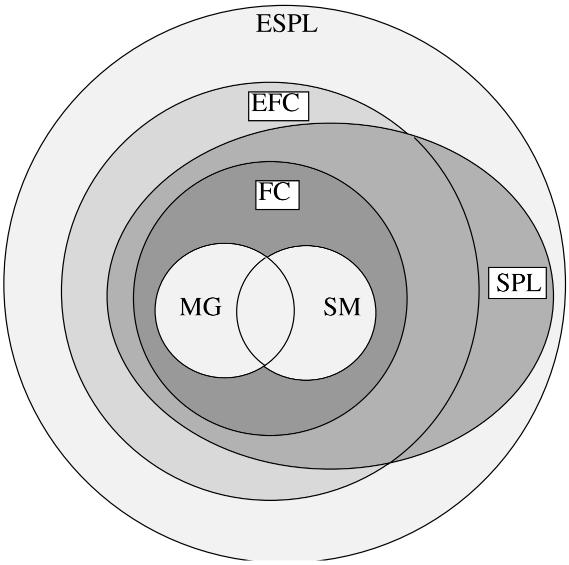


Figure 5.18 Relationship of net classes

Firing of transitions in a strongly connected State Machine obviously does not change the number of tokens in the Place-Transition net and thus the net is bounded.

Theorem 5.11 (Boundedness in State Machines [38]) *Let PN be a live State Machine with* $|\bullet t| = |t\bullet| = 1, \forall t \in T$.
Then PN is bounded.

The former argument for liveness also implies, that a token can always return to a place it started from. So we can always reach the initial marking.

Theorem 5.12 (Home states in State Machines) *If PN is a live and bounded State Machine then* M_0 *is a home state.*

Th. 5.12 directly implies that all reachable markings of a live and bounded State Machine are home states.
We have seen that State Machines are very easy to analyse. Let us investigate the next net class.

Analysis of Marked Graphs

Similar to State Machines we only consider strongly connected Marked Graphs. This implies that each place has exactly one input and one output transition. A typical example of a Marked Graph is shown in Fig. 5.21.
Playing the token game for this Place-Transition net, one realises that, e.g., the token on p_5 seems to cycle around. Note that we have avoided speaking so far

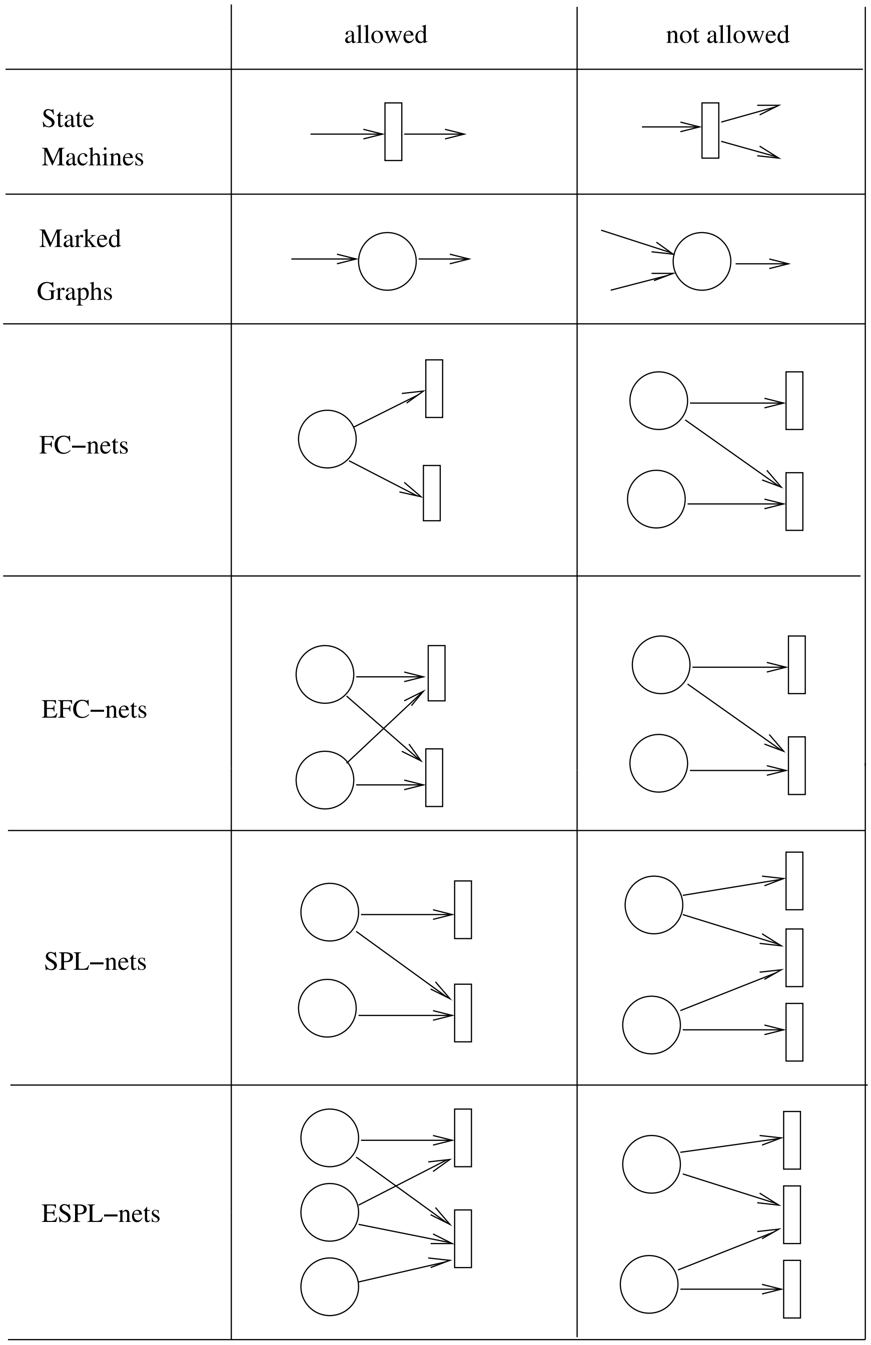


Figure 5.19 Illustrating different net classes

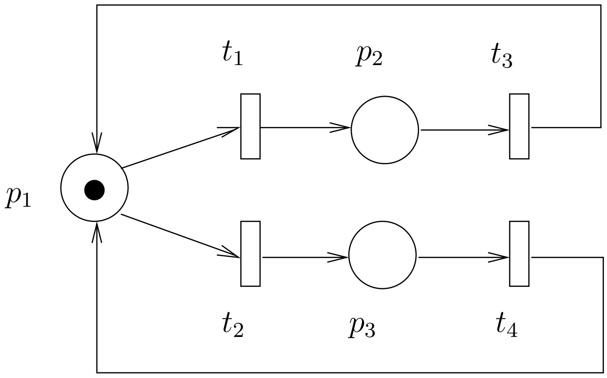


Figure 5.20 Example of a State Machine

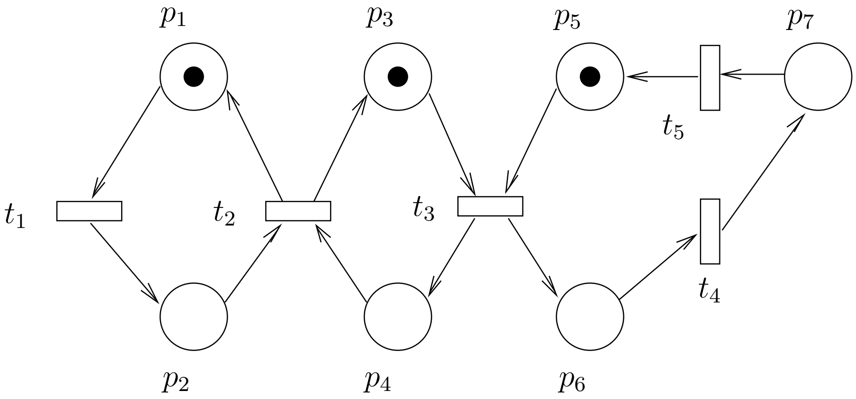


Figure 5.21 An example of a Marked Graph

about tokens moving in a Place-Transition net, since a transition might destroy more or less tokens than it creates. But in this context it makes sense. So the token on p_5 is first fired on p_6 than on p_7 and afterwards returns to p_5. If we modify the initial marking just by removing the token on p_5, we also see that the net is not live. It seems that these cycles play an important role in the context of Marked Graphs. So let us define them:

Definition 5.14 *A* cycle *is a sequence* $x_0, \ldots ,x_k$ *where* $x_i \in P \cup T, 0 \le i \le k$ *and* $x_{i+1} \in x_i \bullet , \forall i \in \{0, \ldots ,k-1\}$ *and* $x_0 = x_k$.
A cycle is called simple *iff* $\forall x_i,x_j \in x_0, \ldots ,x_k : i \neq j \Longrightarrow$
$x_i \neq x_j$ *or* $(i = 0$ *and* $j = k)$.

A simple cycle is a cycle where each element appears once, except for the first and last element. The Marked Graph of Fig. 5.21, e.g., contains amongst others the following simple cycles:
p_1,t_1,p_2,t_2,p_1 and $p_5,t_3,p_6,t_4,p_7,t_5,p_5$.
The following result should now be no surprise for us.

Theorem 5.13 (Liveness in Marked Graphs [150]) *Let PN be a Marked Graph with* $|\bullet p| = |p \bullet| = 1$.
PN is live iff every simple cycle contains a place with at least one token at M_0.

Determining boundedness of a Marked Graph can also be based on the concept of simple cycles.

Definition 5.15 *PN is* covered by simple cycles *iff* $\forall x \in P \cup T : \exists$ *a simple cycle* $x_0, \ldots, x_k$ *with* $x = x_i$ *for some* $i \in \{0, \ldots, k\}$.

Theorem 5.14 (Boundedness in Marked Graphs [150]) *A live Marked Graph PN is bounded iff PN is covered by simple cycles.*

Similar to our discussion concerning State Machines Fig. 5.21 shows that M_0 is a home state. After firing of every transition M_0 is reached again. In fact Marked Graphs unveil the same property as State Machines, i.e.

Theorem 5.15 (Home states in Marked Graphs [125]) *If PN is a live and bounded Marked Graph then* M_0 *is a home state.*

So far we have seen that the two net classes, State Machines and Marked Graphs, are very simple to analyse. In the next subsection we draw attention to EFC-nets, since they are a superset of FC-nets and we can establish some useful analysis techniques for this "extended" class directly.

Analysis of EFC-nets

According to the definition, EFC-nets unveil the possibilities of connecting transitions to their input places as shown in Fig. 5.22.

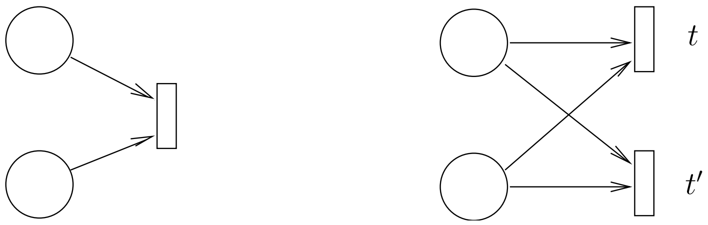


Figure 5.22 Possible connections between transitions and their input places in EFC-nets

With that the following is obvious: If a transition t of an EFC-net is enabled then all transitions which are in structural conflict, i.e. all $t' \in (\bullet t)\bullet$, are enabled as well. So amongst conflicting transitions, we can choose the transition to fire freely.
For the analysis of FC- and EFC-nets we need to understand the terms "deadlock" and "trap".

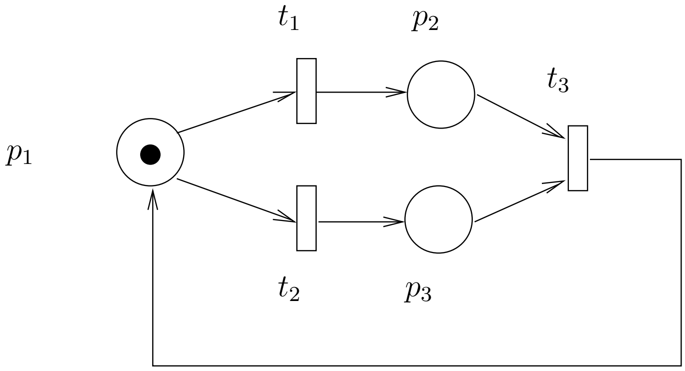


Figure 5.23 non-live FC-net

Definition 5.16

1. *$P' \subseteq P, P' \neq \emptyset$ is a* deadlock[4] *iff* $\bullet P' \subseteq P'\bullet$.
2. *$P' \subseteq P, P' \neq \emptyset$ is a* trap *iff* $P'\bullet \subseteq \bullet P'$.

If a deadlock empties, this subset will remain empty. On the other hand once a trap is marked, it will remain marked.
Consider the FC-net in Fig. 5.23, which has the deadlocks $\{p_1,p_2\}$, $\{p_1,p_3\}$ and $\{p_1,p_2,p_3\}$, which is also the only trap of the net. At the initial marking all deadlocks are marked. Firing t_1 now empties the deadlock $\{p_1,p_3\}$ and thus this deadlock will remain empty. Since $\{p_1,p_3\}$ is empty all transitions in $\{p_1,p_3\}\bullet$ are not enabled and are therefore not live. An analogous argument holds for the deadlock $\{p_1,p_2\}$ if t_2 fires first at the initial marking.
A natural idea is to avoid such situations by ensuring that a deadlock can never be emptied. This is, e.g., the case if the deadlock contains a marked trap, because a trap remains marked in all further reachable markings. For EFC-nets this idea yields a characterising condition for liveness:

Theorem 5.16 (Liveness in EFC-nets [34, 38]) *An EFC-net is live iff every deadlock contains a marked trap at M_0.*

This condition is often called the deadlock/trap-property (dt-property).

Proof.
We will only prove sufficiency.
Assume the EFC-net is not live.
Then $\exists t \in T : M \in R(PN,M_0) : \forall M' \in R(PN,M) : \neg M'[t >$, i.e. we can find a transition t and a marking M, such that t is not enabled in all further reachable markings. The salient property of EFC-nets is now that there must be a fixed place which is and remains empty, i.e.

[4] Other authors refer to deadlocks as *siphons* or *tubes*.

$\exists p \in \bullet t : M'(p) = 0, \forall M' \in R(PN,M)$.
The existence of such a place p is ensured by the fact that all transitions of $(\bullet t)\bullet$ are also not live. Furthermore since p remains empty all its input transitions are not live, i.e.
$\forall \tilde{t} \in \bullet p : \forall M' \in R(PN,M) : \neg M'[\tilde{t} >$.
Let P_{empty} denote the set of empty places in $R(PN,M)$, i.e.
$\forall p \in P_{empty}, M' \in R(PN,M) : M'(p) = 0$
and let T_{dead} be the set of transitions which are not live in $R(PN,M)$, i.e.
$\forall t \in T_{dead}, M' \in R(PN,M) : \neg M'[t >$.
This yields $\bullet P_{empty} \subseteq T_{dead}$ and because of the EFC-net structure we get $P_{empty}\bullet = T_{dead}$. Thus $\bullet P_{empty} \subseteq P_{empty}\bullet$ and P_{empty} is an empty deadlock, which contains no marked trap. □

In [104, 107] efficient polynomial-time algorithms are described for determining the dt-property for FC-nets.
Checking boundedness of EFC-nets is somewhat more complicated. In the following we present only the main theorem without further discussion. Before we can establish this theorem we need the following definitions.

Definition 5.17 *Let $PN = (P,T,I^-,I^+,M_0)$ and $PN' = (P',T',I'^-,I'^+,M'_0)$ be two Place-Transition nets and $X \subseteq P \cup T$.*

1. *PN' is a subnet of PN iff*
   $P' \subseteq P, T' \subseteq T$ and
   $I'^-(p,t) = I^-(p,t), I'^+(p,t) = I^+(p,t), \quad \forall p \in P', t \in T'$, and
   $M'_0(p) = M_0(p), \forall p \in P$.
2. *PN' is a subnet of PN generated by X iff*
   $P' = (X \cap P) \cup \bullet(X \cap T) \cup (X \cap T)\bullet$,
   $T' = (X \cap T) \cup \bullet(X \cap P) \cup (X \cap P)\bullet$, and
   $I'^-(p,t) = I^-(p,t), I'^+(p,t) = I^+(p,t), \quad \forall p \in P', t \in T'$, and
   $M'_0(p) = M_0(p), \forall p \in P$.

Definition 5.18

1. *$PN' = (P',T',I'^-,I'^+,M'_0)$ is a* P-component[5] *of $PN = (P,T,I^-,I^+,M_0)$ iff PN' is a subnet of PN generated by $P' \subseteq P$ and $\forall t' \in T' : |\bullet t' \cap P'| \leq 1$ and $|t'\bullet \cap P'| \leq 1$, where the $\bullet$-notation is with respect to PN'.*
2. *PN is covered by P-components iff $\forall x \in P \cup T : \exists$ a P-component PN' of PN with $x \in P' \cup T'$.*

Theorem 5.17 (Boundedness in EFC-nets [35]) *A live EFC-net PN is bounded iff PN is covered by strongly connected P-components.*

[5] A P-component is related to a State Machine.

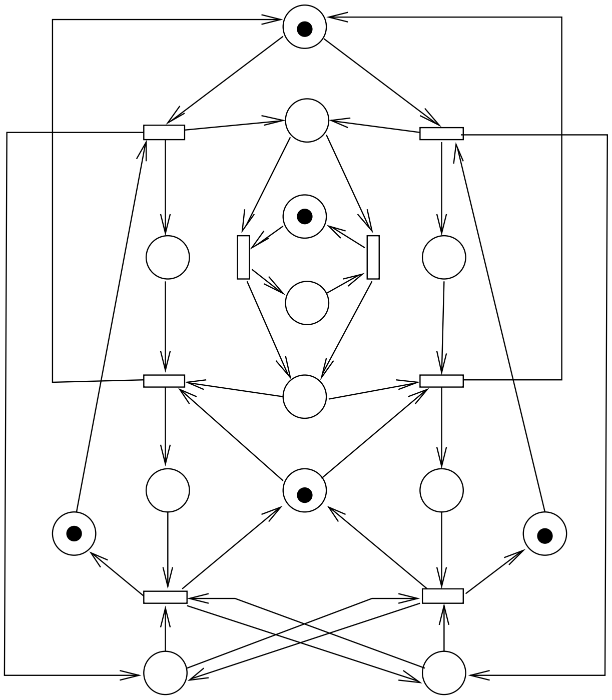


Figure 5.24 Live and bounded Place-Transition net with no home states

Research efforts to determine the existence of home states for EFC-nets have only been undertaken in the recent past. Some authors conjectured that all bounded and live Place-Transition nets have home states, e.g. [124]. This conjecture was proven to be false in 1984, where E. Best and K. Voss [39] published a bounded and live ESPL-net without any home states (cf. Fig. 5.24). Since 1989 it is known that the above-mentioned conjecture holds for EFC-nets.

Theorem 5.18 (Home states in EFC-nets [36, 178]) *If PN is a live and bounded EFC-net then its reachability set contains home states.*

Furthermore we can characterise all home states of a live and bounded EFC-net as follows.

Theorem 5.19 ([36]) *Let PN be a live and bounded EFC-net. Then*
M is a home state iff M marks all traps of PN.

Proof. We will only prove necessity.
Let $\tilde{P} \subseteq P$ be an arbitrary trap of PN and $p \in \tilde{P}$. Since PN is live and bounded, it is strongly connected (cf. Theorem 5.1) and $\exists t \in \bullet p$. Since PN is live $\exists \tilde{M},\tilde{M}' \in R(PN,M_0) : \tilde{M}[t > \tilde{M}'$. $\tilde{M}'$ is a marking which marks $\tilde{P}$. Since M is a home state, we have $\tilde{M}' \rightarrow^* M$ and thus M marks $\tilde{P}$. □

This characterisation of home states can be exploited as follows.

Theorem 5.20 ([36]) *Let PN be a live and bounded EFC-net and $\sigma \in T^*$ be a firing sequence with $\forall t \in T : \#(\sigma,t) > 0$.*[6] *Then M given by $M_0[\sigma > M$ is a home state.*

Proof. After the firing sequence σ has occurred, all traps are marked, since a trap being marked remains marked. □

Th. 5.20 is important for analysing Stochastic Petri nets, discussed in Ch. 8, by means of simulating the net. Once each transition has fired, all further reached markings are recurrent. These are exactly the states (markings) we are interested in when analysing the steady state behaviour of the associated Markov process (cf. Ch. 8). Hence EFC-nets are a convenient class of Place-Transition nets, because there are characterising conditions for boundedness and liveness and furthermore the existence of home states is guaranteed for bounded and live EFC-nets.

For Place-Transition nets with more complex structures such theorems are not known up to now. E.g. it is known that the dt-property is sufficient for ensuring liveness in ESPL-nets, but not necessary (see [38]).

Another analysis technique which has evolved in the last decade to analyse also more complex Place-Transition nets efficiently is the subject of the next section.

Exercise 5.15 *Prove the relations between the net classes as given in Fig. 5.18.*

Exercise 5.16 *Let $Q_1,Q_2 \subseteq P$. Prove the following conjectures:*

1. *Q_1,Q_2 are deadlocks $\Longrightarrow Q_1 \cup Q_2$ is a deadlock.*
2. *Q_1,Q_2 are traps $\Longrightarrow Q_1 \cup Q_2$ is a trap.*

Exercise 5.17 *Prove or disprove:*

1. *If $PN = (P,T,I^-,I^+,M_0)$ is a live EFC-net then $PN = (P,T,I^-,I^+,M)$ is live $\forall M \geq M_0$.*
2. *If $PN = (P,T,I^-,I^+,M_0)$ is a live Place-Transition net then $PN = (P,T,I^-,I^+,M)$ is live $\forall M \geq M_0$.*

Exercise 5.18 *Show that the Place-Transition net of Fig. 5.24 is bounded and live, but has no home states.*

Exercise 5.19 *Prove or disprove:*

1. *If PN is a live and bounded EFC-net then M_0 is a home state.*
2. *Let PN be a live and bounded Place-Transition net. Then M is a home state $\Longrightarrow M$ marks all traps of PN.*

Exercise 5.20 *Prove that if PN is a live and bounded Marked Graph or a live and bounded State Machine then M_0 marks all traps of PN.*

[6] $\#(\sigma,t)$ denotes the number of occurrences of $t \in T$ in $\sigma \in T^*$.

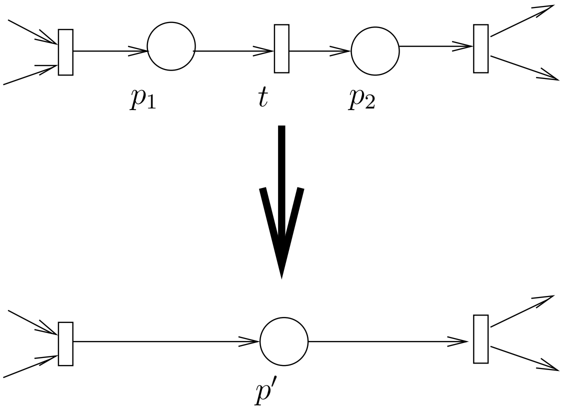


Figure 5.25 Simple reduction rule

5.4.4 Reduction and Synthesis Analysis

Reduction analysis deals with the reduction of the Place-Transition net by replacing subnets of the net by less complex subnets such that several properties remain invariant. Since the reduced Place-Transition net is less complex than the original net, once sufficiently reduced, the analysis can be performed, e.g., on the coverability/reachability tree of the reduced Place-Transition net. A simple example of a reduction is shown in Fig. 5.25.

This particular reduction rule specifies that if two empty places (p_1,p_2) and one transition (t) with $|\bullet p_1| = |p_1 \bullet| = |\bullet p_2| = |p_2 \bullet| = |\bullet t| = |t \bullet| = 1$ and $p_1 \bullet \ni t \in \bullet p_2$ are given, these three elements may be reduced to one empty place p' with $\bullet p' = \bullet p_1$ and $p' \bullet = p_2 \bullet$. It is obvious that properties like boundedness and liveness are not affected by this reduction (transformation) of the net.

Several authors have independently developed several sets of such reduction rules, e.g. [31, 32, 62, 125, 163]. We only describe one example of a typical reduction rule. The interested reader is referred to the literature.

This transformation from [31] removes redundant places by removing arcs from transitions to the place. If all arcs are removed then the place can be removed from the Place-Transition net. A place is redundant “when its marking is always sufficient for allowing firings of transitions connected to it” [31]. This is expressed by the following definition.

Definition 5.19
Let $PN = (P,T,I^-,I^+,M_0)$ be a Place-Transition net. A place $p \in P$ is redundant *iff there exists a subset $Q \subseteq P \setminus \{p\}$ (possibly empty) and a weighting function $V : Q \cup \{p\} \mapsto \mathbb{N}$ such that the following conditions hold:*

1. $\exists b \in \mathbb{N}_0 : V(p)M_0(p) - \sum_{q \in Q} V(q)M_0(q) = b$,
i.e. the weighted marking of p is greater or equal to the weighted marking of the places in Q.

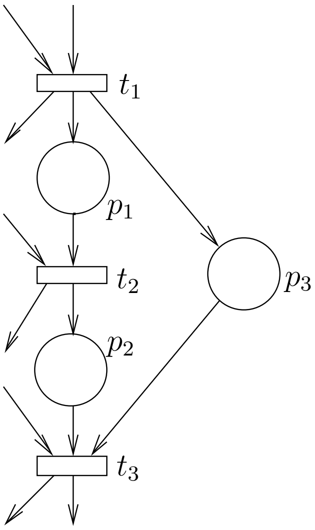


Figure 5.26 Portion of a Place-Transition net with a redundant place

2. $\forall t \in T : V(p)I^-(p,t) - \sum_{q \in Q} V(q)I^-(q,t) \leq b$,
   i.e. firing of a transition removes less or equal tokens on p than from the places in Q with respect to the weight function.

3. $\forall t \in T : \exists c_t \in \mathbb{N}_0$:
   $V(p)(I^+ - I^-)(p,t) - \sum_{q \in Q} V(q)(I^+ - I^-)(q,t) = c_t$,
   i.e. with respect to the weight function there are a greater or equal number of tokens placed onto p than on the places of Q after firing of t.

After having identified the redundant place the transformation can be performed by removing all edges in $\bullet p$ and $p\bullet$. For every transition, $t \in T$, an edge with weight c_t is added from t to p. If all the weights c_t are zero then the place p is isolated and can be removed. Simplification of redundant places preserves properties like boundedness, liveness and the existence of home states (cf. [31]).

Example 5.8 *Figure 5.26 shows a portion of a Petri net with a redundant place p_3. Figure 5.27 shows the same portion of the Place-Transition net after the transformation has been performed. In this example we have $Q = \{p_1,p_2\}$; $V(p) = 1, \forall p \in Q \cup \{p_3\}$; $b = 0$; $c_t = 0, \forall t \in T$, and the place p_3 has been removed.*

Most of these transformation/reduction rules can be applied in the reverse direction, thus increasing the size of the Place-Transition net. This form of building a net is called synthesis of Petri nets. If we start with a Place-Transition net which embraces all positive properties like boundedness and liveness and apply these synthesis rules, we always yield a Place-Transition net with the same properties making the analysis of the target net unnecessary.

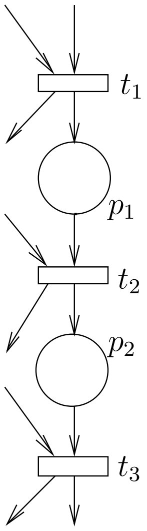


Figure 5.27 Portion of the Place-Transition net after removing the redundant place

Reduction and synthesis techniques are analysis techniques which have not been examined systematically in the literature. Several sets of rules have been published, but it is, e.g., not known how powerful all these published rules are. E.g. are all life and bounded simple-nets completely reducible to an atomic net $(PN = (\{p\},\{t\},I^-,I^+,M_0)$ where $I^-(p,t) = I^+(p,t) = 1$ and $M_0(p) > 0)$? Only for FC-nets this and similar questions have been considered, see e.g. [69, 70, 72]. Furthermore the effort of testing the application conditions is not always considered in the literature. In [62] the computational complexity of several reduction and synthesis rules is examined.

5.5 Further Remarks on Petri Nets

This introduction covers only a part of current Petri net theory. We have concentrated on those properties of a Petri net which are also significant for the performance analysis of the corresponding Stochastic Petri net.
Other properties which are of interest are e.g.

1. fairness, i.e.
   $\forall t \in T : \exists k_t \in \mathbb{N} : \forall t' \in T, \forall \sigma \in T^* : M_0[\sigma >: \#(\sigma,t') - \#(\sigma,t) \leq k_t$, where $\#(\sigma,t)$ denotes the number of occurrences of $t \in T$ in $\sigma \in T^*$.

   Fairness means that each transition will eventually fire and can not be excluded from firing. The sender/receiver system given in Fig. 5.3 is fair, whereas the Place-Transition net in Fig. 5.8 is not fair, since t_2 is not included in the infinite firing sequence $(t_1 t_3)^+$.

2. reversibility, i.e. M_0 is a home state of the Place-Transition net.

3. persistence, i.e. $\forall t_i, t_j \in T, t_i \neq t_j$ and $\forall M \in R(PN, M_0)$:
   $M[t_i >$ and $M[t_j > \Longrightarrow M[t_i t_j >$.

   Persistence means that a transition can only be disabled by firing of itself. Marked Graphs are e.g. a subclass of persistent nets.

4. properties of the net language where a language of a Place-Transition net is given by $\{W(\sigma) | \sigma \in T^* : M_0[\sigma >\}$ with
   $W : T \mapsto$ "set of symbols (words)" and
   $W(t\tilde{\sigma}) := W(t)W(\tilde{\sigma}), \forall t \in T, \tilde{\sigma} \in T^*$.

5. synchronic distance sd of two transitions $t_i, t_j \in T$. This notion is related to fairness and defined as follows:
   $sd(t_i, t_j) := sup\{|\#(\sigma, t_i) - \#(\sigma, t_j)| \;\; |M[\sigma >, \forall M \in R(PN, M_0)\}$.

Finally we turn to the question of whether an algorithm exists to decide whether a Place-Transition net is live or not. This decidability problem remained open for about 20 years until 1981.

In fact it was well known since the early days of Petri net theory that the above mentioned problem is equivalent to the well-known reachability problem which can be described as follows:

Given a Place-Transition net and a marking M, does an algorithm exists to decide whether $M \in R(PN, M_0)$? As we learned in Ex. 5.11, the coverability tree is not suitable for deciding both problems. In 1981 E.W. Mayr [118, 119] and in 1982 R. Kosaraju [109] published algorithms for the reachability problem, i.e. reachability and liveness problem are both decidable. Furthermore it is known that both problems are exponential-space hard with complexity exponential in $|P|, |T|$ and the flow relation $|F|$ (cf. [119]).

Exercise 5.21 (cf. [146]) *This exercise is dedicated to some real life problems in modelling computer systems. It concerns the 'mutual exclusion' problem, which occurs if several parallel processes are sharing common resources. If one process is, e.g., changing the internal state of a resource (consider writing a file), then for all other processes access has to be denied, in order to keep the state of the resource consistent.*

A very common solution is to restrict simultaneous access to a resource by modifying the source code of each process and insert special lines of code.

E.g. a control variable is introduced, which represents the access status of a resource (free or occupied) and each process has to check this control variable before using the resource. The section of code where the resource is used is referred to as the "critical section". The crux of all such solutions is the atomicity or non-atomicity of certain statements. In this exercise we will denote atomic statements (operations) in pointed brackets, e.g. <x := y; >.

In the following, four algorithms are presented in pseudo-code, and you are asked to prove or disprove the correctness of these algorithms using Petri nets.

For simplicity we only look at two processes P_0 and P_1 in the first three algorithms and assume that the presented part of the code of each process is executed infinitely often.

1. *Algorithm:*
   Control variables:
   var flag : array [0..1] of boolean;
   turn : 0..1;
   Initial value of turn is 0 and all flags are false

   Program for process P_i:

   ```
   <flag[i] := true;>
   while <flag[j]> do /* $j = i + 1 \bmod 2$ */
       if <turn = j> then
           begin
               <flag[i] := false;>
               while <turn = j> do <nothing> enddo;
               <flag[i] := true; >
           endif;
   enddo;
   critical section;
   <turn := j>;
   <flag[i] := false;>
   ```

2. *Algorithm:*
   Control variables:
   var flag : array [0..1] of boolean;
   turn : 0..1;
   Initial value of turn is 0 and all flags are false

   Program for process P_i:

   ```
   <flag[i] := true;>
   while <turn ≠ i> do
       while <flag[j]> do <nothing> enddo; /* $j = i + 1 \bmod 2$ */
       <turn := i;>
   enddo;
   critical section;
   <flag[i] := false;>
   ```

3. *Algorithm:*
   This algorithm uses a binary semaphore sem. P(sem) *decreases the value*

of sem *by* 1*, provided it is greater than* 0*, otherwise the process has to wait until* sem *becomes* > 0. V(sem) *increases the value of* sem *by* 1.
Control variables:
var sem : binary semaphore;
Initial value of sem is 1

Program for process P_i*:*

```
< P(sem); >
critical section;
< V(sem); >
```

4. *Algorithm:*
In this problem we have 5 *processes* $P_0, \ldots, P_4$ *sharing common resources. The restriction we have is that if process* P_i *is in its critical section, then (*$P_{i-1} \bmod 5$*) and (*$P_{i+1} \bmod 5$*) are both not able to be or to enter their own critical section.*
This algorithm uses an array of binary semaphores given by

Control variables:
var sem[0..4] : binary semaphore;
Initial value of all semaphores is 1

Program for process P_i*:*

```
< P(sem[i]); >
< P(sem[i+1 mod 5]); >
critical section;
< V(sem[i]); >
< V(sem[i+1 mod 5]); >
```

6 Coloured Petri Nets

You may have noticed that the graphical representation of Petri nets becomes fairly complex if we try to model real life problems. The main reason is that we only have one type of token. In our mutual exclusion example (see Exercise 5.21, page 113) we had to represent each process by a separate subnet, although their respective behaviour is identical. Furthermore we had to represent all values of a variable by different places.

In this chapter we introduce Coloured Petri nets (CPNs) in which a type called the *colour* is attached to a token. CPNs were first defined by K. Jensen (cf. [93]). There are several other Petri net models which distinguishes between individual tokens, e.g. [79, 99, 148], but since Queueing Petri nets (QPNs) [18], discussed in Ch. 10, are based upon CPNs, we need to know only about the latter.

Fig 6.1 illustrates a part of a CPN in which place p_1 is marked by a token of colour a and p_2 is marked by two tokens, one of colour b and one of colour c. How can we describe the firing of transition t? E.g., we may require transition t to be enabled in the marking shown and that it destroys the tokens a and b and creates a token of colour d on p_3. Another possible behaviour of t may be that it destroys tokens a and c and creates a token of colour e on p_3. These two different ways in which t might fire are referred to as the different *modes* in which t fires. In CPNs these modes of a transition are also described by colours, but now attached to the transition. Let us denote the set consisting of all these colours by $C(t)$. So we say, e.g., that transition t fires with respect to colour x (characterising a certain “firing mode” of t) or with respect to colour y. With that in mind the incidence functions of a CPN can be defined similar to the incidence functions of a Place-Transition net (see Def. 5.1 on page 81). E.g., if transition t fires with respect to colour x, tokens a and b will be destroyed and one token of colour d will be created on p_3.

Before defining CPNs formally we need to know about *multi-sets*. Informally a

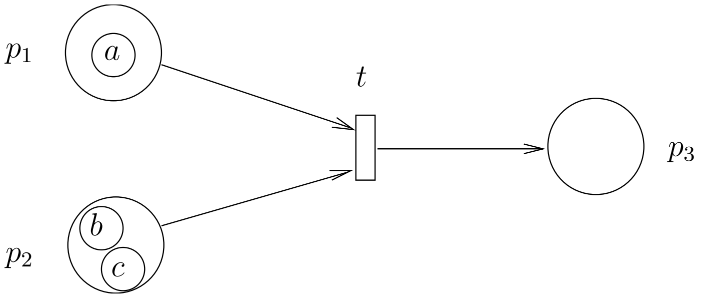


Figure 6.1 A simple CPN

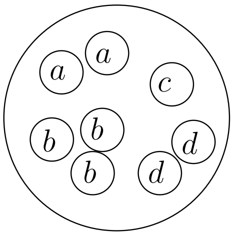


Figure 6.2 A multi-set m

multi-set is the same as a set, except that individual elements may occur more than once and for this reason they are sometimes also called *bags*. For instance if we have a set {a,b,c} and add element b to the set we still have only the set {a,b,c}. However, if we added b to the multi-set {a,b,c} we have the new multi-set {a,b,b,c} with two occurrences of the element b.

Definition 6.1 (Multi-set) *A multiset m, over a non-empty set S, is a function $m \in [S \mapsto \mathbb{N}_0]$. The non-negative integer $m(s) \in \mathbb{N}_0$ is the number of appearances of the element s in the multi-set m.*

Refer to Fig. 6.2. The definition of the multi-set m in that figure is given by

$$m(s) = \begin{cases} 2, & \text{if } s \in \{a,d\} \\ 3, & \text{if } s = b \\ 1, & \text{if } s = c \end{cases}$$

We denote the set of all finite multi-sets over S by S_{MS} and define addition of multi-sets and multiplication with ordinary numbers as follows:

Definition 6.2 $\forall m_1,m_2 \in S_{MS}$ *and* $r \in \mathbb{R}$ [1] *define*
$(m_1 + m_2)(s) := m_1(s) + m_2(s)$
$(rm_1)(s) := rm_1(s)$

Now, remember that the incidence functions of a Place-Transition net reflect the connection between places and transitions. That is, $I^-(p,t)$ describes the number of tokens which are destroyed on place p in case of t fires. The corresponding backward incidence function of a CPN is now an element of

$$I^-(p,t) \in [C(t) \mapsto C(p)_{MS}].$$

The forward incidence function $I^+(p,t)$ is obviously defined similarly.
Let us return to our example. Since t can fire in two different modes, the colour set should consist of (at least) two elements, e.g. $C(t) = \{x,y\}$. The (backward) incidence function $I^-(p_1,t) \in [C(t) \mapsto C(p_1)_{MS}]$ is given by $I^-(p_1,t)(x) = \{a\}$, which is a multi-set, and $I^-(p_1,t)(x)(a) = 1$ denotes the number of elements of colour a in that multi-set.
We define a CPN formally as follows:

[1] $\mathbb{R}$ denotes the set of real numbers.

Definition 6.3 (CPN) A Coloured Petri net *(CPN) is a 6-tuple CPN=(P,T,C,I^-,I^+,M_0), where*

- *P is a finite and non-empty set of places,*
- *T is a finite and non-empty set of transitions,*
- *$P \cap T = \emptyset$,*
- *C is a colour function defined from $P \cup T$ into finite and non-empty sets,*
- *I^- and I^+ are the backward and forward incidence functions defined on $P \times T$ such that*
  $I^-(p,t),I^+(p,t) \in [C(t) \rightarrow C(p)_{MS}], \forall\, (p,t) \in P \times T$,
- *M_0 is a function defined on P describing the initial marking such that $M_0(p) \in C(p)_{MS}, \forall p \in P$.*

The *preset* and *postset* of a transition and place, respectively, of a CPN are defined analogous to that for Place-Transition nets on page 83:

Definition 6.4
The preset $\bullet(p,c)$ of $p \in P$ and $c \in C(p)$ is
$\bullet(p,c) := \{(t,c')|t \in T,c' \in C(t) : I^+(p,t)(c')(c) \neq 0\}$.
The preset $\bullet(t,c')$ of $t \in T$ and $c' \in C(t)$ is
$\bullet(t,c') := \{(p,c)|p \in P,c \in C(p) : I^-(p,t)(c')(c) \neq 0\}$.
The postset $(p,c)\bullet$ of $p \in P$ and $c \in C(p)$ is
$(p,c)\bullet := \{(t,c')|t \in T,c' \in C(t) : I^-(p,t)(c')(c) \neq 0\}$.
The postset $(t,c')\bullet$ of $t \in T$ and $c' \in C(t)$ is
$(t,c')\bullet := \{(p,c)|p \in P,c \in C(p) : I^+(p,t)(c')(c) \neq 0\}$.

Next we can define the behaviour of a CPN, again very much the same way as for Place-Transition nets.

Definition 6.5 (Enabled Transition) *A transition $t \in T$ is enabled in a marking M w.r.t. a colour $c' \in C(t)$, denoted by $M[(t,c') >$, iff $M(p)(c) \geq I^-(p,t)(c')(c), \forall p \in P,c \in C(p)$.*
An enabled transition $t \in T$ may furthermore fire in a marking M w.r.t. a colour $c' \in C(t)$ yielding a new marking M', denoted by $M \rightarrow M'$ or $M[(t,c') > M'$, with
$M'(p)(c) = M(p)(c) + I^+(p,t)(c')(c) - I^-(p,t)(c')(c), \forall p \in P,c \in C(p)$.

Not surprisingly the various properties we defined for Place-Transition nets can also be defined for CPNs where we now denote the reachability set by $R(CPN) := R(CPN,M_0) := \{M|M_0 \rightarrow^* M\}$ where $\rightarrow^*$ is the reflexive and transitive closure of $\rightarrow$.

Definition 6.6 *Let CPN=(P,T,C,I^-,I^+,M_0) be a Coloured Petri net.*

1. *CPN is bounded iff* $R(CPN,M_0)$ *is finite.*

2. *CPN is live iff* $\forall M \in R(CPN,M_0),t \in T,c \in C(t) : \exists M' : M \to^* M'$ *and* $M'[(t,c) >$.

3. *A marking* M' *is called a home state iff* $\forall M \in R(CPN,M_0) : M \to^* M'$.

4. $D \subseteq \bigcup_{p \in P} \bigcup_{c \in C(p)} (p,c)$ *is called deadlock (trap) iff* $\bullet D \subseteq D\bullet$ $(D\bullet \subseteq \bullet D)$.
D is marked at a marking M *iff* $\exists (p,c) \in D : M(p)(c) > 0$.

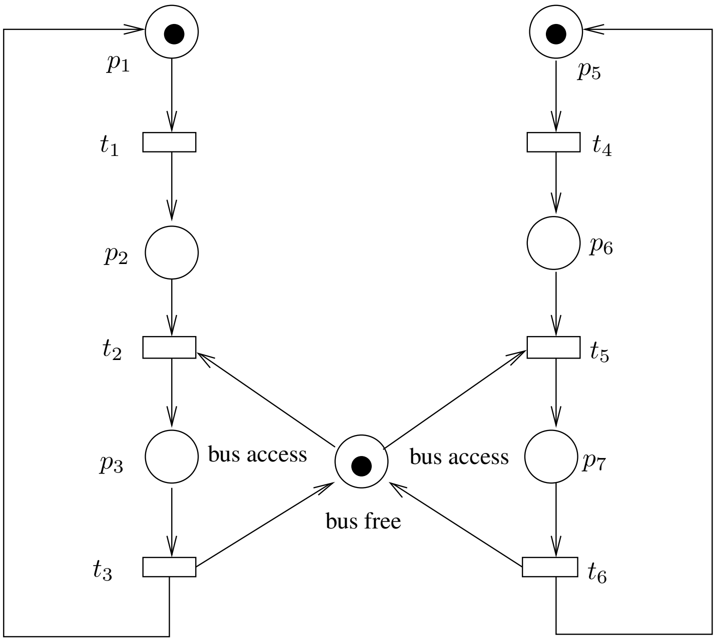


Figure 6.3 Place-Transition net model of a dual multiprocessor system

Example 6.1 *Consider a dual processor system where each processor accesses a common bus, but not simultaneously. Fig. 6.3 illustrates a Place-Transition net model of the system. Places* p_1,p_2,p_3 *represent the states of the first processor while places* p_5,p_6,p_7 *represent the states of the second one.*
However, since the processors are identical, an obvious idea (now that we know about CPNs) is to represent each of the two processors by a different colour token. A CPN model of the same system is illustrated in Fig. 6.4. The formal definition is given by $CPN = (P,T,C,I^-,I^+,M_0)$ *where*

1. $P = \{p_1,p_2,p_3,p_4\}$,

2. $T = \{t_1,t_2,t_3\}$,

3. $C(p_1) = C(p_2) = C(p_3) = \{c_1,c_2\}, C(p_4) = \{\bullet\}$,
   $C(t_1) = C(t_2) = C(t_3) = \{c_1',c_2'\}$.

4. *For the definition of the backward and forward incidence functions it is sufficient to specify only the functional non-zero values.*

$$\begin{array}{lll}
I^-(p_1,t_1)(c_1')(c_1) = 1, & I^-(p_1,t_1)(c_2')(c_2) = 1, & I^-(p_2,t_2)(c_1')(c_1) = 1,\\
I^-(p_2,t_2)(c_2')(c_2) = 1, & I^-(p_4,t_2)(c_1')(\bullet) = 1, & I^-(p_4,t_2)(c_2')(\bullet) = 1,\\
I^-(p_3,t_3)(c_1')(c_1) = 1, & I^-(p_3,t_3)(c_2')(c_2) = 1, & I^+(p_2,t_1)(c_1')(c_1) = 1,\\
I^+(p_2,t_1)(c_2')(c_2) = 1, & I^+(p_3,t_2)(c_1')(c_1) = 1, & I^+(p_3,t_2)(c_2')(c_2) = 1,\\
I^+(p_1,t_3)(c_1')(c_1) = 1, & I^+(p_1,t_3)(c_2')(c_2) = 1, & I^+(p_4,t_3)(c_1')(\bullet) = 1,\\
I^+(p_4,t_3)(c_2')(\bullet) = 1, & &
\end{array}$$

5. $M_0(p_1)(c_1) = M_0(p_1)(c_2) = M_0(p_4)(\bullet) = 1$.

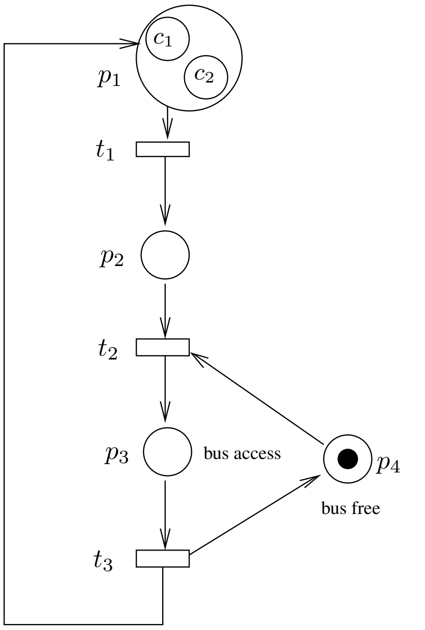


Figure 6.4 Coloured Petri net model of the same dual multiprocessor system

Now that we know about the structure, behaviour and properties of CPNs, we next would like to know how to analyse them. Fortunately, every CPN can be *unfolded* uniquely into a Place-Transition net so that all the concepts relating to CPNs are consistent with those of Place-Transition nets.

A CPN=(P,T,C,I^-,I^+,M_0) is unfolded into a Place-Transition net in the following way:

1. $\forall\, p \in P, c \in C(p)$ create a place (p,c) of the Place-Transition net.
2. $\forall\, t \in T, c' \in C(t)$ create a transition (t,c') of the Place-Transition net.

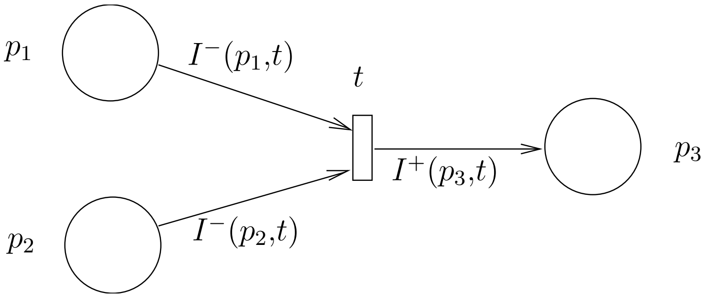


Figure 6.5 The graphical representation of a CPN

3. Define the incidence functions of the Place-Transition net as
$I^-((p,c)(t,c')) := I^-(p,t)(c')(c)$,
$I^+((p,c)(t,c')) := I^+(p,t)(c')(c)$.

4. The initial marking of the Place-Transition net is given by
$M_0((p,c)) := M_0(p)(c), \forall\, p \in P, c \in C(p)$

The unfolded CPN is given by

$$PN = (\bigcup_{p \in P} \bigcup_{c \in C(p)} (p,c), \bigcup_{t \in T} \bigcup_{c' \in C(t)} (t,c'), I^-, I^+, M_0)$$

It should be obvious that while the unfolding of a CPN yields a unique ordinary Petri net the converse is not true. *Folding* (as it is termed) of a Place-Transition net can be done in more than one way.

Example 6.2 *As an example, the unfolded net of Fig 6.4 is displayed in Fig. 6.3.*

It should thus be clear that Coloured Petri nets are only a means of simplifying the graphical representation of a Petri net model of a complex system. We can analyse such models by unfolding the CPN and performing the analysis on the unfolded net.
Coloured Petri nets have undergone several revisions. Jensen[96, 97] uses expressions to specify the incidence functions and markings, but for invariant analysis a function representation as presented here is used. The expression representation is based on the meta language ML [122]. In the same book Jensen also introduces *guards* which can be attached to the transitions of a CPN. A guard determines a boolean value and the corresponding transition is only enabled if this value is true.
For simplicity and to avoid confusion, we will describe the "older" graphical representation presented by Jensen in [93, 94] and only touch on the graphical representation of CPNs using ML constructs and not go into the formal details.
Fig. 6.5 illustrates a part of a CPN where the incidence functions
$I^-(p,t), I^+(p,t) \in [C(t) \mapsto C(p)_{MS}]$ are represented as "arc weights". If we can find mnemonics for these functions, this idea of representing a CPN graphically will be sufficient for a human reader. This is the original idea presented by Jensen in [93].

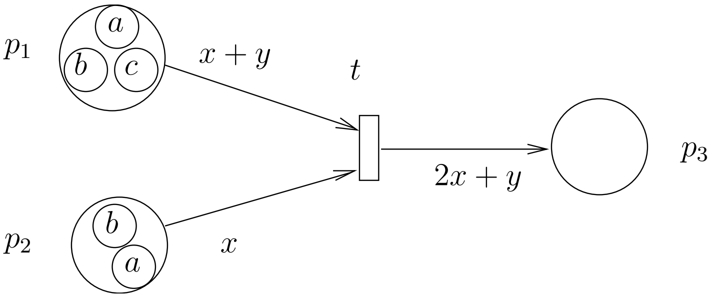


Figure 6.6 CPN with expressions attached to the arcs

Later, concepts of Predicate/Transition nets [79] were adopted for the graphical representation of CPNs. In order to understand this we need to remember the usual concept of binding all or a subset of all free variables in an expression to values. Consider some variables x,y and z and an expression involving these variables, e.g. $2x+y-z$. If we bound these variables to values i,j,k respectively, then the given expression evaluates to $2i+j-k$. In most cases the incidence function of a CPN can now be written using an expression. Consider e.g. the CPN of Fig. 6.6, where we have attached an expression to each arc remembering that the x and y are variables on a multi-set.

Thus the expression $x+y$ states that in case transition t fires, a token of kind x and a token of kind y is destroyed on p_1. To determine whether a transition t is enabled and the result of it firing, the free variables of all expressions attached to the input and output arcs of t must be bound to some values in order to evaluate the expression. An arc expression must evaluate to a multi-set over the colour set of the corresponding place. Thus possible evaluations of the arc expression $x+y$ are $a+a$,$a+b$,$a+c$,$a+d$ etc. with $C(p_1)=\{a,b,c,d\}$. Clearly, for all arc expressions adjacent to a transition we have to bind identical variables to the same value. E.g., if we bound the variable x to a and y to b, this implies that the arc expressions $x+y$, x and $2x+y$ evaluate to $a+b$, a and $2a+b$ respectively. After the evaluation of all arc expressions of a transition, enabling and firing of that transition is similar to that of Place-Transition nets. E.g., if the arc expressions $x+y$ and x evaluate to $a+b$ and a respectively, then t is enabled at a marking M iff there is at least one token of colour a and at least one token of colour b in the marking of p_1 and at least one token of colour a in the marking of p_2. This is the case in Fig 6.6 and if t fires it destroys the tokens a and b on p_1 as well as the token a on p_2 and creates 2 tokens of colour a and one token of colour b on p_3 as prescribed by the arc expression $2x+y$.

If we bound x to c and y to b, then t is not enabled since there is no token of colour c in the marking of p_2 depicted in Fig. 6.6.

If, depending on the binding, we want a transition to behave in more than one way, we can describe this by employing syntactical constructs of programming languages. Jensen in [96, 97] uses the programming language ML for this and we offer the CPN in Fig. 6.7 as an example.

In that example, $C(p_1) = \{a,b,c\}, C(p_2) = \{a,b,c,d\}, C(p_3) = \{r,s\}$ and if x is bound to a and y to b then t is enabled and the arc expression "if ..." evaluates to $2r + s$. Note that this arc expression always evaluates to a multi-set over the colour set of p_3, as before. If x is bound to a and y to c then t is also enabled and in that case, when it fires, no token is created on p_3 as denoted by the expression "empty".

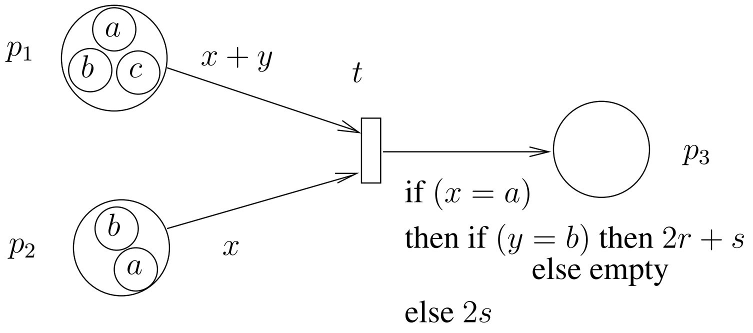


Figure 6.7 CPN with code rather than functional description of the incidence functions

In [96] the reader will find a formal definition of CPNs and its graphical representation which differs somewhat from that described above. In the following we will use the definition of a CPN given in this chapter and use the concepts concerning the graphical representation only to draw CPNs compactly.

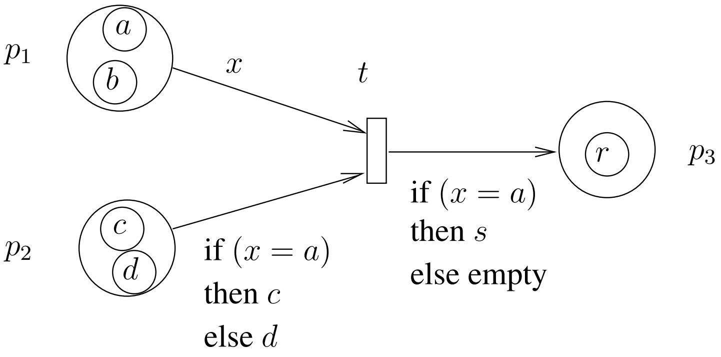


Figure 6.8 $C(p_1) = \{a,b,c\}, C(p_2) = \{b,c,d\}, C(p_3) = \{r,s\}$

Exercise 6.1 *Fold the following sets of places and transitions of the Petri net in Fig. 5.3, page 81:*
$\{p_9,p_{10}\}$ *and* $\{p_i,p_{i+4}\},\{t_i,t_{i+4}\},i \in \{0,\ldots,4\}$.

Exercise 6.2

1. *Give the formal definition (see Def. 6.3) of the CPN displayed in Fig. 6.8.*

2. *Draw the reachability graph of this CPN.*
3. *Determine the unfolded Place-Transition net. What does the "empty"-expression imply for the connection of transitions and places of the unfolded net?*

7 Further Reading

Further introductions on Petri nets are given in [132, 149, 150] and short introductory papers are [125, 130].

A common extension of Petri nets found in the literature is to impose priorities on transitions and/or to introduce so-called inhibitor arcs. If two transitions of different priority are enabled in the sense of ordinary Petri nets then only the higher priority transition may fire. An inhibitor arc connects a place p to a transition and imposes an additional constraint on the enabling of that transition. The transition is now enabled iff the enabling rule of ordinary Petri nets concerning the usual arcs between places and that transition is satisfied and if place p is not marked. In [132] these extensions are discussed, amongst others. Petri nets with inhibitor arcs or priorities have the power of Turing machines and thus several problems, e.g. reachability problem or determining liveness, become undecidable. This is the reason why most Petri net researchers do not consider these extensions, although they might be necessary to model real life problems.

The analysis of net classes, especially of (Extended) Free Choice nets, is the subject of [34, 36, 37, 38, 39, 60, 175] and several results are summarised in [38, 125]. Analysis of Marked Graphs is considered in [78]. This class of nets is also called "Synchronisationsgraphen" or T-graphs. Another name for State Machines is S-graphs. (Extended) Simple nets, also called Asymmetric Choice nets, are dealt with in [38].

A further net class with similar properties like EFC-nets are investigated in [173, 174]. This class is called *Equal Conflict nets* (EC-nets) and allows multiple arc weights. Its definition is:

$\forall t,t' \in T : (\bullet t \cap \bullet t' \neq \emptyset \Longrightarrow I^-(p,t) = I^-(p,t'), \forall p \in P)$.

Reduction and synthesis analysis can be found in several papers. The following papers are concerning reduction and synthesis analysis of Petri nets: [31, 32, 70, 80, 101, 126, 168, 171, 172, 176, 182].

A study on different efficient methods for the analysis of the reachability set (symmetry method, stubborn set method, symbolic model checking, incomplete reachability analysis) can be found in [145]. Progress has been made in the analysis of the reachability set exploiting dynamic priorities[21], structural information for a Kronecker based representation of the state space (e.g. [106] (see also page 175)) or Binary Decision Diagrams (e.g. [123]).

Further Petri net models introducing the concept of individual tokens are described in [79, 99, 148]. The invariant analysis of CPNs was first discussed in [93]. The latest version of CPNs is defined in [96, 97, 98] and it is shown how a functional representation can be used for calculating the invariants of this CPN version. A coloured version of the reduction rules of [31, 32] are defined in [88]. CPNs with an infinite number of colours also have the power of Turing machines

[131].

An international conference with the title “Application and Theory of Petri Nets” started in 1980 and is held annually, e.g. [95, 3, 177, 58, 40, 12, 59, 64, 128, 54, 71]. Further publications on Petri nets can be found in “Advances in Petri Nets” (Lecture Notes in Computer Science, Springer-Verlag, e.g. [44, 45, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 41, 1, 67]).

The German “Gesellschaft für Informatik (GI)” has a “Special Interest Group on Petri Nets and Related System Models”, which publishes a Petri net newsletter to its members half-yearly. For membership contact Gesellschaft für Informatik, email: gs@gi-ev.de (yearly costs: ≈ EUR 10) or visit the URLs http://www.gi-ev.deand http://www.informatik.uni-hamburg.de/TGI/GI-Fachgruppe0.0.1/resp. There is also a Petri net mailing group (see http://www.informatik.uni-hamburg.de/TGI/PetriNets/mailing-lists/). Additional information on Petri nets can be accessed via URL http://www.informatik.uni-hamburg.de/TGI/PetriNets/.

First proposals have been published defining a standard for high-level Petri nets (*High-level Petri Nets - Concepts, Definitions and Graphical Notation*) within the International Organization for Standardisation (ISO). For more information see http://www.informatik.uni-hamburg.de/TGI/PetriNets/standardisation/.

Part III

TIME-AUGMENTED PETRI NETS

Petri nets involve no notion of time, since it is not defined at what point in time a transition will fire. Analysing the performance of a system with a Petri net is thus not possible. Petri nets can only be used for qualitative analysis, i.e., to analyse the functional or qualitative behaviour of a system. For performance or quantitative analysis the temporal (time) behaviour of the system has to be part of the Petri net description.
Since the early 70's several suggestions on incorporating time into Petri nets have been published. In general, there are two possible ways to do this:

1. Specifying sojourn times of tokens on places.

   If tokens are fired onto a place p they become unavailable to all output transitions of p for a certain time interval. Once this time has elapsed the tokens become available and can be consumed by the output transitions of p. Such Petri nets are known as *Timed Places Petri nets (TPPNs).*

2. Specifying a firing delay of enabled transitions.

   Once a transition is enabled, it will fire after a certain time interval has elapsed. These Petri nets are also called *Timed Transitions Petri nets (TTPNs)* which, in turn, can be classified into two groups.

   (a) Preselection models.
   Once a transition is enabled, it selects all tokens it needs to fire on its input places so that they are unavailable to any other transition of the Petri net. The enabled transition waits until its firing time interval has elapsed and then fires immediately. By firing, the transition destroys all its reserved tokens and creates tokens onto its output places according to the firing rule of a Petri net.

   (b) Race models.
   In these Petri nets tokens are not reserved by a transition. Once a transition is enabled, it waits for its firing time to elapse and than fires immediately provided it is still enabled at that time. Since a faster transition might disable other transitions, enabled transitions compete or "race" for tokens; hence the name of this class of Petri net.

These time-augmented Petri nets, TPPN as well as TTPN models, can be classified further depending upon whether the times mentioned are deterministic or stochastic. In the first case the class of such Petri nets is called "Timed Petri Nets" and in the latter they are called "Stochastic Petri Nets".[1]
A further classification depending upon the firing policy of transitions (resampling, age memory, enabling memory) can be found in [4, 5], but this classification is not relevant for exponentially distributed firing delays.

[1] Unfortunately the name of this class coincides with the name of a member of the class: Molloy's SPNs [124].

Since our interest is in time-augmented Petri nets which can be analysed by Markovian techniques, the next sections will consider those time-augmented Petri nets where time is exponentially distributed.

We first of all introduce a wide-spread family of Stochastic Petri nets, not surprisingly called "Stochastic Petri Nets" (SPNs) as well as "Generalized Stochastic Petri Nets" (GSPNs) and then describe a different class called "Queueing Petri Nets" (QPNs). QPNs combine the two ways of incorporating time by specifying a sojourn time of tokens on places as well as a firing delay for enabled transitions. All three classes are race models, i.e., enabled transitions compete for the tokens on their input places. A preselection policy, as we will see later, can be represented using immediate transitions as defined for GSPNs and QPNs.

8 Stochastic Petri Nets

The continuous-time Stochastic Petri net (M.K. Molloy [124], S. Natkin [127]) $SPN = (PN,\Lambda)$ is formed from the Place-Transition net $PN = (P,T,I^-,I^+,M_0)$ by adding the set $\Lambda = (\lambda_1,\ldots,\lambda_m)$ to the definition. λ_i is the, possibly marking dependent, transition rate of transition t_i. I.e., the firing time is exponentially distributed and the distribution of the random variable χ_i of the firing time of transition t_i is given by

$$F_{\chi_i}(x) = 1 - e^{-\lambda_i x}$$

A typical example of a Stochastic Petri net (SPN) is illustrated in Fig. 8.1 (cf. [124]).

In the SPN in the figure, transition t_1 is enabled at $M_0 = (1,0,0,0,0)$. The time elapsed until t_1 fires is exponentially distributed with rate λ_1, i.e. the average time for t_1 to fire is $\frac{1}{\lambda_1}$. Once t_1 has fired, using the firing rule of Place-Transition nets, we obtain marking $M_1 = (0,1,1,0,0)$. At M_1, t_2 and t_3 are concurrently enabled. That is, one of these two transitions will fire next after a certain time has elapsed. If transition t_2 fires first, the SPN changes to marking $M_2 = (0,0,1,1,0)$ and if t_3 fires before t_2, we get the marking $M_3 = (0,1,0,0,1)$. The next marking thus depends on which transition "wins the race". The probability that t_2 fires first is given by:

$$P[t_2 \text{ fires first at } M_1] \quad = \quad P[\chi_2 < \chi_3]$$

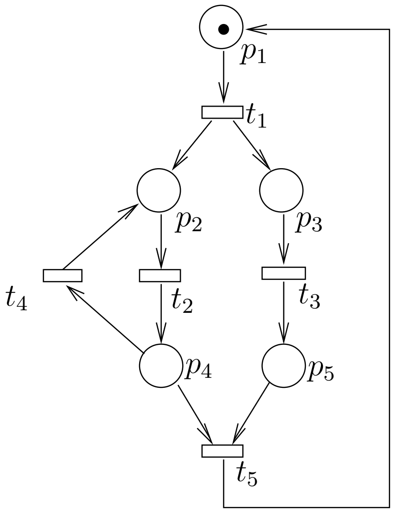


Figure 8.1 A typical Stochastic Petri net

$$\begin{aligned} &= \int_0^\infty (\int_0^x \lambda_2 e^{-\lambda_2 y} dy) \lambda_3 e^{-\lambda_3 x} dx \\ &= \int_0^\infty (1 - e^{-\lambda_2 x}) \lambda_3 e^{-\lambda_3 x} dx \\ &= \frac{\lambda_2}{\lambda_2 + \lambda_3} \end{aligned}$$

and similarly,

$$P\{t_3 \text{ fires first at } M_1\} = \frac{\lambda_3}{\lambda_2 + \lambda_3}.$$

This shows that the probability of changing from marking M_1 to some other marking is independent of the time spent in M_1!
The sojourn time in M_1 is given by the minimum of the independent, exponentially distributed firing times of both transitions, namely:

$$\begin{aligned} P[\min(\chi_2,\chi_3) \leq x] &= P[\chi_2 \leq x \text{ or } \chi_3 \leq x] \\ &= 1 - P[\chi_2 > x \text{ and } \chi_3 > x] \\ &= 1 - e^{-\lambda_2 x} e^{-\lambda_3 x} \\ &= 1 - e^{-(\lambda_2 + \lambda_3)x} \end{aligned}$$

Thus the sojourn time in M_1 is also exponentially distributed with parameter $(\lambda_2 + \lambda_3)$. In combination with the fact that the probability of changing the state is independent of the sojourn time, this implies that a SPN describes a Markov process.
The rate of going from M_1 to say e.g. M_2 is then given, as usual, by:

$$\frac{\lambda_2}{(\lambda_2 + \lambda_3)} \frac{1}{E[\min(\chi_2,\chi_3)]} = \frac{\lambda_2}{(\lambda_2 + \lambda_3)} \frac{1}{\frac{1}{(\lambda_2 + \lambda_3)}} = \lambda_2$$

The performance or quantitative analysis of SPNs can be carried out straightforwardly by analysing the corresponding Markovian process. Consider Fig. 8.2 which depicts the reachability graph of the SPN of Fig. 8.1. Consider each marking in that reachability graph as a state of the corresponding Markov chain (MC) and attach the firing rate λ_i of transition i as an arc label to each transition in the Markov chain, giving rise to Fig. 8.3.
To be more precise: The Markov chain (cf. Sec. 2.3) of an SPN can be obtained from the reachability graph of the associated Place-Transition net PN as follows: The MC state space is the reachability set $R(PN)$ and the transition rate from state M_i to state M_j is given by $q_{ij} = \lambda_k$, the possibly marking-dependent, firing rate of transition t_k from M_i to M_j. If several transitions lead from M_i to M_j then q_{ij} is the sum of the rates of those transitions. Similarly $q_{ij} = 0$ if no transitions lead from M_i to M_j and q_{ii} is determined so as to satisfy $\sum_j q_{ij} = 0$ (cf. Eq. 2.43 on page 50).

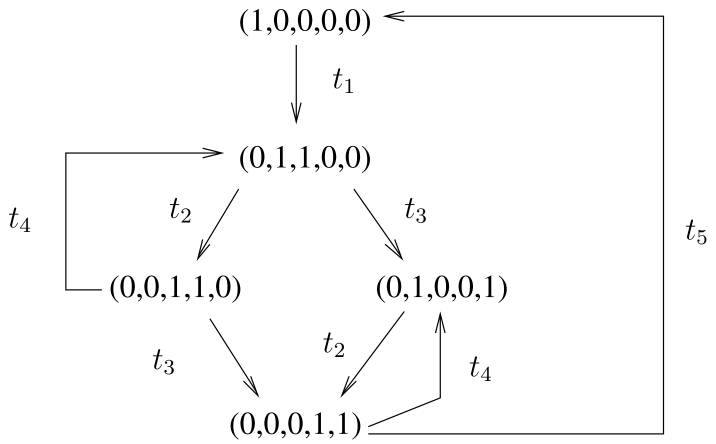


Figure 8.2 Reachability graph of the SPN's underlying Place-Transition net

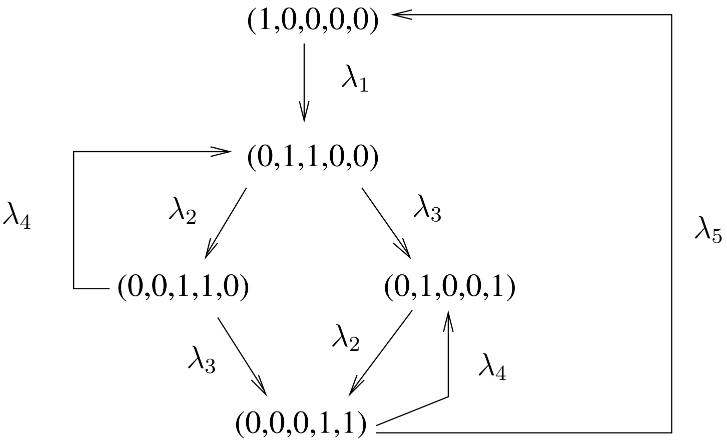


Figure 8.3 Markov chain of the SPN

The square matrix $Q = (q_{ij})$ of order $s = |R(PN)|$ is the matrix Q in Sec. 2.3. As before, the steady state distribution π of the MC is obtained by solving the linear equations

$$\pi Q = 0; \quad \sum_{i=1}^{s} \pi_j = 1$$

From the vector $\pi = (\pi_1, \pi_2, \ldots, \pi_s)$ we can then compute the following performance measures:

Probability of being in a subset of markings: Let $B \subseteq R(PN)$ constitute the markings of interest in a particular Stochastic Petri net. Then the probability of being in a state of the corresponding subset of the MC is given by:

$$P[B] = \sum_{M_i \in B} \pi_i.$$

Mean number of tokens: Let $B(p_i,n)$ be the subset of $R(PN)$ for which the number of tokens in a place p_i is n, i.e. $B(p_i,n) = \{M \in R(PN)|M(p_i) = n\}$. Then the average number of tokens in place p_i is given by:

$$\overline{m_i} = \sum_{n=1}^{\infty}(nP[B(p_i,n)]).$$

Probability of firing transition t_j: Let EN_j be the subset of $R(PN)$ in which a given transition t_j is enabled, i.e. $EN_j = \{M \in R(PN)|M[t_j >\}$. Then, the probability r_j that an observer, who looks randomly into the net, sees transition t_j firing next is given by:

$$r_j = \sum_{M_i \in EN_j} \pi_i(\frac{\lambda_j}{(-q_{ii})})$$

where $(-q_{ii})$ is the sum of transition rates out of M_i.

Throughput at a transition t_j: The throughput at a timed transition is given by it's mean number of firings at steady state:

$$\overline{d_j} = \sum_{M_i \in EN_j} \pi_i \lambda_j$$

Example 8.1 *Consider the Stochastic Petri net shown in Fig.8.1. That SPN displays sequential operation (t_5,t_1), parallel operation (t_2,t_3), forking (t_1), joining (t_5), and conflict (t_4,t_5). Assume mean firing rates $\lambda_1 = 2$, $\lambda_2 = 1$, $\lambda_3 = 1$, $\lambda_4 = 3$, and $\lambda_5 = 2$. Starting with an initial marking of one token in place p_1 and no tokens in the remaining places, the reachability set has five markings or equivalently, there are five states in the Markov chain.*
Solving the Markov chain we obtain the following steady-state marking probabilities:

$$\begin{aligned}
P[M_1] = &\quad P[(1,0,0,0,0)] = \frac{5}{43}\\
P[M_2] = &\quad P[(0,1,1,0,0)] = \frac{8}{43}\\
P[M_3] = &\quad P[(0,0,1,1,0)] = \frac{2}{43}\\
P[M_4] = &\quad P[(0,1,0,0,1)] = \frac{23}{43}\\
P[M_5] = &\quad P[(0,0,0,1,1)] = \frac{5}{43}
\end{aligned}$$

Using the marking probabilities and the number of tokens in each place in a particular marking we can deduce the steady state probabilities of their being μ_i tokens at place p_i. This is known as the token probability density function.

$$P[\mu_1 = 0] = \tfrac{38}{43} \quad P[\mu_1 = 1] = \tfrac{5}{43}$$

$$P[\mu_2 = 0] = \tfrac{12}{43} \quad P[\mu_2 = 1] = \tfrac{31}{43}$$

$$P[\mu_3 = 0] = \tfrac{33}{43} \quad P[\mu_3 = 1] = \tfrac{10}{43}$$

$$P[\mu_4 = 0] = \tfrac{36}{43} \quad P[\mu_4 = 1] = \tfrac{7}{43}$$

$$P[\mu_5 = 0] = \tfrac{15}{43} \quad P[\mu_5 = 1] = \tfrac{28}{43}$$

If we were to assume a different initial marking then we would obtain a different Markov chain. The more tokens we have in the system the larger the Markov chain. If we start with two tokens in place p_1 we would find 14 states in the reachability set. Similarly, if we started with 3 tokens in place p_1, we would find 30 states in the reachability set. This state space explosion *is a familiar problem in practice.*

The most important aspect of SPNs is that the reachability graph of the underlying Place-Transition net and the Markov chain are isomorphic. In other words, the number of states/markings and the connection structure of both graphs are the same. The Markov chain describes the "reachability graph" of the SPN. Therefore all properties of the underlying Place-Transition net also hold for the SPN and vice versa. For that reason qualitative (functional) analysis of an SPN can be done by applying the algorithms and techniques for Place-Transition nets.
Although SPNs embrace these important features, they also exhibit significant problems, which led to their modification to Generalized Stochastic Petri nets (GSPNs). Before we introduce GSPNs we first point out the difficulties of SPNs. Consider the SPN in Fig. 8.4 with firing rates $\Lambda = (1.0,1.0,1.0,\mu,\mu,\mu)$. The corresponding Markov chain is shown in Fig. 8.5.

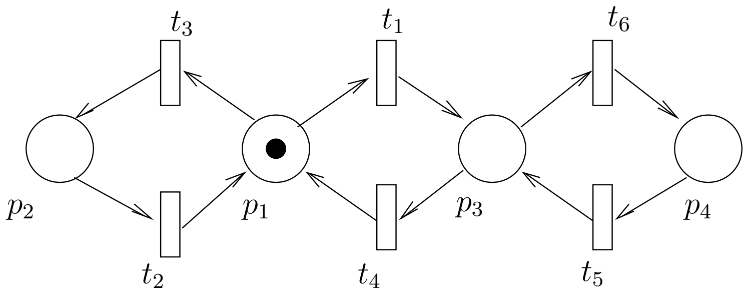


Figure 8.4 Simple SPN

We now want to analyse this continuous-time Markov chain for different values of μ. The infinitesimal generator Q of the MC is given by

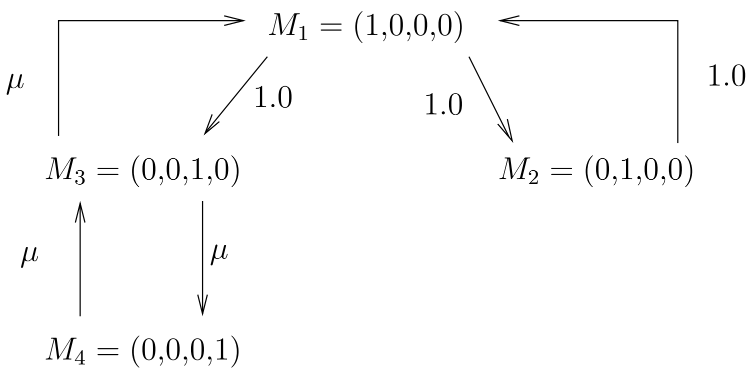


Figure 8.5 Markov chain

$$Q = \begin{pmatrix} -2.0 & 1.0 & 1.0 & 0 \\ 1.0 & -1.0 & 0 & 0 \\ \mu & 0 & -2\mu & \mu \\ 0 & 0 & \mu & -\mu \end{pmatrix}$$

and we have to solve the equation

$$\pi Q = 0 \tag{8.1}$$

yielding the steady-state distribution $\pi = (P[M_1],P[M_2],P[M_3],P[M_4])$.
A simple iterative method (power method) to solve Eq. (8.1) is:

$$\pi^k(aQ + I) = \pi^{k+1} \tag{8.2}$$

where π^k is the approximate steady state distribution in the k-th step of the iteration and a is a factor such that the sum of absolute values of all rows is less or equal to 1.
For our example we choose $a := \frac{1}{1.001}(max\{2,2\mu\})^{-1}$. In general a good choice is[1]

$$a := \frac{1}{1.001}(max_{i=1}^{|R(PN)|}\{|q_{ii}|\})^{-1}$$

Table 8.1 shows the number of iterations needed for several values of μ and the corresponding result for π starting with the vector $(0.25,0.25,0.25,0.25)$. The iteration stops once $\|\pi^{n+1} - \pi^n\| < \epsilon$ where $\|x\| := \sum_{i=1}^n |x_i|$ for $x = (x_1,\ldots,x_n)$ without taking the convergence speed into account for simplicity. The relative error is given here by $\max_{i=1}^4(100\% \times |\frac{\pi_i^{\text{iteration}} - \pi_i^{\text{exact}}}{\pi_i^{\text{exact}}}|)$.

[1] For more information on the numerical analysis of Markov chains see [169].

μ	#iterations	$\pi^{\text{iteration}}$	rel. error
1.0	1	(0.25000,0.25000,0.25000,0.25000)	0.00000
2.0	47	(0.33333,0.33333,0.16667,0.16667)	0.00064
10.0	122	(0.45455,0.45454,0.04546,0.04546)	0.00156
50.0	470	(0.49022,0.49017,0.00980,0.00980)	0.00545
100.0	861	(0.49510,0.49500,0.00495,0.00495)	0.01043
200.0	1574	(0.49761,0.49741,0.00249,0.00249)	0.02045
500.0	3464	(0.49925,0.49875,0.00100,0.00100)	0.05049
1000.0	6226	(0.50000,0.49900,0.00050,0.00050)	0.10060
10000.0	39155	(0.50495,0.49495,0.00005,0.00005)	1.00143
100000.0	161009	(0.55004,0.44995,0.00001,0.00001)	10.01045
200000.0	183245	(0.60010,0.39990,0.00000,0.00000)	20.02048
300000.0	153105	(0.65015,0.34985,0.00000,0.00000)	30.03040
400000.0	88950	(0.70020,0.29980,0.00000,0.00000)	40.04046
500000.0	59	(0.74997,0.25003,0.00000,0.00000)	160.13592
1000000.0	59	(0.74998,0.25001,0.00000,0.00000)	270.28565
10000000.0	59	(0.75000,0.25000,0.00000,0.00000)	2252.93736

Table 8.1 Iterative solution for different μ; $\epsilon = 10^{-6}$

Table 8.1 shows that for increasing μ the number of iterations also increases implying increased CPU-time to obtain the steady-state distribution π. Furthermore the iteration becomes unstable for large values of μ, yielding incorrect results for π. Note that the solution for π is $(\frac{\mu}{2\mu+2},\frac{\mu}{2\mu+2},\frac{1}{2\mu+2},\frac{1}{2\mu+2})$ and thus π tends to $(\frac{1}{2},\frac{1}{2},0,0)$, if μ tends to infinity.
If $\mu = \infty$ markings M_3 and M_4 will be left immediately by the Markov process in Fig. 8.5. So the only significant markings for the steady-state distribution are M_1 and M_2. With this in mind we can reduce the Markov chain to that in Fig. 8.6 before determining the steady-state distribution. Solving this reduced Markov chain iteratively will not lead to complications. Note that the size of the MC is now smaller than the original one.
The last example illustrates that if the firing rates of transitions differ in orders of magnitude, problems in the quantitative analysis of SPNs might occur. For this reason SPNs were extended to the so-called Generalized Stochastic Petri nets (GSPNs).

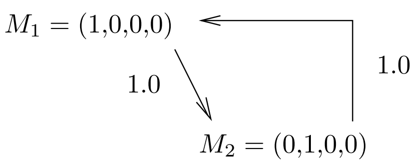


Figure 8.6 Reduced Markov chain

Exercise 8.1

1. *Determine the infinitesimal generator Q of the MC described by the SPN of example 8.1 and check the given steady state distribution.*
2. *Calculate the following quantities:*
   (a) $\overline{m}_j, \forall j \in \{1,\ldots,5\}$, *i.e. the average number of tokens on all places at steady state.*
   (b) $r_j, \forall j \in \{1,\ldots,5\}$, *i.e. the probability of firings of all transitions.*

Exercise 8.2 *Derive Eq.(8.2) from Eq.(8.1).*

9 Generalized Stochastic Petri Nets

Generalized Stochastic Petri nets (GSPNs)(M. Ajmone-Marson, G. Balbo, G. Conte [6, 11, 7]) have two different classes of transitions: *immediate* transitions and *timed* transitions. Once enabled, immediate transitions fire in zero time. Timed transitions fire after a random, exponentially distributed enabling time as in the case of SPNs.

In Fig. 9.1 the Producer-Consumer system is modelled by a GSPN. Timed transitions are drawn by open bars and immediate transitions are drawn by solid bars. Since we assume that producing and consuming an item takes much longer than inserting and removing an item to or from the buffer, we decided to model the activities of production and consumption by timed transitions ($\{t_3,t_6\}$) and the other activities by immediate transitions.

As always, several transitions may be simultaneously enabled at a marking M. If the set of enabled transitions, denoted by $EN_T(M)$, comprises only timed transitions, then the enabled timed transition $t_i \in EN_T(M)$ fires with probability

$$\frac{\lambda_i}{\sum_{j:t_j \in EN_T(M)} \lambda_j}$$

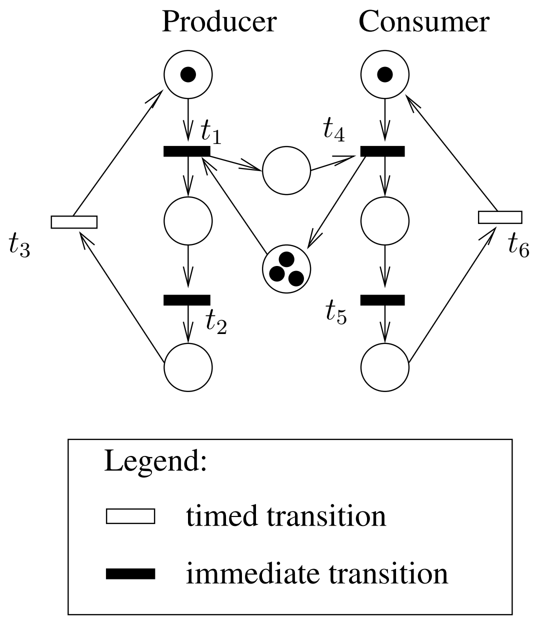


Figure 9.1 Producer-Consumer system with limited buffer capacity

exactly as before for SPNs.
If both timed and immediate transitions are enabled in the sense of Def. 5.4 (see page 84), then by definition, only the immediate transitions are enabled in the GSPN. Thus, in GSPNs, firing of immediate transitions has priority over the firing of timed transitions. So

$$EN_T(M) := \{t \in T|M[t > \text{ and if } \exists t' \in T_2 : M[t' > \text{ then } t \in T_2\}$$

where T_2 denotes the set of immediate transitions. The definition of $EN_T(M)$ reflects that if t is a timed transition then no immediate transition is enabled in the sense of Def. 5.4.
When $EN_T(M)$ has only one immediate transition than that transition fires with probability 1.0. If $EN_T(M)$ comprises several immediate transitions, we have to specify the probability with which each immediate transition $t \in EN_T(M)$ fires at M.
This probability distribution is called a *random switch* or a *switching distribution*. For specification of a probability distribution for concurrently enabled immediate transitions, we actually need to know which transitions contribute to the switching distribution. To get this information usually requires a pre-analysis of the GSPN. This may be complex. Therefore it is often more suitable to define firing weights from which the switching distribution can be determined.[1] So, if we attach firing weights w_i and w_j to two transitions t_i and t_j and only these transitions are both enabled at some marking M, the probability of firing t_i is given by $\frac{w_i}{w_i+w_j}$.
The formal definition of a GSPN is as follows:

Definition 9.1 *A GSPN (cf. [6, 11, 76, 7]) is a 4-tuple*
$GSPN = (PN,T_1,T_2,W)$ *where*

- $PN = (P,T,I^-,I^+,M_0)$ *is the underlying Place-Transition net*
- $T_1 \subseteq T$ *is the set of timed transitions,* $T_1 \neq \emptyset$,
- $T_2 \subset T$ *denotes the set of immediate transitions,*
  $T_1 \cap T_2 = \emptyset$, $T = T_1 \cup T_2$
- $W = (w_1,\dots,w_{|T|})$ *is an array whose entry*[2] $w_i \in \mathbb{R}^+$
  - *is a (possibly marking dependent) rate of a negative exponential distribution specifying the firing delay, when transition* t_i *is a timed transition, i.e.* $t_i \in T_1$ *or*

[1] In practice the specification of these firing weights often requires a form of pre-analysis, e.g. by calculating so-called extended conflict sets [76]. Also confusion situations have to be considered. Confusion occurs if a conflict situation is resolved or produced by a third transition. E.g., after firing t_1t_2 in the Petri net of Fig. 8.1 we reach a marking in which t_4 is enabled and is not in conflict with t_5, since p_5 is empty. But firing t_3 leads to this conflict situation. When defining firing weights we have to take this influence of “third party” transitions into account.

[2] $\mathbb{R}^+$ denotes the set of positive real numbers.

- *is a (possibly marking dependent) firing weight, when transition t_i is an immediate transition, i.e. $t_i \in T_2$.*

Note that if $T_2 = \emptyset$ then the definition of a GSPN coincides with the definition of an SPN, i.e. SPNs $\subset$ GSPNs.

Example 9.1 *The definition of the GSPN which models the Producer-Consumer system is as follows: $GSPN = (PN, T_1, T_2, W)$ where*

- *PN is the underlying Place-Transition net. We omit the definition of that Petri net, since one can obviously derive it from Fig. 9.1.*
- $T_1 = \{t_3, t_6\}, T_2 = \{t_1, t_2, t_4, t_5\}$
- $W = (w_1, w_2, w_3, w_4, w_5, w_6), w_i \in \mathbb{R}^+$.

How do we analyse a GSPN? Note that a GSPN does not directly describe a continuous-time Markov process, since immediate transitions fire in zero time. So the sojourn time in markings which enable immediate transitions is no longer exponentially distributed. Such markings we will call vanishing, because if a random observer looks at the stochastic process of a GSPN, he will never observe such states, although the stochastic process sometimes visits them.
On the other hand, markings which enable timed transitions only will be observed, since the stochastic process sojourns in such markings for an exponentially distributed length of time. So these markings are not left immediately. Therefore we will call them tangible. In the following we will refer to a marking also as a state.

9.1 Quantitative Analysis of GSPNs

Throughout this section we will assume that the GSPN has a finite reachability set. In order to analyse a GSPN we analyse the embedded Markov chain of the corresponding stochastic process (cf. Sec. 2.2.3). Note that from the definition of GSPNs we know that the probability of changing from one marking to another is independent of the time spent in a marking. Thus a GSPN describes a semi-Markov process. Imagine that the firing of transitions take place at points $d_0, d_1, \ldots, d_n, \ldots$ in time. If a timed transition fires at, say, d_k then $d_{k+1} - d_k > 0$, since timed transitions constitute tangible states in which the process sojourns for a certain time interval. If an immediate transition is responsible for a state change at time d_k in the corresponding stochastic process, then we have $d_k = d_{k+1}$.
So we can think of our stochastic process as a discrete-time Markov chain where some points coincide in time. To specify this embedded, discrete-time Markov chain, we have to determine the transition probability from marking or state M_k to M_r. These probabilities are given by

$$P[M_k \to M_r] = \frac{\sum_{i:t_i \in \{t \in T | M_k[t > M_r\} \cap EN_T(M_k)} w_i}{\sum_{j:t_j \in EN_T(M_k)} w_j} \tag{9.1}$$

If $\{t \in T | M_k[t > M_r\} = \emptyset$ we have $P[M_k \to M_r] = 0$. Normally $\{t \in T | M_k[t > M_r\}$ contains at most one element, since otherwise "redundant" transitions $t,t' \in T$ causing the same state change would be present, i.e. $I^-(p,t) = I^-(p,t')$ and $I^+(p,t) = I^+(p,t'), \forall p \in P$.

Note that these probabilities are independent of the time spent in marking M_k, which is why we are allowed to consider an embedded Markov chain. From the embedded Markov chain, we can write the transition probability matrix as an augmented matrix with respect to the sets of vanishing and tangible states:

$$P := \begin{pmatrix} C & D \\ E & F \end{pmatrix} \tag{9.2}$$

where

$$\begin{aligned} C = (c_{ij}),\ c_{ij} &= P[M_i \to M_j; M_i \in \hat{V}, M_j \in \hat{V}], \\ D = (d_{ij}),\ d_{ij} &= P[M_i \to M_j; M_i \in \hat{V}, M_j \in \hat{T}], \\ E = (e_{ij}),\ e_{ij} &= P[M_i \to M_j; M_i \in \hat{T}, M_j \in \hat{V}], \\ F = (f_{ij}),\ f_{ij} &= P[M_i \to M_j; M_i \in \hat{T}, M_j \in \hat{T}]. \end{aligned}$$

and $\hat{T}$ denotes the set of tangible states and $\hat{V}$ the set of vanishing states, $\hat{T} \cap \hat{V} = \emptyset$. C describes the transition probabilities between vanishing states and F specifies the probabilities between tangible states.

The steady state distribution $\tilde{\pi}$ of the embedded Markov chain, provided it exists, is given as usual by

$$\tilde{\pi}P = \tilde{\pi} \text{ and } \sum_{M_i \in \hat{T} \cup \hat{V}} \tilde{\pi}_i = 1 \tag{9.3}$$

From this steady state distribution we can calculate the steady state distribution π of the original stochastic process by weighting the probability $\tilde{\pi}_j$ with the portion of time the process spends in marking M_j.

Obviously π_j is 0 if M_j is a vanishing marking. For tangible markings π_j can be calculated as follows:

$v_{sj} := \frac{\tilde{\pi}_s}{\tilde{\pi}_j}$ determines the mean number of visits (cf. Eq. (2.20) on page 32) to state M_s between two consecutive visits to state M_j. Since $(\sum_{k:t_k \in EN_T(M_s)} w_k)^{-1}$ is the mean time the process will spend in state M_s, the mean time between two consecutive visits to M_j (mean cycle time) is given by:

$$\frac{1}{\tilde{\pi}_j} \sum_{M_s \in \hat{T}} \tilde{\pi}_s \times (\sum_{k:t_k \in EN_T(M_s)} w_k)^{-1}$$

On the other hand, the stochastic process will spend

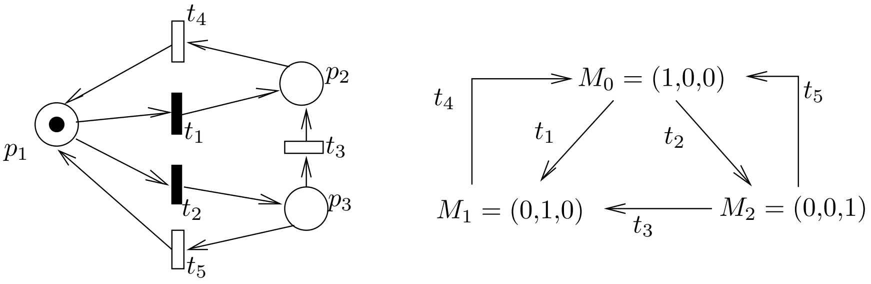


Figure 9.2 A GSPN and its reachability graph

$$\Big(\sum_{k:t_k \in EN_T(M_j)} w_k\Big)^{-1}$$

time units on the average in state M_j.

As stated before, the steady state probability π_j is given by the fraction of time the process spends in marking M_j (see also Eq. (2.35) on page 45). This portion is characterised by the mean time spend in marking M_j divided by the mean cycle time. Thus the steady state distribution π of the stochastic process described by a GSPN is given by

$$\pi_j = \begin{cases} \dfrac{\tilde{\pi}_j \times \Big(\sum\limits_{k:t_k \in EN_T(M_j)} w_k\Big)^{-1}}{\sum\limits_{M_s \in \hat{T}} \tilde{\pi}_s \times \Big(\sum\limits_{k:t_k \in EN_T(M_s)} w_k\Big)^{-1}} & \text{if } M_j \in \hat{T} \\ 0 & \text{if } M_j \in \hat{V} \end{cases} \tag{9.4}$$

Example 9.2 *Consider the GSPN displayed in Fig. 9.2 and define $w_1 = w_2 = 1$ and $w_3 = w_4 = w_5 = 3$. The only vanishing marking is the initial marking M_0. Matrix P is given by*

$$P = \begin{pmatrix} C & D \\ E & F \end{pmatrix} = \left(\begin{array}{c|cc} 0 & \frac{1}{2} & \frac{1}{2} \\ \hline 1 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 \end{array}\right)$$

and solving the global balance equations $\tilde{\pi}P = \tilde{\pi}; \sum_i \tilde{\pi}_i = 1$ yields $\tilde{\pi} = (\frac{4}{9}, \frac{3}{9}, \frac{2}{9})$. Thus the mean cycle time for tangible marking M_1 is $1 \times \frac{1}{3} + \frac{2}{3} \times \frac{1}{3+3} = \frac{4}{9}$ and for marking M_2 this is $\frac{3}{2} \times \frac{1}{3} + 1 \times \frac{1}{6} = \frac{2}{3}$. So the steady state distribution is given by

$$\pi_0 = 0, \qquad \pi_1 = \frac{\frac{1}{3}}{\frac{4}{9}} = \frac{3}{4}, \qquad \pi_2 = \frac{\frac{1}{6}}{\frac{2}{3}} = \frac{1}{4}.$$

The main task in the calculation of the steady state distribution described above, is to solve (9.3). The complexity of this calculation is determined by the number of reachable markings, including the vanishing markings. From the description of immediate transitions we know that the steady state probability of being in a vanishing state is defined to be zero.

We can eliminate vanishing markings from the embedded Markov chain before calculating the steady state distribution in a way similar to that in the example of the SPN in Ch. 8, where the reachability set was reduced to two markings for $\mu \to \infty$. This will lead to a reduction of the number of global balance equations and a more efficient analysis.

To obtain a reduction of the state space we have to determine the transition probabilities between tangible states. On leaving a tangible state M_i, the process might visit one or more vanishing states, $M_r \in \hat{V}$ before finally reaching a tangible marking, say M_j. Thus the reduced embedded Markov chain is specified by transition matrix P':

$$P' := [p'_{ij}], p'_{ij} := f_{ij} + \sum_{M_r \in \hat{V}} e_{ir} P[M_r \longrightarrow^* M_j], \tag{9.5}$$

where $P[M_r \longrightarrow^* M_j]$:= P[the stochastic process starts in state M_r
and reaches the tangible state M_j,
and all states reached in between
are vanishing states], $M_r \in \hat{V}$,$M_j \in \hat{T}$.

$P[M_r \longrightarrow^* M_j]$ can be obtained as follows. Define
$P[M_r \xrightarrow{h} M_j]$:= P[the stochastic process starts in state M_r
and reaches the tangible state M_j after
exactly h steps, and all states reached in between
are vanishing states], $M_r \in \hat{V}$,$M_j \in \hat{T}$.

This yields

$$P[M_r \longrightarrow^* M_j] = \sum_{h=1}^{\infty} P[M_r \xrightarrow{h} M_j].$$

From our discussions on Markov chains we know that, given the matrix P characterising the single step probabilities between all states, the h-step probabilities are given by P^h. From (9.2) we know that the single step probabilities out of vanishing states are described by matrices C and D. Thus the h-step probabilities $P[M_r \xrightarrow{h} M_j]$ are given by $C^{h-1} \times D$.

Note that D specifies the single step probabilities between vanishing and tangible states, and that M_j is a tangible state. So the probability $P[M_r \xrightarrow{h} M_j]$ is simply given by the corresponding element of matrix $C^{h-1} \times D$, namely $(C^{h-1} \times D)_{rj}$. Referring to Eq. (9.5) we can now write matrix P' as follows

$$P' = F + E \times \sum_{h=0}^{\infty} C^h \times D \quad \text{provided} \sum_{h=0}^{\infty} C^h \text{ exists.} \tag{9.6}$$

The steady state distribution $\tilde{\pi}$ of the reduced embedded Markov chain is given by

$$\tilde{\pi} = \tilde{\pi} \times P'; \sum_{M_s \in \hat{T}} \tilde{\pi}_s = 1, \tag{9.7}$$

if a unique solution for $\tilde{\pi}$ exists. The steady state distribution π of the Markovian process is given by Eq. (9.4) for tangible markings.

A prerequisite for the existence of P' is the existence of $\sum_{h=0}^{\infty} C^h$. How can we characterise the existence of this limit?

Define $u_i := 1 - \sum_{j=1}^{|\hat{V}|} c_{ij}$. $u_i > 0$ if the sum in row i of matrix C is less then 1 and $u_i = 0$ if it is 1. With that we can separate the set of vanishing states into two subsets $J_0 := \{M_i | u_i = 0\}$ and $J_1 := \{M_i | u_i > 0\}$. If the stochastic process is in a state $M_i \in J_0$, it can only transit to another vanishing state, being in a state $M_j \in J_1$ there is a positive probability to reach a tangible state. The existence of the limit in Eq. (9.6) can be analysed by inspecting the structure of the reachability set with respect to vanishing states. If the process can always reach tangible states, then this limit exists.

The reader should refer to Sec. 2.1.2 on absorbing Markov chains and transient behaviour where the matrix Q of transition probabilities between transient states correspond to matrix C of the transition probabilities between vanishing states in this section. First we define:

Definition 9.2 (A Trap.) *(cf. [110]) C has no trap* $:\Longleftrightarrow$
$\forall M_i \in J_0 : \exists M_j \in J_1$ *reachable from* M_i*; otherwise C has a trap.*

Theorem 9.1 *(cf. [110]) C has no trap iff* $(I - C)^{-1} = \sum_{h=0}^{\infty} C^h$ *exists.*

Because C consists of the transition probabilities between vanishing states, we speak of a timeless trap. If there is no timeless trap, we can determine matrix P'. Note that a timeless trap can only exist if the GSPN is not live, since all timed transitions are not live.

Another condition for the existence of a steady state distribution is the existence of a unique solution of Eq. (9.7), i.e. in case of a finite reachability set, the GSPN has to contain home states (cf. Sec. 5.3).

Example 9.3 *Consider the GSPN of Fig. 9.3 with* $w_i = i$*. The reachability set comprises the following markings:*
M_0 = (1,1,1,0,1),M_1 = (0,0,2,1,0),M_2 = (0,1,2,0,0),
M_3 = (1,0,1,1,1),M_4 = (2,1,0,0,2),M_5 = (2,0,0,1,2),
M_6 = (0,0,2,0,1),M_7 = (1,0,1,0,2),M_8 = (2,0,0,0,3)
and the corresponding reachability graph is shown in Fig. 9.4.
The markings M_0,M_1,M_2,M_3,M_4,M_5 *are vanishing and the markings* M_6,M_7,M_8 *are tangible. The matrix* P *of the embedded Markov chain is*

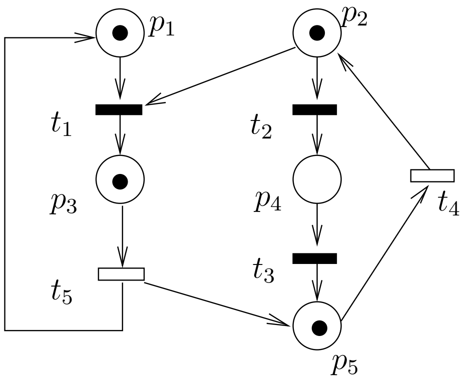


Figure 9.3 $w_i = i$ in the example

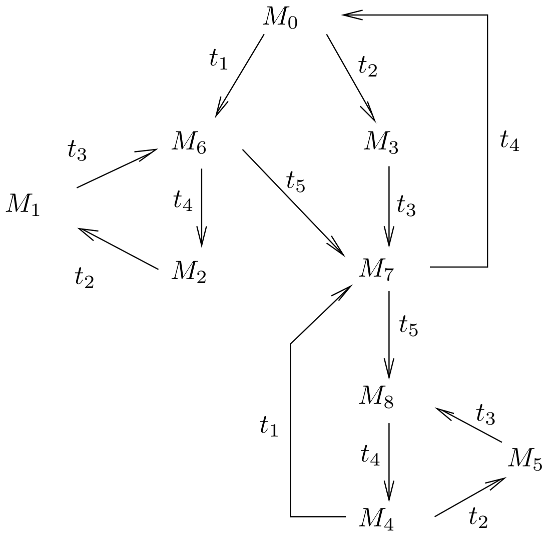


Figure 9.4 Reachability graph of the GSPN

$$P = \begin{pmatrix} C & D \\ E & F \end{pmatrix} = \left(\begin{array}{cccccc|ccc} 0 & 0 & 0 & \frac{2}{3} & 0 & 0 & \frac{1}{3} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{2}{3} & 0 & \frac{1}{3} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 0 & \frac{4}{9} & 0 & 0 & 0 & 0 & \frac{5}{9} & 0 \\ \frac{4}{9} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{5}{9} \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{array} \right)$$

Since matrix

$$(I-C)^{-1} = \begin{pmatrix} 1 & 0 & 0 & \frac{2}{3} & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \frac{2}{3} \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

we get

$$P' = F + E \times (I-C)^{-1} \times D = \begin{pmatrix} \frac{4}{9} & \frac{5}{9} & 0 \\ \frac{4}{27} & \frac{8}{27} & \frac{5}{9} \\ 0 & \frac{1}{3} & \frac{2}{3} \end{pmatrix}$$

Solving

$$\tilde{\pi} P' \;=\; \tilde{\pi} \;\; \textit{and} \;\; \sum_{i=6}^{8} \tilde{\pi}_i = 1$$

yields $\tilde{\pi}_6 = \frac{1}{11}$,$\tilde{\pi}_7 = \frac{15}{44}$,$\tilde{\pi}_8 = \frac{25}{44}$. *Thus the mean number of visits is*
$v_{66} = 1$,$v_{67} = \frac{4}{15}$,$v_{68} = \frac{4}{25}$,$v_{76} = \frac{15}{4}$,$v_{77} = 1$,$v_{78} = \frac{3}{5}$,$v_{86} = \frac{25}{4}$,$v_{87} = \frac{5}{3}$,$v_{88} = 1$.
The sojourn time $t_s(M_i)$ *for tangible marking* M_i *is given by*
$t_s(M_6) = t_s(M_7) = (4+5)^{-1} = \frac{1}{9}$,$t_s(M_8) = \frac{1}{4}$.
Let $\hat{X} = \tilde{\pi}_6 \times t_s(M_6) + \tilde{\pi}_7 \times t_s(M_7) + \tilde{\pi}_8 \times t_s(M_8) = \frac{301}{1584}$,
then we get the following mean cycle times $t_c(M_i)$ *for tangible marking* M_i :
$t_c(M_6) = \frac{1}{\tilde{\pi}_6}\hat{X} = \frac{301}{144}$,$t_c(M_7) = \frac{1}{\tilde{\pi}_7}\hat{X} = \frac{301}{540}$,$t_c(M_8) = \frac{1}{\tilde{\pi}_8}\hat{X} = \frac{301}{900}$,
which gives the steady state distribution for tangible states:
$\pi_6 = \frac{t_s(M_6)}{t_c(M_6)} = \frac{16}{301}$, $\quad \pi_7 = \frac{60}{301}$, $\quad \pi_8 = \frac{225}{301}$.

The steady state distribution π of the GSPN can be employed for the calculation of performance figures as shown in Ch. 8. Only the throughput at an immediate transitions has to be calculated differently, since immediate transitions are only enabled in vanishing states.

Let $r = (r_i)$ be a vector, which determines for a vanishing state M_i the rate at which the state is entered and left immediately at steady state. Since we know the steady state distribution π of tangible states, we can calculate r as follows. Let $\tilde{E}$ be the matrix specifying the rates for a specific state change leaving a tangible state M_i and entering a vanishing state M_j, i.e.

$$\tilde{E} = (\tilde{e}_{ij}),\ \tilde{e}_{ij} = \sum_{k:t_k \in \{t \in T | M_i[t>M_j\} \cap EN_T(M_i)} w_k \;\; ; M_i \in \hat{T}, M_j \in \hat{V}$$

Then

$$r = \pi \times \tilde{E} \times (I-C)^{-1}$$

The **throughput at an immediate transition** $t_j \in T_2$ is then given by

$$\overline{d_j} = \sum_{M_i \in EN_j \cap \hat{V}} r_i \frac{w_j}{\sum_{k:t_k \in EN_T(M_i)} w_k}$$

where $EN_j = \{M \in R(PN) | M[t_j >\}$.

Example 9.4 *For the GSPN of Fig. 9.3 (cf. Ex. 9.3) we have*

$$\tilde{E} = \begin{pmatrix} 0 & 0 & 4 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 & 0 \end{pmatrix}$$

and thus

$$r = \pi \times \tilde{E} \times (I - C)^{-1} = \frac{1}{301}(240,64,64,160,900,600)$$

The throughputs at transition t_1 and t_2 are now

$$\begin{aligned} \overline{d_1} &= \frac{240}{301} \times \frac{1}{3} + \frac{900}{301} \times \frac{1}{3} = \frac{380}{301} \\ \overline{d_2} &= \frac{240}{301} \times \frac{2}{3} + \frac{64}{301} \times 1 + \frac{900}{301} \times \frac{2}{3} = \frac{824}{301} \end{aligned}$$

The throughput at the timed transition t_5 can be calculated as for SPNs which yields

$$\overline{d_5} = (\pi_6 + \pi_7) \times 5 = \frac{380}{301}$$

and the throughput at transition t_4 is $\lambda_4 = 4$, since it is enabled in all tangible markings.

9.2 Qualitative Analysis of GSPNs

In the previous section we saw that qualitative properties, like the existence of home states, are essential preconditions for the existence of a steady state distribution of the stochastic process represented by the GSPN.

For the qualitative analysis of a GSPN we want to exploit the underlying Place-Transition net of the GSPN and use the algorithms presented in Sec. 5.4.

Our interest is therefore in the combined qualitative and quantitative analysis illustrated in Fig. 9.5. First of all, while neglecting time, certain qualitative aspects, such as boundedness and liveness of the Place-Transition net are fully analysed. Then, if the Place-Transition net satisfies all the required qualitative properties, it may be worthwhile to do a quantitative analysis. For this procedure to apply, all qualitative features of the Place-Transition net have to remain valid after the introduction of time. Unfortunately, this is not always the case.

Incorporating time, particularly introducing a priority relation on transitions, changes the properties of the Place-Transition net significantly [17, 19, 20]. In this section we investigate this further.

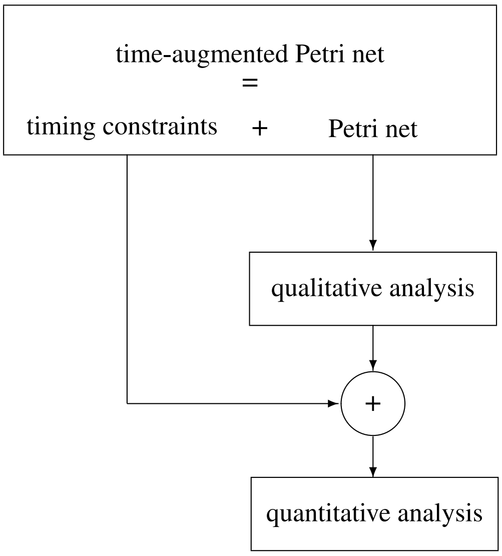


Figure 9.5 Principal analysis procedure for time-augmented Petri nets

In the following discussion we will denote the underlying Place-Transition net of a GSPN by PN. Since firing of immediate transitions has priority on firing of timed transitions the enabling rule in GSPNs is simply given by

Definition 9.3 *A transition $t \in T$ of a GSPN is* enabled *at M, denoted by $M[t>_{GSPN}$, iff $M(p) \geq I^-(p,t), \forall p \in P$ and $(\exists t' \in T_2 : M(p) \geq I^-(p,t'), \forall p \in P \Longrightarrow t \in T_2)$.*

In other words, a transition is enabled iff the usual enabling rule for Place-Transition nets holds and if there exists another enabled immediate transition t', then t must also be an immediate transition, since otherwise the firing of t' would have priority over the firing of t. Or to put it differently, $t \in T$ of a GSPN is enabled at M iff $t \in EN_T(M)$, with $EN_T(M)$ defined on page 140.
With respect to the enabling rule in GSPNs, all definitions concerning the dynamic behaviour of GSPNs are similar to those concerning Place-Transition nets (cf. Def. 5.4 and 5.5). We will use subscripts PN and $GSPN$ to distinguish between the same notions for the underlying Place-Transition net, e.g. $M[t>_{PN}$,$M[t>_{GSPN}$,$M[\sigma>_{PN}$, $M \rightarrow^*_{GSPN} M'$ or $R(GSPN,M_0)$ etc.
We first of all investigate timeless traps in the context of the state space of a GSPN. A timeless trap means that the GSPN can reach a marking whence only immediate transitions can fire in the future.

Definition 9.4 *A GSPN has a* timeless trap, *iff $\exists M \in R(GSPN,M_0)$: $\forall k \in \mathbb{N} : \exists \sigma \in T^*, |\sigma| \geq k$ such that $M[\sigma>_{GSPN}$ and $\forall \tilde{\sigma} \in T^* : M[\tilde{\sigma}>_{GSPN} \Longrightarrow \tilde{\sigma} \in T_2^*$.*

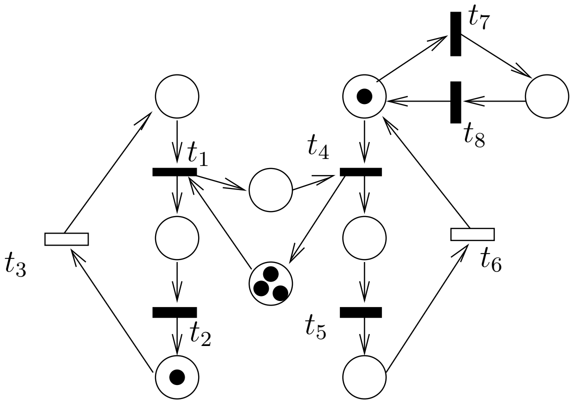


Figure 9.6 A GSPN with a timeless trap

where $|\sigma|$ denotes the length of σ. Since T is a finite set, the existence of a timeless trap implies that some immediate transitions fire infinitely often. Fig. 9.6 depicts an example of a timeless trap which is caused by the firings of the immediate transitions t_7 and t_8. Note that this definition of a timeless trap corresponds directly with that for a trap of the matrix C in Def. 9.2. Timeless traps are avoided by the following condition.

Definition 9.5 *Condition NoTT.* [3]
Condition NoTT $:\Longleftrightarrow \forall \tilde{T} \subseteq T_2 : \tilde{T} \neq \emptyset \Longrightarrow \bullet\tilde{T} \neq \tilde{T}\bullet$.

Condition NoTT prevents the existence of a strongly connected subnet generated by immediate transitions.
Let GSPN be a Generalized Stochastic Petri net with a finite reachability set. Then condition NoTT implies that the GSPN has no timeless trap.

Theorem 9.2 *Let GSPN have a finite reachability set.*
Condition NoTT $\Longrightarrow$ *GSPN has no timeless trap.*

Proof.: Assume there is a timeless trap and let $\tilde{T}$ be the set of immediate transitions which fire infinitely often. Let $\tilde{P}$ be the set of places which are either an input place for $\tilde{T}$ or an output place, but not both, i.e. $\tilde{P} = \left(\bullet\tilde{T} \setminus \tilde{T}\bullet\right) \cup \left(\tilde{T}\bullet \setminus \bullet\tilde{T}\right)$. If $\tilde{T} \neq \emptyset$ we have $\tilde{P} \neq \emptyset$ since otherwise, $\bullet\tilde{T} = \tilde{T}\bullet$ contradicting condition NoTT. There are two cases to consider

1. $\exists\, p \in \bullet\tilde{T} \setminus \tilde{T}\bullet$.
   Since $p \notin \tilde{T}\bullet$ not all transitions of $\tilde{T}$ can fire infinitely often, contradicting our assumption.

[3] NoTT stands for No Timeless Trap.

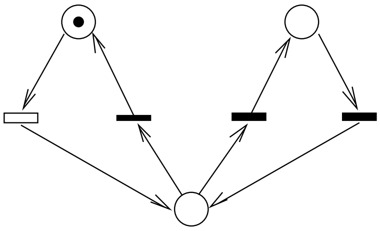


Figure 9.7 GSPN not satisfying condition NoTT

2. $\exists\, p \in \tilde{T}\bullet \setminus \bullet\tilde{T}$.
   Since $p \notin \bullet\tilde{T}$ and all $t \in \tilde{T}$ fire infinitely often, the GSPN has an infinite reachability set, since the number of tokens on p is not bounded, which is again a contradiction. □

Condition NoTT is not necessary to avoid timeless traps, as illustrated by the GSPN in Fig. 9.7. From the proof of Theorem 9.2 it follows that the next theorem holds.

Theorem 9.3 *Condition NoTT* $\Longrightarrow \exists h_0 \in \mathbb{N}: \; C^h = 0, \forall h \geq h_0.$

where C is the matrix given on page 142. Remember that C^h describes the h-step transition probabilities between vanishing states. From the proof of Th. 9.2 we know that the number of firings of immediate transitions is always bounded. The theorem essentially tells us that when condition NoTT applies, the number of steps between vanishing states do not exceed limit h_0. This implies that there are no loops between vanishing states in the state space of the GSPN. This property is particularly important when eliminating vanishing states while generating the state space. It is only necessary to multiply the probabilities along every path, starting at a particular tangible state M_i and leading to a particular tangible state M_j, and sum all these path probabilities to determine the single step probability between these two tangible states.
Condition NoTT can be verified very easily with the algorithm in Fig. 9.8.
Incorporating time also changes other properties of a Place-Transition net. E.g. an unbounded Place-Transition net may yield a bounded GSPN (see Fig. 9.9). Fortunately the reverse does not present a similar problem.

Theorem 9.4 *PN bounded* $\Longrightarrow$ *GSPN bounded.*

Proof. Obvious, since $R(GSPN,M_0) \subseteq R(PN,M_0)$. □

Other properties unveil greater problems. E.g., there is no relationship between the liveness of a GSPN and its underlying Place-Transition net PN. Fig. 9.10 shows a non-live GSPN, whose PN is live. The immediate transition t^* is only enabled at the initial marking which is the only marking where places p_1 and p_2

```
$T_H := T_2$
while $\bullet T_H \neq T_H \bullet$ do
begin
    $P_H := (\bullet T_H \setminus T_H \bullet) \cup (T_H \bullet \setminus \bullet T_H)$
    $\forall t \in T_H$ do
        if $(\bullet t \cap P_H) \cup (t \bullet \cap P_H) \neq \emptyset$
        then $T_H := T_H \setminus \{t\}$
end
if $T_H = \emptyset$
    then Condition NoTT is satisfied
    else Condition NoTT is not satisfied
```

Figure 9.8 Algorithm for checking condition NoTT

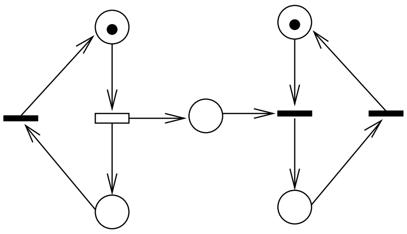


Figure 9.9 GSPN bounded, PN not bounded

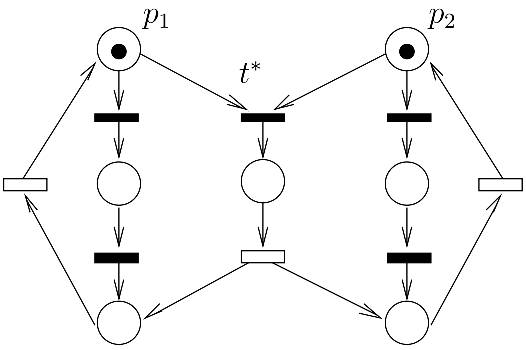


Figure 9.10 PN live, GSPN not live

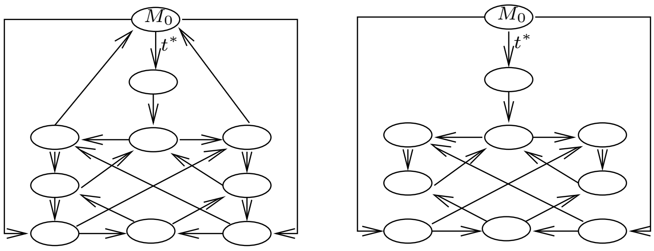


Figure 9.11 Structure of reachability graphs: PN (left); GSPN (right)

are marked simultaneously. The reachability graphs of PN and GSPN are shown in Fig. 9.11.

Fig. 9.12, on the other hand, depicts a live GSPN with a non-live underlying Petri net. After firing t_2 and t_3 the Place-Transition net is dead, but the firing sequence t_2t_3 can not occur in the GSPN, since the firing of t_1 has priority on the firing of t_2. The only conclusion we can establish for a GSPN whose underlying Place-Transition net is live is the following:

Lemma 9.1 *Let GSPN be a Generalized Stochastic Petri net whose underlying Place-Transition net is live. Then* $\forall M \in R(GSPN,M_0) : \exists t \in T : M[t >_{GSPN}$.

Proof. Since the enabling rule of GSPNs is a restriction of the enabling rule of Place-Transition nets, we know $R(GSPN,M_0) \subseteq R(PN,M_0)$. Liveness of the underlying Place-Transition net implies that $\forall M \in R(GSPN,M_0) : \exists t \in T : M[t >$. If $t \in T_2$ then also $M[t >_{GSPN}$ holds and our proof is complete. If $t \in T_1$ then $\neg M[t >_{GSPN}$ will only hold if there is an immediate transition which prevents the enabling of t, i.e. $\exists t' \in T_2 : M[t' >_{GSPN}$ so that t' is enabled at M according to the enabling rules of GSPNs. □

So in such GSPNs there is always some enabled transition, although there might be no live transition (see Fig. 9.13). Lemma 9.1 only implies that total deadlocks cannot occur.

Furthermore, although the PN contains home states, this may not be true for the GSPN as shown in Fig. 9.14. The reachability graph of the GSPN, see Fig. 9.15, now contains two terminal strongly connected components (cf. Def. 5.7 and Th. 5.4 on page 94).

All these examples illustrate that the analysis procedure in Fig. 9.5 has to be modified or possibly completely changed to usefully combine qualitative and quantitative analysis. What possibilities do we have to cope with this basic problem? One way is to modify existing Petri net algorithms to render them suitable for the analysis of time-augmented nets. This may not be so useful, as Petri net theory and existing software tools could then no longer be used directly. Furthermore, all future research results in the Petri net area would have to be adapted to time-augmented Petri nets.

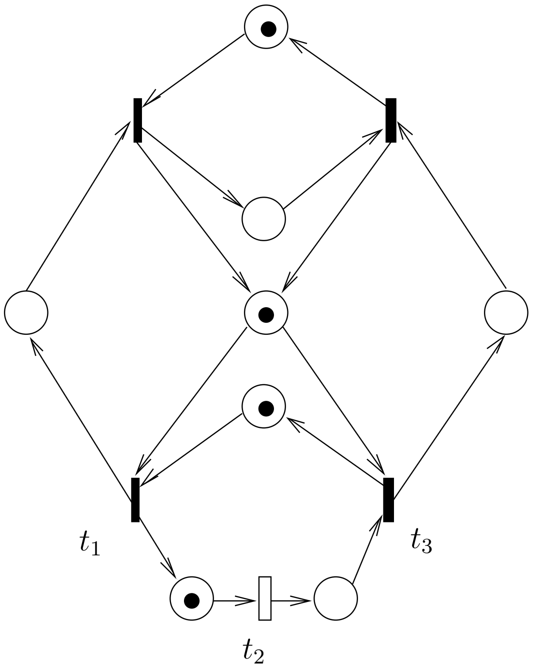


Figure 9.12 GSPN live, PN not live

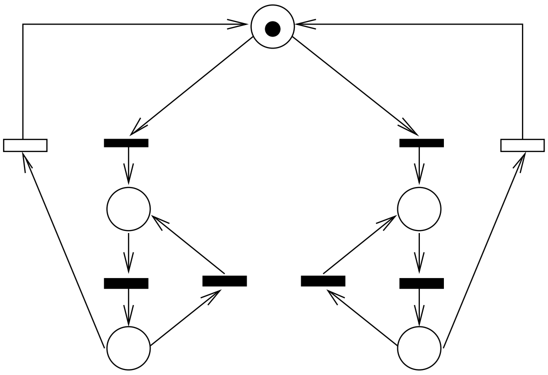


Figure 9.13 GSPN with no live transitions

Another way is to determine suitable restrictions for an integration of time, such that the results of a qualitative analysis remain valid for a quantitative analysis as well. The latter idea leads to the analysis procedure illustrated in Fig. 9.16. The great benefit of this is that the standard theory of Petri nets remains unaffected. Finding such restrictions for general net structures is difficult. We restrict ourselves to EFC-nets. Another reason for this restriction is that this class of Petri nets has been studied exhaustively, and many qualitative properties can be characterised by conditions which can be tested efficiently (cf. Sec. 5.4.3).

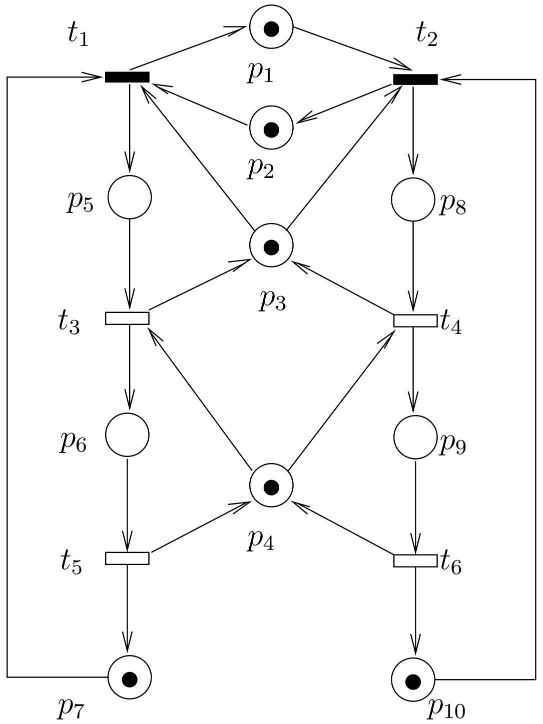


Figure 9.14 GSPN with no home states

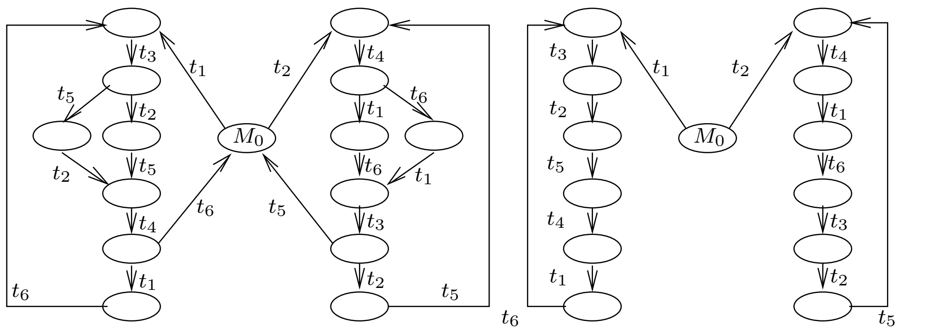


Figure 9.15 Reachability graphs: PN (left); GSPN (right)

9.2.1 Qualitative Analysis of EFC-GSPNs

Definition 9.6 *A GSPN is an* EFC-GSPN *iff its underlying Place-Transition net is an EFC-net.*

Since EFC-nets have the structures shown in Fig. 5.22 on page 105, it seems obvious to insist that conflicts may only occur between transitions of the same kind. When this condition, which we shall call EQUAL-Conflict, holds for an EFC-GSPN, all positive properties of the underlying PN remain valid.

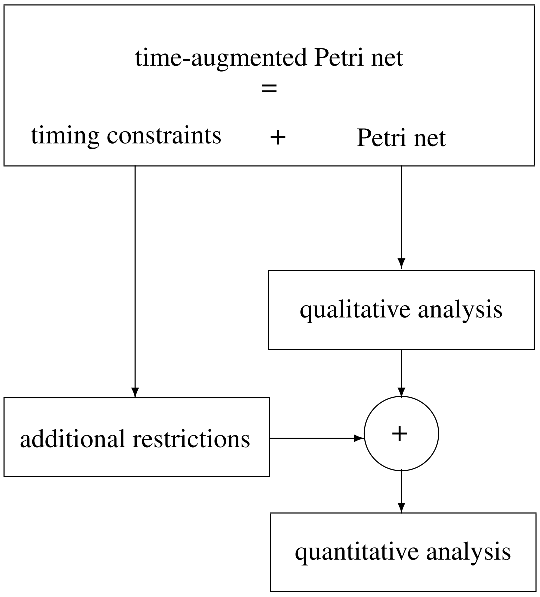


Figure 9.16 Modified analysis procedure for time-augmented Petri nets

Definition 9.7 *A* $GSPN = (PN,T_1,T_2,W)$ *satisfies condition* EQUAL-Conflict*, iff* $\forall t,t' \in T : \bullet t \cap \bullet t' \neq \emptyset \Longrightarrow \{t,t'\} \subseteq T_1$ *or* $\{t,t'\} \subseteq T_2$*.*

We will show that this condition ensures the absence of timeless traps and that liveness and the existence of home states extend to the GSPN.

Theorem 9.5 ([17, 19, 20]) *If we are given an EFC-GSPN whose underlying Place-Transition net is live and bounded, the following holds:*

1. *Condition EQUAL-Conflict* $\Longrightarrow$ *GSPN has no timeless trap.*
2. *Condition EQUAL-Conflict* $\Longleftrightarrow$ *GSPN live.*
3. *Condition EQUAL-Conflict* $\Longrightarrow$ *GSPN has home states.*

Proof. Since the proof of part 3 is lengthy we will only prove the first two parts.
Proof of part 1: Assume the GSPN has a timeless trap, i.e.
$\exists M \in R(GSPN,M_0)$:
$\forall k \in \mathbb{N} : \exists \sigma \in T^*, |\sigma| \geq k$ such that $M[\sigma >_{GSPN}$ and
$\forall \tilde{\sigma} \in T^* : M[\tilde{\sigma} >_{GSPN} \Longrightarrow \tilde{\sigma} \in T_2^*$.
Since PN is bounded Th. 9.4 ensures boundedness of the GSPN. Thus $R(GSPN,M_0)$ is finite and there exists a terminal strongly connected component (cf. Def. 5.7) $C_M \subseteq R(GSPN,M)$. Define $\tilde{T} := \{t \in T | \exists M' \in C_M : M'[t >_{GSPN}\}$ and $\tilde{P} := \bullet\tilde{T}$.

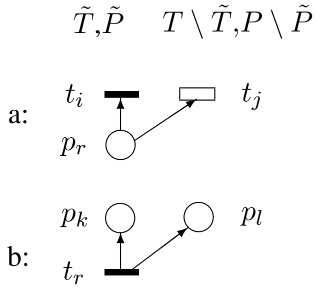


Figure 9.17 Illustration of part 1 of the proof

$\tilde{T}$ can be referred to as the set of live transitions with regard to C_M. Because of our assumption $\tilde{T} \subseteq T_2$ holds. Since the Place-Transition net is live and bounded, it is strongly connected according to Th. 5.1. So one of the following two cases holds (see Fig. 9.17):

a: $\exists p_r \in \tilde{P} : p_r\bullet \supseteq \{t_i,t_j\}$ and $t_i \in \tilde{T}, t_j \in T \setminus \tilde{T}$.
The assumption of a timeless trap yields $t_i \in T_2$. If $M'[t_i >_{GSPN}$ for some marking $M' \in C_M$, the EFC-net structure and condition EQUAL-Conflict imply $M'[t_j >_{GSPN}$ contradicting $t_j \notin \tilde{T}$.

b: $\exists t_r \in \tilde{T} : t_r\bullet \supseteq \{p_k,p_l\}$ and $p_k \in \tilde{P}, p_l \in P \setminus \tilde{P}$.
Since $t_r \in \tilde{T}$, t_r fires infinitely often with regard to C_M and our assumption of a timeless trap. Because $p_l \notin \tilde{P} = \bullet\tilde{T}$, place p_l is not bounded, contradicting the boundedness of the GSPN.

Thus our assumption is not valid, which completes the proof of part 1.

Proof of part 2:

if: Assume the GSPN is not live.
Then $\exists t \in T : \exists M \in R(GSPN,M_0) : \forall M' \in R(GSPN,M) : \neg M'[t >_{GSPN}$.
Since the net is an EFC-GSPN there exists a place $p \in \bullet t$ which is empty in all markings of $R(GSPN,M)$. Note that from part 1 we know that the GSPN has no timeless trap. This further implies that all input transitions of that place are not live with regard to $R(GSPN,M)$, i.e.

$$\forall t \in \bullet p : \neg M'[t >_{GSPN}, \forall M' \in R(GSPN,M). \tag{9.8}$$

Define $T_{dead} := \{t \in T | \neg M'[t >_{GSPN}, \forall M' \in R(GSPN,M)\}$ and $P_{empty} := \{p \in P | M'(p) = 0, \forall M' \in R(GSPN,M)\}$.
Because of the EFC-net structure we have $P_{empty}\bullet = T_{dead}$ and furthermore $\bullet P_{empty} \subseteq T_{dead}$ from (9.8). Thus we have $\bullet P_{empty} \subseteq P_{empty}\bullet$ and P_{empty} is an empty deadlock, which contradicts the dt-property, since the Place-Transition net is a live EFC-net (cf. theorem 5.16 on page 106).

only if: Assume that condition EQUAL-Conflict does not hold, i.e. $\exists t,t' \in T : \bullet t \cap \bullet t' \neq \emptyset$ and $\{t,t'\} \not\subseteq T_1$ and $\{t,t'\} \not\subseteq T_2$.
Since the GSPN is an EFC-GSPN, obviously t and t' can not be both live, contradicting the liveness of the GSPN. □

Note that in part 3 of Th. 9.5, because of the EFC-net structure, boundedness and liveness of the Place-Transition net implies the existence of home states for the Place-Transition net as well. [19, 20] shows that Th. 9.5 also holds for EC-nets (cf. page 125) and extends the results to general net structures and multiple levels of priorities for immediate transitions.

9.3 Further Remarks on GSPNs

The difficulties with the qualitative analysis of GSPNs apply to nearly any kind of time-augmented Petri net. This is an intrinsic problem which cannot be avoided. Since immediate transitions have priority over timed transitions, GSPNs are equivalent to Turing machines and thus several properties, such as liveness, are undecidable for general GSPNs (cf. [132]).
There are yet further disadvantages concerning the modelling of systems using GSPNs.
GSPNs, like Place-Transition nets, do not distinguish between individual tokens so that the graphical representation very soon becomes complex. Defining a coloured version of GSPNs (Coloured GSPNs, CGSPNs) can be done easily by folding places and transitions of the same kind. The firing rates or firing weights are then associated with the specific colour of a transition. Analysis of such a CGSPN can, e.g., be done by unfolding the net and analysing the GSPN as usual.

Definition 9.8 *A Coloured GSPN (CGSPN) is a 4-tuple*
$CGSPN = (CPN,T_1,T_2,W)$ *where*

- $CPN = (P,T,C,I^-,I^+,M_0)$ *is the underlying Coloured Petri net.*
- $T_1 \subseteq T$ *is the set of timed transitions,* $T_1 \neq \emptyset$*,*
- $T_2 \subset T$ *is the set of immediate transitions,* $T_1 \cap T_2 = \emptyset$*,*
  $T = T_1 \cup T_2$*,*
- $W = (w_1,\ldots,w_{|T|})$ *is an array whose entry* w_i *is a function of* $[C(t_i) \mapsto \mathbb{R}^+]$ *such that* $\forall c \in C(t_i) : w_i(c) \in \mathbb{R}^+$
  - *is a (possibly marking dependent) rate of a negative exponential distribution specifying the firing delay with respect to colour c, if* $t_i \in T_1$ *or*

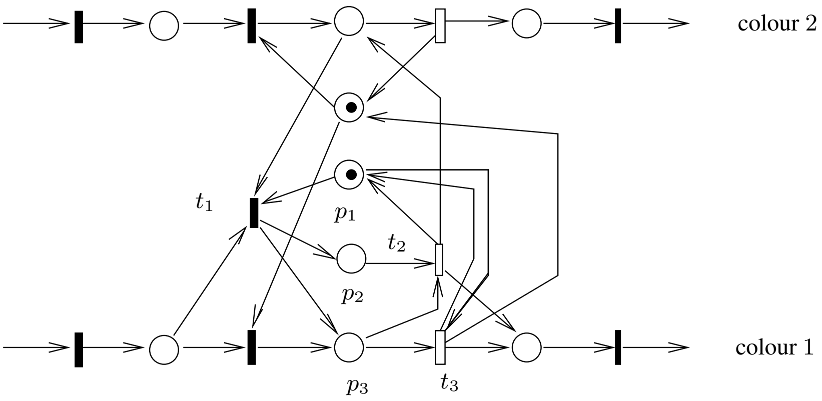


Figure 9.18 2 class -/M/1-"preemptive-resume priority" queue modelled by a GSPN; $w_2 = w_3$

- *is a (possibly marking dependent) firing weight with respect to colour c, if $t_i \in T_2$.*

A more severe disadvantage of GSPNs is the difficulties which arise if one models queues, which we discussed in Ch. 3. Modelling scheduling strategies is particularly difficult.

Fig 9.18 illustrates this point. The GSPN models a queue which serves two colours of tokens according to the pre-empt resume priority strategy (cf. Sec. 3.8). A newly arriving token with higher priority (colour 1) pre-empts the service of the token being served at that moment. The pre-empted token will continue his task, at the point of interruption, after completion of service of all tokens with higher priority. Transition t_1 is enabled if pre-emption is to occur. Firing of t_1 pre-empts tokens of colour 2. The markings of p_1 and p_2 keep track of this event. This is the reason why the service of a token of colour 1 is modelled by two timed transitions, t_2 and t_3, with identical parameters. Note that this model of a pre-emptive priority scheduling strategy with resumption is only correct if exponentially distributed service times are assumed. Other service time distributions require more complex models.

Also simple scheduling strategies, like FCFS, might not be easy to represent using plain GSPN elements. Consider a queue where 2 colours of tokens arrive according to exponential distributed interarrival times and whose service times are exponentially distributed. If the scheduling strategy is FCFS, i.e. our system is an M/M/1-FCFS queue, we have to encode the colour of the token in each position of the queue. Assume that an upper bound for the number of tokens in the queue is given, e.g. 3, then the GSPN in Fig. 9.19 would model the queue accurately.

Transitions $t_1$and t_2 model the arrival of a token of either colour and transitions t_3 and t_4 model the service of the token in front of the queue. The places p_{i1} and p_{i2} represent position i of the queue and the place p_{i0} ensures that this position is occupied by at most one token. Since entering a position is modelled by

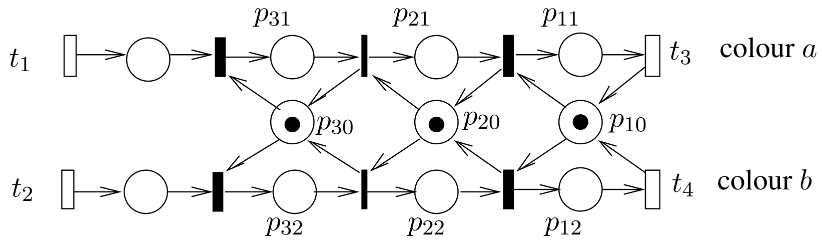


Figure 9.19 GSPN model of a FCFS queue

immediate transitions, a token entering position 3 of the queue will immediately advance to position 2 and 1, if they are free.

This way of modelling a FCFS queue with GSPN elements works fine if an upper bound for the number of tokens is known *a priori*. Suppose, however, that we want to perform several experiments with different initial markings. This will necessitate a modification of the GSPN model of the queue for each experiment. If there is no upper bound known beforehand it is even more difficult. Additionally, if the service time of a queue is specified by, e.g. a Coxian distribution, it becomes just about impossible to model such a queue with a GSPN.

In an attempt to resolve these difficulties, GSPNs were extended to Queueing Petri nets (QPNs).

Exercise 9.1

1. *Consider a GSPN with a finite reachability set. Convince yourself that for such a GSPN the existence of home states is necessary and sufficient for the existence of the steady state distribution, provided there are no timeless traps.*

2. *Determine the steady state probability of the Producer-Consumer example 9.1 for* $w_1 = 12.3, w_2 = 0.56, w_3 = 2.0, w_4 = w_5 = 32.1, w_6 = 2.0.$

Exercise 9.2 *Give the formal counterparts of Def. 5.4 and 5.5 for GSPNs.*

Exercise 9.3

- *Apply the algorithm given in Fig. 9.8 to the GSPN of Fig. 9.6.*
- *Prove that the GSPN given in Fig. 9.7 does not have a timeless trap.*
- *Design an algorithm for the construction of the reduced embedded MC where vanishing states are eliminated on the fly.*

Exercise 9.4

1. *Prove that the GSPN of Fig. 9.13 has no live transitions.*

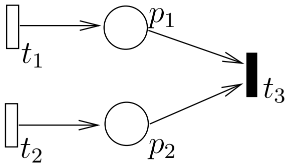


Figure 9.20 An unbounded GSPN model

2. *Prove the statements in the text concerning liveness of the GSPNs in Fig. 9.10 and 9.12.*

Exercise 9.5 *Show that the GSPN of Fig. 9.14 has no home states and that for the underlying Place-Transition net home states do exist.*

Exercise 9.6 *Prove or disprove each conjecture:*
Given a GSPN satisfying condition EQUAL-Conflict. Then

1. *PN live $\Longrightarrow$ GSPN live.*
2. *GSPN live $\Longrightarrow$ PN live.*
3. *PN has home states $\Longrightarrow$ GSPN has home states.*

Exercise 9.7

1. *Design a GSPN model for a queue which serves 3 colours of tokens according to a 2-stage Coxian distribution and the scheduling strategy depicted in Fig. 9.18.*
2. *Design a GSPN representation of a M/C/1-LCFSPR queue with a 2-stage Coxian distribution and 2 colours of tokens, where at most 4 tokens are present. LCFSPR stands for Last Come First Served Preemptive Resume, which means that a newly arriving token preempts the token in service. After completing service of the new arrived token, the preempted token continues at the point of interruption.*

Exercise 9.8 *Consider the unbounded GSPN of Fig. 9.20 with*
$w_1 = \lambda, w_2 = \mu, w_3 = 1$. *Determine the corresponding steady state distribution and conditions for its existence.*
Hint: Have a look at Fig. 3.5 and Eqs. 3.5 and 3.6 on page 62.

10 Queueing Petri Nets

Queueing Petri Nets (QPNs; F. Bause, H. Beilner, P. Kemper [18, 22, 29]) try to eliminate the problem of representing scheduling strategies by integrating the concept of queues (cf. Ch. 3) into a coloured version of GSPNs (CGSPNs). This is done by partioning the set of places into two subsets: *queueing places* and *ordinary places*.

A queueing place (cf. Fig. 10.1) consists of two components: the queue and a depository for tokens which have completed their service at this queue. Tokens, when fired onto a queueing place by any of its input transitions, are inserted into the queue according to the scheduling strategy of the queue. Tokens in a queue are not available for the transitions. After completion of its service, the token is placed onto the depository. Tokens on this depository are available to all output transitions of the queueing place. An enabled timed transition will fire after an exponentially distributed time delay and an immediate transition fires immediately as in GSPNs.

Since we are now no longer forced to describe scheduling strategies with only GSPN elements, the description of systems is simplified. Since the arrival process is determined by the firings of the input transitions of a queueing place, we will omit its specification in the Kendall notation of a queue (cf. Sec. 3).

Example 10.1 *Consider the FCFS queue with* 2 *colours of tokens as described in Sec. 9.3 (see also Fig. 9.19). Using queueing places we have no problem to describe a model even for an unlimited number of tokens. Fig. 10.2 shows the corresponding QPN employing the shorthand notation for queueing places as given in Fig. 10.1.*

The definition of a QPN is as follows:

Definition 10.1 *A Queueing Petri net (QPN) is a triple*
$QPN = (CGSPN, P_1, P_2)$ *where*

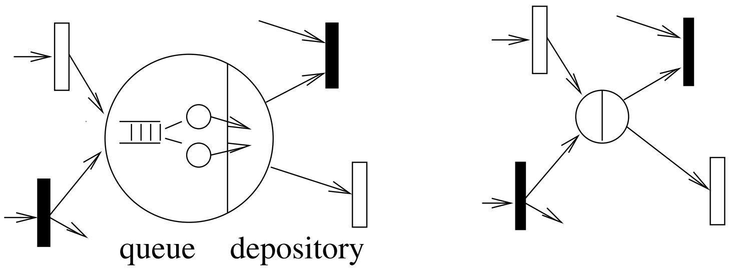


Figure 10.1 A queueing place and its shorthand notation

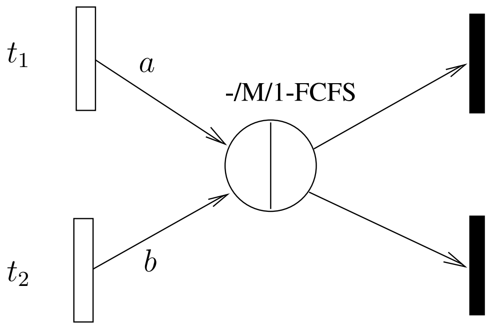


Figure 10.2 QPN model of a FCFS queue with 2 colours of tokens

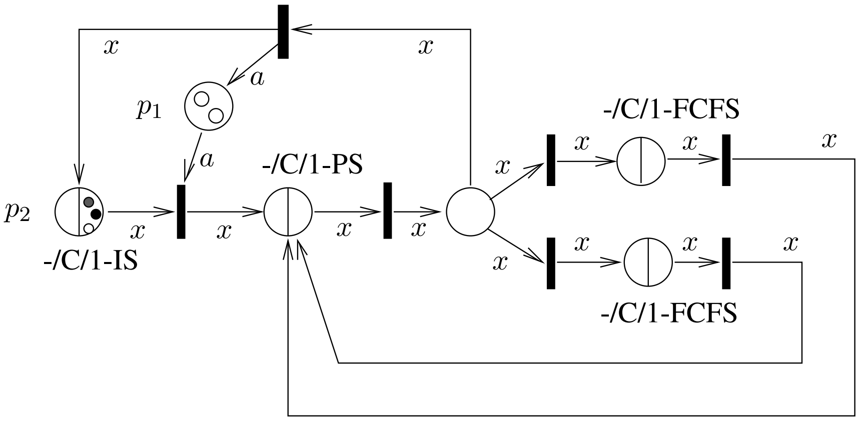


Figure 10.3 QPN model of a central server system with memory constraints

- *$CGSPN$ is the underlying Coloured $GSPN$,*
- *$P_1 \subseteq P$ is the set of queueing places and*
- *$P_2 \subseteq P$ is the set of ordinary places, $P_1 \cap P_2 = \emptyset$,$P = P_1 \cup P_2$.*

Note that if $P_1 = \emptyset$ the QPN describes a CGSPN so that the following relationship holds: CGSPNs $\subset$ QPNs.

Example 10.2 *Fig. 10.3 shows an example of a central server system with memory constraints. Place p_2 represents several terminals, where users start jobs after a certain thinking time. These jobs request service at the CPU (represented by the -/C/1-PS queue where C stands for a Coxian distribution) and two I/O systems (represented by the -/C/1-FCFS queues). To enter the subsystem (CPU + I/O systems) each job has to allocate a certain amount of memory. For simplicity the memory size needed by each job is assumed to be the same, which is represented by a token of colour a on place p_1.*

10.1 Quantitative Analysis of QPNs

Since we have introduced queueing places, which consist of queues and depositories, a marking M of a QPN consists of two parts $M = (n,m)$ where n specifies the state of all queues and m is the marking of the underlying CGSPN. For a queueing place $p \in P_1$, $m(p)$ denotes the marking of the depository. The initial marking M_0 of the QPN is given by $M_0 = (O,m_0)$, where O is the state describing that all queues are empty and m_0 is the initial marking of the CGSPN. Similar to GSPNs the firing of immediate transitions has priority over the firing of timed transitions and the service of tokens in queues. Thus, as in GSPNs, the reachability set of a QPN comprises vanishing and tangible states. If at least one immediate transition is enabled, M is a vanishing state and tangible otherwise. Thus

$$\begin{aligned} EN_T(M) := \{(t,c)| \quad & \qquad\qquad m[(t,c) > \text{ and} \\ & \text{if } \exists t' \in T_2, c' \in C(t') : m[(t',c') > \text{ then } t \in T_2\} \end{aligned}$$

where $m[(t,c) >$ denotes the usual enabling in CPNs (cf. Def. 6.5). $EN_T(M)$ is the set of enabled transitions at a marking M with respect to a colour $c \in C(t)$, keeping the priority of immediate transitions in mind.

QPNs can be analysed like GSPNs by calculating the steady state distribution of the reduced embedded Markov chain. For the remaining discussion we consider only QPNs with finite reachability sets. To simplify notation let us assume that all queues of the QPN are of type -/M/1-PS and that the first $|P_1|$ places are queueing places. The service rate of a colour-c token in the queue of a queueing place p is defined as $\mu(p,c)$. A possible state descriptor for such a queue is a function $n(p) : C(p) \mapsto \mathbb{N}_0$ where $n(p)(c)$ denotes the number of colour-c tokens in the queue.

State transitions occur due to the arrival or the service of tokens. With respect to a service of a colour $c \in C(p)$ token the state $n(p)$ changes to

$$n'(p)(\tilde{c}) = \begin{cases} n(p)(\tilde{c}) - 1 & \text{if } \tilde{c} = c \\ n(p)(\tilde{c}) & \text{otherwise} \end{cases}$$

The corresponding rate is given by

$$r(n(p)(c)) = \frac{n(p)(c)}{\sum_{c' \in C(p)} n(p)(c')} \mu(p,c)$$

Recall that each token is served with $\frac{1}{\sum_{c' \in C(p)} n(p)(c')}$ of the server's capacity and that $n(p)(c)$ colour-c tokens are in the queue. If the queue is empty, i.e. $\sum_{c' \in C(p)} n(p)(c') = 0$, we define $r(n(p)(c)) = 0$.

The possible state transitions of the QPN can be described in the following way. Let $M = (n,m)$ be a marking of the QPN. A directly reachable marking $M' = (n',m')$ is defined as follows. If $n(p)(c) > 0$ for some $p \in P_1, c \in C(p)$ then M' with respect to a service in place p according to colour c is given by

$$n'(\tilde{p})(\tilde{c}) = \begin{cases} n(\tilde{p})(\tilde{c}) & \text{if } \tilde{p} \neq p \text{ or } \tilde{c} \neq c \\ n(\tilde{p})(\tilde{c}) - 1 & \text{otherwise} \end{cases} \tag{10.1}$$

$$m'(\tilde{p})(\tilde{c}) = \begin{cases} m(\tilde{p})(\tilde{c}) & \text{if } \tilde{p} \neq p \text{ or } \tilde{c} \neq c \\ m(\tilde{p})(\tilde{c}) + 1 & \text{otherwise} \end{cases} \tag{10.2}$$

and we denote this case by $M[(p,c) >_{QPN} M'$. Eq. (10.1) says that a token of colour c has left the queue in p and Eq. (10.2) tells us that this token is now on the depository of p. If $(t,c') \in EN_T(M)$, then M' is given by

$$n'(p)(c) = n(p)(c) + I^+(p,t)(c')(c), \forall p \in P_1, c \in C(p) \tag{10.3}$$

$$m'(p)(c) = \begin{cases} m(p)(c) - I^-(p,t)(c')(c) & \text{if } p \in P_1 \\ m(p)(c) + I^+(p,t)(c')(c) - I^-(p,t)(c')(c) & \text{otherwise} \end{cases} \tag{10.4}$$

This means, tokens fired onto a queueing place are inserted into the queue (cf. Eq. (10.3)); the tokens at the depository are removed and ordinary places are marked according to the usual firing rule of CPNs (cf. Eq. (10.4)). We will denote this case by $M[(t,c') >_{QPN} M'$.

To specify the transition probabilities of the QPN's embedded Markov chain[1] we now have to take also the service rates into consideration. For a marking $M = (n,m)$ define

$$Z(M) = \sum_{p \in P_1} \sum_{c \in C(p)} r\left(n(p)(c)\right)$$

which is the sum of all rates out of M due to services in all queueing places of P_1. Furthermore define

$$T(M) = \{t \in T | \exists c \in C(t) : (t,c) \in EN_T(M)\}$$

With that the transition probability between marking M_k and M_r are defined as $P[M_k \rightarrow M_r] =$

[1] Note that this technique of using the embedded Markov chain is only applicable if the transition probabilities out of a marking (state) are independent of the time spend in that state (semi-Markov process). This imposes a restriction on the allowed queues of the QPN.

$$\begin{cases} \dfrac{\sum\limits_{(t_i,c)\in EN_T(M):M_k[(t_i,c)>_{QPN}M_r} w_i(c)}{\sum\limits_{(t_j,c')\in EN_T(M_k)} w_j(c')} & \text{if } T(M)\subseteq T_2 \\[2em] \dfrac{\sum\limits_{(t_i,c)\in EN_T(M):M_k[(t_i,c)>_{QPN}M_r} w_i(c)}{\sum\limits_{(t_j,c')\in EN_T(M_k)} w_j(c') + Z(M_k)} & \text{if } T(M)\subseteq T_1 \\[2em] \dfrac{r(n(p)(c))}{\sum\limits_{(t_j,c')\in EN_T(M_k)} w_j(c') + Z(M_k)} & \begin{array}{l}\text{if } T(M)\cap T_2=\emptyset \\ \text{and} \\ M_k[(p,c)>_{QPN} M_r\end{array} \\[2em] 0 & \text{otherwise} \end{cases} \tag{10.5}$$

The first expression in Eq. (10.5) is the probability that an enabled immediate transition fires. The second and the third expressions are, respectively, the probabilities that a timed transition fires and that a service completion in a queueing place occurs. The probabilities in the latter cases (cf. page 131) are given by the rate causing the state transition divided by the sum of all rates leading out of that marking.

Having determined the probabilities $P[M_k \to M_r]$ we can proceed as in the GSPN case by defining the matrices C,D,E and F (cf. Eq. (9.2) on page 142) and solving the global balance equations of the reduced embedded Markov chain

$$\tilde{\pi} = \tilde{\pi} \times P'; \sum_{M_s\in\hat{T}} \tilde{\pi}_s = 1.$$

where P' is given by Eq. (9.6) and $\hat{T}$ is the set of tangible states (cf. page 142).

Again we see that a steady state distribution of the QPN exists iff there is no timeless trap and the reachability set of the QPN has home states.

Given the steady state distribution $\tilde{\pi}$ of the reduced embedded Markov chain, the mean number of visits to marking M_s between two consecutive visits of marking M_j is $v_{sj} = \frac{\tilde{\pi}_s}{\tilde{\pi}_j}$. Thus the mean cycle time $t_c(M_j)$, i.e. the mean time between two consecutive visits of M_j, is given by

$$t_c(M_j) = \frac{1}{\tilde{\pi}_j} \sum_{M_s\in\hat{T}} \tilde{\pi}_s \, (\sum_{(t_k,c)\in EN_T(M_s)} w_k(c) + Z(M_s))^{-1}$$

and since the mean sojourn time t_s in marking $M_j \in \hat{T}$ is

$$t_s(M_j) = (\sum_{(t_k,c)\in EN_T(M_j)} w_k(c) + Z(M_j))^{-1}$$

we obtain the steady state distribution π of the QPN by

$$\pi_j = \begin{cases} \frac{t_s(M_j)}{t_c(M_j)} & \text{if } M_j \in \hat{T} \\ 0 & \text{if } M_j \in \hat{V} \end{cases} \tag{10.6}$$

as in the case of the quantitative analysis of GSPNs.

Example 10.3 *Consider the QPN of Fig. 10.4 with*
$\mu(p_1)(a) = 3, \mu(p_1)(b) = 2, w_1(a) = 3, w_1(b) = 2, w_2(a) = w_2(b) = 1.$
Define $\delta_c(\tilde{c}) = \begin{cases} 1 & \textit{if } \tilde{c} = c \\ 0 & \textit{otherwise} \end{cases}$

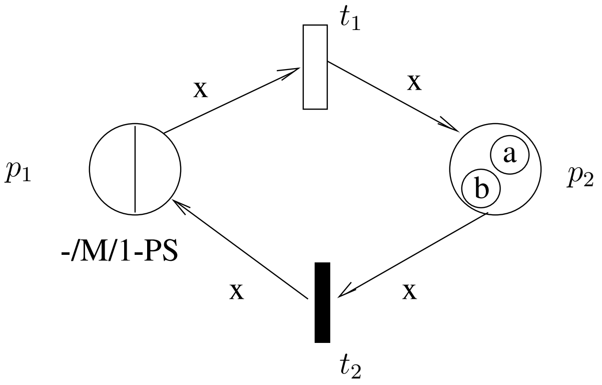


Figure 10.4 $C(p_1) = C(p_2) = C(t_1) = C(t_2) = \{a,b\}$

The reachability set comprises the following markings:

$$\begin{aligned} M_0 &= ((0),(0,\delta_a+\delta_b)), & M_1 &= ((\delta_a),(0,\delta_b)), & M_2 &= ((\delta_b),(0,\delta_a)), \\ M_3 &= ((0),(\delta_b,\delta_a)), & M_4 &= ((0),(\delta_a,\delta_b)), & M_5 &= ((\delta_a+\delta_b),(0,0)), \\ M_6 &= ((\delta_a),(\delta_b,0)), & M_7 &= ((\delta_b),(\delta_a,0)), & M_8 &= ((0),(\delta_a+\delta_b,0)) \end{aligned}$$

and the corresponding reachability graph is shown in Fig. 10.5. For a marking, e.g. $M_5 = ((\delta_a + \delta_b),(0,0))$ *the first component denotes* $n(p_1)$ *which is here the sum of two functions* δ_a *and* δ_b*. The second component of* M_5*, that is* $(0,0)$*, denotes the marking of the underlying CGSPN.*
The markings M_0,M_1,M_2,M_3,M_4 *are vanishing and the markings* M_5,M_6,M_7,M_8 *are tangible. The matrix* P *of the embedded Markov chain is*

$$P = \begin{pmatrix} C & D \\ E & F \end{pmatrix} = \left(\begin{array}{ccccc|cccc} 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & \frac{2}{5} & \frac{3}{5} & 0 \\ 0 & \frac{2}{5} & 0 & 0 & 0 & 0 & 0 & 0 & \frac{3}{5} \\ 0 & 0 & \frac{3}{5} & 0 & 0 & 0 & 0 & 0 & \frac{2}{5} \\ 0 & 0 & 0 & \frac{3}{5} & \frac{2}{5} & 0 & 0 & 0 & 0 \end{array}\right)$$

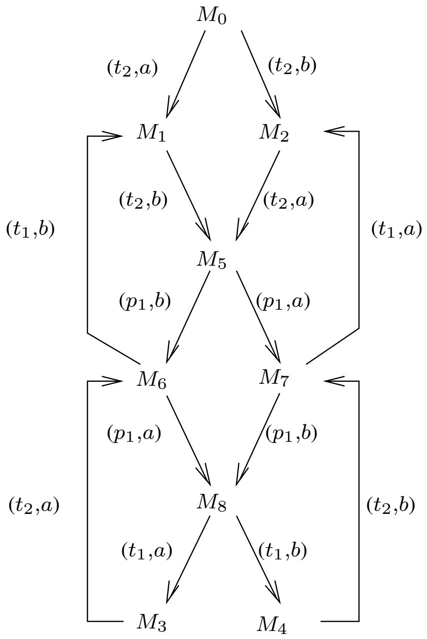


Figure 10.5 Reachability graph

Since matrix

$$(I-C)^{-1} = \begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

we get

$$P' = F + E \times (I-C)^{-1} \times D = \begin{pmatrix} 0 & \frac{2}{5} & \frac{3}{5} & 0 \\ \frac{2}{5} & 0 & 0 & \frac{3}{5} \\ \frac{3}{5} & 0 & 0 & \frac{2}{5} \\ 0 & \frac{3}{5} & \frac{2}{5} & 0 \end{pmatrix}$$

Solving

$$\tilde{\pi} P' \;=\; \tilde{\pi} \; \textit{and} \; \sum_{i=5}^{8} \tilde{\pi}_i = 1$$

yields $\tilde{\pi}_5 = \tilde{\pi}_6 = \tilde{\pi}_7 = \tilde{\pi}_8 = \frac{1}{4}$*. Thus* $v_{kj} = 1, \forall j,k \in \{5,6,7,8\}$*. The sojourn time for tangible markings is given by* $t_s(M_5) = (\frac{1}{2} \times 3 + \frac{1}{2} \times 2)^{-1} = \frac{2}{5}$ *and* $t_s(M_j) = (3+2)^{-1} = \frac{1}{5}, \forall j \in \{6,7,8\}$*. With that the cycle time is given by* $t_c(M_j) = 1, \forall j \in \{5,6,7,8\}$*, which yields the steady state distribution for tangible states:* $\pi_5 = \frac{2}{5}$ *and* $\pi_j = \frac{1}{5}, \forall j \in \{6,7,8\}$.

As we have seen, QPNs allow to describe queues in a convenient way without having to specify queues by pure Petri net elements. An appropriate state descriptor for a $-/M/1-FCFS$ queue serving several colours of tokens would be a string in $(C(p))^*$, where p is the queueing place. A state $va \in (C(p))^*, a \in C(p)$, would be changed to bva if a token of colour $b \in C(p)$ arrives or alternatively to v if the token of colour a completes its service.
However the complexity of the quantitative analysis, determined by the size and structure of the reachability set, is still the same as that obtained by modelling the queue with CGSPN elements.

10.2 Qualitative Analysis of QPNs

As we have seen, qualitative properties of a QPN are essential for a successful quantitative analysis of the net. Because QPNs are based on CPNs, we are again interested in employing the analysis procedure shown in Fig. 9.5. Since CGSPNs $\subseteq$ QPNs, all negative examples given in Sec. 9.2 hold for QPNs as well.
First let us investigate timeless traps. If we neglect the specific values of the rates and consider only the priority relation imposed on the firing of transitions, the enabling rule in QPNs is given by

Definition 10.2 *A transition $t \in T$ of a QPN is enabled at a marking M with respect to a colour $c \in C(t)$, denoted by $M[(t,c) >_{QPN}$, iff $(t,c) \in EN_T(M)$.*

According to this enabling rule all definitions for Place-Transition nets can be defined similarly for QPNs. A timeless trap is now defined as follows:

Definition 10.3 *A QPN has a timeless trap, iff $\exists M \in R(QPN,M_0)$:*
$\forall k \in \mathbb{N} : \exists \sigma \in T_C^* : |\sigma| \geq k$, such that $M[\sigma >_{QPN}$ and
$\forall \tilde{\sigma} \in T_C^* : M[\tilde{\sigma} >_{QPN} \Longrightarrow \tilde{\sigma} \in (T_2)_C^*$,
where $Y_C := \cup_{y \in Y} \cup_{c \in C(y)} (y,c)$ for $Y \subseteq P \cup T$.

As in GSPNs timeless traps cannot occur if condition NoTT (cf. Def. 9.5) holds for QPNs. For QPNs we are able to establish a generalised version of this condition: If an immediate transition has at least one queueing place as an input place, this transition cannot contribute to a timeless trap, since tokens fired onto a queueing place are only available to the output transitions after being served. To serve a token in a queueing place the QPN has to be in a tangible state. Since we are now dealing with Coloured Petri nets all colours of a transition have to be considered, because they characterise the different "firing modes" of a transition.

Definition 10.4 (Condition NoTT for QPNs)
Condition NoTT $:\Longleftrightarrow \forall \tilde{T} \subseteq (T_2)_C \setminus (P_1)_C\bullet : \quad \bullet\tilde{T} \neq \tilde{T}\bullet$

For the •-notation of these sets, see Def. 6.4 on page 118. Condition NoTT ensures the absence of timeless traps for QPNs with finite reachability sets and the proof is similar to the one given for Th. 9.2.

Theorem 10.1 *Let QPN have a finite reachability set.*
Condition NoTT (for QPNs) $\Longrightarrow$ *QPN has no timeless trap.*

Obviously Th. 9.3 also holds in this context and condition NoTT can be verified by a similar algorithm as given in Fig. 9.8. We start initially with $T_H := (T_2)_C \setminus (P_1)_C\bullet$ and use the •-notation according to Def. 6.4.
Assuming that tokens are not created in queues, which holds for all queues we have learned about so far, we get the following result directly:

Theorem 10.2 *CPN bounded* $\Longrightarrow$ *QPN bounded*

The converse is, as in the case of GSPNs, not true, which is shown by Fig. 9.9. For the derivation of further results, especially on liveness and existence of home states, we restrict ourselves to QPNs which have an EFC-net structure. Remember that we have established several useful results for this class of PNs in Sec. 5.4.3.

10.2.1 Qualitative Analysis of EFC-QPNs

The structure of a QPN is determined by a Coloured Petri net. Unfolding even a strongly connected CPN might yield several isolated Place-Transition nets. Fig. 10.6 shows a simple example of this. We know that a timed transition of a CGSPN "unfolds" into a set of timed transitions of the GSPN according to Def. 9.8. Each of the isolated Place-Transition nets has at least one transition, which is derived by unfolding a timed transition, or there is at least one place, which is derived by unfolding a queueing place. We will denote this property by $(*)$. With that we can exploit the results for EFC-GSPNs.
Taking the colours of the underlying CPN into consideration, condition EQUAL-Conflict can be defined for QPNs as well.

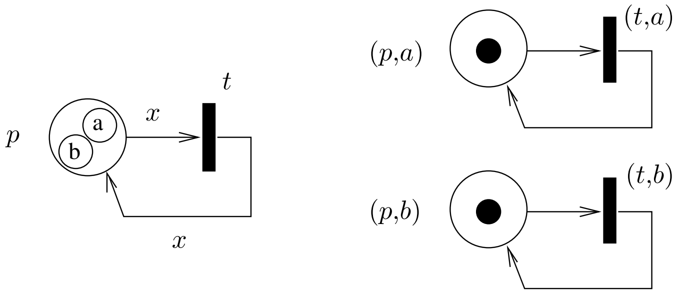


Figure 10.6 Unfolding a simple CPN

Definition 10.5 *A QPN satisfies condition EQUAL-Conflict, iff*
$\forall t,t' \in T : \bullet(\{t\}_C) \cap \bullet(\{t'\}_C) \neq \emptyset \Longrightarrow \{t,t'\} \subseteq T_1$ *or* $\{t,t'\} \subseteq T_2$.

If you think of the unfolded Petri net of the CPN underlying the QPN, this condition describes the same condition as that in Def 9.7 and it is again of particular importance for EFC-nets.

Definition 10.6 *A QPN is an EFC-QPN iff its underlying unfolded CPN consists of EFC-nets.*

Remember that unfolding a CPN might yield several isolated Place-Transition nets. For an EFC-QPN these Place-Transition nets must all be EFC-nets. According to property $(*)$ all these EFC-nets have at least one transition, derived by unfolding a timed transition, or at least one place derived by unfolding a queueing place. If we assume that the CPN is bounded and live (cf. Def. 6.6 on page 118) this means that all EFC-nets are strongly connected as implied by Th. 5.1. With that we easily see that condition EQUAL-Conflict implies the absence of timeless traps.

Theorem 10.3 *Given an EFC-QPN whose underlying CPN is live and bounded, then*
Condition EQUAL-Conflict $\Longrightarrow$ *QPN has no timeless traps.*

Proof. Although the proof is similar to the proof of Th. 9.5 we will go more into the details; just to show how the unfolding of a net can be used for this purpose.
Assume the QPN has a timeless trap, i.e. $\exists M \in R(QPN,M_0)$:
$\forall k \in \mathbb{N} : \exists \sigma \in T_C^* : |\sigma| \geq k$, such that $M[\sigma >_{QPN}$ and
$\forall \tilde{\sigma} \in T_C^* : M[\tilde{\sigma} >_{QPN} \Longrightarrow \tilde{\sigma} \in (T_2)_C^*$.
Since T is a finite set, there is at least one element $(\hat{t},\hat{c})$ whose number of occurrences in σ is not bounded.
Since the CPN is bounded, Th. 10.2 ensures boundedness of the QPN implying the finiteness of $R(QPN,M_0)$. Thus there exists a terminal strongly connected subset $C_M \subseteq R(QPN,M)$. Define
$\tilde{T} := \{(t,c) | \exists M' \in C_M : M'[(t,c) >_{QPN}\}$. The assumption implies $\tilde{T} \subseteq (T_2)_C$ and since CPN is live and bounded, the corresponding unfolded net consists of strongly connected Place-Transition nets. Let us consider the corresponding Place-Transition net which comprises $(\hat{t},\hat{c})$. Because of $(*)$ there is at least one transition in the net, which does not contribute to the timeless trap and thus is not a member of $\tilde{T}$. Define $\tilde{P} := \bullet\tilde{T}$, then one of the following two cases holds with respect to the Place-Transition net being considered:

1. $\exists (p_r,c_r) \in \tilde{P}$:
   $(p_r,c_r)\bullet \supseteq \{(t_i,c_i),(t_j,c_j)\}$ and $(t_i,c_i) \in \tilde{T},(t_j,c_j) \in T_C \setminus \tilde{T}$.
   The assumption of a timeless trap yields $t_i \in T_2$. Thus if $M'[(t_i,c_i) >_{QPN}$ for some marking $M' \in C_M$, the EFC-net structure and condition EQUAL-Conflict imply $M'[(t_j,c_j) >_{QPN}$ contradicting $(t_j,c_j) \notin \tilde{T}$.

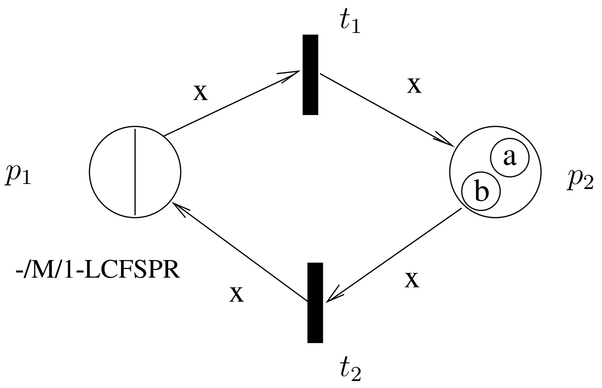


Figure 10.7 Non live QPN

2. $\exists(t_r,c_r) \in \tilde{T}$:
   $(t_r,c_r)\bullet \supseteq \{(p_k,c_k),(p_l,c_l)\}$ and $(p_k,c_k) \in \tilde{P},(p_l,c_l) \in P_C \setminus \tilde{P}$.
   Since $(t_r,c_r) \in \tilde{T},t_r$ fires infinitely often with respect to $c_r \in C(t_r)$. Since $(p_l,c_l) \notin \tilde{P} = \bullet\tilde{T}$, the place p_l is not bounded with respect to colour $c_l \in C(p_l)$, contradicting the boundedness of the QPN. □

Considering the theorems we have established for EFC-GSPNs one would also expect that condition EQUAL-Conflict is sufficient for liveness in the QPN case. Unfortunately this is not the case, since it might now be possible that the scheduling strategy prevents tokens from being served. Fig. 10.7 shows an example of this. A token being served at the queueing place p_1 immediately returns to this place and preempts the token in service. Thus the token of colour a or the token of colour b will never complete its service. If at least one token of each colour of the queueing place has the opportunity of being served in every state of the queue, such situations cannot occur. This gives rise to the following definition.

Definition 10.7 *A QPN satisfies condition EQUAL-Service iff* $\forall p \in P_1$*: the service time distribution of the queue in* p *is Coxian and if* $|C(p)| > 1$ *then the scheduling strategy is PS or IS.*

If $|C(p)| = 1$ we can choose an arbitrary scheduling strategy. We only have to demand that the server stays never idle while there are customers present, which is one of the main features of so-called *work conserving* scheduling strategies [102]. Note that all scheduling strategies we have looked at in Ch. 3 satisfy this requirement.

Condition EQUAL-Service states that the queue can be analysed by means of Markovian techniques, since the service time is exponentially distributed in each stage of service of a token and that for queues serving several colours of tokens all tokens are served simultaneously.[2] With that we can exploit Th. 9.5 to establish the following:

[2] [18] gives a general definition of condition EQUAL-Service also comprising further scheduling strategies.

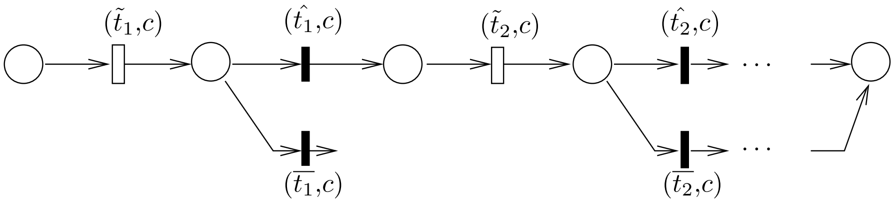


Figure 10.8 GSPN representation of a Coxian distributions

Theorem 10.4 *Given an EFC-QPN satisfying Condition EQUAL-Service and whose underlying CPN is live and bounded, then*

1. *Condition EQUAL-Conflict* $\Longleftrightarrow$ *QPN is live*
2. *Condition EQUAL-Conflict* $\Longrightarrow$ *QPN has home state.*

Proof. Since for each queueing place p the service distribution is Coxian and in case of $|C(p)| > 1$ the scheduling strategy is PS or IS, each queue can be unfolded and the service of each token colour can be represented by the GSPN subnet shown in Fig. 10.8. $(\tilde{t_i},c)$ represents the exponential service time distribution for a token in stage i of its service, $(\hat{t_i},c)$ represents entering stage $(i+1)$ of service and $(\overline{t_i},c)$ the completion of service. Note that the rates and the firing weights of the transitions $(\tilde{t_i},c)$,$(\hat{t_i},c)$,$(\overline{t_i},c)$ are now possibly marking dependent. The resultant GSPN satisfies condition EQUAL-Conflict and the statement of this theorem follows from Th. 9.5. □

If condition EQUAL-Service does not hold, we have already seen that liveness does not extend from the CPN to the QPN, even for simple net structures. Fig. 10.7 also shows that home states might not exist. But even if the QPN is live, condition EQUAL-Service is essential for preserving the existence of home states. Fig 10.9 shows a further example of a QPN with no home states. The queues in the queueing places p_2 and p_5 serve arriving tokens according to the FCFS scheduling strategy and the service times are exponentially distributed. If transition t_1 (t_3) is enabled in the QPN, the token on the input place p_1 (p_4) is "transferred" to place p_2 (p_5) and a token of the complementary colour is created on place p_3 (p_6). Transitions t_2 and t_4 "merge" tokens of the same colour. Serving tokens according to the FCFS rule yields two strongly connected subsets in the reachability set of the QPN, such that home states do not exist. These subsets are characterised by the sequence of tokens in both queues: either "a" ahead of "b" or "b" ahead of "a".

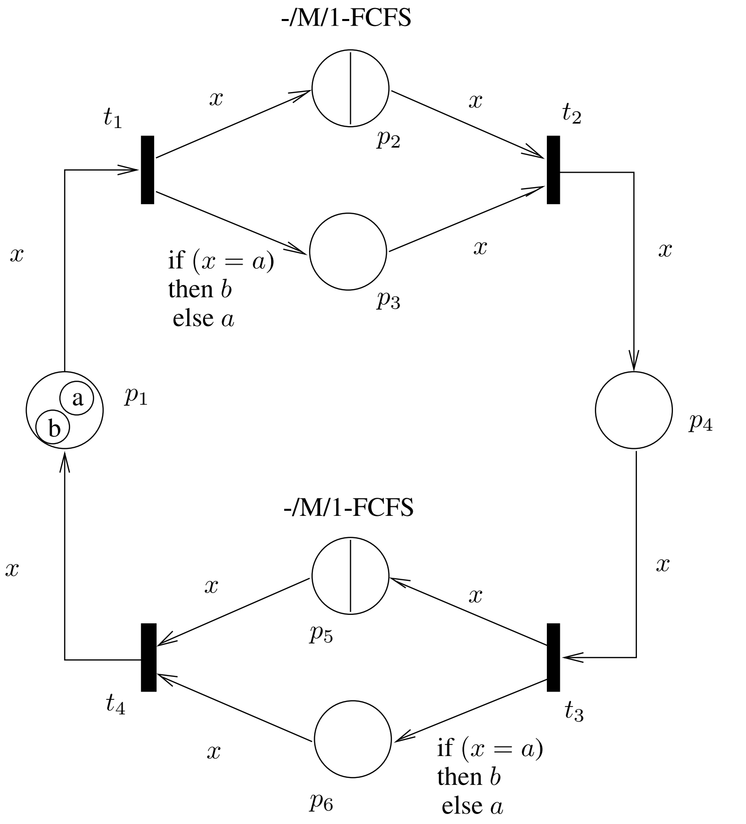


Figure 10.9 QPN with no home states

10.3 Some Remarks on Quantitative Analysis

As we have seen, introducing time into Place-Transition nets might change the behaviour and thus the properties of the net significantly. Most of the results presented in Sec. 9.2 and 10.2 only hold for nets with a restricted structure. Most nets in practice unveil a more complex net structure and thus only some properties, like boundedness or P-invariants of the Place-Transition net, extend to the time-augmented net. Nevertheless, a qualitative analysis of the Place-Transition net is often useful to gain insight into the behaviour of the net. Since it is often desirable that the "functioning" of the net does not depend on specific timing constraints, like e.g. in mutual exclusion problems, such an analysis should be performed first. Unfortunately, determining essential properties for a quantitative analysis, must often only be done by inspecting the reachability set of the time-augmented net.

Even if the time-augmented Petri net satisfies all positive qualitative properties for a quantitative analysis, determining a solution of the global balance equations becomes intractable for large nets. Hence several authors (cf. [13, 25, 26, 43, 65, 89, 111, 179]) have investigated time-augmented Petri nets with product-

form solutions, similar to product-form queueing networks [102] and developed algorithms based on the product-form property [164, 165].
Another approach is to structure the model hierarchically, e.g. by place refinement, and to exploit the hierarchical structure also for the quantitative analysis such that models with millions of states can be analysed, see e.g. [27]. The general idea is similar to the one used for the analysis of superposed GSPNs [63], where the net is composed of several components interacting via a set of synchronised transitions ST. Considering these components in isolation gives reachability sets RS^i whose individual size is usually much smaller than the size of the reachability set of the whole net. The infinitesimal generator Q of the whole reachability set RS can then be expressed by a Kronecker based representation of the following form (cf. [105])

$$Q = \bigoplus Q^i_l + \sum_{t \in ST} \bigotimes Q^i_t$$

where the Q^i_* are $(|RS^i| \times |RS^i|)$-matrices.
The advantage of this representation is a shift from space to time concerning the complexity of iterative solution techniques for solving the global balance equations. In many cases the sizes of the reachability sets of the components are much smaller than the corresponding state-transition matrix Q of the whole GSPN. Thus the information on the state space structure is stored in a set of smaller matrices and an element of Q can be determined by some additional calculations (cf. [52]). This approach has its greatest benefits when all matrices Q^i_* fit into main memory whereas Q would not. Then the additional operations specified by the Kronecker operators can be neglected in contrast to time-consuming swapping operations of the computing equipment. A similar Kronecker based representation can be defined for nets with specific hierarchical structures, thus profiting from the same effect.

Exercise 10.1 *Calculate the steady state distribution of the central server model of Fig. 10.3. Assume that all rates and firing weights are* 2.0.

Exercise 10.2 *Determine the reachability sets of the QPN of Fig. 10.9 and its underlying CPN.*

11 Further Reading

Introductory papers on Stochastic Petri nets are, for example, those by Ajmone-Marsan [2] and Pagnoni [129].
In this introduction we have only discussed some of the most important time-augmented Petri net models, which can be analysed using Markovian analysis techniques. There are several other models integrating timing aspects into Petri nets. We will briefly describe some of them using the classification of time-augmented Petri nets given on page 129:

Timed Places Petri Nets:

Timed Petri Nets: In [166] J. Sifakis attaches real values $z_i \in \mathbb{R}^+$ to each place p_i of the Petri net. The constant z_i determines the dwelling time of a token on that place. After this time has expired the token becomes available to all output transitions of p_i. Enabled transitions have to fire immediately. In [167] it is shown that this model is equivalent to the TTPN model of C. Ramchandani. In [166, 167] a relationship between the initial marking, the dwelling times z_i and the firing frequencies of transitions is established.

A modified version of the TPPN model by Sifakis is investigated in [55].

Stochastic Petri Nets: C.Y. Wong, T.S. Dillon and K.E. Forward [181] define a stochastic version of the TPPN model by Sifakis, where the dwelling times of tokens are now characterised by exponential distributions such that Markovian analysis techniques are applicable.

Timed Transitions Petri Nets:

Preselection models:

Timed Petri Nets:

- One of the first models considering Petri nets and "time" was developed by C. Ramchandani [144]. The constant firing time, i.e. the time delay between enabling and firing, of a transition $t_i \in T$ is characterised by a real variable $z_i \in \mathbb{R}^+$. If transitions are in conflict the one to reserve tokens first is chosen randomly. It might be possible that after a transition has reserved tokens it becomes enabled again. Thus a transition might be enabled simultaneously to itself. In [143] marked graph structures of this TTPN model are investigated and, similar to [144], the minimum cycle time of transitions, i.e. the minimum time between two consecutive firings of a transition, is calculated (see also [125]).

- R.R Razouk and C.V. Phelps [147] define a model similar to that of Merlin (see Merlin's TTPNs). The behaviour of a transition is determined by a pair of real values $(z_{min}, z_{max}), z_{min} \in \mathbb{R}^+$,
$z_{max} \in \mathbb{R}^+, z_{min} \leq z_{max}$; if a transition is enabled at time τ it must not reserve tokens before time $\tau + z_{min}$. After reservation it fires exactly at time $\tau + z_{max}$.

Stochastic Petri Nets: W.M. Zuberek [185, 186] associates a rate, specifying an exponentially distributed firing time, and a probability $c(t)$ with each transition. In case of a conflict $c(t)$ determines the probability with which transition t reserves tokens on the input places. Zuberek also extends the underlying Petri net by introducing inhibitor arcs.

Race models:

Timed Petri Nets: P. Merlin [121] attaches to each transition a pair of real values $(z_{min}, z_{max}), z_{min} \in \mathbb{R}^+$,
$z_{max} \in \mathbb{R}^+, z_{min} \leq z_{max}$ and defines the behaviour as follows: If a transition is enabled at time τ then it must not fire before time $\tau + z_{min}$. Afterwards the transition may fire at any point in time in the time interval $[\tau + z_{min}, \tau + z_{max}]$ provided it is still enabled. At latest at time $\tau + z_{max}$ the transition has to fire. Merlin shows that his model is adequate for modelling time-outs of communication protocols, which is also investigated in [61]. In [33] the analysis of this TTPN model is considered.

Stochastic Petri Nets:

- M. Ajmone-Marsan and G. Chiola [8, 9] extend the GSPN model by introducing transitions with a deterministic firing time (Deterministic Stochastic Petri nets; DSPNs). If at all markings amongst timed transitions (i.e. transitions with an exponentially distributed firing time) only one deterministic transition is enabled, the Petri net can be mapped onto a Semi-Markov process. The analysis of DSPNs is also considered in [10, 81, 84, 113]. The analysis of DSPNs and also more general Non-Markovian Stochastic Petri Nets is considered in [82].
- J.B. Dugan [66] defines Extended Stochastic Petri nets (ESPNs) by allowing firing times of transitions to be specified by an arbitrary continuous time distribution. As in SPNs each transition is a timed transition. Amongst inhibitor arcs the underlying Petri net is extended by further arcs: counter-alternate and probabilistic arcs.

Since 1985 an international workshop on Timed and Stochastic Petri nets is held biennial. The title of the first workshop in 1985 was “Timed Petri Nets” [134], which afterwards changed to “Petri Nets and Performance Models” [135, 136, 137, 138, 139, 140, 141, 142]. Some publications on Timed and Stochastic Petri Nets can also be found in the annual international conference “Application and Theory of Petri Nets” and in the “Advances in Petri Nets”.
Further information on QPNs can be accessed via WWW using URL http://ls4-www.informatik.uni-dortmund.de/QPN/
Tool support is available for several Stochastic Petri net formalisms (e.g. GSPNs [50, 51], DSPNs [83, 85, 184, 183, 112, 114, 113], QPNs [27, 30]), see also [73, 74]. A list on Petri net tools can be accessed via WWW using URL http://www.informatik.uni-hamburg.de/TGI/PetriNets/tools.
A proposal has been published defining a common interface description also for Stochastic Petri nets [28, 47], thus exhibiting the possibility to exchange model descriptions between different sets of tools.

12 Application Examples

In this chapter we present two examples of modelling systems using Stochastic Petri nets. We will employ the QPN formalism, since it comprises SPNs and GSPNs, and also offers convenient description facilities for queueing situations. In Sect. 12.1 we address the well-known problem of sharing resources by different processes. In Sect. 12.2 a node of a DQDB network is modelled by a QPN subnet. In this section we will use the more general definition of QPNs including so-called immediate queueing places as given in [18].

12.1 Resource Sharing

This example considers the general problem of modeling sharing of processors/machines by different processes.

Imagine that each process is given by a process schemata specified by an ordinary Place-Transition net. Since scheduling in QPN models is described by queueing in places, obviously activities sharing resources should be attached to the places of the Place-Transition net. Assume, e.g., the following description of process schemata.

A user specifies a series of n simulation runs, which might be executed in parallel (see Fig. 12.1). After termination of all simulation runs a special program is invoked, collecting the simulation results and preparing them for printing. Finally the results are printed. The second process schemata specifies a user who prepares a LaTeX-document for printing by starting two sequential calls of this typesetting program (just to 'get cross-references right'). For both process schemata we have to specify additionally the service time distribution of each task, e.g. the parameters of a Coxian distribution and the weights of immediate transitions. For simplicity we will neglect these parameters here.

After definition of the load, we have to determine the machines executing these tasks. In this example we consider a small network comprising three workstations and one printer. Each workstation works off all tasks in a processor sharing manner and the printer services jobs in order of their arrival.

Mapping of these processes (tasks) to processors (machines) is defined statically by the users. The user initiating the simulation runs, e.g., decides to dedicate n_i runs to workstation i ($n = n_1 + n_2 + n_3$) and starting the collector task at workstation 3. The user of LaTeX on the other hand independently decides to use workstation 3 for all his tasks. Attaching distinct colours to the places and transitions of both process schemata now yields the QPN model given in Fig. 12.2. Activities represented by places of the P/T-net are 'mapped' to the

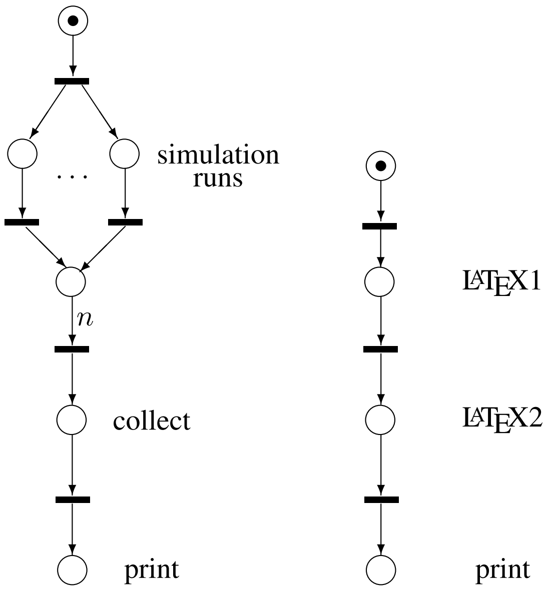


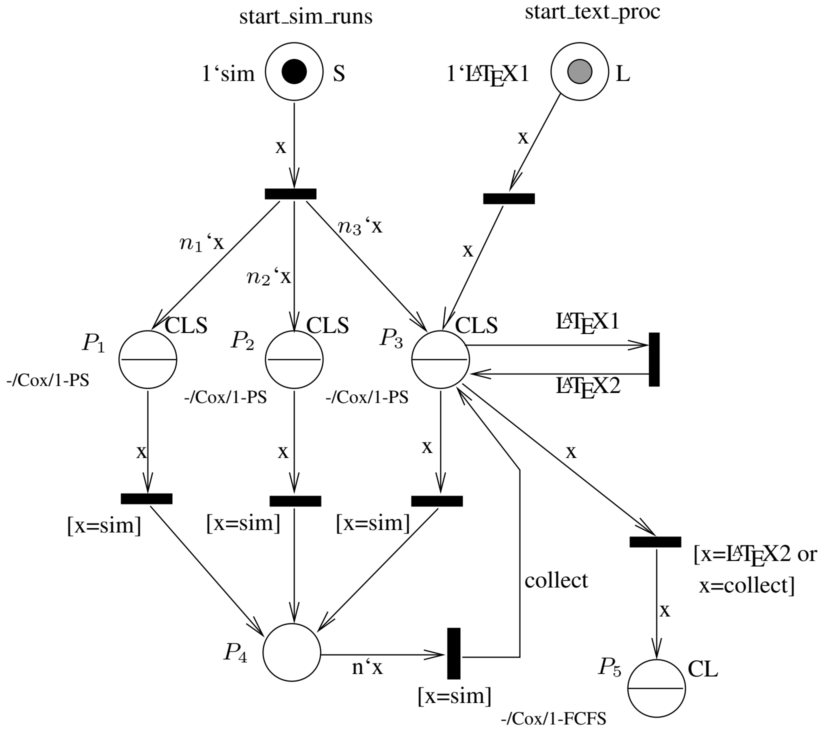
Figure 12.1 Two process schemata

corresponding timed queueing places of the QPN. Places $P_i, i = 1,2,3$, model the three workstations and place P_5 the printer. The most interesting queueing place is $P3$, where n_3 simulation runs and afterwards a collector job compete with a LaTeX job (LaTeX1 or LaTeX2) for service at the CPU. Note that this QPN models an absorbing Markov chain (cf. Sect. 2.1.2) and the mean time until absorption reflects the mean time until both results have been printed.

12.2 Node of a DQDB network

The Distributed Queue Dual Bus (DQDB) protocol has been proposed as the subnetwork for the IEEE 802.6 MAN for the interconnection of Local Area Networks, servers, workstations and other devices. The basis of a DQDB network is a pair of unidirectional buses as illustrated in Fig. 12.3. Nodes are connected to both buses by a logical OR-writing tap and a reading tap. The *Head Station* (frame generator) continuously generates frames. Each frame is subdivided into slots of equal size and each slot in turn has several fields. The *End Station* (slave frame generator) terminates the forward bus and removes all incoming frames and generates slots at the same transmission rate and of the same sort on the other bus.

Data to be transmitted, is split up into segments of fixed size fitting on a slot's information field. Each slot can be allocated either to isochronous traffic (called



colour C = with collect;
colour L = with LaTeX1 | LaTeX2;
colour S = with sim;
colour CS = with C | S;
colour CL = with C | L;
colour LS = with L | S;
colour CLS = with C | L | S;

Figure 12.2 QPN model of two processes sharing resources

pre-arbitrated slots or PA-slots) or non-isochronous traffic (called *queued arbitrated* slots or QA-slots). Access to QA-slots is controlled by the following protocol, which operates on both buses symmetrically, so it is sufficient to consider only one direction.

In the following we assume data segments to be sent in the direction of the D-bus and requests in the direction of the R-bus (cf. Fig. 12.3). Segments arriving at a node are placed into a so called 'local queue' which is served in FCFS manner. The first segments (only) at all non-empty local queues for one bus constitute a distributed queue whose service discipline is FCFS. This distributed queue does not represent any existing physical queue, but is only established virtually by means of the following mechanism. Each local queue can be in two states: idle

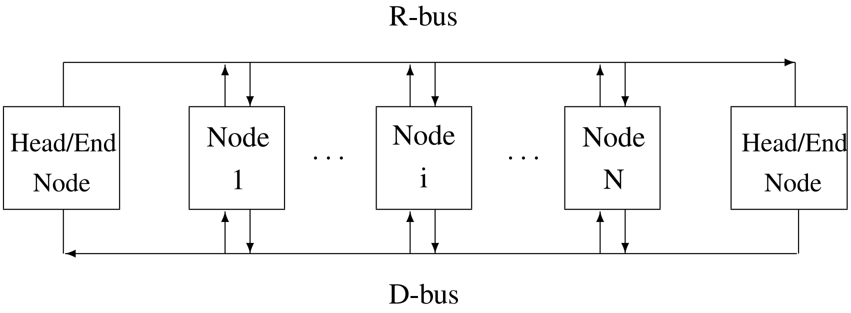


Figure 12.3 DQDB architecture

or countdown. If no segment is waiting for transmission the corresponding local queue is in the idle state. If a segment arrives at a node for transmission on the D-bus, an empty local queue changes to its countdown state. For each segment to be send, a request is transmitted on the opposite bus, thus informing upstream nodes (with regard to the direction of the D-bus) to let through an empty slot on the D-bus for its own waiting segment. A request can only be transmitted if the corresponding bit in the slot on the R-bus is not set, otherwise this indicates the request of a downstream node. Each node keeps track of its own pending requests by incrementing a request queue counter (RQC). When a local queue is idle, it keeps count of the number of outstanding requests from downstream nodes via a so-called request counter. For each empty slot passing on the D-bus this counter is decremented, accounting in this way for a served segment of a downstream node on the D-bus. If a local queue changes to its countdown state due to the arrival of a segment, the current value of its request counter is copied to a so-called countdown counter. This special counter is decremented whenever an empty slot passes on the D-bus until it reaches 0. Following this instant, the node transmits its own segment in the first empty slot on the D-bus.

For this example we employ the more general definition of QPNs [18]. Pure scheduling aspects can be described by immediate queueing places. In contrast to timed queueing places, tokens on those places can be viewed as being “served” immediately. The service in immediate queueing places has priority over the scheduling/service in timed queueing places and firing of timed transitions, similar to the priority of the firing of immediate over timed transitions.

A simple example of an immediate queueing place is given in Fig. 12.4. Here the definition of the scheduling strategy is as follows: serve tokens immediately when present according to FCFS, but only if the depository is empty. If there is a token at the depository all tokens are blocked in their current position.[1] Since the depository is emptied by transition t in Fig. 12.4, this transition determines the “real” service rate. We will now use this type of immediate queueing place, which will be called “FCFS with blocking” for the modelling of a node in a

[1] Similar to immediate transition also probabilities have to be specified for the “service” in immediate queueing places. For the special strategy chosen here the specific values of these probabilities do not matter. A formal definition of this strategy is given in [18].

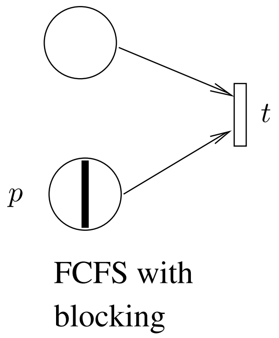


Figure 12.4 An immediate queueing place

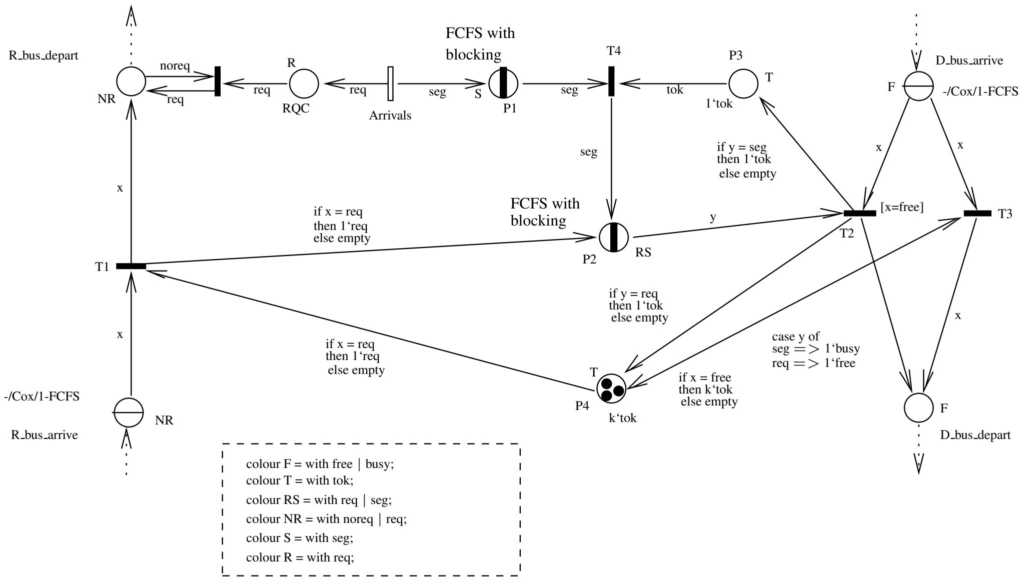


Figure 12.5 QPN model of a DQDB node

DQDB network.

Fig. 12.5 presents a QPN model for an arbitrary node of a DQDB network. The definition of the colour sets and the initial marking is located near the corresponding places. E.g. place $P4$ has colour set T and will initially contain k tokens of type tok. The guard of a transition is given in brackets. E.g. transition $T2$ is enabled if the free variable x is bound to *free* (*[x = free]*).

The R- and D-bus are represented by immediate transitions $T1$ and $T2$,$T3$ resp. Since DQDB mainly deals with two queues: the local queue at each node and the distributed queue consisting of all first segments at all non-empty local queues, these queues are modelled explicitly. Part of the distributed queue is modeled by the immediate queueing place $P2$. The state of this queue reveals the information that node has about the actual distributed queue due to arriving requests on the

R-bus. Note that e.g. 'Node 1' does not have any information on the state of the distributed queue, because it does not know of segments to be transmitted at other nodes. The basic idea of the QPN model is to represent the value of the request counter by inserting each request into the FCFS queue at place $P2$. This queue might also contain one segment (the first of the local queue) at most, which is ensured by place $P3$. All other segments have to wait in the immediate queueing place $P1$, whose queue also obeys an FCFS scheduling strategy with blocking. Place $P4$ is complementary to $P2$ thus ensuring that the number of tokens in $P2$ will never exceed k = 'maximum value of request counter' + 1 (first segment of the local queue). Service of the queue is performed by firing of immediate transition $T2$, if an empty slot on the D-bus arrives at this node. In case of serving a segment at $P2$ the formerly empty slot is marked busy. If $P2$ is empty (implying $P4$ contains k tokens of type *tok*) or a busy slot (token of type *busy*) arrives, $T3$ fires immediately.

Several models of such nodes can be interconnected by immediate transitions to constitute the R- and D-bus yielding a model of the whole DQDB network.

13 Solutions to Selected Exercises

Solution to Ex. 1.5

$$kp(1-p)^{k-1}$$

Solution to Ex. 1.1

- Probability of the same birthday for at least two persons is

$$p = 1 - \frac{365!}{365^n(365-n)!}$$

- $p \leq 0.5$ for $n \leq 22$

Solution to Ex. 1.2 In this case there are two mutually exclusive hypotheses:

G_1 “The student knows the correct answer”, and

G_2 “The student is guessing”

with probabilities $P\{G_1\} = p$ and $P\{G_2\} = 1-p$. We observed, event E, that the student had the answer wrong. The conditional probabilities of E, given the hypotheses, are

$$P\{E|G_1\} = 0 \quad P\{E|G_2\} = 1 - \frac{1}{m}$$

According to the Theorem of Total Probability,

$$P\{E\} = (1 - \frac{1}{m})(1-p).$$

Using Bayes Theorem we get the answer

$$\begin{aligned} P\{G_2|E\} &= \frac{P\{E|G_2\}P\{G_2\}}{P\{E|G_1\}P\{G_1\} + P\{E|G_2\}P\{G_2\}} \\ &= 1 \end{aligned}$$

which is to be expected.

Solution to Ex. 2.1

- The stochastic process changes state every day and since we know that two sunny days never follow one another the transition probabilities are not strictly state independent. Assuming that the process is Markovian is thus an approximation only.

- If we let S,R and D represent a sunny, rainy and dull day respectively, then we can define a new stochastic process by considering the states SS,SR,SD etc. which would represent the weather on two consecutive days. We need to determine new transition probabilities as well. A Markovian process will approximate this new process more closely than in the previous case.

Solution to Ex. 2.5 Draw the process as a discrete time Markov chain with 4 states of which state 0 is one absorbing state (the other is clearly C). Then

$$Q = \begin{pmatrix} 0 & p \\ q & 0 \end{pmatrix}$$

and

$$R = \begin{pmatrix} q & 0 \\ 0 & p \end{pmatrix}$$

From these we can compute

$$N = \begin{pmatrix} \frac{1}{1-pq} & \frac{p}{1-pq} \\ \frac{q}{1-pq} & \frac{1}{1-pq} \end{pmatrix}$$

and the steady state probability matrix NR. From the latter the probability of starting in state 1 and ending in state 0 is given by

$$n_{10} = \frac{q}{1-pq}.$$

The expected duration of the game is given from Th. 2.5 by

$$\frac{1+p}{1-pq}$$

Solution to Ex. 2.8 A car spends 64 percent of its time in *Town 2*.

Solution to Ex. 2.9 There are 1.62 computers broken on the average and they break down at a mean rate of 2.76 ($= 2 \times \frac{8}{21} + 4 \times \frac{6}{21} + 6 \times \frac{3}{21}$) per day.

Solution to Ex. 3.3 For random variables $X_1, \ldots, X_n$ we have

$$E[\sum_i X_i] = \sum_i E[X_i]$$

so that the mean arrival rate to the central computer is $\frac{10}{15}$. Since the average number in the central computer is 5, the average time in the central computer is thus 7.5 seconds.

Solution to Ex. 3.4

$$\begin{aligned}\pi_k &= (k+1)\rho^k \pi_0 \\ &= (k+1)\rho^k (1-\rho)^2 \\ N &= \frac{2\rho}{1-\rho}\end{aligned}$$

Solution to Ex. 3.11 Introducing a *speed vector* c_i, which describes the speed of the server when i customers are present, allows one to write the solution to both parts (a) and (b) using one formula:

$$P[(n_1,\ldots,n_K)] = P[(0,\ldots,0)]\frac{n!}{\prod_{i=1}^{n} c_i}\prod_{i=1}^{K}(\frac{\rho_i^{n_i}}{n_i!}) \tag{13.1}$$

where

$$\begin{aligned}\rho_i &= \frac{\lambda_i}{\mu_i} \\ \rho &= \sum_{i=1}^{K}\rho_i \qquad (< 1, \text{ergodicity condition}) \\ n &= \sum_{i=1}^{K} n_i \\ P[(0,\ldots,0)] &= (\sum_{j=0}^{\infty}\frac{\rho^j}{\prod_{i=1}^{j} c_i})^{-1}\end{aligned}$$

For PS we have $c_i = 1,\forall i$, giving:

$$P[(n_1,\ldots,n_K)] = (1-\rho)n!\prod_{i=1}^{K}\frac{\rho_i^{n_i}}{n_i!}$$

and for the IS case $c_i = i,\forall i$, we have

$$P[(n_1,\ldots,n_K)] = e^{-\rho}\prod_{i=1}^{K}\frac{\rho_i^{n_i}}{n_i!}$$

The mean number of customers in the latter case is

$$N = \sum_{n_1=0}^{\infty}\ldots\sum_{n_K=0}^{\infty}(\sum_{j=1}^{K} n_j)P[(n_1,\ldots,n_K)] = \rho$$

Eq. 13.1 is also valid for an M/Cox/m queue by setting $c_i = \min(i,m)$.

The reader should note that the steady state probability distribution of each of these queues, is independent of the particular Coxian distribution of the corresponding service time and only depends on the mean value.

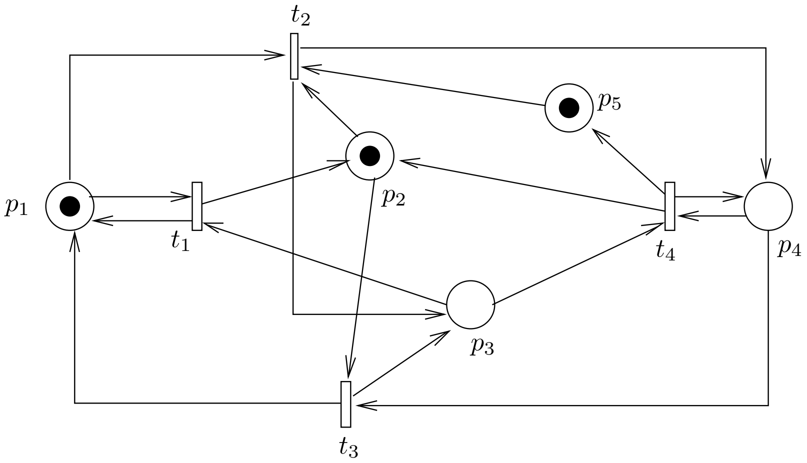


Figure 13.1 Counterexample for "Boundedness ⇒ Pos. P-Invariant covering"

Solution to Ex. 5.11 The coverability tree is not sufficient to determine liveness. Fig. 5.16 displays two Petri nets which have the same coverability tree. The Petri net displayed on the left side is not live (e.g. $M_0[t_1t_2 > M'$ yields a marking M' in which no transition is enabled) whereas the Petri net on the right side is live.

Solution to Ex. 5.14 The safe Place-Transition net of Fig. 13.1 is not covered by positive P-invariant, since there is no positive P-invariant covering place p_5.

Solution to Ex. 5.17

1. This statement follows directly from theorem 5.16, since a trap is marked in every marking M covering the given marking M_0. This property of an EFC-net is also known as the liveness monotonicity property of EFC-nets.
2. Fig. 9.12 (cf. [150]) shows a counterexample. If the token at the place in $\bullet t_2$ is removed the Petri net is live. With the displayed initial marking, which covers the former marking, the net is not live.

Solution to Ex. 5.19

1. Fig. 13.2 (cf. [39]) shows a counterexample.
2. Since the given proof of theorem 5.19 does not use the fact that the Petri net is an EFC-net, this statement is valid as well. The EFC property is needed for the converse implication.

Solution to Ex. 5.21 The first and the third algorithm are correct implementations. The fourth algorithm is an incorrect solution for the famous dining

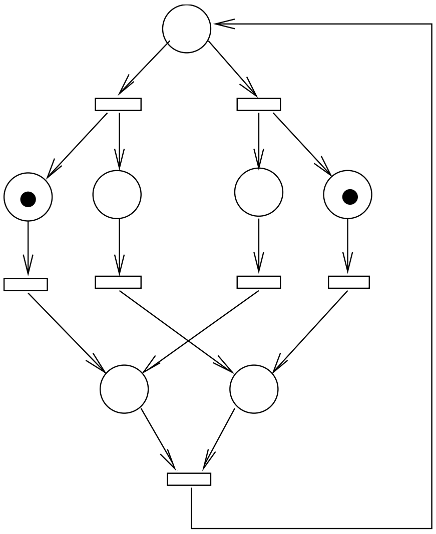


Figure 13.2 EFC-net whose initial marking is not a home state

philosopher problem. The system might reach a deadlock state, e.g. if all involved processes perform their "P(sem[i])"-operation simultaneously. The second algorithm is also incorrect, since both processes might enter the critical region simultaneously. One can see that as follows:

Statements of process P_0	Statements of process P_1
	flag[1] := true;
	turn = 0 $\Longrightarrow$ enter while loop
	flag[0] = false $\Longrightarrow$ skip inner while loop
flag[0] := true;	
turn = 0 $\Longrightarrow$ **enter critical region**	
	turn := 1;
	turn = 1 $\Longrightarrow$ skip while loop
	enter critical region

Solution to Ex. 9.6 All conjectures do not hold, since we have the following counterexamples:

1. see Fig. 9.10,
2. see Fig. 9.12,
3. see Fig. 9.14.

Solution to Ex. 9.8 This simple unbounded GSPN is quite interesting, since the corresponding Markov process has no steady state distribution irrespective of the values of λ and μ.

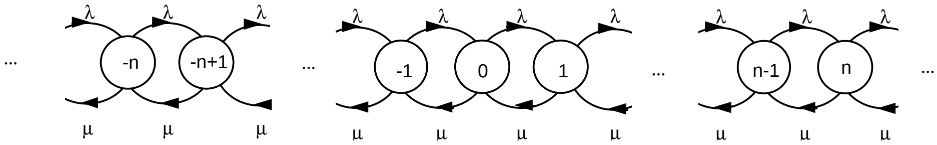


Figure 13.3 Markov chain for unbounded GSPN

The most important step proving this is in finding an appropriate state representation for the infinite Markov chain. Since transition t_3 is immediate, we have that whenever $M(p_1) > 0$, $M(p_2) = 0$ holds and similarly $M(p_2) > 0$ implies $M(p_1) = 0$. Thus $M(p_1) - M(p_2)$ is a sufficient state descriptor and the potentially two dimensional Markov chain of the GSPN can be represented in one dimension as shown in Fig. 13.3.

The global balance equations are given by

$$\pi_i(\lambda + \mu) = \lambda\pi_{i-1} + \mu\pi_{i+1} \qquad i = -\infty,\ldots,\infty \tag{13.2}$$

which rewrites as

$$\lambda(\pi_i - \pi_{i-1}) = \mu(\pi_{i+1} - \pi_i) \qquad i = -\infty,\ldots,\infty \tag{13.3}$$

If system (13.3) could be solved subject to $0 \le \pi_i \le 1, \forall i$ and $\sum_{i=-\infty}^{\infty} \pi_i = 1$, then a stationary distribution would in fact exist and would be given by this solution. However, it is not difficult to verify that such a solution cannot exist.

Define $\tau_i = \pi_{i+1} - \pi_i$. With that Eq. (13.3) rewrites to

$$\tau_i = \frac{\lambda}{\mu}\tau_{i-1}$$

giving

$$\tau_i = (\frac{\lambda}{\mu})^i \tau_0$$

implying

$$\pi_{i+1} - \pi_i = (\frac{\lambda}{\mu})^i(\pi_1 - \pi_0) \qquad i = -\infty,\ldots,\infty$$

We can now distinguish the following cases:

1) $\pi_1 = \pi_0$ implying $\pi_{i+1} = \pi_i, \forall i$ showing that $\sum_{i=-\infty}^{\infty} \pi_i = 1$ can not be satisfied.

2) $\pi_1 \ne \pi_0$. Consider

$$\pi_{n+1} - \pi_{-n} = \sum_{i=-n}^{n} (\pi_{i+1} - \pi_i) = \sum_{i=-n}^{n} (\frac{\lambda}{\mu})^i(\pi_1 - \pi_0)$$

If $\pi_1 > \pi_0$ we can conclude $\pi_{n+1} > n(\pi_1 - \pi_0)$ contradicting $\pi_{n+1} \leq 1$ and assuming $\pi_1 < \pi_0$ implies $\pi_{n+1} < n(\pi_1 - \pi_0) + 1$ leading to a further contradiction for arbitrarily large n, namely $\pi_{n+1} < 0$.[1]

The reader should note that all conclusions hold (and thus no steady state distribution exists) irrespective of the values of λ and μ!

The unbounded GSPN of Fig. 9.20 can, e.g., be used to describe the core transshipping activities in logistic nodes (cf.[24]). Vehicles arrive at a rate of λ delivering a unit to the store (p_1) of the logistic node, whereas vehicles wanting to load a unit are arriving at a rate of μ and have to wait (p_2) when their demands cannot be satisfied because of an empty storage. The time for loading and unloading etc. is neglected in the model.

In [23] this effect is described in more detail and it is also shown that a simulation of such a system runs the immanent danger of leaving this effect uncovered and assuming a stationary process.

[1] Note that in $\sum_{i=-n}^{n}(\frac{\lambda}{\mu})^i \ldots$ we have n terms with a factor of $\alpha_1 := \frac{\lambda}{\mu}$ and similarly n terms with the reciprocal factor $\alpha_2 := \frac{\mu}{\lambda}$. Thus $\alpha_1 \geq 1$ or $\alpha_2 \geq 1$ holds irrespective of the values of λ and μ.

Bibliography

[1] G.A. Agha, F. De Cindio, and G. Rozenberg, editors. *Concurrent Object-Oriented Programming and Petri Nets, Advances in Petri Nets 2001*, volume 2001. Lecture Notes in Computer Science, Springer-Verlag, 2001.

[2] M. Ajmone-Marsan. Stochastic Petri nets: An elementary introduction. In *Advances in Petri Nets*, pages 1–29. Lecture Notes in Computer Science, Vol. 424, Springer-Verlag, 1989.

[3] M. Ajmone-Marsan, editor. *Proceedings of the 14th International Conference on Application and Theory of Petri Nets, Chicago (USA)*, volume 691. Lecture Notes in Computer Science, Springer-Verlag, June 1993.

[4] M. Ajmone-Marsan, G. Balbo, A. Bobbio, G. Chiola, G. Conte, and A. Cumani. On Petri nets with stochastic timing. In *Proceedings of the International Workshop on Timed Petri Nets, Torino*, pages 80–87, 1985.

[5] M. Ajmone-Marsan, G. Balbo, A. Bobbio, G. Chiola, and A. Cumani. The effect of execution policies in the semantics and analysis of Stochastic Petri nets. *IEEE Transactions on Software Engineering*, 15(7):832–846, 1989.

[6] M. Ajmone-Marsan, G. Balbo, and G. Conte. *Performance Models of Multiprocessor Systems*. MIT Press Series in Computer Science, 1986.

[7] M. Ajmone-Marsan, G. Balbo, G. Conte, S. Donatelli, and G. Franceschinis. *Modelling with Generalized Stochastic Petri Nets*. Wiley Series in Parallel Computing. John Wiley and Sons, 1995.

[8] M. Ajmone-Marsan and G. Chiola. On Petri nets with deterministic and exponential transition firing times. In *Proceedings of the 7th European Workshop on Application and Theory of Petri Nets, Oxford*, pages 151–165, 1986.

[9] M. Ajmone-Marsan and G. Chiola. On Petri nets with deterministic and exponentially distributed firing times. In G. Rozenberg, editor, *Concurrency and Nets. Advances in Petri Nets*, pages 132–145. Springer-Verlag, 1987.

[10] M. Ajmone-Marsan and G. Chiola. Improving the efficiency of the analysis of DSPN models. In *Proceedings of the 9th European Workshop on Application and Theory of Petri Nets, Venice*, pages 183–201, 1988.

[11] M. Ajmone-Marsan, G. Conte, and G. Balbo. A class of Generalised Stochastic Petri Nets for the performance evaluation of multiprocessor systems. *ACM Transactions on Computer Systems*, 2:93–122, 1984.

[12] P. Azema, editor. *Proceedings of the 18th International Conference on Application and Theory of Petri Nets, Toulouse (France)*, volume 1248. Lecture Notes in Computer Science, Springer-Verlag, June 1997.

[13] G. Balbo, S.C. Bruell, and M. Sereno. Arrival theorems for product-form stochastic Petri nets. In *SIGMETRICS*, pages 87–97, May 1994.

[14] S. Balsamo and V. de Nitto-Persone. A survey of product-form queueing networks with blocking and their equivalences. *Annals of Operation Research*, 48:31–61, 1994.

[15] F. Baskett, K. M. Chandy, R. R. Muntz, and F. G. Palacios. Open, closed and mixed networks of queues with different classes of customers. In *Journal of the ACM 22*, pages 248–260, 1975.

[16] B. Baumgarten. *Petri-Netze. Grundlagen und Anwendungen.* BI Wissenschaftsverlag, Mannheim, 1990.

[17] F. Bause. *Funktionale Analyse zeitbehafteter Petri-Netze*. DeutscherUniversitätsVerlag, 1992.

[18] F. Bause. Queueing Petri nets: a formalism for the combined qualitative and quantitative analysis of systems. In *Proceedings of the 5th International Workshop on Petri Nets and Performance Models, Toulouse (France)*, pages 14–23. IEEE, October 1993.

[19] F. Bause. Petri nets and priorities. Technical Report 564, Fachbereich Informatik der Universität Dortmund (Germany), 1995.

[20] F. Bause. On the Analysis of Petri Nets with Static Priorities. *Acta Informatica*, 33(7):669–685, 1996.

[21] F. Bause. Analysis of Petri Nets with a dynamic priority method. In P. Azema and G. Balbo, editors, *Proc. of 18th International Conference on Application and Theory of Petri Nets, Toulouse, France, June 1997*, volume LNCS 1248, pages 215–234, Berlin, Germany, June 1997. Springer-Verlag.

[22] F. Bause and H. Beilner. Eine Modellwelt zur Integration von Warteschlangen- und Petri-Netz-Modellen. In *Proceedings of the 5th GI/ITG-Fachtagung, Messung, Modellierung und Bewertung von Rechensystemen und Netzen*, pages 190–204. Braunschweig, Gesellschaft für Informatik (GI), Germany, September 1989.

[23] F. Bause and H. Beilner. Intrinsic problems in simulation of logistic networks. In *11th European Simulation Symposium and Exhibition (ESS99), Simulation in Industry, Erlangen (Germany)*, pages 193–198, October 1999.

[24] F. Bause and H. Beilner. A short note on synchronisation in open systems. In *Petri Net Newsletters, No. 57*, pages 9–12. Special Interest Group On Petri Nets and Related System Models, Gesellschaft für Informatik (GI), Germany, 1999.

[25] F. Bause and P. Buchholz. Aggregation and disaggregation in product form Queueing Petri nets. In *Proceedings of the Seventh International Workshop on Petri Nets and Performance Models, June 3-6, 1997, Saint Malo, France*, pages 16–25. IEEE Computer Society, 1997.

[26] F. Bause and P. Buchholz. Queueing Petri nets with product form solution. *Performance Evaluation*, 32(4):265–299, 1998.

[27] F. Bause, P. Buchholz, and P. Kemper. QPN-Tool for the specification and analysis of hierarchically combined Queueing Petri Nets. In H. Beilner and F. Bause, editors, *Quantitative Evaluation of Computing and Communication Systems*, volume LNCS 977, pages 224–238. Springer-Verlag, Berlin, 1995.

[28] F. Bause and H. Beilner (eds.). Performance tools - model interchange formats. Technical Report 581, Fachbereich Informatik der Universität Dortmund (Germany); also presented at the Joint International Conference on "Modelling Techniques and Tools for Computer Performance Evaluation" and "Measuring, Modelling and Evaluating Computing and Communication Systems", Sept. 1995, Heidelberg (Germany), 1995.

[29] F. Bause and P. Kemper. Queueing Petri nets. In *Proceedings of the 3rd Fachtagung Entwurf komplexer Automatisierungssysteme, Braunschweig*. Technische Universität Braunschweig, May 1993.

[30] F. Bause and P. Kemper. QPN-Tool for the qualitative and quantitative analysis of Queueing Petri Nets. In G. Haring and G. Kotsis, editors, *Proc. of the 7th International Conference on Computer Performance Evaluation, Modelling Techniques and Tools, Vienna (Austria), LNCS 794*, pages 321–334. Springer-Verlag, Berlin, 1994.

[31] G. Berthelot. Checking properties of nets using transformations. In *Advances in Petri Nets*, pages 19–40. Springer-Verlag, 1985.

[32] G. Berthelot. Transformations and decompositions of nets. In *Petri Nets: Central Models and Their Properties, Advances in Petri Nets, Part I*, pages 359–376. Springer-Verlag, 1986.

[33] B. Berthomieu and M. Menasche. A state enumerative approach for analyzing time Petri nets. In W. Brauer, editor, *Applications and Theory of Petri Nets. Selected Papers from the 3rd European Workshop on Applications and Theory of Petri Nets, Varenna*. Informatik-Fachberichte. No. 66, 1982.

[34] E. Best. Structure theory of Petri nets: The free choice hiatus. In *Advances in Petri Nets, Part I*, pages 168–205. Springer-Verlag, 1986.

[35] E. Best and J. Desel. Partial order behaviour and structure of Petri nets. Technical Report 373, Arbeitspapiere der GMD, Gesellschaft für Mathematik und Datenverarbeitung, Sankt Augustin, Germany, 1989.

[36] E. Best, J. Desel, and J. Esparza. Traps characterize home states in free choice systems. *Theoretical Computer Science*, 101:161–176, 1992.

[37] E. Best and M.W. Shields. Some equivalence results for free choice nets and simple nets on the periodicity of live free choice nets. In *Proceedings of the 8th Colloquium on Trees and Algebra and Programming*, pages 141–154. Lecture Notes in Computer Science, 159, 1983.

[38] E. Best and P.S. Thiagarajan. Some classes of live and safe Petri nets. In *Advances in Petri Nets*, pages 71–94. Springer-Verlag, 1987.

[39] E. Best and K. Voss. Free choice systems have home states. *Acta Informatica*, 21:89–100, 1984.

[40] J. Billington, editor. *Proceedings of the 17th International Conference on Application and Theory of Petri Nets, Osaka (Japan)*, volume 1091. Lecture Notes in Computer Science, Springer-Verlag, June 1996.

[41] J. Billington, editor. *Application of petri nets to communication networks, Advances in Petri Nets 1999*, volume 1605. Lecture Notes in Computer Science, Springer-Verlag, 1999.

[42] G. Bolch, S. Greiner, H. de Meer, and K.S. Trivedi. *Queueing Networks and Markov Chains*. John Wiley and Sons, 1998.

[43] R.J. Boucherie. A characterisation of independence for competing markov chains with applications to stochastic Petri nets. In *in [138]*, pages 117–126, 1993.

[44] W. Brauer, W. Reisig, and G. Rozenberg, editors. *Petri Nets: Applications and Relationships to Other Models of Concurrency. Advances in Petri Nets 1986, Part II*, volume 255. Lecture Notes in Computer Science, Springer-Verlag, 1987.

[45] W. Brauer, W. Reisig, and G. Rozenberg, editors. *Petri Nets: Central Models and Their Properties. Advances in Petri Nets 1986, Part I*, volume 254. Lecture Notes in Computer Science, Springer-Verlag, 1987.

[46] S.L. Brumelle. A generalization of $L = \lambda W$ to moments of queue length and waiting times. *Operations Research*, 20:1127–1136, 1972.

[47] P. Buchholz and P. Kemper. APNNed – a net editor and debugger within the APNN toolbox. In J. Desel, P. Kemper, E. Kindler, and A. Oberweis, editors, *Forschungsbericht: 5. Workshop Algorithmen und Werkzeuge für Petrinetze*. Universität Dortmund, Fachbereich Informatik, 1998.

[48] P.J. Burke. The output of a queueing system. *Operations Research*, 4:699–704, 1956.

[49] M.L. Chaudhry and J.G.C. Templeton. *A First Course in Bulk Queues*. John Wiley and Sons, 1983.

[50] G. Chiola. A graphical Petri net tool for performance analysis. In *Proceedings of the 8th European Workshop on Application and Theory of Petri Nets, Zaragoza (Spain)*, pages 317–331, 1987.

[51] G. Chiola. GreatSPN 1.5 software architecture. *Computer Performance Evaluation*, pages 121–136, 1992.

[52] G. Ciardo and A. S. Miner. A data structure for the efficient kronecker solution of GSPNs. In *Proc. 8th Int. Workshop on Petri Net and Performance Models (PNPM'99), 8-10 October 1999, Zaragoza (Spain)*, pages 22–31, 1999.

[53] E. Cinlar. *Introduction to Stochastic Processes*. Prentice Hall, 1975.

[54] J.-M. Colom and M. Koutny, editors. *Proceedings of the 22nd International Conference on Application and Theory of Petri Nets, Newcastle upon Tyne (UK)*, volume 2075. Lecture Notes in Computer Science, Springer-Verlag, June 2001.

[55] J.E. Coolahan and N. Roussopoulos. A timed Petri net methodology for specifying real-time system requirements. In *Proceedings of the International Workshop on Timed Petri Nets, Turino*, pages 24–31, 1985.

[56] P.J. Courtois and P. Semal. Bounds for transient characteristics of large or infinite Markov chains. In W.J. Stewart, editor, *Numerical Solutions of Markov Chains*, pages 413–434. Marcel Dekker, Inc., 1990.

[57] D.R. Cox. A use of complex probabilities in the theory of complex processes. *Proc. Cambridge Phil. Soc.*, 51:313–319, 1955.

[58] G. de Michelis and M. Diaz, editors. *Proceedings of the 16th International Conference on Application and Theory of Petri Nets, Turin (Italy)*, volume 935. Lecture Notes in Computer Science, Springer-Verlag, June 1995.

[59] J. Desel, editor. *Proceedings of the 19th International Conference on Application and Theory of Petri Nets, Lisboa (Portugal)*, volume 1420. Lecture Notes in Computer Science, Springer-Verlag, June 1998.

[60] J. Desel and J. Esparza. *Free Choice Petri Nets*. Cambridge University Press, 1995.

[61] M. Diaz. Petri net based models in the specification and verification of protocols. In G. Rozenberg, editor, *Petri Nets: Applications and Relationship to Other Models of Concurrency. Advances in Petri Nets. Proceedings of an Advanced Course, Bad Honnef*, pages 135–170. Lecture Notes in Computer Science, Vol. 255, Springer-Verlag, 1986.

[62] Stephen R. Donaldson. Algorithms and complexity of Petri net transformations. Master's thesis, Department of Computer Science, University of Cape Town, 1993.

[63] S. Donatelli. Superposed generalized stochastic petri nets: definition and efficient solution. In R. Valette, editor, *Lecture Notes in Computer Science; Application and Theory of Petri Nets 1994, Proceedings 15th International Conference, Zaragoza, Spain*, volume 815, pages 258–277. Springer-Verlag, 1994.

[64] S. Donatelli, editor. *Proceedings of the 20th International Conference on Application and Theory of Petri Nets, Williamsburg, Virginia, (USA)*, volume 1639. Lecture Notes in Computer Science, Springer-Verlag, June 1999.

[65] S. Donatelli and M. Sereno. On the product form solution for stochastic Petri nets. In *Proceedings of the 13th International Conference on Application and Theory of Petri Nets, Sheffield (UK)*, pages 154–172, 1992.

[66] J.B. Dugan. *Extended Stochastic Petri Nets: Applications and Analysis*. PhD thesis, Department of Electrical Engineering, Duke University, 1984.

[67] H. Ehrig, G. Juhas, J. Padberg, and G. Rozenberg, editors. *Unifying Petri Nets, Advances in Petri Nets 2001*, volume 2128. Lecture Notes in Computer Science, Springer-Verlag, 2001.

[68] S. Eilon. A simpler proof of $L = \lambda W$. *Operations Research*, 17:915–916, 1969.

[69] J. Esparza. Synthesis rules for Petri nets, and how they lead to new results. In J.C.M. Baeten and J.W. Klop, editors, *Proceedings of CONCUR'90: Theories of Concurrency: Unification and Extension*, pages 183–198. Springer-Verlag, Berlin, 1990.

[70] J. Esparza. Reduction and synthesis of live and bounded free choice nets. Technical report, Hildesheimer Informatikberichte, Institut für Informatik, Universität Hildesheim, 1991.

[71] J. Esparza and C. Lakos, editors. *Proceedings of the 23nd International Conference on Application and Theory of Petri Nets, Adelaide (Australia)*, volume 2360. Lecture Notes in Computer Science, Springer-Verlag, June 2002.

[72] J. Esparza and M. Silva. Top-down synthesis of live and bounded free choice nets. In *Proceedings of the 1tth International Conference on the Application and Theory of Petri Nets, Paris*, pages 63–83, 1990.

[73] F. Feldbrugge. *Petri Net Tool Overview 1989*, chapter in [157], pages 151–178. Lecture Notes in Computer Science, Springer-Verlag, 1990.

[74] F. Feldbrugge. *Petri Net Tool Overview 1992*, chapter in [161], pages 169–209. Lecture Notes in Computer Science, Springer-Verlag, 1993.

[75] W. Feller. *An Introduction to Probability Theory and its Applications*, volume 1. Academic Press, 1968.

[76] G. Balbo G. Chiola, M. Ajmone-Marsan and G. Conte. Generalized stochastic Petri nets: A definition at the net level and its implications. *IEEE Transactions on Software Engineering*, 19(2):89–107, February 1993.

[77] E. Gelenbe and G. Pujolle. *Introduction to Queueing Networks*. John Wiley & Sons, 1987.

[78] H.J. Genrich and K. Lautenbach. Synchronisationsgraphen. *Acta Informatica*, 2:143–161, 1973.

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

[80] H.J. Genrich and P.S. Thiagarajan. A theory of bipolar synchronization schemes. *Theoretical Computer Science*, 30:241–318, 1984.

[81] R. German. New results for the analysis of deterministic and stochastic Petri nets. In *Proc. IEEE International Performance and Dependability Symposium, Erlangen (Germany)*, pages 114–123, 1995.

[82] R. German. *Performance Analysis of Communication Systems: Modeling with Non-Markovian Stochastic Petri Nets.* John Wiley and Sons, 2000.

[83] R. German, C. Kelling, A. Zimmermann, and G. Hommel. TimeNET: a toolkit for evaluating non-markovian stochastic Petri nets. *Performance Evaluation*, 1995.

[84] R. German and C. Lindemann. Analysis of stochastic Petri nets by the method of supplementary variables. *Performance Evaluation*, 20:317–335, 1994.

[85] R. German and J. Mitzlaff. Transient analysis of deterministic and stochastic Petri nets with TimeNET. In H. Beilner and F. Bause, editors, *Quantitative Evaluation of Computing and Communication Systems*, volume LNCS 977, pages 209–223. Springer-Verlag, Berlin, 1995.

[86] R. Goodman. *Introduction to Stochastic Models*. Benjamin/Cummings Publishing Company, Inc, 1988.

[87] W.J. Gordon and G.F. Newell. Closed queueing networks with exponential servers. *Operations Research*, 15(2):254–265, 1967.

[88] S. Haddad. Generalization of reduction theory to coloured nets. In *Proceedings of the 9th European Workshop on Application and Theory of Petri Nets, Venice*, 1988.

[89] W. Henderson and P.G. Taylor. Embedded processes in stochastic Petri nets. *IEEE Transactions on Software Engineering*, 17:108–116, 1991.

[90] R.A. Howard. *Dynamic Probabilistic Systems. Volume I: Markov Models. Volume II: Semi-Markov and Decision Processes*. John Wiley & Sons, Inc., 1971.

[91] J.R. Jackson. Networks of waiting lines. *Oper. Res.*, 5:518–521, 1957.

[92] N.K. Jaiswal. *Priority queues*. Academic Press, 1968.

[93] K. Jensen. Coloured Petri nets and the invariant method. *Mathematical Foundations on Computer Science, Lecture Notes in Computer Science*, 118:327–338, 1981.

[94] K. Jensen. Coloured Petri nets. In G. Rozenberg, editor, *Petri Nets: Central Models and Their Properties. Advances in Petri Nets. Proceedings of an Advanced Course Lecture Notes in Computer Science*, volume 254, No. 1, pages 248–299. Springer-Verlag, Berlin, 1986.

[95] K. Jensen, editor. *Proceedings of the 13th International Conference on Application and Theory of Petri Nets, Sheffield (UK)*, volume 616. Lecture Notes in Computer Science, Springer-Verlag, June 1992.

[96] K. Jensen. *Coloured Petri Nets; Basic Concepts, Analysis Methods and Practical Use*. EATCS Monographs on Theoretical Computer Science, Vol. 1 Basic Concepts, 1997.

[97] K. Jensen. *Coloured Petri Nets; Basic Concepts, Analysis Methods and Practical Use*. EATCS Monographs on Theoretical Computer Science, Vol. 2 Analysis Methods, 1997.

[98] K. Jensen. *Coloured Petri Nets; Basic Concepts, Analysis Methods and Practical Use*. EATCS Monographs on Theoretical Computer Science, Vol. 3 Practical Use, 1997.

[99] K. Jensen and G. Rozenberg, editors. *High-level Petri Nets. Theory and Application.* Springer-Verlag, 1991.

[100] J.W. Jewell. A simple proof of $L = \lambda W$. *Operations Research*, 15:1109–1116, 1967.

[101] R. Johnsonbaugh and T. Murata. Additional methods for reduction and expansion of marked graphs. *IEEE Transactions on Circuit and Systems*, 28(10):1009–1014, 1981.

[102] K. Kant. *Introduction to Computer System Performance Evaluation*. McGraw-Hill, Inc., 1992.

[103] J.G. Kemeny and J.L. Snell. *Finite Markov Chains*. Springer-Verlag, New York, 1960.

[104] P. Kemper. Linear time algorithm to find a minimal deadlock in a strongly connected free-choice net. In M. Ajmone-Marsan, editor, *Proceedings of the 14th International Conference on Application and Theory of Petri Nets*. Springer-Verlag, Berlin, 1993.

[105] P. Kemper. Numerical analysis of superposed GSPNs. *IEEE Transactions on Software Engineering*, 22(9):615–628, September 1996.

[106] P. Kemper. Reachability analysis based on structured representations. In *Lecture Notes in Computer Science; Proc. 17th International Conference in Application and Theory of Petri Nets (ICATPN'96), Osaka, Japan*, volume 1091, pages 269–288. Springer-Verlag, June 1996.

[107] P. Kemper and F. Bause. An efficient polynomial-time algorithm to decide liveness and boundedness of free choice nets. In K. Jensen, editor, *Proceedings of the 13th International Conference on Application and Theory of Petri Nets*, volume LNCS 616, pages 263–278. Springer-Verlag, Berlin, 1992.

[108] L. Kleinrock. *Queueing Systems. Volume 1: Theory*. John Wiley and Sons, 1975.

[109] R. Kosaraju. Decidability of reachability in vector addition systems. In *Proceedings of the 14th Annual ACM Symposium on Theory of Computing*, pages 267–280, 1982.

[110] R. Lal and U.N. Bhat. Reduced systems in Markov chains and their applications in queueing theory. *Queueing Systems*, 2:147–172, 1987.

[111] A.A. Lazar and T.G. Robertazzi. Markovian Petri net protocols with product form solution. *Performance Evaluation*, 12:67–77, 1991.

[112] C. Lindemann. DSPNexpress: a software package for the efficient solution deterministic and stochastic Petri nets. *Performance Evaluation*, 1995.

[113] C. Lindemann. *Performance Modelling with Deterministic and Stochastic Petri Nets*. John Wiley & Sons, 1998.

[114] C. Lindemann, A. Thümmler, A. Klemm, M. Lohmann, and O.P. Waldhorst. Quantitative system evaluation with dspnexpress 2000. In *Proc. 2nd Int. Workshop on Software and Performance, Ottawa (Canada)*, pages 12–17, 2000.

[115] J.D.C. Little. A simple proof of $L = \lambda W$. *Operations Research*, 9:383–387, 1961.

[116] K. Marshall and R. Wolff. Customer average and time average - queue lengths and waiting times. *J. of Appl. Probability*, 8:535–542, 1971.

[117] J. Martinez and M. Silva. A simple and fast algorithm to obtain all invariants of a generalized Petri net. In *Application and Theory of Petri Nets, Selected Papers from the First and Second European Workshop on Application and Theory of Petri Nets*. Informatik-Fachberichte, No. 52, Springer-Verlag, 1981.

[118] E.W. Mayr. An algorithm for the general Petri net reachability problem. In *Proc. of the 13th Annual ACM Symp. on Theory of Computing*, pages 238–246, 1981.

[119] E.W. Mayr. An algorithm for the general Petri net reachability problem. *SIAM, Journal Comput.*, 13(3):441–460, August 1984.

[120] J. McKenna. A generalization of littles's law to moments of queue lengths and waiting times in closed, product-form queueing networks. *J. Appl. Prob.*, 26:121–133, 1989.

[121] P. Merlin. *A Study of the Recoverability of Computing Systems*. PhD thesis, Department of Information and Computer Science, University of California, Irvine (USA), 1974.

[122] R. Milner, R. Harper, and M. Tofte. *The definition of standard ML*. MIT Press, 1990.

[123] A.S. Miner and G. Ciardo. Efficient reachability set generation and storage using decision diagrams. In *Proc. of the 20th Int. Conf. on Application and Theory of Petri Nets 1999, Williamsburg, VA (USA). Lecture Notes in Computer Science Vol. 1639*, pages 6–25, 1999.

[124] M.K. Molloy. *On the Intergration of Delay and Throughput Measures in Distributed Processing Models*. PhD thesis, University of California, 1981.

[125] T. Murata. Petri nets: Properties, analysis and applications. *Proceedings of IEEE*, 77(4):541–580, April 1989.

[126] T. Murata and J.Y. Koh. Reduction and expansion of live and safe marked graphs. *IEEE Transactions on Circuit and Systems*, 27(1):68–71, 1980.

[127] S. Natkin. *Les Reseaux de Petri Stochastiques et leur Application a l'Evaluation des Systemes Informatiques*. PhD thesis, CNAM, Paris, 1980.

[128] M. Nielsen, editor. *Proceedings of the 21st International Conference on Application and Theory of Petri Nets, Aarhus (Denmark)*, volume 1825. Lecture Notes in Computer Science, Springer-Verlag, June 2000.

[129] A. Pagnoni. Stochastic nets and performance evaluation. In *Petri Nets: Central Models and Their Properties. Advances in Petri Nets*, pages 460–478. Lecture Notes in Computer Science, No. 254, 1986.

[130] J.L. Peterson. Petri nets. *ACM Computing Surveys*, 9:242–252, September 1977.

[131] J.L. Peterson. A note on coloured Petri nets. *Information Processing Letters*, 11:40–43, Aug. 1980.

[132] J.L. Peterson. *Petri Nets and the Modelling of Systems*. MIT Press Series in Computer Science, 1981.

[133] C.A. Petri. *Kommunikation mit Automaten*. PhD thesis, Universität Bonn, 1962.

[134] *Proceedings of the International Workshop on Timed Petri Nets, Turino (Italy)*. IEEE Computer Society Press, July 1985.

[135] *Proceedings of the 2nd International Workshop on Petri Nets and Performance Models, Madison (USA)*. IEEE Computer Society Press, 1987.

[136] *Proceedings of the 3rd International Workshop on Petri Nets and Performance Models, Kyoto (Japan)*. IEEE Computer Society Press, 1989.

[137] *Proceedings of the 4th International Workshop on Petri Nets and Performance Models, Melbourne (Australia)*. IEEE Computer Society Press, 1991.

[138] *Proceedings of the 5th International Workshop on Petri Nets and Performance Models, Toulouse (France)*. IEEE Computer Society Press, 1993.

[139] *Proceedings of the 6th International Workshop on Petri Nets and Performance Models, Durham (USA)*. IEEE Computer Society Press, 1995.

[140] *Proceedings of the 7th International Workshop on Petri Nets and Performance Models, Saint Malo (France)*. IEEE Computer Society Press, 1997.

[141] *Proceedings of the 8th International Workshop on Petri Nets and Performance Models, Zaragoza (Spain)*. IEEE Computer Society Press, 1999.

[142] *Proceedings of the 9th International Workshop on Petri Nets and Performance Models, Aachen (Germany)*. IEEE Computer Society Press, 2001.

[143] C.V. Ramamoorthy and G.S. Ho. Performance evaluation of asynchronous concurrent systems using Petri nets. *IEEE Transactions on Software Engineering*, 6(5):440–449, 1980.

[144] C. Ramchandani. *Analysis of Asynchronous Concurrent Systems by Timed Petri Nets*. PhD thesis, Department of Electrical Engineering, MIT (USA), July 1973.

[145] M. Rauhamaa. A comparative study of methods for efficient reachability analysis. Technical Report Research Report, Helsinki University of Technology, September 1990.

[146] M. Raynal. *Algorithms for mutual exclusion*. MIT Press, 1986.

[147] R.R. Razouk and C.V. Phelps. Performance analysis using Petri nets. In *Proceedings of the IFIP WG 6.1 4th International Workshop on Protocol, Specification, Testing, and Verification*, pages 561–576, Skytop Lodge (USA), 1984.

[148] W. Reisig. Petri nets with individual tokens. In *Applications and Theory of Petri Nets. Selected Papers from the 3rd European Workshop on Applications and Theory of Petri Nets, Varenna (Italy)*. Informatik-Fachberichte, No. 66, Springer-Verlag, 1982.

[149] W. Reisig. *Petri Nets. An Introduction*, volume 4. EATCS Monographs on Theoretical Computer Science, Springer-Verlag, 1985.

[150] W. Reisig. *Petrinetze. Eine Einführung*. Springer Verlag, 1986.

[151] T.G. Robertazzi. *Computer Networks and Systems: Queueing Theory and Performance Evaluation*. Springer, 1990.

[152] S.M. Ross. *Introduction to Probability Models*. Academic Press, fourth edition, 1989.

[153] G. Rozenberg, editor. *Advances in Petri Nets 1984*, volume 188. Lecture Notes in Computer Science, Springer-Verlag, 1985.

[154] G. Rozenberg, editor. *Advances in Petri Nets 1985*, volume 222. Lecture Notes in Computer Science, Springer-Verlag, 1986.

[155] G. Rozenberg, editor. *Advances in Petri Nets 1987*, volume 266. Lecture Notes in Computer Science, Springer-Verlag, 1987.

[156] G. Rozenberg, editor. *Advances in Petri Nets 1988*, volume 340. Lecture Notes in Computer Science, Springer-Verlag, 1988.

[157] G. Rozenberg, editor. *Advances in Petri Nets 1989*, volume 424. Lecture Notes in Computer Science, Springer-Verlag, 1990.

[158] G. Rozenberg, editor. *Advances in Petri Nets 1990*, volume 483. Lecture Notes in Computer Science, Springer-Verlag, 1991.

[159] G. Rozenberg, editor. *Advances in Petri Nets 1991*, volume 524. Lecture Notes in Computer Science, Springer-Verlag, 1991.

[160] G. Rozenberg, editor. *Advances in Petri Nets 1992*, volume 609. Lecture Notes in Computer Science, Springer-Verlag, 1992.

[161] G. Rozenberg, editor. *Advances in Petri Nets 1993*, volume 674. Lecture Notes in Computer Science, Springer-Verlag, 1993.

[162] G. Rozenberg, editor. *Advances in Petri Nets 1998*, volume 1499. Lecture Notes in Computer Science, Springer-Verlag, 1998.

[163] V.M. Savi and X. Xie. Liveness and boundedness analysis for Petri nets with event graph modules. In *Proceedings of the 13th International Conference on Application and Theory of Petri Nets*, pages 328–347. Springer-Verlag, 1992.

[164] M. Sereno and G. Balbo. Computational algorithms for product form solution stochastic Petri nets. In *Proceedings of the 5th International Workshop on Petri Nets and Performance Models, Toulouse (France)*, pages 98–107. IEEE Computer Society Press, Oct. 1993.

[165] M. Sereno and G. Balbo. Mean value analysis of stochastic petri nets. *Performance Evaluation*, 29(1):35–62, 1997.

[166] J. Sifakis. Use of Petri nets for performance evaluation. In *Measuring, Modeling and Evaluation of Computer Systems, Proceedings of the 3rd International Workshop on Modeling and Performance Evaluation of Computer Systems, Amsterdam*, pages 75–93, 1977.

[167] J. Sifakis. Performance evaluation of systems using Petri nets. In W. Brauer, editor, *Application and Theory of Petri Nets, Selected Papers from the First and Second European Workshop on Application and Theory of Petri Nets*, pages 307–319. Informatik-Fachberichte, No. 52, 1980 and 1981.

[168] Y. Souissi and G. Memmi. Compositions of nets via a communication medium. In *Proceedings of the 10th Workshop on Application and Theory of Petri Nets, Bonn (Germany)*, pages 292–311, 1989.

[169] W.J. Stewart. *Introduction to the numerical solution of Markov chains*. Princeton University Press, 1994.

[170] S. Stidham. A last word on $L = \lambda W$. *Operations Research*, 22:417–421, 1974.

[171] I. Suzuki and T. Murata. Stepwise refinement of transitions and places. In W. Brauer, editor, *Application and Theory of Petri Nets. Selected Papers of the First and Second Workshop on Application and Theory of Petri Nets*, pages 136–141. Informatik-Fachberichte, No. 52, 1980.

[172] I. Suzuki and T. Murata. A method of stepwise refinement and abstraction of Petri nets. *Journal on Computing and System Sciences*, 27(1), 1983.

[173] E. Teruel and M. Silva. Liveness and home states in equal conflict systems. In *Proceedings of the 14th International Conference on Application and Theory of Petri Nets, Chicago (USA)*, pages 415–432, 1993.

[174] E. Teruel and M. Silva. Well-formedness of equal conflict systems. In *Proceedings of the 15th European Workshop on Application and Theory of Petri Nets, Zaragoza (Spain)*, pages 491–510, 1994.

[175] P.S. Thiagarajan and K. Voss. In praise of free choice nets. In *Advances in Petri Nets*, pages 438–454. Springer-Verlag, 1984.

[176] R. Valette. Analysis of Petri nets by stepwise refinement. *Journal of Computer and Systems Sciences*, 18:35–46, 1979.

[177] R. Valette, editor. *Proceedings of the 15th International Conference on Application and Theory of Petri Nets, Zaragoza (Spain)*, volume 815. Lecture Notes in Computer Science, Springer-Verlag, July 1994.

[178] W. Vogler. Live and bounded free choice nets have home states. In *Petri Net Newsletters*, pages 18–21. Special Interest Group On Petri Nets and Related System Models, Gesellschaft für Informatik (GI), Germany, 4 1989.

[179] I.Y. Wang and T.G. Robertazzi. Service stage Petri net models with product form. *Queueing Systems*, 7:355–374, 1990.

[180] W. Witt. A review of $L = \lambda W$ and extensions. *Queueing Systems*, 9:235–268, 1991. A correction note for this paper was published 1992 in Vol. 12, pp. 431-432 of the same journal.

[181] C.Y. Wong, T.S. Dillon, and K.E. Forward. Timed place Petri nets with stochastic representation of place time. In *Proceedings of the International Workshop on Timed Petri Nets, Turino*, pages 96–103, 1985.

[182] S.S. Yau and C.R. Chou. Control flow analysis of distributed computing system software using structured Petri net models. In *Proceedings of the Workshop on Future Trends of Distributed Systems in the 1990s, HongKong*, pages 174–183, 9 1988.

[183] A. Zimmermann, J. Freiheit, R. German, and G. Hommel. Petri net modelling and performability evaluation with timenet 3.0. In *11th Int. Conf. on Modelling Techniques and Tools for Computer Performance Evaluation (TOOLS'2000)*, volume LNCS 1786, pages 188–202. Springer-Verlag, 2000.

[184] A. Zimmermann, R. German, J. Freiheit, and G. Hommel. Timenet 3.0 tool description. In *Int. Conf. on Petri Nets and Performance Models (PNPM'99), Zaragoza (Spain)*, 1999.

[185] W.M. Zuberek. Timed Petri nets and preliminary performance evaluation. In *Proceedings IEEE 7th Annual Symposium on Computer Architecture*, pages 89–96, 1980.

[186] W.M. Zuberek. Performance evaluation using extended Petri nets. In *Proceedings of the International Workshop on Timed Petri Nets, Turino*, pages 272–278, 1985.

Index