Università Ca' Foscari di Venezia

Dipartimento di Informatica
Dottorato di Ricerca in Informatica, 21° Ciclo

(A.A. 2005/2006 – 2007/2008)

Tesi di dottorato: TD-2009-2
Settore scientifico disciplinare: INF/01

On the relations among product-form stochastic models

Andrea Marin
Matr. 955255

Tutore del dottorato
Simonetta Balsamo

Coordinatore del dottorato
Annalisa Bossi

January, 2009

Author's Web Page: http://www.dsi.unive.it/~marin

Author's e-mail: marin@dsi.unive.it

Author's address:

Dipartimento di Informatica
Università Ca' Foscari di Venezia
Via Torino, 155
30172 Venezia Mestre – Italia
tel. +39 041 2348411
fax. +39 041 2348419
web: http://www.dsi.unive.it

To my family and my friends.
Alla mia famiglia ed ai miei amici.

Abstract

Product-form stochastic models are characterized by a Markovian stochastic process that fulfills a set of properties that allows an efficient steady state analysis. According to this approach, the model is decomposed into several components. Each of these components has an underlying stochastic process that is in general much simpler than the joint one. The product-form property states that the steady state probabilities of the joint process can be expressed as the normalized product of the steady state proabilities of its interacting components. Product-form stochastic models are widely used for performance evaluation purposes in the study of communication protocols, software or hardware architectures. Product-form stochastic models can be defined using several high-level formalisms. For example BCMP theorem provides a product-form solution for a class of Markovian queueing networks. In this thesis we use two results, the $M \Rightarrow M$ property and the Reversed Compound Agent Theorem (RCAT) to explore the relations among several product-form model classes belonging to different formalisms: queueing networks (QN), stochastic Petri nets (SPN), generalized stochastic Petri nets (GSPN), Markovian Process Algebra (MPA). We identify new classes of product-form GSPNs, and we prove that previous results on product-form SPNs can be studied using RCAT. From a practical point of view we show how to map multiclass queueing stations of BCMP types into GSPNs with a finite structure maintaining a strong equivalence relation (in particular the average performance indices and the product-form property are preserved). As a consequence we are able to study hybrid models in product-form where their interaction and compositions are formally defined by GSPNs. An algorithm is defined in order to translate BCMP QNs into GSPNs according to the previous theoretical results. The algorithm can be easily extended to allow the modeler to specify new stations using GSPNs.

Sommario

I modelli stocastici in forma prodotto sono caratterizzati da processi Markoviani che godono di alcune proprietà che consentono un'analisi della distribuzione stazionaria di stato efficiente. Questa tecnica prevede che si individuino alcune componenti del modello, a ciascuna delle quali è associato un processo stocastico molto più semplice di quello di partenza. La proprietà di forma prodotto stabilisce che la distribuzione stazionaria di stato del processo congiunto può essere calcolata come normalizzazione del prodotto delle distribuzioni stazionarie delle singole componenti opportunamente riparametrizzate. I modelli stocastici in forma prodotto sono molto usati nell'ambito della valutazione delle prestazioni dei sistemi, come ad esempio nello studio dei protocolli di comunicazione o delle architetture hardware o software. I modelli in forma prodotto possono essere specificati usando diversi formalismi ad alto livello. Per esempio il teorema BCMP fornisce una soluzione in forma prodotto per una classe di reti di code Markoviane. In questa tesi usiamo due risultati, la proprietà $M \Rightarrow M$ e il Reversed Compound Agent Theorem (RCAT) per esplorare le relazioni tra alcune classi di modelli in forma prodotto: reti di code (QN), reti di Petri stocastiche (SPN), reti di Petri stocastiche generalizzate (GSPN), ed algebre di processo Markoviane (MPA). Inoltre identifichiamo nuove classi di GSPN in forma prodotto, e dimostriamo che precedenti risultati per SPN possono essere studiati mediante RCAT. Da un punto di vista applicativo mostriamo come sia possibile associare ad ogni rete di code multiclasse BCMP una GSPN con struttura finita che garantisca una forte relazione di equivalenza. Di conseguenza siamo in grado di definire una tecnica di analisi di modelli ibridi (cioé espressi mediante formalismi differenti) con forma prodotto per i quali la composizione e l'interazione sono formalmente specificati mediante GSPN. Infine definiamo un algoritmo in grado di trasformare una rete di code BCMP in una GSPN sfruttando i precedenti risultati teorici. L'algoritmo è modulare ed è facilmente estendibile. Sotto un profilo teorico i nuovi risultati aiutano la comprensione delle condizioni che caratterizzano i processi stocastici associati a modelli in forma prodotto, aprendo la strada ad ulteriori progressi in questo importante ambito di ricerca.

Acknowledgments

First of all I would like to thank my supervisor: Simonetta Balsamo. I am very grateful to her for several reasons. First of all, she has suggested me to start this experience and then she has supported and motivated me especially in some difficult moments. Finally, she has taught me a lot and has given me several ideas about possible topics of research in this area.

I thank my official referees, Ramon Puigjaner and Gianfranco Balbo, for their valuable feedbacks and their constructive criticism in reviewing this Thesis.

Special thanks go to Peter Harrison, for the interesting discussions we had during my visit in London. Also, I cannot forget the work done and the time spent with Maria Grazia Vigliotti during my sojourn there.

Samuel Rota Bulò has been a great colleague as well as a good friend. I have appreciated the time spent in room 15, and the scientific conversations with him.

Thanks to all my students and school colleagues that have always supported me and appreciated (at least I hope) my work.

Finally, I would like to thank my family and my friends (especially Anna, Massimo, Ivan, Mattia, Michele, Alessia, Gianni and Anna) that have been patient with me and have trusted and supported me all along this experience. I think one day they will eventually appreciate Petri Nets as I do, even if that day seems too far, by now. In particular my father and my mother have always encouraged me even in the worst moments.

During this last year I have lost too many people that I loved: Eugenio, Rinaldo and Grazia. This thesis is also dedicated to them.

Contents

List of Figures

Preface

In this thesis we present some results previously published with Simonetta Balsamo and some others not yet published and developed in cooperation with Simonetta Balsamo and Peter Harrison. I want to thank my co-authors for the ideas and the support provided for the definition of these results. In each of this works Andrea Marin has given an original e concrete contribution.

This thesis consists of two parts. The first part introduces the formalisms used in the following and the main results on their product-forms are presented. The second part illustrates the contributions of this work.

Chapter 1 introduces the Markovian stochastic models. We formally define what we mean for product-form models by giving a definition at the continuous time Markov chain level. We also introduce some results on product-form solutions for interacting Markov chains. Chapter 2 briefly recalls the Markovian queueing network (QN) models and presents the main theorem on product-form QNs, i.e., BCMP theorem. Moreover we illustrate the properties on the queueing model that imply the BCMP product-form. These properties can be expressed in terms of a characterization of the scheduling discipline or in terms of properties of the underlying CTMC. Special attention is devoted to the $M \Rightarrow M$ property. The review is based on the published paper [12]. Chapter 3 introduces Stochastic Petri Nets and Generalized Stochastic Petri nets and illustrates the results on product-forms for these formalisms. The last introductory part is Chapter 4 that briefly illustrates Markovian process algebra and the main results for the product-forms. In this chapter a special attention is devoted to the presentation of RCAT and ERCAT theorems that are widely applied in the following chapters.

The second part of the thesis illustrates the original contributions. Chapter 5 presents a yet unpublished result obtained by a joint work with Peter Harrison. A class of well-known SPNs in product-form is proven to be in product-form also by RCAT. The consequences of this relation are twofold. First we define a new approach to find SPN product-form solutions for a class of SPNs. Then we obtain an improvement of the modularity of traditional product-form SPNs. For example, we can study two product-form SPNs in isolation and then combine them obtaining a new product-form model. The analysis of the latter model is based on the solutions of its sub-models. We also prove that we can combine a product-form SPN belonging to this class with other formalisms in product-form by RCAT (e.g. G-networks). In this chapter we show several examples of analysis of product-form SPNs pointing out the differences of the technique based on RCAT and the one based on the well-known results. We also show some examples of SPN model compositions in product-form, but we refer to Chapter 7 for an application within a hybrid formalism.

In Chapter 6 we define a GSPN model corresponding to each BCMP station queueing discipline. We define and prove a strong equivalence relation between the GSPN models and the corresponding BCMP queueing stations. In this chapter we assume the system in isolation, under class independent Poisson arrivals, while in Chapter 7 we show that this equivalence holds also within product-form queueing networks. The results presented in this chapter have been published in [15, 16].

Chapter 7 shows that the underlying idea of this thesis allows for the analysis of product-form hybrid models. Basically, we compose models (even expressed in different formalisms) if they fulfill the $M \Rightarrow M$ property or they satisfy RCAT conditions. In this chapter we show that the models defined in Chapter 3 satisfy both $M \Rightarrow M$ property and RCAT conditions (with some restrictions). We illustrate several examples of hybrid modelling and their corresponding GSPN models. We also provide a result on the relation between the fairness of the probabilistic queueing disciplines with exponential servers without preemption and the $M \Rightarrow M$ property. In other words we prove that such a queueing discipline satisfies $M \Rightarrow M$ property only if every customer in the queue has the same probability of entering in service after a job completion. Some of the results illustrated in this Chapter are published in [15, 16] and other ones have been accepted for publication.

The last part showing original results is Chapter 8. Here we define an algorithm that transforms a BCMP QN into a GSPN. The properties of the algorithm an its computational complexity are studied. The contents of the chapter are based on the published work [13].

For the sake of readability Appendix A provides the proves of most of the theorems enunciated in Chapters 6 and 7. Appendix B shows some calculations on the examples illustrated in Chapter 5.

Finally, we present the conclusions of the work.

Introduction

Stochastic models have been widely used to derive both qualitative and quantitative properties of artificial and natural systems. In this thesis we mainly focus on stochastic models for computer systems such as hardware or software architectures or communication protocols. Stochastic models are based on the concept of stochastic process. A stochastic process is a family of random variables X_t where t is a parameter running over a suitable index set T, called time. The set of values that can be assumed by X_t (its domain) is the state space of the stochastic process. The wide application field of stochastic models derives from several reasons. First of all, under appropriate assumptions, deterministic systems can be analyzed using stochastic models. The main idea of this approach is that the probabilistic behavior of the model can be used in order to make up for a lack of knowledge (see [116] for a discussion of this topic). A second reason of the luck of stochastic model analysis is the recent theoretical advances in this filed joint with a high availability of powerful computers (or networks of computers) that can perform fast and precise calculations. From a theoretic point of view, Markov Processes play a pivotal role in this framework (see Chapter 1 for a brief review). Andrey Markov (1856-1922) introduced the concept of Markov Process, i.e., a stochastic process that yields the Markov property. Informally, the Markov property states that, given the present state of a model, future states are independent of the past states. In other words the description of the present state catches all the information needed to predict the future states of the stochastic process. A Markov process whose state space is a discrete set is called Markov Chain. If the time set T is discrete we have a discrete time Markov chain (DTMC) whereas if the time set T is continuous we have a continuous time Markov chain (CTMC).

Other important stochastic processes that are strictly related to the Markov processes are the semi-Markov processes and the Markov reward processes whose analysis is based on the renewal theory. For a presentation of these processes see [105]. The main strength of these processes is that the residence time in a state has not to be exponentially distributed and, in case of Markov reward processes a set of metrics can be associated with states or state transitions.

Although simple systems such as simple communication protocols can be described in terms of CTMCs or DTMCs (see for example [37]), this can quickly become a difficult task in case of complex systems with several interacting components. Specifically, the computational complexity of solving a stochastic process grows with the number of process states. This number can exponentially depend on the number of components of the system modeled by the process. In order to overcome this problem, a set of high level formalisms has been defined. Thanks to

these formalisms the modeler can describe a complex system by a relatively compact model. However, in order to perform a qualitative or quantitative analysis, in general, one has to derive and analyze the stochastic process associated with the high level model. Then the results obtained by this low level analysis have to be interpreted in terms of the high level model components. A representative example is the analysis of a queueing system [77, 38]. Queueing systems are used to represent a system in which a finite or infinite set of customers compete for a set of resources. Customers arrive to the queueing center according to a stochastic process, compete for the resources according to a queueing discipline, and finally are served and the service time is usually represented by a random variable. Customers can be all identical (single class queue) or clustered into classes (multiclass queue). The analysis of a model expressed with this formalism just needs to specify the model parameters (i.e. the arrival processes per class, the service time distribution, the number of resources, the queueing discipline). This high level definition makes the model more compact than the one given by directly specifying the stochastic process of the system. However, the stochastic process underlying even a relatively simple model can be very complex with a high number of states. Several exact results are known for some types of high level models, especially in queueing theory [77].

In this thesis we focus on the following aspects:

- We just consider stochastic models whose stochastic processes are CTMCs. This include a wide class of high level formalisms, such as stochastic Petri nets [92], generalized stochastic Petri nets [89], Markovian queueing networks [22, 74, 77, 12], Markovian Process Algebras [67, 20, 68].

- We focus on the analysis of the long-term behavior of the model. This means that we are interested in studying the stochastic process of the model when $t \to \infty$. Under certain conditions some models exhibit a stationary behavior when $t \to \infty$, i.e., the probability of observing a given state becomes stable and independent of the initial model state.

Dealing with models with large state spaces. In many practical cases a system consists of several components that interact in some ways. A good modeling technique is defining the sub-models of the system, each of which corresponds to a system component. Then the modeler defines how the sub-models interact. Even if this approach has many strengths, it puts in evidence one of the main limits of stochastic modeling. In fact, even if we assume that each sub-model has its own stochastic process, it is obvious that this stochastic process can depend on the states of the other sub-models. Therefore the joint process has a potential number of states given by the product of all the possible states of each stochastic process generated by the sub-models. Let us call $M(t)$ the state of the model at time t, and $M_1(t), \ldots, M_N(t)$ the state of the sub-models at the same time. In the trivial case the sub-models are independent, therefore the probability of observing a joint

state $\mathbf{m} = (m_1, \ldots, m_N)$ at time t is simply given by $\Pr\{M(t) = m\} = \Pr\{(M_1(t) = m_1\} \cdots Pr\{M_N(t) = m_N\}$. However, this case is not very interesting because the sub-models could be studied in isolation. On the other side there are also examples of models that consist of non-independent sub-models, but the following relation holds:

$$t \to \infty \quad \Pr\{M(t) = m\} \propto \; g_1(m_1) \cdots g_N(m_N). \tag{1}$$

In other words, the steady state probabilities of the joint process can be calculated as a normalized product of the steady state probabilities of the sub-models with an appropriate parameterization defined by function g_i. We say that the models whose steady state distributions can be expressed by (1) are models in product-form, or with a product-form solution. Let us consider the Jackson queueing network model class. Informally a Jackson queueing network consists of several queueing centers, with negative exponential distributed service times, external arrivals according to a Poisson process, and a probabilistic routing among the queueing centers. It has been proved in [73] that this class of Markovian queueing networks is in product-form, i.e., a pseudo arrival rate for each service center can be determined by the analysis of the routing of the network, and then the steady state probabilities can be calculated as the product of the steady state probabilities of each queueing center parameterized with the pseudo arrival rate and considered in isolation.

Product-form models play an important role in stochastic modeling especially when an exact analysis of the steady state is desired. This is due to at least two reasons:

- In the general case, obtaining a steady state distribution given a CTMC of n states requires $\mathcal{O}(n^3)$ operations. It is clear that the analysis of a model considering the sub-models is much more efficient.
- Some product-form models fulfill a set of interesting properties that allow the definition of algorithms that can calculate the desired performance indices of the model very efficiently. A number of these algorithms have been defined for product-form queueing networks (see [12] for a review) and product-form stochastic Petri nets ([39, 111]). Another property of product-form queueing networks is known as the Norton's theorem that allows for a hierarchical composition of product-form queueing networks that preserves the product-form property.

Motivations. The work presented in this thesis aims to explore the relations among the classes of product-form models defined in the literature. In fact, as already explained, although it is interesting to interpret the product-form conditions at the high level formalism in which a model is defined, the product-form property is related to the underlying CTMC. Let us make an example. One of the main results for product-form queueing networks is the BCMP theorem [17]. The theorem states a product-form solutions for queueing networks in which the queueing centers have

specific scheduling disciplines. It is worthwhile understanding how these disciplines influence the CTMC of the model in order to satisfy the product-form properties. Many authors worked in this direction producing several results [76, 94, 32, 98, 33]. More recently some new surprising results about product-forms and new formalisms appeared in literature. For example in [63, 39] product-form conditions and solution have been defined for stochastic Petri nets, in [6] product-form is studied for generalized stochastic Petri Nets, and in [70, 69, 110, 57, 59, 61, 62] product-form is studied for PEPA models. Moreover, new results concerning the properties of CTMCs in product-form have been recently published. For years the only well-known product-forms were characterized by local balance, quasi reversibility and linear traffic equations. These properties are described in Chapter 2. However, the definition of product-form G-networks [50] showed an example in which even if these conditions do not hold the CTMC is in product-form. The results presented in [59, 61, 62] confirmed this idea and generalized it.

From a theoretical point of view, we try to interpret most of these results under the viewpoint of a unique formalism, i.e., the generalized stochastic Petri nets. The choice of this formalism is due to its high expressivity, rigorous semantic, and the large availability of analysis and simulation tools. As a practical consequence, we aim to define a theoretical framework for a hybrid formalism definition in which product-forms can be identified.

Main contributions. In this thesis we mainly focus on the relations among the product-form model classes identified by BCMP queueing networks [17], Coleman, Henderson et al. stochastic Petri nets [63, 39]. In particular we consider the product-form composition of models based on two results. The first one is the $M \Rightarrow M$ property defined by Muntz in [94] for the queueing networks. Informally, a multiclass queueing center fulfills the $M \Rightarrow M$ property if under class independent Poisson arrivals, it exhibits class independent Poisson departures (see Chapter 2 for a review). Moreover, in [94] the author proves that a network of queueing centers that fulfill the $M \Rightarrow M$ property, with probabilistic routing, open or closed, has a product-form solution. Note that even if external arrivals to a network of queues occur according to a Poisson process, it is not true that internal arrivals to the service centers are Poisson processes if in the net has cycles. In order to apply this result to other formalisms than queueing networks, we must give a correct and meaningful interpretation of *arrival process* or *departure process*. In fact GSPNs do not implement the notion of customers but the tokens that can represent either customers or resources. The other result that we widely use in the thesis is reversed compound agent theorem (RCAT) proved in [59]. Although this theorem is defined in terms of cooperation of PEPA agents, it can be straightforwardly used to study other formalisms. Chapter 4 reviews this theorem and its extensions [61, 62]. Let us informally get the intuition of RCAT theorem. As we have said before, two interacting processes are considered in product-form if the steady state probabilities of the joint process can be expressed

as product of the steady state probabilities of the basic processes appropriately parameterized. For instance, consider a tandem of exponential queues as shown by Figure 1. It is known that the model is in product-form. However, if the first queue

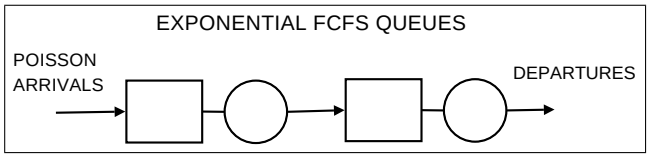

Figure 1: A tandem of two FCFS queues with single server, and exponentially distributed service time.

(the one with the external arrivals) can be studied in isolation because its stochastic process is not influenced by the process of the second queue, this cannot be said for the second queue. In fact, some transitions in the stochastic process of the second queue considered in isolation have unknown rates, because they are synchronized with the departures of the customers from the first queue. In this case, the parameterization of the second queue requires to determine these missing rates. RCAT derives this parameterization by the analysis of the reversed processes in isolation. In the specific case of the tandem queues, the missing parameter is the reversed rate of the transitions corresponding to a job completion in the stochastic process of the first queue. Burke's theorem [28] states the departure process from the first queue is a Poisson process with rate equals to the arrival process rate. This can be used as a parameter for studying the second queue in isolation. Note that in the birth and death process (see Chapter 1 for a brief review) of the first queue, the reversed rate of the transitions corresponding to a job completion is exactly the arrival rate.

In this thesis we show and prove the following results:

- We show that the well-know class of product-form stochastic Petri nets defined in [63, 39], with some restrictions, can be studied by RCAT theorem and its extensions. On one hand this result establishes a clear relation between the product-form for stochastic Petri nets and the one identified by RCAT. On the other hand there are at least three practical consequences of this result. First, the class of stochastic Petri nets in product-form by RCAT can be composed with any other model satisfying the same theorem. This is a big step toward a hybrid modeling formalism in which product-form solutions can be studied. In fact, RCAT has been successfully used to study exponential queues [59], BCMP queueing networks [58], networks with negative customers in product-form (G-Networks) [60] and other models.

  We show that the compositionality property of the stochastic Petri nets defined by Coleman, Henderson et al., results really enhanced. In fact the composition of two models obtained by RCAT can still be studied by RCAT.

  Finally, we note that also the hierarchical modeling results enhanced by the RCAT approach. In fact, one can build a model consisting of several sub-

models each of which satisfying RCAT conditions. Then the whole model satisfies RCAT conditions itself.

Using RCAT, the technique to find the product-form solution is different from the well-known one presented in [39]. Indeed, we start the net analysis by verifying the structural conditions, and then we build a partition of the net places in order to decompose it in several building blocks. The building blocks have a straightforward solution by RCAT and then the whole net is studied as composition of the building blocks.

- We define a set of GSPN models each of which corresponds to a multiclass BCMP station type. Note that this is not a trivial operation because multiclass queueing networks cannot be studied as state machines. In fact the queueing disciplines influence the steady state probabilities and the conditions for the product-form [17, 32]. Roughly speaking, this means that it is not sufficient to count the number of customers in the station in order to have an equivalent GSPN model. This problem has already been studied in [7] where GSPN models whose stochastic processes are isomorphic to the correspondent BCMP stations are defined. This ensures the equivalence but unluckily the models have an infinite structure (for open systems) in order to represent the queueing discipline. In our approach we relax the equivalence terms but we propose to model BCMP station types with finite structured GSPN models. The defined equivalence terms preserve the average performance indices of the models.

  It is worthwhile noting that these GSPN models do not belong to any of the well-known GSPN product-form classes.

- The next step studies how these models can be combined maintaining a product-form solution. In order to achieve this we show that:

  - The $M \Rightarrow M$ property holds for the defined GSPN models.
  - RCAT theorem conditions hold for the GSPN models.

  Therefore we can use them in a wide class of models. The $M \Rightarrow M$ property allows one to compose these models with other ones for which the $M \Rightarrow M$ holds. We explain the exact terms of this composition in the following parts of the thesis. Basically, these compositions of product-form models originate a BCMP-like product-form, i.e., with a linear system of traffic equations and the whole stochastic process is quasi-reversible. RCAT based compositions allow for the hybrid modeling discussed above, i.e., we can compose the defined GSPN models with other product-form models such as G-networks or Coleman, Henderson et al. product-form stochastic Petri nets without batch token movements and with state independent transition rates. In this context we also show some examples of GSPN non product-form models that can be

approximated using RCAT by a product-form model. In fact, we take advantage of the fact that RCAT does not use the global balance equations to decide whether the composition of two models is in product-form, but it requires some conditions concerning the structures of the CTMCs. Therefore if these structural conditions are not satisfied one can think to modify the model behavior in order to meet RCAT requirements. In this way, the resulting modified model can be seen as a product-form approximation of the original one. However, we will see that giving exact bounds for this approximation is not an easy task.

- Finally, we define an algorithm that translates a BCMP queueing network into an equivalent (in the sense specified above) GSPN model. The algorithm is defined such that modularity is really enhanced. In particular we use interfaces in order to identify the places of a GSPN model where external tokens arrive to or where internal tokens depart from. This algorithm can be a base to develop a tool that translates hybrid models in product-form into a GSPN in product-form.

I

Stochastic models in product-form: formalisms and state of the art

1

Stochastic models

> We have knowledge of the past, but we can't control it. We can control the future, but we have no knowledge of it.
>
> Claude Shannon

1.1 Introduction

In this chapter we present a brief introduction to stochastic models. A stochastic model is characterized by a stochastic process. By studying this process the analyst can derive a set of model properties such as liveness, performance parameters and model checking issues. In this chapter we mainly focus on Markov stochastic processes.

A stochastic process is a set of random variables $\{X(t)|t \in T\}$ defined over the same probability space and indexed by the parameter t, called time. The process random variables take values in set Γ, called state space of the process. Both set T and state space Γ can be either discrete or continuous. The process is called continuous-time or discrete-time if the time parameter t is continuous or discrete, respectively. A discrete-time process is usually denoted by $\{X_n|n \in T\}$. If the state space Γ is discrete then the process is called discrete-space or chain, otherwise the process is called continuous-space. The probabilistic behavior of a stochastic process is defined by the joint probability distribution function of the random variables $X(t_i)$ for any set of times $t_i \in T, 1 \leq i \leq n, n \geq 1$, denoted by $Pr\{X(t_1) \leq x_1; X(t_2) \leq x_2; \ldots; X(t_n) \leq x_n\}$, where $x_i \in \Gamma$.

1.2 Markovian stochastic models

Hereafter we consider discrete-space Markov processes, also called Markov chains. Let us define a discrete-time Markov chain. A discrete-time Markov process is a process whose state at step $n+1$ only depends on the state probability at step n and

is independent of the previous history. This is known as the Markov property. The conditional probability distribution of the process satisfies the following condition:

$$Pr\{X_{n+1} = j | X_0 = i_0; X_1 = i_1; \ldots; X_n = i_n\} = Pr\{X_{n+1} = j | X_n = i_n\}, \quad (1.1)$$

for all $n > 0$, and $j, i_0, i_1, \ldots, i_n \in \Gamma$.
Similarly, a continuous-time process is said to be a Markov process if it satisfies the following condition:

$$Pr\{X(t) = j | X(t_0) = i_0; X(t_1) = i_1; \ldots; X(t_n) = i_n\} = Pr\{X(t) = j | X(t_n) = i_n\}, \quad (1.2)$$

for all set of times $t_0 < t_1 < \ldots < t_n < t$, and $n > 0$, $j, i_0, i_1, \ldots, i_n \in \Gamma$. Note that, because of the Markov property, the residence time of the process in each state is distributed according to either the geometric or the negative exponential distribution respectively for discrete-time or continuous-time Markov processes. If the one-step conditional probability on the right-hand side of formula (1.1) is independent of time n, then the Markov chain is homogeneous. Then, we define the transition probability from state i to state j as $p_{ij} = Pr\{X_{n+1} = j | X_n = i\}$, and the matrix of state transition probabilities $\mathbf{P} = [p_{ij}]$, where $p_{ij} \in [0, 1]$, $\sum_j p_{ij} = 1$, $\forall i, j \in \Gamma$. The stationary behavior of the Markov process can be evaluated if the process satisfies some conditions. Informally, a Markov process is said to be irreducible if every state can be reached from any other state. Each state can be transient or recurrent, and it is said to be positive recurrent if the average return time to the state is finite. An ergodic Markov chain is irreducible and formed by positively recurrent aperiodic states. Let $\boldsymbol{\pi} = [\pi_0 \pi_1 \pi_2 \ldots]$ denote the stationary state probability vector, where $\pi_j = Pr\{X = j\}$ is the stationary probability of state $j \in \Gamma$. Then for homogeneous ergodic discrete-time Markov chains we can compute the stationary probabilities $\boldsymbol{\pi}$ as follows [77]:

$$\boldsymbol{\pi} = \boldsymbol{\pi}\mathbf{P}, \quad (1.3)$$

with the normalizing condition $\sum_j \pi_j = 1$. This is called system of global balance equations.

Let us now consider a continuous-time Markov chain . The Markov chain is homogeneous if the conditioned probability on the right-hand side of formula (1.2) is independent of time t_n, but only depends on the interval width $(t - t_n)$. In other words we can write the state transition probability from state i to state j only depending on the interval width s as follows:

$$p_{ij}(s) = Pr\{X(t_n + s) = j | X(t_n) = i\}, \quad \forall i, j \in \Gamma, \forall t_n \geq 0.$$

Hence, we have a width dependent state transition probability matrix $\mathbf{P}(s) = [p_{ij}(s)]$. Then we can define a rate transition probability matrix $\mathbf{Q} = [q_{ij}], i, j \in \Gamma$, also called process infinitesimal generator, as follows:

$$\mathbf{Q} = \lim_{s \to 0} \frac{\mathbf{P}(s) - \mathbf{I}}{s}.$$

The stationary behavior of the continuous-time Markov chain can be evaluated for homogeneous ergodic chain. The stationary state probabilities $\boldsymbol{\pi} = [\pi_0 \pi_1 \pi_2 \ldots]$, where $\pi_j = Pr\{X = j\}$ for each state $j \in \Gamma$, can be computed by solving the following system of global balance equations:

$$\boldsymbol{\pi}\mathbf{Q} = \mathbf{0}, \tag{1.4}$$

with the normalizing condition $\sum_j \pi_j = 1$.

For the special classes of birth and death processes it is possible to derive a closed-form solution of the stationary state probability $\boldsymbol{\pi}$ defined by system (1.3) for discrete-time and system (1.4) for continuous-time processes, respectively. A birth and death Markov process has state space $\Gamma = \mathbb{N}$ and the only non-zero state transitions are those from any state i to states $i-1$, i, $i+1$, $\forall i \in \Gamma \setminus \{0\}$, and only to state 1 from state 0. Hence, the transition state probability matrix $\mathbf{P}$ for discrete-time, or the transition rate matrix $\mathbf{Q}$ for continuous-time process, is tridiagonal. Let us denote the rates of matrix $\mathbf{Q}$ for a continuous-time birth and death Markov chain as follows: $q_{i\ i+1} = \lambda_i$, $i \geq 0$ and $q_{i\ i-1} = \mu_i$, $i \geq 1$. Then the stationary state probability $\boldsymbol{\pi}$ can be calculated as follows:

$$\pi_i = \pi_0 \prod_{j=0}^{i-1} \frac{\lambda_j}{\mu_{j+1}}, \tag{1.5}$$

for $i \geq 0$, and where π_0 is given by the normalizing condition, i.e.,

$$\pi_0 = \Big[\sum_{i=0}^{\infty} \prod_{j=0}^{i-1} \frac{\lambda_j}{\mu_{j+1}} \Big]^{-1}.$$

This solution holds under the stability condition. A sufficient condition for the stationary solution is that there exists a state $k_0 > 0 : \lambda_k < \mu_k$, $\forall k > k_0$ [77]. Several basic queueing systems can be analyzed by birth and death Markov processes.

Continuous Time Markov Chains (CTMC) have been widely used in the performance evaluation field in order to derive several model properties (for example [77, 76, 116, 45, 38]).

1.3 Product-form interacting Markov chains

Product-form is a property strictly related to the CTMC underlying a stochastic model. Therefore, every product-form class can be seen in terms of interacting Markov chains. However, we usually distinguish the product-form model classes according to the formalism that is used to define the model. This is because one tries to characterize a product-form model class in terms of structural conditions or high-level concepts that are easier to understand by a modeler. For example, in queueing theory it comes natural to express the product-form conditions in terms of

the occupancy of the queueing station or its arrival process, and so on. In general, these concepts cannot straightforwardly be used for different formalisms (what is the number of customers of a given class in a Markov chain?). As a consequence, in classifying a product-form model we consider the formalism under which the conditions (and the solutions) are expressed. One of the aims of this thesis is to investigate the relations among the CTMCs of these product-form classes.

In this section we review the product-form conditions for models expressed in terms of interacting Markov chains. Let us formally introduce the problem. We consider a set of N CTMCs with state spaces $\Gamma_1, \ldots, \Gamma_N$. We can define an interaction among them such that the joint process is still a CTMC and its state space Γ is $\Gamma \subseteq \Gamma_1 \times \ldots \Gamma_N$. Let $\pi_i(\gamma_i)$ be the steady state probability of state $\gamma_i \in \Gamma_i$ with $1 \leq i \leq N$. Then we say that the CTMCs are in product-form if for all $\gamma \in \Gamma$, where $\gamma = (\gamma_1, \ldots, \gamma_N)$, we have that:

$$\pi(\gamma) = \frac{1}{G} \prod_{i=1}^{N} \pi_i(\gamma_i), \tag{1.6}$$

where G is the normalizing constant.

1.3.1 Boucherie's product-form

Boucherie's product-form results are presented in [25]. The author introduces the theory in the context of competing Markov chains and then he shows its application to stochastic Petri nets as a generalization of Lazar-Robertazzi results [84]. We briefly review Boucherie's theory on competing Markov chains. Let us consider a set of K competing CTMCs with spaces $\Gamma_1, \ldots, \Gamma_K$ and transition rates $q_k(\gamma, \epsilon)$ with $\gamma, \epsilon \in \Gamma_k$. Let I be an index set. For each k, let A_{ki}, $i \in I$, be a set of mutually exclusive sets such that:

- $A_{ki} \neq \emptyset$,
- $A_{ki} \subset \Gamma_k$,
- $\bigcup_{i \in I} A_{ki} = \Gamma_k$.

Suppose that the k-th CTMC uses the resource $i \in I$ in the state γ, then according to this notation we have that $\gamma \in A_{ki}$. We say that CTMCs k_1 and k_2 compete over resource $i \in I$ if in the joint state space there is not any state in which k_1 and k_2 can simultaneously be in states γ_{k_1} and γ_{k_2}, where both these states use the resource $i \in I$. In other words the set $\{(\gamma_{k_1}, \gamma_{k_2}) : \gamma_{k_1} \in A_{k_1 i}, \gamma_{k_2} \in A_{k_2 i}\}$ is empty. Let C_{ki} denote the set of CTMCs that compete with CTMC k on the resource $i \in I$.

The transition rates of the joint process with state space $\Gamma = \prod_{k=1}^{K} \Gamma_k$ are defined as follows:

$$q(\gamma, \gamma') = \sum_{k=1}^{K} q_k(\gamma_k, \gamma_{k'}) \prod_{\substack{r=1 \\ r \neq k}}^{K} \mathbf{1}(\gamma_r = \gamma_{r'}) \mathbf{1}(\text{if } \gamma_r \in A_{ri} \text{ then } k \notin C_{ri}), \tag{1.7}$$

where $\gamma = (\gamma_1, \ldots, \gamma_K)$, $\gamma' = (\gamma'_1, \ldots, \gamma'_k)$ and $\mathbf{1}(\cdot)$ is the indicator function.

In this framework the CTMCs compete over a set of resources labelled by a set I. In every joint state $\gamma \in \Gamma$ there cannot be two CTMCs using the same shared resource. So we can partition a state space Γ_k into I sets A_{ki} such that $\cup_{i\in I} A_{ki} = \Gamma_k$ and $A_{ki} \neq \emptyset$. If CTMCs k_1 and k_2 compete on resource i then in the state space Γ of the joint CTMC there cannot be a state γ where components $\gamma_{k_1} \in A_{k_1 i}$ and $\gamma_{k_2} \in A_{k_2 i}$.

Under these assumptions it can be proved [25] that the steady state probabilities are in product-form. Note that condition (1.7) is such that:

- Given two adjacent states $\gamma, \gamma' \in \Gamma$, then just a single CTMC k changes its state, i.e., states γ and γ' differ just by their k-th component which changes from γ_k to γ'_k. The transition rate from γ to γ' is determined only by the transition rate between the states γ_k and γ'_k of the k-th CTMC.

- Suppose that CTMCs k and k' conflict on resource $i \in I$. Then, if one of the two CTMCs is in a state belonging to the set A_{ki} (respectively $A_{k'i}$) then the other CTMC cannot change its state at all. This probably is the main limit of this product-form class.

Example 1 *Let us consider two CTMCs with state spaces* $\Gamma_1 = \{a_1, b_1, c_1\}$ *and* $\Gamma_2 = \{a_2, b_2, c_2\}$. *The transition rate matrices* $\mathbf{Q}_1$ *and* $\mathbf{Q}_2$ *are:*

$$\mathbf{Q}_1 = \mathbf{Q}_2 = \begin{bmatrix} 0 & v & 0 \\ 0 & 0 & v \\ v & 0 & 0 \end{bmatrix},$$

with v *a positive real constant. The state transition diagram is depicted in Figure 1.1 for* $i = 1, 2$.

If we consider the two processes in isolation, their stationary probabilities are $\pi_i(\gamma_i) = \frac{1}{3}$ *for* $\gamma_i = a_i, b_i, c_i$ *and* $i = 1, 2$. *Let us now define the competition. Suppose* $I = \{1, 2\}$ *and* $A_{k1} = \{a_k, c_k\}$, $A_{k2} = \{b_k\}$ *and* $C_{12} = \{2\}$, $C_{22} = \{1\}$, $C_{k1} = \emptyset$, *with* $k = 1, 2$. *In Figure 1.1 we use white color to fill the states which use resource* 1 *and grey for the state which uses resource* 2. *Note that competition is just over resource* 2.

The joint process of $\Gamma = \Gamma_1 \times \Gamma_2$ *with the transition rates defined by Formula (1.7) has* 8 *reachable states. We can conclude that the stationary probability of each state is the same, i.e.,* $\frac{1}{8}$. *Figure 1.2 shows the composed process* S.

It is easy to verify that $\pi(\gamma) = \frac{1}{8}$ *for all* γ *satisfies the set of GBEs for the CTMC on space* Γ.

Boucherie's product-form condition is very interesting for various reasons. First of all, it gives the conditions directly on the Markov Chains so the result can be used as framework for studying the product-form conditions for different stochastic model

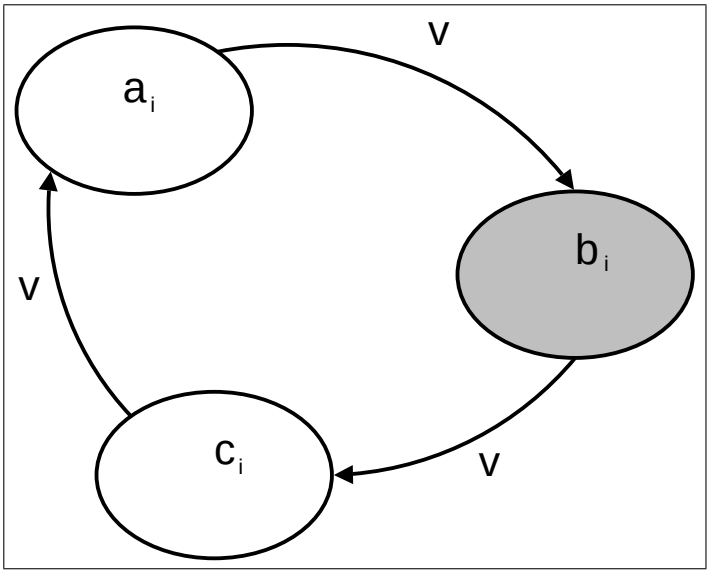

Figure 1.1: Example of CTMC

formalisms, such as stochastic process algebra and stochastic Petri nets (SPN). If we apply Boucherie's approach to SPNs, as pointed out in the original work [25], we note a second strength of the result. In fact, if we consider Coleman et al. product-form for SPNs [63, 39] the authors aim to give a stationary solution which is expressed in terms of product of functions of the number of tokens in each place. Boucherie's framework, on the other hand, is more suitable for hierarchical modelling than that presented in [63, 39]. In fact, the idea is that given a set of SPNs (let us call them agent), with their underlying CTMCs, one can define a set of sufficient conditions for the product-form solution on the interaction of the agents. However, given a SPN, an automatic identification of Boucherie's product-form is not an easy task.

It is worthwhile exploring the practical consequences of Boucherie's conditions for the product-form. In the following we comment them one by one:

- The conditions on the single CTMC. These are very general, indeed it is just asked them to be ergodic and to have a stationary distribution.

- The conditions on the composed process state space. The exclusion mechanism appears to be very flexible. Defining a partition on the state space is a really general approach. In fact, several interacting systems can be represented by this approach. In the original paper the author gives an example where he shows how the exclusion mechanism can model the availability of multiple resources.

- The conditions on the composed process transition rates. We think that this is a key-condition for the product-form and we focus on its meaning. Suppose

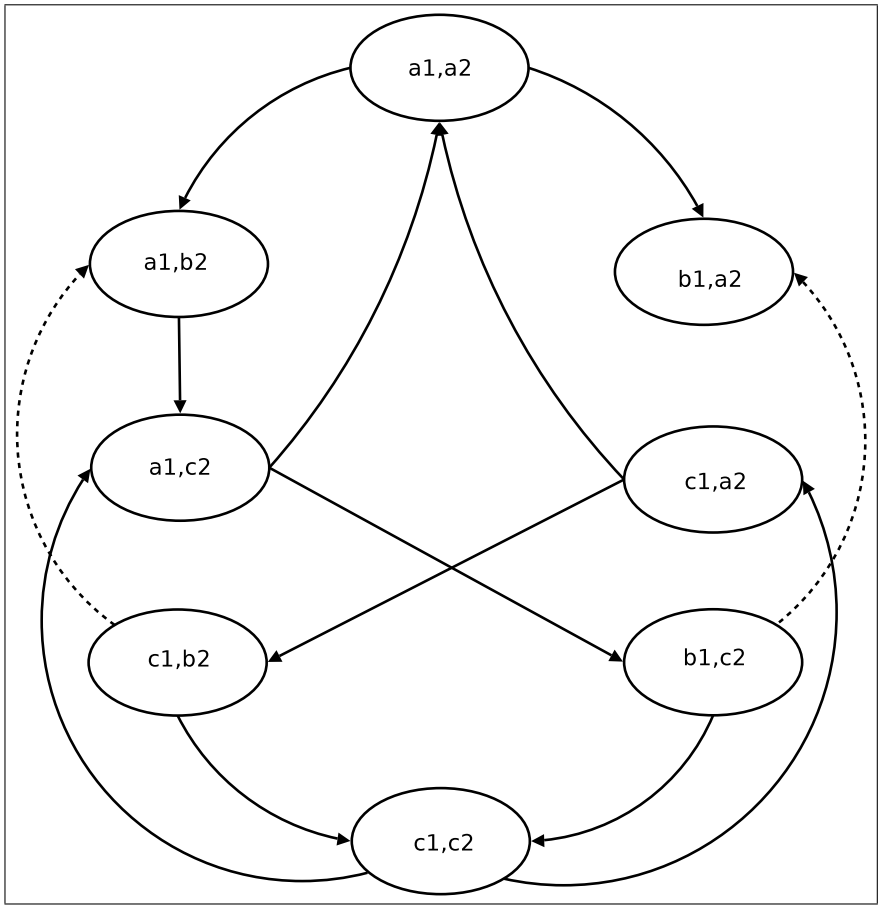

Figure 1.2: Composed process with state space Γ. In Boucherie's definition the dotted lines are not present, however, as discussed in the conclusions, for many practical cases they should be modeled. All the transition rates are equal to $v > 0$.

that CTMCs k, k' conflict on resource i. Then when one of the two CTMCs is in a state belonging to the set A_{ki} or $A_{k'i}$ the other Markov chain cannot change its state at all. The following example aims to illustrate by a practical case of study the Bocuherie product-form requirements.

Example 2 *Let us consider two identical processes which perform the following tasks:*

$$\underbrace{think}_{a} \rightarrow \underbrace{print}_{b} \rightarrow \underbrace{input\ data}_{c} \rightarrow think \rightarrow \cdots$$

The printing state requires to use of the shared resource printer *so there is a competition on that resource. This is basically the CTMC of Example 1 where it is assumed that all the tasks have an exponential distributed duration with the same mean. The point is that Boucherie's framework requires that when a process is printing the other*

one is stopped, independently of the state it is in. There are no reasons, from a modelling point of view, for stopping a process which is performing an input action when the other process is printing. In Figure 1.2, by adding the dotted arcs, one obtains the process how it should be *from a modelling point of view. But the Boucherie's product-form is lost.*

1.3.2 Stochastic Automata Networks in product-form

In this paragraph we briefly recall product-form stochastic automata networks. As we are not using these models in the rest of this work, we just cite the main results in this field.

Stochastic Automata Networks (SAN) are introduced in [100] to model distributed parallel algorithms. SANs use a very low level formalism in which every component is an annotated Markov Chain, therefore all the well-known product-forms can be mapped into this formalism. A SAN consists of a number of individual stochastic automata that operate more or less independently one of each other [100]. Every automaton of the SAN is characterized by a state that changes on the time t. The state of the SAN is given by the state of every stochastic automaton that it consists of.

The automata of a SAN can interact in two ways:

1. The rate at which a transition occurs in an automaton can depend on the state of a set of automata of the SAN.

2. A transition in one automaton can force a transition to occur in one ore more other automata.

Note that it is possible to represent these interaction types by PEPA, so we cover this topics in the following chapters.

SANs product-form conditions are studied for example in [47, 48] in the case of interactions of type 1.

1.4 Conclusions

In this chapter we have recalled the main definitions of Markov Chains both in discrete-time (DTMC) and in continuous-time (CTMC). Moreover, we have introduced the concept of interacting Markov Chains and the definition of product-form. It is worthwhile pointing out that the product-form is a property that is defined on the CTMC underlying a model, whatever formalism is used to define it. However, working at the CTMCs level has some limitations. First of all the conditions can be hard to be interpreted semantically. For example, if we consider a queueing model (formally introduced in the next chapter) we expect that the conditions for the product-form are expressed in terms of queueing theory concepts, such as customers, arrival events, queueing disciplines and so on. Understanding a condition

on the CTMC at a higher level can be a real hard task for a modeler. However, from a theoretical point of view, it can be useful to understand the product-form conditions expressed for higher level formalisms in terms of properties of the associated CTMC. This analysis allows us to study the interaction of several product-form model classes defined for different formalisms. In the following chapters we show that, under appropriate conditions, product-form models expressed by different formalisms can be combined originating a hybrid modeling formalism that preserves the product-form property of its components.

2

Queueing networks

2.1 Introduction

Queueing network models have been extensively applied to represent and analyze resource sharing systems, such as production, communication and computer systems. They have proved to be a powerful and versatile tool for system performance evaluation and prediction. A queueing network model (QN) is a collection of service centers representing the system resources that provide service to a collection of customers that represent the users. Customers compete for the resource service and they possibly wait to be served in the queue into the service centers, according to the queueing discipline. The analysis of QNs consists in evaluating a set of performance measures, such as resource utilization, throughput and customer response time. The dynamic behavior of a QN can be described by a set of random variables that define a stochastic process. Under some constraints on the QN, it is possible to define an associated underlying stochastic Markov process, and to compute the QN performance indices by its solution.

The popularity of QNs for system performance evaluation is due to a good balance between a relative high accuracy in the performance results and the efficiency in model analysis and evaluation. In this framework, the class of product-form networks has played a fundamental role. Product-form QNs have a simple closed form expression of the stationary state distribution that allow us to define efficient algorithms to evaluate average performance measures with polynomial time complexity in the number of model components.

QNs extend the basic queueing systems that are stochastic models first introduced to represent the entire system by a single service center. Queueing systems have been first applied to analyze congestion in telephonic systems and then to study congestion in computer and communication systems [77, 53, 81, 118, 85, 74]. A QN represents a congestion and resource sharing systems as a network of interacting service centers whose analysis often provides quite accurate prediction of their performance. Despite of several assumptions of the class of queueing networks, they have been observed to be very robust models [115]. QNs can be analyzed by analytical methods or by simulation. Simulation is a general technique of wide application, but its main drawback is the potential high development and computational cost

to obtain accurate estimates. Moreover, interpreting simulation outcomes can be quite difficult [82]. Analytical methods require the model to satisfy a set of assumptions and constraints and are based on a set of mathematical relationships that characterize the system behavior.

Jackson [73] introduced product-form QNs for open exponential networks and Gordon and Newell [54] for closed exponential networks. They introduce several assumptions on the model characteristics and provide a simple closed-form expression of the stationary state distribution and some average performance indices. This class of models was then extended to include several interesting and useful characteristics to represent more complex system. These features include different types of customers of the networks, various queueing disciplines (i.e., the scheduling algorithms of the waiting queues), a class of state-dependent service rates, a class of state-dependent probabilistic routing among the service centers and some constraints on the population of the subnetworks. The main result concerning product-form QNs in known as the BCMP theorem and was presented by Baskett, Chandy, Muntz and Palacios [17]. It defines the well-known class of BCMP QNs with product-form solution for open, closed or mixed models with multiple classes of customers, various service disciplines and service time distributions and some types of load-dependent functions for the arrival process and the customers service time. The stationary state distribution is expressed as the product of the distributions of the single service centers with appropriate parameters and, for closed networks, with a normalizing constant.

Various computational algorithms can be applied to analyze and to evaluate the performance indices of product-form QNs. The relevance of these solution algorithms is twofold. First, they provide a powerful tool in the efficient analysis of large QN models, and the analyst can choose the appropriate and most convenient algorithm depending on the type of model. Second, these algorithms provide a basis for approximate solution methods of more general network models with and without product-form. The most relevant solution algorithms for closed networks are the Convolution Algorithm [29] and the Mean Value Analysis [103]. They provide the evaluation of average performance indices with a polynomial space and time computational complexity in the network dimension that is the number of service centers and the network population. Product-form networks with multiple classes of customers are more difficult to analyze. Various types of customers define the customer classes in the network that are gathered into chains. Both Convolution and MVA algorithms have been extended to multiclass networks [101, 107, 79, 26], but their cost grows exponentially with the number of customer classes or chains. Other algorithms for multiclass QNs have been proposed. The tree Convolution and tree MVA algorithms for multiple chain networks are based on a tree data structure to optimize the algorithm computation [80, 119, 72]. Other algorithms for multiple chain QNs with several types of customers are Recursion by Chain Algorithm (RECAL) [41, 42], Mean Value Analysis by Chain [40] and Distribution Analysis by Chain (DAC) [44]. Their computational complexity is polynomial with the number

of classes of customers, but exponential in the number of service centers.

The main computational algorithms for QNs have been integrated in various software tools for performance modelling and analysis that include user friendly interfaces based on different languages to take into account the particular field of application, e.g., computer networks, computer systems. This allows not expert users to apply efficient performance modelling techniques [85, 106, 118, 30]. More recently, the solution algorithms for QNs have been integrated with model specification techniques to provide tools for the combined functional and quantitative system analysis [113].

Product-form networks fulfill various interesting properties. The insensitivity property states that the analytical results, i.e., the stationary state distribution and the average performance indices, depend on the service time requirements only through their average. Similarly, the performance indices depend on the customers routing only through the average visit ratio to each service center [17, 32, 33, 112, 123]. Another important property of product-form QN models is that aggregation methods yield exact results. The aggregation theorem, introduced by Chandy, et al. [31], allows for the replacement of a subnetwork with a single service center, so that the new aggregated network has the same behavior of the original one in terms of a set of performance indices. From the performance viewpoint, exact aggregation allows us to apply the hierarchical system design process by relating the performance indices of the models at different levels in the hierarchy [85]. In a bottom-up analysis of systems represented by a succession of QNs exact aggregation defines the next model. Similarly, in a hierarchical top-down design of system with given performance requirements, the inverse process of disaggregation or development of the network can be applied to define a more detailed model with the same performance indices [10]. Aggregation is an efficient technique when applied to the analysis of nearly complete decomposable systems. Informally, such a system can be decomposed into subsystems whose internal interactions are much higher than the interactions among the subsystems [43]. Exact aggregation for product-form QNs provides a basis for approximate solution methods of more general non-product-form network models [87].

More recently, further research has been devoted to the extension of the class of product-form network models and to its characterization. Some interesting new features have been defined such as the G-Networks, that are QNs with positive and negative customers proposed by Gelenbe [50] that can be used to represent special dynamic of actual systems. Some other more complex models include various functions of state-dependent routing and several special cases of QNs with finite capacity queues, finite population constraints and blocking [2, 8, 24, 55, 78, 117, 120]. Nelson in [96] discusses the mathematics leading to the product-form results and the properties of the stochastic process underlying the network model. Some extensions of product-form QN are presented in [120]. Product-form solution has been extended to QNs with batch arrivals and batch services [64, 65] that are also related to discrete-time QN models.

Some extensions of non product-form QNs have been proposed to represent special classes of systems and communication models, such as Layered Queueing Networks and Extended QN to represent more complex system, e.g., with simultaneous resource possession, finite capacity queues and blocking, and fork and join [81, 108, 3, 104, 99, 8].

We shall now provide an introduction to QN models applied to represent and analyze resource sharing systems. In particular we consider the class of product-form QNs, the main analytical methods to derive a significant set of performance indices, some relevant QNs properties and their applications to system performance evaluation.

2.2 Basic queueing systems

The simplest queueing network consists of a single service center that models the entire system. Basic queueing systems have been defined in queueing theory and applied to analyze congestion systems. The analysis of queueing systems relies on the theory of stochastic processes [38, 77, 81, 108, 74, 118]. Under appropriate independence and exponential assumptions on the model random variables, it is possible to define a continuous-time Markov processes associated with the queueing system. Then, queueing system analysis is usually based on the solution of the underlying Markov process.

A simple queueing system or service center is illustrated in Figure 2.1. The system models the flow of customers as they arrive, wait in the queue if the server is busy serving another customer, receive service, and eventually leave the system. For example, a uniprocessor computer system can be modeled by a simple queueing system where the program to be executed are the customers, the processes ready for execution are in the queue, the processor is the server whose service models program execution. To describe the behavior of a queueing system in time, we have

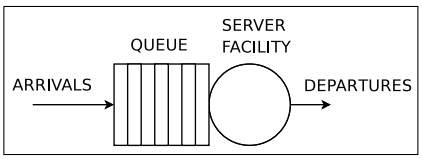

Figure 2.1: A queueing system.

to specify five basic characteristics: 1) the arrival process, 2) the number of servers, 3) the service process, 4) the service or queueing discipline and 5) the system or queue capacity.

1. The *arrival process* to a queueing system describes the behavior of customers arrivals. We define the interarrival time as a random variable representing the time between two consecutive arrivals. The mean arrival rate is the average

number of arrivals per unit of time, and is denoted by λ. The Poisson process is often assumed as the arrival process. This corresponds to an exponential interarrival distribution.

2. The set of identical parallel *servers* can simultaneously service the customers. Each server may correspond to a physically or logically separate service facility of the system, with a common queue shared by all customers.

3. The *service process* describes the customer service. We define the service time as a random variable representing the time spent for a customer service whose mean is $1/\mu$, where μ can be interpreted as the mean service rate.

4. The *queueing discipline* describes the scheduling algorithm for the customers in the queue. If a customer arrives at the system at a time the server(s) is unavailable to provide service to it, it is forced to wait in the queue temporarily until it can start receiving service. If there is more than one customer waiting in the queue at a time the server becomes available, one of the customers in the queue is selected to start receiving service. The customers' selection for service is referred to as the queueing discipline. Queueing discipline may depend on the arrival time, the customer priority and the possible service already given to the customer. Classical examples of queueing disciplines depending only on the arrival time are First Come First Served (FCFS) and Last Come First Served (LCFS). A queueing discipline is work-conserving if there is no artificial creation or loss of work in the system.

5. Finally, the *system capacity* is the upper limit on the number of customers (waiting for and receiving service) in the system. Most analytical studies require the queue size to be infinite, i.e., large enough to accommodate all arriving customers. However, systems have often finite resources imposing an upper bound on the number of customers that can be waiting in the queue simultaneously.

The Kendall's notation $A/B/X/Y/Z$ describes the queueing process of a single queueing system where A indicates the arrival process, B the service process, X the number of parallel servers, Y the system capacity, and Z the service discipline. The simplified notation $A/B/X$ describes a queueing system with infinite capacity and FCFS queueing discipline. Arrival and service processes are denoted by symbols of probability distributions, e.g., D for Deterministic, M for exponential, G for general distribution. For example $M/M/1$ denotes the system with Poisson (Markov) arrival process, exponential (Markov) service process and a single server and $M/G/1$ the same system except for the service time that has a general or arbitrary distribution.

The analysis of a queueing system aims to evaluate a set of performance indices, including the following ones:

- n: the number of customers in the system, i.e. in the queue and being served

- w: the number of customers in the queue
- t_r: the customer response time
- t_w: the customer waiting time
- U: system utilization, that is the percentage of time the system is busy serving,
- X: throughput, i.e., the average number of customers served per unit of time.

Queueing system analysis usually evaluates the latter two average performance indices and the probability distribution or the first moments of random variables n and w, and possibly t_r and t_w.
Let s denote the number of customers in service and let t_s denote the service time, where $E[t_s] = 1/\mu$ is the average service time. Then we can write $n = w + s$ and $t_r = t_w + t_s$. Let N denote the average number of customers in the system and R the mean response time, i.e., $N = E[n]$ and $R = E[t_r]$. Hence:

$$\begin{aligned} N &= E[w] + E[s] \\ R &= E[t_w] + E[t_s]. \end{aligned}$$

Queueing systems are analyzed by defining an associated discrete-space continuous-time stochastic process, whose state include the system population n. Under independent and exponential assumptions we can define an underlying continuous-time Markov chain whose stationary solution is given by Formula (1.4) [77]. Other performance indices can be derived by the stationary state probability and the basic relations, such as Little's law. For some queueing systems, such as the $M/M/1$ and $M/M/m$ systems, the underlying Markov process is a birth and death Markov process, which yields the simple closed-form solution of the stationary state probabilities given by Formula (1.5). Hence, these queueing systems can be easily analyzed and the average performance indices show simple analytical expressions.

M/M/1

The M/M/1 queueing system has Poisson independent arrivals, exponential service time distribution, one server and FCFS discipline. Let λ denote the arrival rate, μ the service rate and let $\rho = \lambda/\mu$ denote the traffic intensity. The system state is defined by n, the customer population and the stationary state probability π_n can be computed by the underlying Markov process that is a birth and death process with constant rates λ and μ. If the system satisfies the stability condition, from Formula (1.5) we immediately obtain:

$$\pi_n = \rho^n(1-\rho) \quad n \geq 0. \tag{2.1}$$

The M/M/1 is stable if the $\rho < 1$, i.e., if the arrival rate is less than the service rate, $\lambda < \mu$. Hence, by the state probability and by Little's law we can derive

other performance indices, such as the average population, the mean response time, system throughput and utilization as follows:

$$N = \frac{\rho}{1-\rho} \tag{2.2}$$

$$R = \frac{1/\mu}{1-\rho} \tag{2.3}$$

$$U = 1 - \pi_0 = \rho \tag{2.4}$$

$$X = \lambda. \tag{2.5}$$

M/M/m

The M/M/m queueing system extends system M/M/1 to m servers under the same exponential and independence assumption for the arrival and service processes, with arrival rate λ and service rate μ for each independent server. Let us define $\rho = \lambda/(m\mu)$. The system state is defined by the customer population as for the M/M/1 system and the associated Markov process is still a birth and death process with constant birth rate λ and variable state-dependent death rate $\mu_n = \min\{n, m\}\mu$ for state $n \geq 0$. If the system satisfies the stability condition, that is if $\rho < 1$, then by Formula (1.5) we obtain:

$$\pi_n = \begin{cases} \dfrac{(m\rho)^n}{n!}\pi_0 & \text{if } 1 \leq n \leq m \\ \dfrac{m^m \rho^n}{m!}\pi_0 & \text{if } n > m \end{cases} \tag{2.6}$$

where:

$$\pi_0 = \Big[\sum_{k=0}^{m-1} \frac{(m\rho)^k}{k!} + \frac{(m\rho)^m}{m!}\frac{1}{1-\rho}\Big]^{-1}.$$

Hence we can derive other performance indices, such as the average population and the mean response time as follows:

$$N = m\rho + \pi_m \frac{\rho}{(1-\rho)^2}$$

$$R = \frac{1}{\mu} + \frac{\pi_m}{m\mu(1-\rho)^2}.$$

M/M/∞

The M/M/∞ queueing system has Poisson independent arrivals with arrival rate λ and an unlimited number of independent exponential servers, each with service rate μ. As a consequence the customers never queue and we can immediately observe that the mean response time is equal to the mean service time, i.e., $R = 1/\mu$. Let $\rho = \lambda/\mu$. The Markov process associated with M/M/∞ is a birth and death process

with constant birth rate λ and variable state-dependent death rate $\mu_n = n\mu$. The system is always stable, and the stationary state probability is given by:

$$\pi_n = \frac{\rho^n}{n!} e^{-\rho} \quad k \geq 0, \tag{2.7}$$

from which we obtain the average population $N = \rho$.

We can apply this type of analysis to various types of M/M/1 queueing systems by defining and solving the associated Markov chain such as for example the M/M/1/B system with finite capacity B and the M/M/1//K system with finite population K. For these and similar basic queuing systems we can derive closed form expressions for the stationary state probability π_n and other average performance indices [38, 77].

2.2.1 Coxian distribution

Many results for single queueing systems can be obtained by assuming exponential distributions for the service and the arrival times. However, for modeling purposes this can be a constraint that should often be relaxed. We shall now introduce a class of quite general distributions used in the analysis of QNs. Coxian distributions are defined as a linear combination of exponential variables, and can be represented by a network of exponential stages. Coxian distributions play an important role in QNs for two reasons: first they are used to model the service time distributions for some station types (from 2 to 4) in BCMP QNs [17], as we shall see in Section 2.3.3. A second reason is that Coxian distribution has a rational Laplace transform and can approximate any distribution arbitrarily closely [77].

Figure 2.2 illustrates a queueing system whose service time is modeled by a Coxian distribution with L exponential stages. There can be at most one customer in stages 1 to L at any time. Customers enter the service via stage 1. The service time at node ℓ, $1 \leq \ell \leq L$, is exponentially distributed with mean $1/\mu_\ell$. A customer completing its service at stage ℓ leaves the system with probability b_ℓ or proceeds to stage $\ell + 1$ with probability a_ℓ, $(a_\ell + b_\ell = 1, 1 \leq i \leq L - 1)$. After stage L, the customer leaves the system with probability 1 $(b_L = 1)$. This service distribution is referred to as a Coxian distribution with L stages. This framework allows an arbitrary distribution that does not have the Markovian property to be approximated by a Coxian distribution that has the Markovian property. For example, consider the service process of Figure 2.2 with $b_\ell = 0$, $1 \leq \ell \leq L-1$ and $\mu_\ell = \mu$, $1 \leq \ell \leq L$. This represents the Erlang distribution with L stages. Its coefficient of variation is equal to $1/\sqrt{\lambda}$ and as $L \to \infty$, it goes to zero approximating a deterministic distribution.

Coxian random variable distribution can be seen as a special case of Phase-type distributions. Informally, given a CTMC with $k + 1$ states of which states $1, \ldots, k$

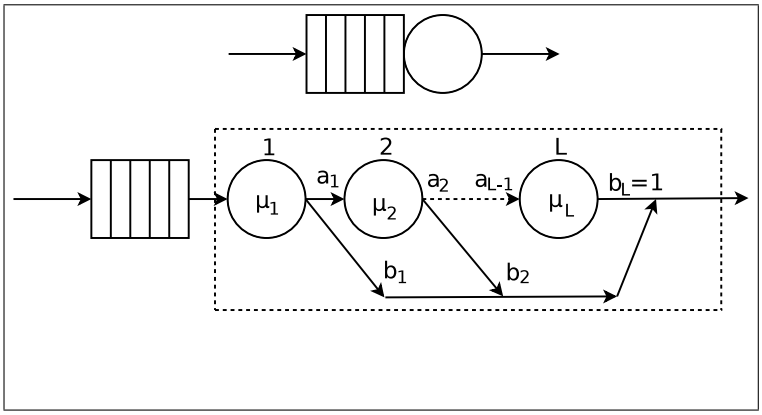

Figure 2.2: A queueing system with a Coxian server with L stages.

are transient and state $k+1$ is absorbing, a continuous phase-type distribution can be defined as the random variable that gives the time until absorption in the CTMC. Every phase-type distribution can be represented by a generalized stage-type distribution, i.e., a series-parallel network of exponential stages [97].

2.2.2 Queueing disciplines

Queueing system behavior and performance indices depend on various system parameters and specifically on the scheduling or queueing discipline. Customer scheduling may depend on the arrival time, like FCFS and LCFS discipline or may be independent of the time arrival and service demand, such as the Random scheduling. Computer system processor scheduling often use Round-Robin discipline where each customer is served for a fixed quantum of time δ. If we consider the quantum size δ much smaller that the average service time then for $\delta \to 0$ we can define the Processor Sharing (PS) discipline. In a system with PS discipline and service rate μ, when there are n customers waiting in the queue, each customer receives the service with rate μ/n.
The scheduling discipline where all the customers are immediately served by a free server is called Infinite Server (IS). Examples of IS or delay service centers are terminal components in timesharing computer systems. The queueing algorithm may depend on the service time required by the customer and possibly the service already given to the customer. Examples are the Shortest Processing Time First, the Shortest Remaining Processing Time First (SRPTF).
The algorithms may also depend on the customer priority that may be defined by some abstract classification of the customers or may depend on the service time. Priority discipline can be with or without preemption. The latter type applies priority scheduling when the server is assigned to a customer after an idle period or at the service completion, and the service is never interrupted. Preemption priority allows a customer with higher priority then the one currently in service to interrupt

that service and to be served. Note that in this case the customers with low priority do not affect the behavior of the customers with high priority.

In the following we mainly focus on the following disciplines FCFS, LCFSPR (LCFS with preemptive resume, i.e., the work done for a preempted customer must not be repeated), PS and IS. Queueing networks whose nodes have such queueing disciplines can be efficiently analyzed under special conditions, as we describe in the next Section.

2.3 Queueing networks

In this section we introduce QN models. First in Sections 2.3.1 and 2.3.2 we define the model whose analysis is based on the solution of the associated Markov stochastic process. Then in Section 2.3.3 we focus on the class of product-form QNs and we review BCMP theorem. Section 2.3.3 discusses the characterization of product-form QNs and reviews both BCMP extensions and non BCMP QNs.

2.3.1 Model definition

A *queueing network* (QN) is a collection of service centers (or *stations* or *nodes*) that serve a set of customers. The customers move among the stations according to a given routing. If the total number of customers, i.e., the *population*, in the network is constant then the network is *closed*. If customers may arrive from (depart to) places outside the network then the network is *open*. Informally, a QN is defined by a set of M service centers $\Omega = \{1, \ldots, M\}$, the set of customers and the network topology.

Each service center is defined by:

- the number of servers. We usually suppose independent and identical servers;
- the service rate. Each server can serve a customer with a speed which can be either constant or dependent on the station state.
- the queueing discipline. The customers in the service center wait to be served according to the scheduling discipline, as introduced in Section 2.2.2.

Customers are described by:

- their total number for closed models,
- the arrival process to each service center for open models,
- the service demand to each service center. The service demand of the customer is expressed in units of service. The service rate of each server is given by units of service / units of time. Hence, we usually consider that the *service time* combines these two parameters as follows: let α be the customer service demand and β the server service rate, then the ratio α/β is the service time.

The network topology models the customer behavior among the interconnected service centers. We assume a non-deterministic behavior represented by the following probabilistic model. In a QN with M stations, when a customer completes its service in station i it immediately exits the node and moves to station j with probability p_{ij}, with $1 \leq i, j \leq M$. For open networks, the customer may also exit the QN from station i with probability p_{i0}. Then the customer behavior in the QN is represented by the *routing probability matrix* $\mathbf{P} = [p_{ij}]$, $1 \leq i, j \leq M$, where $\sum_{j=1}^{M} p_{ij} = 1$ for each station i.

A QN is *well-formed* if it has a well-defined long-term customer behavior, i.e.:

- a closed QN is well-formed if every station is reachable from any other with a nonzero probability;
- for open QNs we can add a virtual station 0 that represents the external behavior, that is it generates external arrivals and absorbs all departing customers, so obtaining a closed QN. Thus an open QN is well-defined by referring to the closed QN definition. Note that for open networks the well-formed property does not imply that the network is stable but just that it has a well-defined long-term customer behavior. Therefore, when we close an open network to decide this property, we can assume a population of just one customer.

In simple queueing models we assume that all the customers are statistically identical, i.e., the service times and the routing probabilities are independent of the customer identity. However, modelling real systems can require to identify different categories of customers that define both the service time and the routing probabilities. To this aim we introduce QNs with multiple types of customers, by defining the concepts of class and chain. In the following we use classes in the global sense (see for example [74, 17, 108, 81]) as opposed to the local sense (see for example [102, 34]). A *chain* forms a permanent categorization of customers, i.e., a customer belongs to the same chain during its whole activity in the network. A *class* is a temporary classification of customers, i.e., a customer can switch from a class to another (usually with a probabilistic behavior) during its activity in the network. The customer service time in each station and the routing probabilities usually depend on the class it belongs to. So we can have multiclass single chain QNs or multiclass and multiple chain QNs. In the following $\mathcal{R}$ denotes the set of classes of the QN, R the number of classes, $\mathcal{C}$ the set of chains and C the number of chains. In a well-formed QN, classes can be partitioned into chains, such that there cannot be a customer switch from classes belonging to different chains. The probabilistic behavior of customers in a well-formed QN is represented by C routing probability matrices $\mathbf{P^{(c)}}$, one for each chain c. A customer that completes its service at station i and class r, either leaves the system with probability $p_{ir,0}^{(c)}$ or immediately moves to a station j in class s with probability $p_{ir,js}^{(c)}$, $1 \leq i, j \leq M$, $r, s \in \mathcal{R}$ and r, s belongs to the same chain c. For multiple chain QNs, we distinguish *open chains*, i.e., if arrivals from and departures to the outside are allowed, and *closed chains*,

i.e., when there is a finite number of customers. Let $K^{(c)}$ denote the population of a closed chain $c \in \mathcal{C}$ and let $p^{(d)}_{0,ir} > 0$ denote the routing probability of an external arrival to station i, class r of open chain $d \in \mathcal{C}$. A QN is said to be open if all its chains are open, closed if they are all closed, and mixed otherwise. We can apply an algorithm [74] that checks whether a multiclass and multiple chain QN is well-formed, given set Ω and the routing matrices, and it defines a partition of the set $E = \{(i,r) : r \in \mathcal{R}, 1 \leq i \leq M\}$ into C ergodic chains. We summarize the notation for QN classes and chains as follows:

- $\mathcal{R}$ is the set of classes and $R = |\mathcal{R}|$
- $\mathcal{R}_i = \{r : \exists j, s, c \ : \ p^{(c)}_{js,ir} > 0 \vee p^{(c)}_{0,ir} > 0 \ r, s \in \mathcal{R}, \ c \in \mathcal{C}, \ 1 \leq j \leq M\}$ is the set of classes served by station i, so $\mathcal{R} = \bigcup_{i=1}^{M} \mathcal{R}_i$,
- $E_c = \{(i,r) | r \in \mathcal{R}_i, 1 \leq i \leq M, \text{class } r \text{ belongs to chain } c\}$,
- $\mathcal{R}^{(c)}_i = \{r \in \mathcal{R}, (i,r) \in E_c\}$ is the set of the classes served by station i belonging to chain c, $1 \leq i \leq M$ and $1 \leq c \leq C$, so $\mathcal{R}_i = \bigcup_{c=1}^{C} \mathcal{R}^{(c)}_i$,
- $\mathcal{R}^{(c)} = \bigcup_{i=1}^{M} \mathcal{R}^{(c)}_i$ is the set of all classes belonging to chain c. We use $r^{(c)}$ to point out that a class r belongs to the set $\mathcal{R}^{(c)}$.

Example. Figure 2.3 shows an example of a two node multiclass and multiple chain QN. The QN has $R = 3$ classes and $C = 2$ chains. Chain 1 is open and formed by classes 1 and 2, while chain 2 is closed. Figure 2.4 sketches how to identify chains of the QN by the analysis of the strong connected components of a directed graph, whose nodes denote the couples (station, class) and arrows are determined by the QN routing matrix. Then we have $\mathcal{R} = \{1,2,3\}$, $\mathcal{R}_1 = \{1,2,3\}$, $\mathcal{R}_2 = \{1,3\}$, $E_1 = \{(1,1),(1,2),(2,1)\}$, $E_2 = \{(1,3),(2,3)\}$, $\mathcal{R}^{(1)}_1 = \{1,2\}$, $\mathcal{R}^{(1)}_2 = \{1\}$.

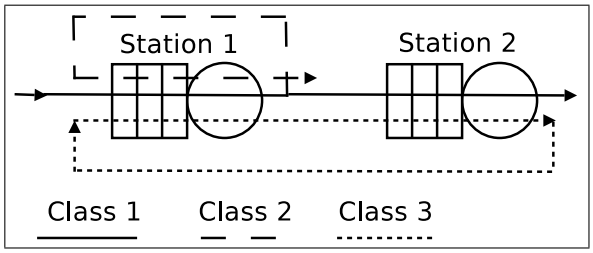

Figure 2.3: Example of multiclass and multiple chain queueing network.

A chain c, $1 \leq c \leq C$, is said to be a single-class if it is has just one class, so that there is not class switching inside the chain. In the QN of Figure 2.3, Chain 1 is *not* single class, while Chain 2 is single class. When all the chains of a QN are single-class, then we say that the QN is single class and multiple chain instead of multiple chain with a single class for chain. In this case, the notation can be

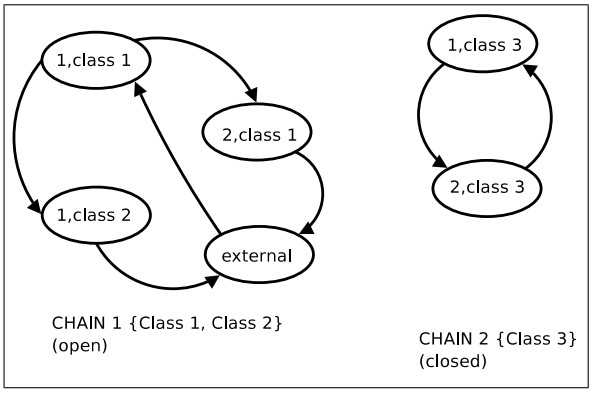

Figure 2.4: Example of a class graph for a multiclass and multiple chain queueing network.

simplified, e.g., an element of the probability routing matrix $\mathbf{P}^{(c)}$, $1 \leq c \leq C$, can be written as $p_{ij}^{(c)} = p_{ir,jr}^{(c)}$, where r is the only class of chain c. $p_{ij}^{(c)}$ represents the probability for a chain c customer of going to station j after being served by station i. Figure 2.5 illustrates and example of single class and multiple chain QN with two chains and three stations.

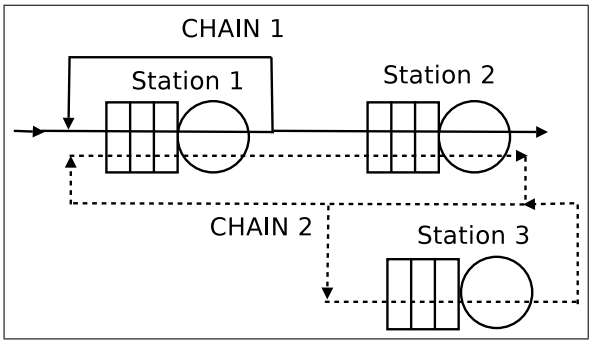

Figure 2.5: Single class and multiple chain queueing network.

In the following we suppose that the QNs are well-formed and admit a stationary behavior, i.e., in the long-run no class can be permanently empty or have an unlimited growth of the number of customers with non-null probabilities.

2.3.2 Markovian queueing networks

Consider a QN with M stations, $\Omega = \{1, \ldots, M\}$ and R classes. Let $n_{ir}(t)$ be the number of customers of class r at the station i at time t, $n_i(t) = \sum_{r=1}^{R} n_{ir}(t)$ the total number of customers at station i at time t and $\mathbf{n_i}(t) = (n_{i1}(t), \ldots, n_{iR}(t))$. Let the state of the QN at time t be $\mathbf{n}(t) = (\mathbf{n_1}(t), \ldots \mathbf{n_M}(t))$. If the stochastic process that describes the evolution of the state $\mathbf{n}(t)$ of a QN is a Markov process, then we say that the QN is Markovian. Hereafter we consider just Markovian QNs whose underlying

stochastic processes are discrete-space time-homogeneous ergodic Markov chains. As we are interested in studying the steady-state performance indices, we can ignore the time parameter t in the state definition. A Markovian QN requires independence and exponential assumptions of the random variables that represent the state. If we consider independent Poisson arrivals for open chains and exponential service time distributions whose rate can depend only on the state of the system then we can define an associated Markov chain with state $\mathbf{n}$. Let $\pi(\mathbf{n})$ denote the stationary probability of state $\mathbf{n}$ and let $\mathbf{Q} = [q_{\mathbf{n},\mathbf{n}'}]$ be the infinitesimal generator of the Markov chain, where $q_{\mathbf{n},\mathbf{n}'}$ denotes the transition rate from state $\mathbf{n}$ to state $\mathbf{n}'$. If the QN is stable, the Markov chain has a steady-state solution. The steady state probabilities $\boldsymbol{\pi}$ are defined as the normalized solution of the system of global balance equations (1.4).

Other performance indices of the QN are derived from the stationary state distribution of the process. Unfortunately, the generality of this approach is limited by its computational complexity. One can easily observe that the process state space cardinality, i.e., the number of states and of global balance equations, often makes the solution of the system intractable. More precisely, for an open network the process state space is infinite and we can obtain an exact solution only in some special cases, when the matrix $\mathbf{Q}$ shows a particular regular structure. For example some QNs with special structure can be analyzed by matrix-geometric technique [97]. For a closed network the process state space grows exponentially with the network parameters that are the number of service centers, customers and customers types. For example, for a single class exponential QN with M service centers and K customers the state space cardinality is $\binom{M+K-1}{K}$. Hence a direct solution of the QN by the underlying Markov process becomes soon prohibitively expensive.

Note that Markovian QNs can be defined also by relaxing exponential conditions for the time distribution. By using Coxian or Phase-type distributions and by a more detailed and appropriate state definition, it is still possible to define the underlying Markov chain. However, this further increases the state space complexity and its cardinality.

In the next section we introduce the class of QNs with product-form that shows a simple closed form of the stationary probability.

2.3.3 BCMP Product-form queueing networks

Product-form QNs provide precise and detailed results in terms of performance indices such as queue length distribution, average response time, resource utilization and throughput. These performance indices are evaluated for each component and for the overall network. Product-form network analysis is based on a set of assumptions on the system parameters that lead to a closed-form expression of the stationary state distribution. Consider a single class and single chain QN with M service centers. Let $\mathbf{n} = (n_1, \ldots, n_M)$ be its state, and n_i the number of customers at station i, and finally $n = \sum_{i=1}^{M} n_i$ the network population for $1 \leq i \leq M$. Product-

form QNs show a closed-form of the joint queue length distribution π that is defined by the associated Markov process, as follows:

$$\pi(\mathbf{n}) = \frac{1}{G} d(n) \prod_{i=1}^{M} g_i(n_i), \tag{2.8}$$

where G is a normalizing constant, function d is defined in terms of network parameters and g_i is a function of n_i and depends on the type of service center i, $1 \leq i \leq M$. For open networks $G = 1$, whereas for closed networks $d(n) = 1$. Product-form QNs have been first introduced by Jackson for open QNs in [73], and by Gordon and Newell for closed QNs in [54]. Both these models require exponential service time distributions, Poisson arrivals for Jackson networks, and consider only single class and single chain QNs. BCMP theorem [17] extends these classes of QNs to open, closed and mixed, multiclass and multiple chain QNs. It also considers non-exponential service time distributions for certain scheduling disciplines.

Product-form QNs can be efficiently analyzed by algorithms with a polynomial time computational complexity in the number of network components. This class of models allows for a good balance between a relative high accuracy in the performance results and the efficiency in model analysis and evaluation. Moreover, product-form networks fulfill several interesting properties such as insensitivity and exact aggregation that greatly influenced the application of this class of models as a powerful tool for performance evaluation. We shall now define the class of BCMP QNs, and then we discuss some properties and possible characterizations of product-form QNs.

BCMP theorem [17] characterizes a wide class of QNs with product-form. Before stating the main result of the theorem we introduce the hypothesis and the model definition. In the following we refer to a QN which satisfies the following assumptions as a BCMP QN. For the sake of clarity, we first present the BCMP theorem for multiple chain, single class networks. In order to simplify the notation, we assume that the class number and the chain number are the same, e.g., we can write $c \in R_i$ to identify a chain c served by node i. Then, we consider multiple chain and multiclass networks.

Service center types.

The network consists of M service stations:

$$\Omega = \{1, \ldots, M\}.$$

The number of classes and the number of chains are the same $R = C$ and each chain can be open or closed. We refer to the notation introduced in Section 2.3.1. BCMP theorem considers four types of service centers:

Type 1: FCFS Service discipline and exponentially distributed chain-independent service time.

For types 2, 3 and 4 stations, the service time distributions have rational Laplace transforms (see Section 2.2.1) and the average service rate can depend on the state of each customer chain.

Type 2: PS Service discipline.

Type 3: IS service centers.

Type 4: LCFSPR Service discipline.

We first assume a constant service rate. Let $\mu_i^{(c)}$ denote the service rate of station i for chain c customers, $1 \leq i \leq M$ and $c \in \mathcal{C}$. For type 1 service centers $\mu_i^{(c)} = \mu_i$, as the service time is chain independent.

State vector.

BCMP theorem gives a product-form solution for states with different levels of detail. Let $\mathbf{n} = (\mathbf{n_1}, \ldots, \mathbf{n_M})$ denote the network state, where:

- $\mathbf{n_i} = (n_i^{(1)}, \ldots, n_i^{(C)})$ is the occupancy vector at station i, $1 \leq i \leq M$,
- $n_i = \sum_{c=1}^{C} n_i^{(c)}$ is the number or customers at station i, $1 \leq i \leq M$,
- $n = \sum_{i=1}^{M} n_i$ is the total number of customers in the network,
- $n^{(c)} = \sum_{i=1}^{M} n_i^{(c)}$ is the number of customers of chain c in the network, $1 \leq c \leq C$. Note that $n^{(c)} = K^{(c)}$ if c is a closed chain.

Arrivals to open chains.

For open chains, customers arrive to the network from an external source. There are two possible state dependencies for the arrival process. In the first case the total arrival process to the network is a Poisson process with parameter $\lambda(n)$ where n is the total number of customers in the network. Arrivals are distributed among the classes according to the routing probabilities. Let $p_{0i}^{(c)}$ denote the probability of an external arrival to node i and open chain c, then by the decomposition property of Poisson processes, the arrival process to station i and chain c is a Poisson process with rate $\lambda(n)p_{0i}^{(c)}$, with $\sum_{i=1}^{M} \sum_{c \in \mathcal{R}_i} p_{0i}^{(c)} = 1$. In the second case the arrival processes to different open chains are independent Poisson processes whose rates depend on the total number of customers of the associated chain, i.e., $\lambda_c(n^{(c)})$ where $1 \leq c \leq C$ and c is an open chain. The arrival process to station j, $1 \leq j \leq M$ is a Poisson process with rate $\lambda_c(n^{(c)})p_{0j}^{(c)}$. The routing probabilities satisfy the normalizing condition $\sum_{i=1}^{M} p_{0i}^{(c)} = 1$. For a closed chain c, we set $p_{0i}^{(c)} = 0$ for every station $i = 1, \ldots, M$ and let $K^{(c)}$ denote the constant chain population, i.e., $n^{(c)} = K^{(c)}$ for all states $\mathbf{n}$.

Traffic equations.

We first define the set of expected number of visits or (relative) throughput for each node i and chain c, denoted by $e_i^{(c)}$. These values are obtained as the solution of the following C systems of linear equations, called traffic equations:

$$e_j^{(c)} = \sum_{i=1}^{M} e_i^{(c)} p_{i,j}^{(c)} + p_{0,j}^{(c)} \quad j = 1, \ldots, M \;\; 1 \leq c \leq C \tag{2.9}$$

These systems uniquely define the solution $e_j^{(c)}$ if chain c is open, while they are not uniquely determined if chain c is closed. If chain c is open $e_j^{(c)}$ represents the expected number of visits (visit ratio) for a customer of chain c at station i. If chain c is closed, then we replace one equation for a station i such that $\mathcal{R}_i^{(c)} \neq \emptyset$ with $e_i^{(c)} = 1$, and we obtain a set of linear independent equations. Then the solution $e_j^{(c)}$ represents the relative visit ratio of a customer belonging to chain c to station j for each visit to station i. Let us define the $\rho_i^{(c)} = e_i^{(c)}/\mu_i^{(c)}$ for each station $i = 1, \ldots, M$ and chain $c \in \mathcal{R}_i^{(c)}$, let $\rho_i^{(c)} = 0$ if $c \notin \mathcal{R}_i^{(c)}$.

We now state the BCMP theorem:

Theorem 1 (BCMP theorem, single class, multiple chain [17]) *Let Ω be a BCMP QN under stability conditions. Then the following steady state probability holds:*

$$\pi(\mathbf{n}) = \frac{1}{G} d(\mathbf{n}) \prod_{i=1}^{M} g_i(\mathbf{n_i}), \tag{2.10}$$

where $d(\mathbf{n}) = \prod_{a=0}^{n-1} \lambda(a)$ if the arrival rate depends on the total number of customer in the system, or $d(\mathbf{n}) = \prod_{c=0}^{C} \prod_{a=0}^{n^{(c)}-1} \lambda_c(a)$. Functions $g_i(\mathbf{n_i})$ are determined as follows:

- *For type 1, type 2 and type 4 stations:*

$$g_i(\mathbf{n_i}) = n_i! \Big[\prod_{c=1}^{c} \frac{1}{n_i^{(c)}!} (\rho_i^{(c)})^{n_i^{(c)}} \Big], \tag{2.11}$$

  considering $\mu_i^{(c)} = \mu_i$ for type 1 stations.

- *For type 3 stations:*

$$g_i(\mathbf{n_i}) = \prod_{c=1}^{C} \frac{1}{n_i^{(c)}!} (\rho_i^{(c)})^{n_i^{(c)}}. \tag{2.12}$$

Sketch of the proof of BCMP theorem presented in [17]. The proof given in [17] is based on a detailed definition of the network state and by substitution of the product-form expression into the global balance equations of the Markov continuous-time process underlying the QN. If i is a type 1 station then its state is represented by vector $\mathbf{b_i} = (b_{i1}, \ldots, b_{in_i})$ where n_i is the number of customers in the station and b_{ij} is the number of the class of the j-th customer in the FCFS order. If i is a type 2 or 3 station $\mathbf{b_i} = (\mathbf{b_{i1}}, \ldots, \mathbf{b_{iR}})$ where $\mathbf{b_{ir}}$ is a vector whose components represent the number of customers of class r at a certain stage of service. If i is a type 4 station, its state is similar to the one introduced for type 1 stations but each vector component always stores the customer service stage (note that the discipline has resume feature). In order to prove the theorem it is shown that:

1. The stationary probability distribution for the detailed state (that we omit for the sake of clarity) is correct by substitution on the global balance equations system.

2. The stationary probability distribution for the state $\mathbf{n}$ defined above can be obtained as marginal distribution of the aggregation of the detailed states.

Let us now consider stations with load-dependent service rates. BCMP theorem identifies three kinds of state-dependent service rates:

Type A: The service rate of a customer at station i depends on the total number of customers n_i. Let $x_i(n_i)$ be a positive function which gives the relative service rate at station i, i.e., $x_i(1) = 1$, such that the actual service rate for a class r customer at station i is $x_i(n_i)\mu_{ir}$. Function x_i is also called capacity function. Then function g_i defined by Equation (2.11) or (2.12) must be multiplied by factor:

$$\prod_{a=1}^{n_i} \frac{1}{x_i(a)}.$$

Type B: The service rate of a class r customer at station i depends on $n_i^{(c)}$. Define $y_i^{(c)}(n_i^{(c)})$ as a positive capacity function, similarly to x_i definition in the previous case. Then function g_i definition must be multiplied by factor:

$$\prod_{c \in \mathcal{R}_i} \prod_{a=1}^{n_i^{(c)}} \frac{1}{y_i^{(c)}(a)}.$$

Note that this state-dependent service rate violates the BCMP hypothesis for type 1 stations, that is it applies only for types 2, 3 and 4.

Type C: The service rate of a customer at station i depends on the number of customers in several stations. Let $\mathcal{H} \subseteq \Omega$ be a subset of the station set, and

$n_H = \sum_{h\in\mathcal{H}} n_h$. Define $z_H(n_H)$ to be a positive capacity function which represents the relative service rate when $n_H = 1$. Then the product $\prod_{h\in\mathcal{H}} g_i(\mathbf{n_i})$ in Equation (2.10) becomes:

$$\Big[\prod_{h\in\mathcal{H}} g_i(\mathbf{n_i})\Big] \prod_{a=1}^{n_H} \frac{1}{z_H(a)}.$$

Note that state-dependent service rates can be combined giving a great flexibility to BCMP QN expressive power. For example, in order to model a PS (or LCFSPR) with different mean service rates for class, with m constant rate servers, it suffices to set $x_i(n_i) = \min\{m, n_i\}/n_i$ and $y_i^{(c)}(n_i^{(c)}) = n_i^{(c)}$.

BCMP theorem for multiclass and multiple chain QNs

We now state a more general form of the BCMP theorem, as presented in [17], for QNs where a customer can move within a chain by class switching. In order to represent class switching we have to modify the notation as follows:

- The routing matrices $\mathbf{P}^{(c)}$ assume the general form $p_{ir,js}^{(c)}$ for nodes i, j, classes r and s in chain c as introduced in Section 2.3.1.
- Service rate at station $i = 1, \ldots, M$ is relative to the class (for types 2, 3 and 4 stations), so we use $\mu_{ir}^{(c)}$,
- There are C independent traffic equation systems which define the (relative) visit ratio for class and station, $e_{ir}^{(c)}$ for $i = 1, \ldots, M$ and $r \in \mathcal{R}_i^{(c)}$, $1 \le c \le C$, as follows:

$$e_{jr}^{(c)} = \sum_{i=1}^{M} \sum_{s\in\mathcal{R}_i^{(c)}} e_{is}^{(c)} p_{is,jr}^{(c)} + p_{0,jr}^{(c)}. \tag{2.13}$$

  Let us define $\rho_{ir}^{(c)} = e_{ir}^{(c)}/\mu_{ir}^{(c)}$ for station i, class r belonging to chain c.

- State $\mathbf{n} = (\mathbf{n_1}, \ldots, \mathbf{n_M})$ components are the occupancy vectors for classes, i.e., $\mathbf{n_i} = (\mathbf{n_i^{(1)}}, \ldots, \mathbf{n_i^{(C)}})$ where $\mathbf{n_i^{(c)}}$ is a vector whose components $n_{ir}^{(c)}$ represent the number of customers of class r (belonging to chain c) at station i, for $r \in \mathcal{R}_i^{(c)}$ and $1 \le i \le M$ and $1 \le c \le C$.

Then for a multiple chain and multiclass QN, Theorem 1 still holds by using the following definitions of function g_i:

$$g_i(\mathbf{n_i}) = n_i! \prod_{c=1}^{C} \prod_{r\in\mathcal{R}_i^{(c)}} \frac{1}{n_{ir}^{(c)}!} (\rho_{ir}^{(c)})^{n_{ir}^{(c)}}, \tag{2.14}$$

for a service center i of type 1, 2 and 4, and:

$$g_i(\mathbf{n_i}) = \prod_{c=1}^{C} \prod_{r \in \mathcal{R}_i^{(c)}} \frac{1}{n_{ir}^{(c)}!} (\rho_{ir}^{(c)})^{n_{ir}^{(c)}}, \tag{2.15}$$

for a type 3 service center i.

These QNs can have load-dependent service centers. The capacity function for state-dependent service rates of type B can be reformulated considering the class population instead of the chain population at the node, i.e., the capacity function can be $y_{ir}^{(c)}$ for station i, and $r \in \mathcal{R}_i^{(c)}$.

In order to evaluate the QN performance indices we can simplify the computation of the BCMP product-form solution as follows:

Aggregation by chain. If Ω is a multiclass and multiple chain QN, under certain assumptions we can compute the steady state probability on an aggregate state. Consider an aggregated vector $\mathbf{n_a}$ where $n_{ai}^{(c)} = \sum_{r \in \mathcal{R}_i} n_{ir}^{(c)}$ for $i = 1, \ldots, M$ and $c = 1, \ldots, C$. In the general case, as the service rate can depend on the class of the customer, we can express the aggregated probability $\pi_a(\mathbf{n_a})$ as sum of probabilities $\pi(\mathbf{n})$ as follows:

$$\pi_a(\mathbf{n_a}) = \sum_{\substack{\mathbf{n} | \sum_{r \in \mathcal{R}_i^{(c)}} n_{ir}^{(c)} = n_{ai}^{(c)} \\ i=1,\ldots,M \wedge c=1,\ldots,C}} \pi(\mathbf{n}).$$

If for each station the service rate is load-independent or depends on the network state according to type A, C or B (formulated on the chain population and not the class population), then BCMP theorem for multiclass and multiple chain QNs can be simplified to single class and multiple chain case. In fact formulas (2.10), (2.11) and (2.12) hold by considering the (relative) visit ratio at station i, $1 \leq i \leq M$ and chain c as the sum of the visit ratios for all the classes belonging to the same chain as station i: $e_i^{(c)} = \sum_{r \in \mathcal{R}_i^{(c)}} e_{ir}^{(c)}$, and the service rate per chain is the weighted sum of the service rate of the classes belonging to the same chain: $\mu_i^{(c)} = \sum_{r \in \mathcal{R}_i^{(c)}} e_{ir}^{(c)} \mu_{ir}^{(c)} / e_i^{(c)}$.

Aggregation for open networks by node population. If all chains of the network are open, arrival rates λ are constant and capacity functions $x_i(n_i)$ are of type A, then it is possible to simplify the steady state distribution expression for the aggregated state $\mathbf{n}' = (n_1, \ldots, n_M)$ where n_i is the total number of customer at station i, $n_i = \sum_{c=1}^{C} \sum_{r \in \mathcal{R}_i^{(c)}} n_{ir}^{(c)}$. In fact it is possible to study each station in isolation considering $\rho_i = \sum_{c=1}^{C} \sum_{r \in \mathcal{R}_i^{(c)}} \lambda \cdot \rho_{ir}^{(c)}$ and capacity function x_i. These results provide a convenient way for normalizing probabilities and checking if the stability condition are satisfied (i.e. $\forall i = 1, \ldots, M : \rho_i < 1$).

2.3.4 Characterization of BCMP-like queueing networks

Product-form QNs allow for a derivation of a set of performance indices without generating and solving the underlying Markov processes and the system of global balance equations. The characterization of the classes of product-form QNs is an interesting task. Under some assumptions (e.g., non-priority scheduling, infinite queue capacity, non-blocking factors, state-independent routing probabilities), it is possible to give conditions on the service time distributions and on the station queueing disciplines to determine whether a well-formed QN has a BCMP-like product-form solution. We consider the following properties which are strictly related to product-form: *local balance*, $M \implies M$, *quasi-reversibility*, *station balance*.

Local balance property.

This property states that the effective rate at which the system leaves state ξ due to a service completion of a chain r customer at station i, equals the effective rate at which the system enters state ξ due to an arrival of chain r customer to station i. This result can be also generalized for multiclass and multiple chain networks. It can be shown that, for some queueing disciplines, local balance holds even when service time distributions are represented by a network of exponential stages [98, 32]. In this case the state must include the stage at which a customer is being served. Note that:

- If a steady state probability distribution $\boldsymbol{\pi}$ satisfies the local balance equations (LBEs), then it satisfies also the global balance equations, but the opposite is not true, i.e., LBEs are a sufficient condition for network solution.
- Solving LBEs is computationally more efficient than solving global balance ones even if it still requires to handle the set of reachable states (which can be a problem for open chains or networks). However if we need to prove that a steady state formula is correct, it can be simpler to check if it verifies LBEs.
- The local balance is a property of a station embedded in a QN. The states we are considering are still states of the network.

$M \implies M$ property.

This property is introduced in [94] and it is defined for a single queueing system. An open queueing system satisfies this property if under independent Poisson arrivals for each class of customers, the departure processes are also independent Poisson processes. Let us consider an isolated queueing station with R classes and a state space Γ. Customers of class r arrive to the system according to independent Poisson processes with rate λ_r. Let $\pi(\xi)$ be the steady state probability of state ξ with $\xi \in \Gamma$, let $|\xi|_r$ be the number of customers of class r in the station when the state is ξ. Define

the set $\mathcal{S}_r^+ = \{\xi' : |\xi'|_r = |\xi|_r + 1\}$. Then the $M \implies M$ property holds if:

$$\forall \xi \in \Gamma \quad \sum_{\xi' \in \mathcal{S}_r^+} \frac{\pi(\xi') q_{\xi'\xi}}{\pi(\xi)} = \lambda_r, \tag{2.16}$$

where $q_{\xi'\xi}$ is the transition rate between state ξ' and ξ. Note that:

- $M \implies M$ property considers the station in isolation. Thus, it can be used to decide whether a station with specific queueing discipline and service time distribution, can be embedded in a product-form QN (see for example [1, 86]). If each station of a QN satisfies the $M \implies M$ property then it has a product-form solution.
- If a network (a chain) is open and each of the station fulfills the $M \implies M$ properties, than the network itself satisfies the $M \implies M$ property [94].
- In a QN with service centers with non-priority scheduling disciplines property $M \implies M$ holds for every station if and only if local balance holds [74, 22].

Quasi-reversibility property.

A queueing system exhibits the quasi-reversibility property if the queue length at a given time t is independent of the arrival times of customers after t and of the departure times of customer prior to t. It is possible to prove (e.g., in [76]) that a QN whose stations are quasi-reversible has a product-form solution. Note that:

- Quasi-reversibility property is defined for isolated stations.
- It is easy to prove (see for example [74]) that all arrival streams to a quasi-reversible system should be independent and Poisson, and all departure streams should be independent and Poisson. In other words, a system is quasi-reversible if and only if it exhibits the $M \implies M$ property.

Station balance property.

This property is introduced in [32] and discussed in [98] and [33]. A scheduling discipline satisfies the station balance property if the service rates at which the jobs in a position of the queue are served are proportional to the probability that a job enters that position. We call these kinds of scheduling disciplines *symmetric disciplines*. Note that also this property is defined for an isolated station, and it is a sufficient condition for product-form (e.g., FCFS does not fulfill the station balance).

Figure 2.6 shows the relation between these properties:

It is worthwhile trying to characterize product-form QNs with higher level properties, e.g., properties of the scheduling discipline. We can summarize some observations:

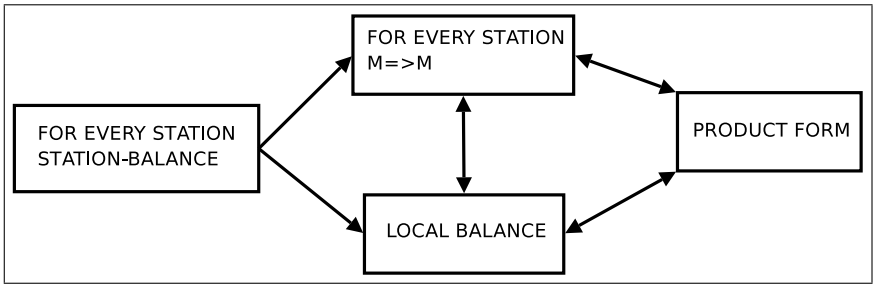

Figure 2.6: Relations between properties related to product-form for nonpriority and work-conserving service centers.

- Every service center with: A) a queueing discipline which is work-conserving and independent of the service time requirement of the customers, and B) an exponentially distributed service time, fulfills local balance property.

- For symmetric disciplines the QN steady state probabilities only depend on the mean of the service time distributions and on the values of their (relative) visit ratio. This is know as the *insensitivity property* of product-form networks [32]. Moreover, we can say that the routing matrix influences the QN performance parameters only for the computation of the visit ratios, i.e., two networks with the same stations and different routing matrices are equivalent if they have the same (relative) visit ratio for each station.

- The symmetric disciplines are the only ones that give product-form solutions if the service time distribution is not exponential [33].

- A symmetric discipline starts serving a customer as soon as it enters in the system, i.e., symmetric discipline are always preemptive discipline.

2.3.5 Other non-BCMP product-forms

Various extensions of the class of BCMP product-form networks have been proposed. They include state-dependent routing [78, 117, 24], i.e., the definition of routing probabilities are special functions that may depend on the state of the entire network or of subnetworks and/or single service centers. This allows the formalism to model systems with more complex features such as dynamic load balancing algorithms or adaptive routing strategies. Such models usually assume some additional constraints on the network parameters and a special structure of the routing state-dependent functions. For example, Towsley [117] considered closed QNs where the routing for some service centers may be a rational function of the queue length of the service centers belonging to a downstream subnetwork with a particular topology, called parallel subnetwork. Boucherie and VanDijk have proposed an extension to a more complex state-dependent routing by considering a more detailed definition of routing

functions dependent on the state of subnetworks called clusters and the state of service centers [24]. The model assumes that the service centers are partitioned into a set of subnetworks that are linked by a state-dependent routing. Then the routing function between two service centers i and j that respectively belong to two disjoint subnetworks I and J has the following expression: $p_{i0}(I)p'_{IJ}p_{0j}(J)$, where $p_{i0}(I)$ and $p_{0j}(J)$ are routing functions internal to subnetworks I and J, respectively, and p'_{IJ} denotes the routing between subnetworks. This model can be useful to represent hierarchical and decomposable systems. Extensions of BCMP networks to include different service disciplines have been derived. Le Boudec proved the product-form solution for QNs with multiple-server nodes with concurrent class of customers that allow to represent special systems [86].

Various special classes of non BCMP QNs have been proved to have product-form solution under particular constraints. These QNs may represent some special system characteristics, such as for example, finite capacity queues, population constraints and positive and negative customers.

QNs with finite capacity queues, subnetwork population constraints and blocking have product-form solution in some special cases [2, 8, 55, 120]. Various blocking types that describe different behaviors of customer arrivals at full capacity service centers and server activities in the network have been defined. For several special combinations of network topology, types of service centers and blocking mechanisms one can derive a product-form solution for the stationary state distribution. Moreover, one can derive various equivalence properties between product-form networks with and without blocking and between networks with different blocking type, as discussed in [8]

Another extension of QNs with product-form is the class of networks proposed by Gelenbe [50] (G-networks) with positive and negative customers that can be used to represent special system behaviors [51]. For example, negative customers may represent commands to delete some transactions in databases or in a distributed computer system due to inconsistency or data locking. A negative customer arriving to a service center reduces the total queue length by one if the queue length is positive and it has no effect otherwise. Negative customers do not receive service. A customer moving between service centers can become either negative or remain positive. Such a QN has product-form solution under exponential and independence assumptions and with a Markovian routing and the solution is based on a set of non linear traffic equations of the customers. G-networks also deal with multiclass of customers [46]. Some further extensions have been introduced in order to extend the modeling power of G-networks, such as the introduction of reset-customers [52], or network state-dependent service rate and routing intensities, triggered batch signal movement [49].

Product-form solution has been extended to QNs with batch arrivals and batch services [64, 65] that are also related to discrete-time QN models. The model evolution is described by a discrete-time Markov chain and assumes special expressions for the probability of batch arrivals and departures and correlated batch routing.

The product-form solution is based on a generalized expression of the traffic equations and the quasi-reversibility property of the network. The product-form solution holds for continuous-time and discrete-time QNs.

3

Stochastic models based on Petri nets

Gonzo: Let's synchronize our watches.
Scooter: We don't have any watches.
Gonzo: That's okay, I don't know what synchronize means anyway.

Jim Henson's Muppet Babies

3.1 Introduction

In this chapter we introduce a very versatile stochastic modelling formalism, i.e., Generalized Stochastic Petri Nets (GSPNs). This formalism is based on Petri Nets (PNs) firstly introduced by Carl Adam Petri in 1962. Petri nets are directed bipartite graphs consisting of places and transitions. Arcs connect the places to the transitions and vice-versa. The state of the system is represented by a marking, i.e., a non-negative vector whose dimension is equal to the number of places of the net. Basic Petri nets have been widely used in the modelling field because they combine a great expressivity power with a set of formal methods of analysis. For example, it is possible to decide if a deadlock can occur, or if the set of all possible states is finite. A review of analysis methods available for basic Petri nets can be found in [95]. Several extensions have been proposed for Petri nets. The introduction of a special type of arc, the inhibitor arc, increases the expressive power of the formalism to be Turing-equivalent. In general, a Petri net model is non-deterministic. Therefore, an immediate extension is the definition of a probabilistic Petri net, i.e., models that in case of non-determinism exhibits a probabilistic behavior. For performance evaluation purposes the introduction of the time parameter is essential. This extension introduces several semantic problems [74] whose solutions originate different formalisms. In this context we focus only on Stochastic Petri Nets (SPNs)

as defined by Molloy in [92] and on Generalized Stochastic Petri Nets (GSPNs) as defined in [36].

Section 3.2 introduces basic PN formalism and analysis. Section 3.3 introduces Generalized Stochastic Petri nets formalism and defines SPN formalism as a special case. Then the problems of the analysis are illustrated, with special attention for the state space explosion problem. Section 3.4 illustrates the fundamental well-known results about (G)SPN with product-form solution.

3.2 Basic Petri Nets (PNs)

In this section we briefly introduce the Petri net model class and cite some relevant properties.

3.2.1 Petri net model definition

A Petri net is a tuple $N = (\mathcal{P}, \mathcal{T}, I(\cdot,\cdot), O(\cdot,\cdot), \mathbf{m_0})$ where:

- $\mathcal{P} = \{P_1, \ldots, P_M\}$ is the set of M places,
- $\mathcal{T} = \{t_1, \ldots, t_N\}$ is the set of N transitions,
- $I(t_i, P_j) : \mathcal{T} \times \mathcal{P} \to \mathbb{N}$ is the input function, $1 \leq i \leq N$, $1 \leq j \leq M$,
- $O(t_i, P_j) : \mathcal{T} \times \mathcal{P} \to \mathbb{N}$ is the output function, $1 \leq i \leq N$, $1 \leq j \leq M$,
- $\mathbf{m_0} \in \mathbb{N}^M$ represents the initial state of the GSPN, i.e., the number of tokens in each place at the initial state.

For each transition t_i let us define the input vector $\mathbf{I}(t_i)$ and the output vector $\mathbf{O}(t_i)$ as follows: $\mathbf{I}(t_i) = (i_1, \ldots, i_M)$ where $i_j = I(t_i, P_j)$, $\mathbf{O}(t_i) = (o_1, \ldots, o_M)$ where $o_j = O(t_i, P_j)$. We say that transition t_a is enabled by marking $\mathbf{m}$ if $m_i \geq I(t_a, P_i)$ for each $i = 1, \ldots, M$. Note that a marking can enable more than one transition. In this case, if the firing of a transition disables the others, we say that the transitions are in conflict, and the behavior of the net is not deterministic. When an enabled transition fires it changes the state $\mathbf{m}$ of the net to $\mathbf{m'}$ where $\mathbf{m'} = \mathbf{m} - \mathbf{I}(t_i) + \mathbf{O}(t_i)$. Given the initial state $\mathbf{m_0}$ the set of all possible states of the model is called reachability set and is denoted by $RS(\mathbf{m_0})$. The incidence matrix $\mathbf{A}$ of a PN is a $N \times M$ matrix whose elements are defined as $a_{ij} = O(t_i, P_j) - I(t_i, P_j)$.

3.2.2 Petri net analysis

Describing a system as a Petri net allows for a formal analysis of its behavioral properties. We can classify PN properties in two classes:

- Marking dependent properties, or behavioral properties, i.e., those properties that depend on the initial marking.

- Marking independent properties, or structural properties, which only depend on the structure, i.e., the combination of places and transitions of the net.

Behavioral properties

A fundamental behavioral property of a system is reachability: given a marking $\mathbf{m}$, determine whether $\mathbf{m} \in RS(\mathbf{m_0})$, i.e., check whether $\mathbf{m}$ is reachable via a firing sequence from the initial marking. A related problem is coverability: given a marking $\mathbf{m}$ determine if there exists a reachable marking $\mathbf{m'} \in RS(\mathbf{m_0})$ such that $\mathbf{m} \leq \mathbf{m'}$; in other words, check whether a marking is reachable such that each place contains at least as many tokens as in $\mathbf{m}$. In general, the problem of deciding if a state is part of the reachability set of a net is EXPSPACE.

The net PN is said to be bounded if, in each place during the evolution of the net, the number of tokens will never exceed a finite number k, formally, if for all $\mathbf{m} \in RS(\mathbf{m_0})$ and $P_i \in \mathcal{P}$, there exists $k \in \mathbb{N}$ such that $m_i \leq k$, where m_i denotes the i-th component of the marking vector $\mathbf{m}$. Net PN is called live if, starting from any reachable marking, any transition in the net can be fired, possibly after some further firings. Formally, for all $\mathbf{m} \in RS(\mathbf{m_0})$ and $t_i \in T$, there exists $\mathbf{m'} \in RS(\mathbf{m})$ such that t_i is enabled at $\mathbf{m'}$. A net has a deadlock (or dead state) if it is possible to reach a marking in which no transition is enabled.

The *reachability graph* of net PN with initial state $\mathbf{m_0}$ is a graph whose vertices are the elements of the reachability set $RS(\mathbf{m_0})$. If there is a transition t that changes the marking from $\mathbf{m}$ to $\mathbf{m'}$ ($\mathbf{m} \xrightarrow{t} \mathbf{m'}$) then we have an arc in the reachability graph from $\mathbf{m}$ to $\mathbf{m'}$. As the reachability graph can be infinite, it is often useful to consider the *coverability tree*. Roughly speaking, this is a tree where nodes are labelled by markings and a node labelled by $\mathbf{m}$ has a child labelled by $\mathbf{m'}$ for any possible firing $\mathbf{m} \xrightarrow{t} \mathbf{m'}$. In order to keep the tree finite, markings can use the symbol ω to represent an unbounded number of tokens (infinity).

However, the coverability tree contains less information than the reachability graph, therefore its analysis may only provide bounds, rather than exact results.

The *state space explosion* is a problem originated by the fact that even a structurally small PN can have a reachability set with a high cardinality. In this case the decision of properties based on the reachability graph or the coverability tree becomes computationally unfeasible.

Structural properties

The term structural (or static) analysis of the net, usually refers to an analysis based on the structure, rather than on the behavior of a net, which does not require the

construction of the reachability set. Structural properties only depend on the net topology and they can be generally studied through the incidence matrix of the net.

Let PN be a Petri net. Roughly speaking, a T-invariant (transition invariant) of PN is an M-dimensional vector in which each component represents the number of times that a transition should fire to take the net from a state $\mathbf{m}$ back to state $\mathbf{m}$ itself. Let us formally define T-invariant $X = (x_1, \ldots, x_N)$:

Definition 1 (T-Invariant) *A T-invariant of a PN is a vector* $\mathbf{X}$ *that satisfies:*

$$\mathbf{A}^T \cdot \mathbf{X} = \mathbf{0}, \quad \mathbf{X} \neq 0 \quad x_i \in \mathbb{N} \tag{3.1}$$

where $\mathbf{A}^T$ *denotes the transposed matrix,* $\cdot$ *the ordinary multiplication of matrices, and* x_i *the* i*-th component of the T-invariant.*

Note that the existence of a T-invariant just means that the system can potentially cycle on $\mathbf{m}$. However, starting from a generic state, we are not sure that there will actually exist a firing sequence that allows the net to cycle. The support of a T-invariant X is the set of transitions corresponding to non-zero entries of X and is denoted by $||X||$. A T-invariant X is minimal if there is not any other T-invariant X' such that $x'_i \leq x_i$ for all $i = 1, \ldots, N$. A support is minimal if no proper non-empty subset of the support is also a support of a T-invariant. The minimal support T-invariant is the unique minimal T-invariant corresponding to a minimal support.

Definition 2 (P-invariant or S-invariant) *A P-invariant or S-invariant (place invariant) is a* M *dimensional vector which can be calculated as a solution of the following equation:*

$$\mathbf{A} \cdot \mathbf{Y} = \mathbf{0} \tag{3.2}$$

If a P-invariant $\mathbf{Y} = (y_1, \ldots, y_M)$ with all positive components exists, then the net is called conservative since the weighted sum of the tokens remains constant during the evolution of the net, i.e., it is constant for each marking of its reachability set:

$$\mathbf{m} \in RS(\mathbf{m_0}) \;\Rightarrow\; \sum_{i=1}^{M} y_i m_i = k,$$

where k is a constant.

The set of places corresponding to non-zero entries in a S-invariant $\mathbf{Y}$ is called support of the S-invariant. A support is *minimal* if no proper non-empty subset of the support is also a support. An S-invariant $\mathbf{Y}$ is called *minimal* if there is no other S-invariant $\mathbf{Y}^*$ such that $y_i^* < y_i$ for all places $P_i \in \mathcal{P}$. For each minimal support of an invariant there exists a unique minimal S-invariant called *minimal support S-invariant.*

Another important concept on PNs is the sufficient place sets.

Definition 3 (Sufficient place set) *A set of places $\mathcal{Q} \subset \mathcal{P}$ is a sufficient place set if the markings of the places in $\mathcal{Q}$ are sufficient to define the marking of the whole SPN.*

A relevant role in the structural analysis of a Petri net is played also by traps and siphons. A trap $\mathcal{S}$ is a subset of places, $\mathcal{S} \subseteq \mathcal{P}$, such that for any transition $t \in \mathcal{T}$, $O(t, P)I(p, t) \geq 0$. For Petri nets with unary weights only, this $P \in \mathcal{S}$ means that the transitions consuming tokens in $\mathcal{S}$ are a subset of those producing tokens in $\mathcal{S}$. As a consequence the overall number of tokens in a trap can only increase, and if it is initially marked (each P in $\mathcal{S}$ contains at least one token) then it will be marked in any reachable marking. Dually, a siphon $\mathcal{S}$ is a subset of places $\mathcal{S} \subseteq \mathcal{P}$ such that for any transition $t \in \mathcal{T}$, $I(t, P)O(t, P) \geq 0$, with $P \in \mathcal{S}$.

If a siphon is token free under some marking, then it will remain token free in all successor markings. Besides having an interest in themselves, structural properties can be helpful in the behavioral analysis of a net. For instance, a necessary condition for a marking $\mathbf{m}$ to be reachable is that the equation

$$\mathbf{A}^T \cdot \mathbf{X} + \mathbf{m_0} = \mathbf{m} \tag{3.3}$$

admits a positive solution. Structural boundedness, i.e., boundedness with respect to any possible initial marking, can be characterized in terms of the existence of an m-vector of positive integers such that $\mathbf{A} \cdot \mathbf{Y} \leq \mathbf{0}$. For some subclasses of Petri nets, behavioral properties can be characterized in purely structural terms. For instance, Condition (3.3) becomes also sufficient for reachability when dealing with acyclic net.

3.3 Generalized Stochastic Petri Nets

Basic PN is a non-deterministic formalism because when two or more transitions are enabled it is not specified which one has to fire. Stochastic Petri Nets (SPNs) associate a negative exponentially distributed random time with every transition. Therefore if more than one transition is enabled, the one with the lowest associated random time fires. When we introduce the time factor in SPNs we should discuss the firing semantic. In this work we just say we refer to the atomic firing semantic (see [74, 4] for a discussion on other semantics), i.e., the tokens are instantaneously removed from the input places and put into the output places when the transition fires.

SPNs are very important for some reasons:

- Their reachability set coincides with the one of the associated PN. Therefore all the analysis discussed in Section 3.2 can be applied.
- It can be shown that the stochastic process associated with the marking changes is a CTMC. The intuition is that the sojourn time in a marking is

given by the minimum of a set of negative exponential random variables, that is again a negative exponential random variable.

In this section we present the formalism for Generalized Stochastic Petri Nets. This formalism is a generalization of SPNs that admits immediate transitions, inhibitor arcs and transition priorities.

Definition 4 *A GSPN is a tuple defined as:*

$$GSPN = (\mathcal{P}, \mathcal{T}, I(\cdot,\cdot), O(\cdot,\cdot), H(\cdot,\cdot), \Pi(\cdot), w(\cdot,\cdot), \mathbf{m_0})$$

where:

- $\mathcal{P} = \{P_1, \ldots, P_M\}$ *is the set of* M *places,*
- $\mathcal{T} = \{t_1, \ldots, t_N\}$ *is the set of* N *transitions (both immediate and timed),*
- $I(t_i, P_j) : \mathcal{T} \times \mathcal{P} \to \mathbb{N}$ *is the input function,* $1 \leq i \leq N$, $1 \leq j \leq M$,
- $O(t_i, P_j) : \mathcal{T} \times \mathcal{P} \to \mathbb{N}$ *is the output function,* $1 \leq i \leq N$, $1 \leq j \leq M$,
- $H(t_i, P_j) : \mathcal{T} \times \mathcal{P} \to \mathbb{N}$ *is the inhibition function,* $1 \leq i \leq N$, $1 \leq j \leq M$,
- $\Pi(t_i) : \mathcal{T} \to \mathbf{N}$ *is a function that specifies the priority of transition* t_i, $1 \leq i \leq N$,
- $\mathbf{m} \in \mathbb{N}^M$ *denotes a marking or state of the net, where* m_i *represents the number of tokens in place* P_i, $1 \leq i \leq N$,
- $w(t_i, \mathbf{m}) : \mathcal{T} \times \mathbb{N}^M \to \mathbb{R}$ *is the function which specifies for each timed transition* t_i *and each marking* $\mathbf{m}$ *a state dependent firing rate, and for immediate transitions a state dependent weight,*
- $\mathbf{m_0} \in \mathbf{N}^M$ *represents the initial state of the GSPN, i.e. the number of tokens in each place at the initial state.*

Analogously to PNs, for each transition t_i let us define the input vector $\mathbf{I}(t_i)$, the output vector $\mathbf{O}(t_i)$ and the inhibition vector $\mathbf{H}(t_i)$ as follows: $\mathbf{I}(t_i) = (i_1, \ldots, i_M)$ where $i_j = I(t_i, P_j)$, $\mathbf{O}(t_i) = (o_1, \ldots, o_M)$ where $o_j = O(t_i, P_j)$ and $\mathbf{H}(t_i) = (h_1, \ldots, h_M)$ where $h_j = H(t_i, P_j)$. Function $\Pi(t_i)$ associates a priority to transition t_i. If $\Pi(t_i) = 0$ then t_i is a timed transition, i.e., it fires after an exponentially distributed firing time with mean $1/w(t_i, \mathbf{m})$, where $\mathbf{m}$ is the marking of the net. If $\Pi(t_i) > 0$ then t_i is an immediate transition and its firing time is zero. We say that transition t_a is enabled by marking $\mathbf{m}$ if $m_i \geq I(t_a, P_i)$ and $m_i < H(t_a, P_i)$ for each $i = 1, \ldots, M$ and no other transition of higher priority is enabled. In the following we consider just two priority levels, 0 (timed transitions) and 1 (immediate transitions). Hence, when an immediate transition is enabled all the timed

ones are disabled. The firing of transition t_i changes the state of the net from $\mathbf{m}$ to $\mathbf{m} - \mathbf{I}(t_i) + \mathbf{O}(t_i)$. The reachability set $RS(\mathbf{m_0})$ of the net is defined as the set of all markings that can be reached in zero or more firings from $\mathbf{m_0}$. Note that it is *not* the case that the reachability set of a GSPN is equal to the reachability set of the corresponding PN. We say that marking $\mathbf{m}$ is tangible if it enables only timed transitions and it is vanishing otherwise. For a vanishing marking $\mathbf{m}$ let $\mathcal{T}_\alpha$ be the set of enabled immediate transitions. Then the firing probability for any transition $t_i \in \mathcal{T}_\alpha$ and any state $\mathbf{m}$ is denoted by $p(t_i, \mathbf{m})$ and is defined as follows:

$$p(t_i, \mathbf{m}) = \frac{w(t_i, \mathbf{m})}{\sum_{t_j \in \mathcal{T}_\alpha} w(t_j, \mathbf{m})}. \tag{3.4}$$

As for SPNs, given a tangible marking $\mathbf{m}$ the transition with the lowest associated stochastic time fires. Sometimes it can be useful to associate a probabilistic output vector to a transition. In this case we denote a possible output vector of transition t_i by $\mathbf{O}_j(t_i)$, and its probability by $d_{i,j}$ where $\sum_j d_{i,j} = 1$. Note that this is not a real extension to the model definition. In fact, the probabilistic behavior of a transition firing can be obtained by the use of immediate transitions in a trivial way.

A GSPN is represented by a graph with the following conventions: timed transitions are white filled boxes, immediate transitions are thin black filled boxes, places are circles, if $I(t_i, P_j) > 0$ we draw an arrow from P_j to t_i labelled with $I(t_i, P_j)$, if $O(t_i, P_j) > 0$ we draw an arrow from t_i to P_j labelled with $O(t_i, P_j)$, if $H(t_i, P_j) > 0$ we draw an circle ending line from P_j to t_i labelled with the value of $H(t_i, P_j)$, the marking $\mathbf{m}$ is represented by a set of m_j filled circles representing the tokens in place P_j for each $j = 1, \ldots, M$. We omit the arc labels if they are equal to 1.

3.3.1 (G)SPN analsys

GSPN analysis consists in finding the steady state probability for each tangible marking of the reachability set. The stochastic process associated with a GSPN model is a semi-Markov process. Indeed, the residence time in a general state can be either exponentially distributed (tangible states) or deterministically zero (vanishing states). However, it is possible to eliminate the vanishing states with efficient algorithms (see for example [89]) and therefore one has to study a CTMC. In the following we will straightforwardly refer to the CTMC associated with a GSPN model.

In the CTMC associated with a GSPN the state sojourn times are computed from the mean transition delays of the net.

3.4 Product-form (G)SPN

The analysis of a CTMC associated with a GSPN or a SPN can be very hard because of the state space explosion problem. As illustrated for queueing networks

(see Section 2.3.3) and for competing and cooperating Markov chains (see Section 1.3), several research efforts have been devoted to identify conditions for product-form (G)SPNs. We can summarize these results by identifying three classes of product-form models:

Boucherie's class Its definition is based on Boucherie's product-form for competing Markov chains [25] presented in Section 1.3.1. In the same paper [25] the author shows the application to SPN product-form of the theory of competing Markov chains in product-form. The main idea is to consider the stochastic processes generated by a set of SPN models as the competing Markov chains, and modelling the blocking mechanism required by the product-form conditions by opportune places and arcs. Informally, the presence of a token in a place associated with a resource represents the fact that the resource is available at a given time. When one of the SPNs uses the resource then it removes the token from that place. The modeler has to pay attention to introduce opportune arcs in order to model the blocking mechanism illustrated in Section 1.3.1 that can result not natural.

Coleman, Henderson et al. class This product-form has been introduced for SPNs in [63, 39]. Some results in this thesis are strictly related to this model class, therefore we review it later in this chapter. The main idea is that SPN models in this product-form have a stationary distribution that can be expressed as normalized product of functions dependending on the number of tokens in each place. Note that the approach is different from the previous case. We can say that Coleman, Henderson et al. product-form conditions depend on the whole net configuration (structure and rates) so it does not allow for a hierarchical approach to the model definitions. In our opinion, the main drawbacks of this product-form are three. The first concerns with the algorithms defined for determining the normalization constant. In [39] and [111] a convolution and a MVA algorithm are presented, however, testing the conditions for their application requires the generation of the reachability set of the model. The second limitation is that the approach is not hierarchical neither compositional. For example, adding a new place to the net structure with some connection arcs requires to re-analyze the whole net. The third problem is related to an interpretation of a condition for the product-form. As we illustrate in the following sections, one of the conditions requires to check the rank of a matrix that depends both on the net structure and on the transition rates. The fact that the product-form depends on the transition rates should not be surprising: BCMP theorem [17] requires that the transition rates corresponding to the customer departures form an FCFS queueing station must be class independent, i.e, they must have the same rate. However, in the BCMP case we have a strong intuition of the meaning of this condition, while the same cannot be said for the SPN product-form. In Chapter 5 we try to analyze this condition deeper and to overcome the problem of compositionality.

Balbo et al. class This class of product-form models deals with GSPN, i.e., it considers the presence of immediate transitions in the net structure. It is defined in [6] where the authors introduce a set of conditions for the application of an algorithm that reduces the GSPN to a SPN in Coleman, Henderson et al. product-form.

3.4.1 Coleman, Henderson et al. product-form SPNs

In this part of the section we review the main results for product-form SPNs [63, 39]. These results are used in Chapter 5 to show the relations between this class of product-form models and the one given by ERCAT [59, 61] that will be reviewed in the next chapter. Moreover, we show that we can enhance the compositionality of this approach by the analysis of the reversed Markov process.

For the sake of clarity we introduce the theorem in two phases. First we introduce the conditions and then we illustrate the main theorem.

COND1: Conditions for Coleman, Henderson et al. product-form.

- The theorem is given for SPNs (without immediate transitions and inhibitor arcs), it deals with probabilistic transition output vectors, and state dependent firing rates. As transitions are all timed we assume all along this section $\Pi(t_i) = 0$ for all $t_i \in \mathcal{T}$ and, as there are not inhibitor arcs, we assume $H(t_i, P_j) = 0$ for all $t_i \in \mathcal{T}$, $P_j \in \mathcal{P}$.

- The firing rate of a transition must be in the following form:

$$w(t_i, \mathbf{m}) = \frac{\psi(\mathbf{m} - \mathbf{I}(t_i))\chi_i}{\phi(\mathbf{m})}, \tag{3.5}$$

  where χ_i is a non-negative constant that depends just on transition t_i, function ψ is a non-negative function, and function ϕ is a positive function. ψ and ϕ play the role of potential functions. They can model a wide class of load-dependent firing rates (for a deep analysis see [63]). We say that transition $t_i \in \mathcal{T}$ is enabled if and only if $\psi(\mathbf{m} - \mathbf{I}(t_i)) > 0$.

- There cannot be two transitions with the same input vector. Often we can transform a SPN that does not satisfy this condition into one that satisfies it by fusing the transitions with the same input vector in one transition and using probabilistic output vectors. However, as pointed out in [63], this cannot be done if the transitions with the same input vector depend on the state of the net in different ways (because function ψ and ϕ depend only on the state of the net and the transition input vector).

- Let B_i be the number of output vectors of transition t_i, then we have that $\sum_{j=1}^{B_i} d_{i,j} = 1$. For each transition $t_i \in \mathcal{T}$ and $\mathbf{O}_j(t_i)$, $1 \leq j \leq B_i$, there must

exist exactly one transition $t_s \in \mathbf{T}$ such that $\mathbf{O}_j(t_i) = \mathbf{I}(t_s)$. We write $E_j(t_i)$ to denote t_s, and $p(t_i, t_s)$ to denote the output vector probability $d_{i,j}$. Vice versa, for every transition t_s we have at least one output vector j of a transition $t_i \in \mathcal{T}$ such that $\mathbf{I}(t_s) = \mathbf{O}_j(t_i)$ for $1 \leq j \leq B_i$.

- The following system of traffic equations must have a solution $\mathbf{f} = (f_1, \ldots, f_N)$ with $N = |\mathcal{T}|$:

$$\chi_i f_i = \sum_s \chi_s f_s p(t_i, t_s) \quad 1 \leq i, s \leq N \tag{3.6}$$

under the condition $f_i > 0$ for all $t_i \in \mathcal{T}$. Note that it is possible to prove that if $\mathbf{f}$ and $\mathbf{f}'$ are solutions of system (3.6) then there exists $k > 0$ such that $\mathbf{f} = k\mathbf{f}'$, i.e., the solutions differ for a constant.

Before stating the main theorem we still need some more notions. Let us define vector $\mathbf{C}$ as follows:

$$\mathbf{C} = \begin{bmatrix} \log\left(\frac{f_1}{f_{E_1(t_1)}}\right) \\ \vdots \\ \log\left(\frac{f_1}{f_{E_{B_1}(t_1)}}\right) \\ \vdots \\ \log\left(\frac{f_N}{f_{E_1(t_N)}}\right) \\ \vdots \\ \log\left(\frac{f_N}{f_{E_{B_N}(t_N)}}\right) \end{bmatrix}. \tag{3.7}$$

Note that vector $\mathbf{C}$ is independent of the particular solution $\mathbf{f}$ of system (3.6) because all the solutions differ for a positive constant.

The main theorem. Now we can state the main theorem. Recall that $M = |\mathcal{P}|$, $N = |\mathcal{T}|$ and ϕ is the function that appears in the firing rate definition (3.5).

Theorem 2 (Coleman, Henderson et al. product-form [63, 39]) *Let S be a SPN that satisfies structural conditions COND1 and with incidence matrix $\mathbf{A}$. Then S has the following product-form solution:*

$$\pi(\mathbf{m}) = \phi(\mathbf{m}) \prod_{i=1}^{M} y_i^{m_i}, \tag{3.8}$$

if and only if:

$$rank(\mathbf{A}) = rank([\mathbf{A}; \mathbf{C}]), \tag{3.9}$$

where $[\mathbf{A};\mathbf{C}]$ *denotes the matrix obtained by augmenting incidence matrix* $\mathbf{A}$ *of column* $\mathbf{C}$. *In this case we can obtain* y_i *for* $i = 1, \ldots, M$ *by solving the system:*

$$-\mathbf{A}\begin{bmatrix} \log(y_1) \\ \vdots \\ \log(y_M) \end{bmatrix} = \mathbf{C}. \tag{3.10}$$

In [63, 39, 56] some examples of Coleman, Henderson et al. analysis can be found. In Chapter 5 we review most of these examples by applying our technique.

The normalizing constant. It is out of the scope of this work presenting and analyzing the algorithms defined for the computation of the normalizing constant. However, we think that it is useful to review the conditions for the application of those algorithms. The first algorithm for normalizing the probabilities of product-form SPNs is presented in [39] and is a convolution algorithm. In [111] a Mean Value Analysis algorithm (MVA) is presented. In both these cases the algorithm can be used for the computation of the normalizing constant only for a subset of the class of Coleman, Henderson et al. product-form SPNs. The first assumption concerns function $\phi(\mathbf{m})$ definition that has to be in the form:

$$\phi(\mathbf{m}) = \prod_{i=1}^{M} \phi_i(m_i).$$

In this case the steady state probabilities can be expressed as:

$$\pi(\mathbf{m}) = K \prod_{i=1}^{M} \left[\phi_i(m_i) y_i^{m_i}\right].$$

This condition is simple to decide.

The second condition is harder to check because it predicates on the reachability set of the SPN. Given the initial state $\mathbf{m}_0$, and a let $\mathbf{Y}$ be a S-invariant then a state $\mathbf{m}$ has to be reachable if and only if $\mathbf{Ym} = \mathbf{Ym}_0$. In general, in order to check this condition, the state space of the SPN must be generated. For some classes of SPNs it trivially holds: this happens for marked graphs (SPNs in which every place has at most one incoming and one outgoing arc) and state machines (SPNs in which every transition has at most one incoming and one outgoing arc).

3.5 Conclusions

In this chapter we have reviewed the main results on product-form SPNs and GSPNs. We have mainly focused on those results that we are going to use in the following of this thesis, i.e., the product-form SPN class defined in [63, 39]. We think it is worthwhile to point out the different approaches that arise from Boucherie's product-form

and Coleman, Henderson et al. one. In the former case the modeler defines a set of SPN sub-models with known steady state probability distributions. Then these sub-models can be composed in a way so that the resulting model exhibits a steady state solution in product-form. In the latter product-form class the modeler defines the whole SPN and the theorem states the conditions under which the steady state probabilities can be expressed as product of functions depending on the number of tokens in each place. Boucherie's product-form appears to have a strong compositional property, while Coleman, Henderson et al. class is monolithic. In fact, in the latter case, adding an arc or a simple place to the net requires a new analysis and the product-form could be lost. Moreover, the problem of giving an interpretation of the rank condition (3.9) from a modeling point of view is still open. A deep discussion about this is one of the main topics of the following chapters.

4

Markovian Process Algebra

4.1 Introduction

In this chapter we briefly introduce the process algebra and review some relevant properties. Process algebra has been widely used as modeling formalism for performing functional analysis of concurrent systems. In the field of performance evaluation several extensions have been proposed in order to be able to model the temporal behavior of systems. The main strength of this formalism is the combination of a well-defined semantic model and compositionality. We mainly focus on the Performance Evaluation Process Algebra (PEPA). A model defined in PEPA has an underlying CTMC that can be algorithmically derived. Product-forms have been studied also for PEPA models. Many results defined for queueing networks and stochastic Petri nets have been reformulated using this formalism. However, we think that the most innovative results are the RCAT, ERCAT and MARCAT theorems that are recalled here in the following sections.

The chapter is structured as follows: Section 4.2 introduces the standard process algebra, Section 4.3 introduces some timed extensions with special attention devoted to PEPA, Section 4.5 illustrates the main well-known results about product-form PEPA models.

4.2 Basic process algebra

Process algebras are abstract languages that have been widely used for the design and specification of concurrent systems. The most used process algebras are:

- Calculus of Communicating Systems (CCS) defined by Milner in [91].
- Communication Sequential Processes (CSP) defined by Hoare in [71].

Process algebra models (either CCS and CSP) have been used to establish the correct behavior of concurrent systems. In fact, several qualitative properties can be derived such as the freedom from deadlock.

A process algebra model consists of a collection of *agents* that can perform some atomic actions. The actions can represent sequential behaviors of the agents,

communications or synchronizations among them. The major distinction between CCS and CSP is on the definition of the communication actions. It is out of the scope of this thesis presenting a formal review of CCS or CSP syntax and semantic, however, we informally introduce the basic notions of the calculus and then describe the communication techniques defined for CCS and CSP. An agent P is defined according to the following syntax:

$$\begin{aligned} P ::= \quad & 0 \\ | \quad & a.S \\ | \quad & S + Q \\ | \quad & S||Q \\ | \quad & S \setminus M \\ | \quad & S[a_1/a_0, \ldots] \\ | \quad & 0 \end{aligned}$$

where S and Q are agents, a and a_i denote a label, M is a set of labels. The prefix operator $(a.S)$ models an agent that after performing action a behaves like S. Using the choice operator $(S + Q)$ we model that the agent behaves as S or Q. In the parallel composition $(S||Q)$ S and Q proceed in parallel (possibly communicating). The restriction $(S \setminus M)$ hides the set of labels M of S from outside agents. The relabelling $(S[a_1/a_0, ...])$ replaces in S the label a_0 by a_1, more than one replacement can be specified. Finally, the null agent (0) is the agent that cannot act and basically can be thought as a deadlock.

In CCS the communication is defined between two agents. Suppose that an agent performs an action a, then the communication occurs when the other agent performs a complementary action $\overline{a}$. The resulting communication action has the special label τ that denotes an internal action invisible to the environment. In CSP there is not the concept of complementary action. Basically, the synchronization occurs between two agents that perform an action with the same label. Note that the joint action remains visible to the environment, therefore other concurrent processes can reuse it to communicate. This leads to a multiway synchronization.

Most of Stochastic Process Algebras (SPAs) use a communication definition that is similar to that defined in CSP.

For both CCS and CSP a structured operational semantic has been defined. This is based on a labelled transition system. This makes possible the construction of a derivative graph, namely a graph in which the vertices are the language terms and the arcs are the transitions.

Bisimulation is a binary equivalence relation that can be applied to process algebra models. Roughly speaking, two systems are bisimilar if they match each other's moves. In the bisimulation style of equivalence an agent is characterized by its actions and, in general, the analysis of the derivative graphs is required. If the internal actions are considered observable then we have a strong equivalence, otherwise we have a weak equivalence.

Both CCS and CSP allow one to derive several qualitative properties of the modelled systems. In order to perform quantitative analysis several timed extensions of these formalisms have been introduced.

4.3 Timed process algebra

As in classical process algebras time is abstracted away it is not possible to derive performance measures of the models. The formalism extensions that we present in this section aim to introduce the timing characteristics and/or the probabilities of different behaviors in the model description. Without this quantified information it is not possible to derive quantitative measures such as expected response time or throughput. In the last years several extensions to process algebras have been defined in order to deal with quantitative analysis. We can identify three classes of extensions:

- Timed process algebras. The main idea under this extension is to add a duration to every action of a process algebra, i.e., the operator $\alpha.P$ becomes $(\alpha, t).P$ where t denotes the time required for action α. These extensions have been proposed for different languages: ACP, CSP, CCS and LOTOS [66].

- Probabilistic process algebras. In this case the main idea consists in defining a new semantic for the *choice* operator. Informally, $S = P + Q$ in a standard process algebra means that process S can behave either as P or Q. In probabilistic process algebras this non-determinism becomes a probabilistic choice. Also in this case the probabilistic extension has been introduced for ACP, CCS, CSP and LOTOS [66].

- Stochastic process algebras. In this case an action requires a random time to be performed. *Timed processes and performance analysis* (TIPP) is a stochastic process algebra that extends CSP [67]. Action durations are modelled by exponential random variables, therefore the underlying stochastic process is a Markov process. Another process algebra based on a CSP extension is *Performance evaluation process algebra* (PEPA) defined by Hillston in [68]. Also in this case the action duration are exponentially distributed. *Extended Markovian Process Algebra* (EMPA) has been introduced by Bernardo et al. in [20] and provides constructs to represent immediate transitions and non-determinism. A complete definition of EMPA semantic can be found in [21].

Adding temporal information to a Process algebra influences the following analysis [66]

- functional behavior (e.g. liveness or deadlocks),

- temporal behavior (e.g. throughput, waiting times, reliability),

- combined properties (e.g. probability of timeout).

We focus on Performance Evaluation Process Algebra (PEPA) that is a Markovian Process Algebra (MPA). It extends the classic process algebras because in PEPA every action has a duration that is modelled by an exponentially distributed random variable. The goal of the formalism definition is being able to obtain a Continuous Time Markov Chain (CTMC) given any PEPA model. A PEPA model consists of a set of components that can interact according to a small number of combinators: prefix, choice, parallel composition and abstraction. We briefly recall this formal model description in Section 4.4. Thanks to the great flexibility of this formalism and the available software tools, PEPA has recently become quite popular in the performance analysis field. However, even a system consisting of simple interacting components may generate a very large CTMC and the exact analysis of the system performance by PEPA can soon become unfeasible. For this reason recently many research efforts have been devoted to study product-forms for PEPA that could possibly allow for more efficient solution algorithms [70, 57, 110, 59, 61]. Roughly speaking, a product-form model has a steady-state distribution that can be calculated by product of the steady state distributions of its components, even if the components are not stochastically independent. Note that defining efficient solution algorithms for product-models can be a non-trivial task. For example, if the CTMC has a finite number of states, the product-form solution has to be normalized that, in its basic definition, requires to compute a summation over all the possible system states of the process. Therefore, this is computationally expensive and can cause numerical stability problems.

4.4 Performance Evaluation Process Algebra

In this section we introduce PEPA formalism and we recall the associated notation. PEPA models are based on the description of component interactions. Each component has an associated set of actions. Let $\mathcal{A}ct$ be the set of all actions, then $a \in \mathcal{A}ct$ is a pair (α, r) where $\alpha \in \mathcal{A}$ is the type of the action (and $\mathcal{A}$ the set of action types) and $r \in \mathbb{R}^+$ is the parameter of the negative exponential distribution. In the following we briefly recall the PEPA operational semantic presented in details in [68]. The *prefix* combinator models the sequential behavior, i.e., the component $(\alpha, r).P$ carries out the activity (α, r) in Δt time and then behaves as component P. Δt is an exponentially distributed random variable with parameter r. The *choice* combinator $+$, i.e. $P + Q$, models a component that behaves as component P or as component Q. When a component can perform an activity (α, r) we say that the activity with type α is enabled. Let us denote all the activities enabled in P by $\mathcal{A}ct(P)$. Then $\mathcal{A}ct(P+Q) = \mathcal{A}ct(P) \uplus \mathcal{A}ct(Q)$, where $\uplus$ denotes the multiset union. The first completed activity determines if the component behaves as P' or Q', where P' is the component which results from P completing the activity, and similarly Q'. The *cooperation* combinator models the synchronization and the cooperation among

components. We use the following notation: $P \underset{\mathcal{L}}{\bowtie} Q$ where $\mathcal{L}$ is a set of action types. All the activities in P and Q whose type is not in $\mathcal{L}$ are called individual and are not affected by the operator. On the other hand, the shared activities can be carried on only when they are enabled in both the components P and Q. This can cause a component to block waiting for the other. When the activity is enabled in both the components it is carried on with the rate of the slowest. This ensures that $P \underset{\mathcal{L}}{\bowtie} Q$ has an exponentially distributed state resident time. One of the activities contributing to the cooperation can be passive, i.e., it has an unspecified rate $(\alpha, \top)$. In this case the other activity determines the rate of the cooperation. If $\mathcal{L} = \emptyset$ the components carry on their activities independently. We call this case pure parallel combinator and we denote it by $P||Q$. The last combinator is the *abstraction*. We use the syntax $P/\mathcal{L}$ where P is a component and $\mathcal{L}$ a set of types. All the activities in P with type in $\mathcal{L}$ cannot be carried out in cooperation with other components, that is they are hidden and assume the unknown type τ. However, they still require a time to be completed. The hidden activities can be thought as internal processes of the component.

PEPA models analysis. In this paragraph we outline the steps needed to study a PEPA model. In order to obtain the performance measures of the system modelled by PEPA formalism, one needs to build the associated CTMC. The first step is to build the derivation graph of the model according to any of the algorithms available for the process algebra. The derivation graph describes all the possible evolutions of each component of the model. Noting that every transition of the derivation graph is associated with one or more exponentially distributed random times, it is easy to see how it can be mapped into a CTMC. Therefore, once the CTMC is defined one can derive its steady-state solution by exact or approximate techniques. Similarly to other formalisms, the main problem of this approach is that the number of the CTMC states tends to grow exponentially with the complexity of the model, hence its analysis can soon become unfeasible.

4.5 PEPA models in product-form

We can reformulate the product-form property in PEPA context as follows. Suppose P and Q are two interacting components, then we say that their stationary probability distribution is in product-form if it can be expressed as product of functions that that depend on the state of P or on the state of Q.

The investigation of product-form solutions for PEPA models can be classified as follows:

- Models with a reversible CTMC.
- Models with a quasi-reversible CTMC.

- Models based on the Boucherie's product-form [25].
- Models based on the Coleman, Henderson et. al. product-form [63, 39].
- Models based on RCAT theorem and extensions [59, 61]

In the following we briefly describe the former models and will spend more time for the latter one.

4.5.1 Reversible models.

The first product-form we illustrate is based on the reversibility of the associated CTMC. In [69] the authors give syntactical conditions which ensure that a PEPA model has a product-form solution. The basic idea is to use complementary types of activities, say α and $-\alpha$, that is, if it is possible to leave a state due to an activity of type α then in the arrival state there must be an activity of type $-\alpha$ which allows the system to get back to the previous state. α and $-\alpha$ forms a reverse pair. Clearly these conditions are quite strict. As quasi-reversible models are based on the same idea, we review it in the following paragraph.

4.5.2 Quasi-reversible models.

Quasi-reversible PEPA models are studied in [57]. This approach consists of two steps:

- The authors call QR-component a PEPA component whose CTMC is quasi-reversible. Then they introduce a subset of QR-components called input/output components that can be recognized syntactically. Informally we can say that input/output components play the same role of queueing stations in a product-form queueing network. It is important to note that the syntactical conditions are very useful in practice. In fact it is possible to check if a PEPA component is an input/output component without generating its derivation graph and CTMC.
- Then more complex components are studied as combinations of QR-components. Therefore the authors give a set of sufficient syntactical conditions on the co-operation operator that ensure that the Markov chain of the whole process is still quasi-reversible. They distinguish between closed interactions and open interactions, which can informally be associated with closed and open queueing networks, respectively. Note that these results are valid for QR-components class of models and not only for input/output model class.

It is worthwhile noting that, as the authors point out, the QR-component set is larger than the input/output component set. Let us review how input/output components are defined. A PEPA component P enables a reverse pair $(\alpha, -\alpha)$ if (α, r) is an

enabled activity in P and for every derivate component P' such that $P \overset{(\alpha,r)}{\rightarrow} P'$ there exists $(-\alpha, s)$ such that $P' \overset{(-\alpha,s)}{\rightarrow} P$, where r and s are positive real numbers or $\top$. If a PEPA component mimes a queueing network, we can think that the reverse pair can be associated with the arrival and completion events.

A PEPA component is an input/output component if it enables only two activities: the passive $(\alpha, \top)$ and the active $(-\alpha, r)$ where $r \in \mathbb{R}^+$ which forms a reverse pair.

Example 3 (Modelling M/M/1/FCFS stations) *In this example we model a queueing center with FCFS discipline by input/output PEPA models. We proceed by steps: first we consider a single class queueing center, then a multiclass one as defined in BCMP theorem [17]. It is well-know that the CTMC associated with a BCMP QN is quasi-reversible [75] hence we can try to model BCMP networks by QR-components. Before giving the PEPA component definition, we want to point out that for multiclass queueing stations the queueing discipline influences the stationary distributions and hence the performance indices [32, 33, 98].*

- *Single class queue. This case is trivial. In fact by the insensitivity property [74] we can represent the state of the system with the number of customers in queue or being served at a given time. Let λ be the arrival rate and μ the service rate of the $M/M/1/FCFS$ queue, with $\lambda, \mu \in \mathbb{R}^+$. Le us assume the system to be in a generic state different from the initial one, i.e., the empty station. We have to consider two possible events: a customer arrival, and a service completion. These two events form a reverse pair in the PEPA component. Let P_0 be the initial component, then:*

$$\begin{aligned} P_0 &\overset{\text{def}}{=} (arrive, \top).P_1 \\ P_n &\overset{\text{def}}{=} (arrive, \top).P_{n+1} + (serve, \mu).P_{n-1} \end{aligned}$$

  If we want the arrival process to be a Poisson process with rate λ then we can define an arrival component $A \overset{\text{def}}{=} (arrive, \lambda).A$ and the cooperation as $S \overset{\text{def}}{=} P_0 \underset{arrive}{\bowtie} A$. It is trivial to prove that P_n is an input/output component, thus it can be syntactically recognized as a QR-component. It is worthwhile noting that we have been able to define an input/output component because the state of this queueing system can be defined as the total number of customers that are present regardless to their arrival order. The equivalence can be stated for the steady state probabilities of observing a given number of customers in the station and is based on the idea that all the customers are identical. If one is interested in the response time distribution such implicit representation of the FCFS discipline is not appropriate.

- *Multiple class queue. In this case the state of the system cannot be represented just by the number of customers for each class because the arrival order must*

be considered. In fact the customers are not all identical. In this case we do not have an input/output component and the motivation is simple. Consider a generic component $Q_{\overline{n}} = Q_{(n_1,\overline{n'})}$ *modelling a station with* R *classes and FCFS discipline.* $\overline{n}$ *is a vector such that* n_i *is the class of the* i*-th oldest customer present in the station, where* $1 \leq n_i \leq R$*. The service completion event takes the PEPA component to* $Q_{\overline{n'}}$ *but it can be the case that the state* $Q_{\overline{n}}$ *cannot be reached by the state* $Q_{\overline{n'}}$ *because of a customer arrival.*

Therefore, even if BCMP QNs can certainly be modeled by QR-components, to the best of our knowledge they have not been modelled by input/output components yet. As a consequence their product-form solution cannot be decided by a syntactical (structural) analysis.

It should be clear that every Jackson [73] and Gordon-Newell [54] queueing network can be modelled by a set of input/output components because these networks require all the customers to be statistically identical.

4.5.3 Coleman, Henderson et al. product-form

Coleman, Henderson et. al. have studied a product-form for Stochastic Petri Nets in [63, 39] which is based on the definition of a routing process on the CTMC associated with the model. In [110] the author applies this idea to obtain a class of product-form PEPA models. In this case, the conditions for product-form require to solve a linear system of equations. We point out some aspects of this technique:

- It can be used to model Jackson and Gordon-Newell queueing networks.
- From a modelling point of view, it is not clear the meaning of the conditions based on the existence of the solution of a linear system derived from the syntactic structure and rates of the model.

4.5.4 Boucherie's product-form

Another product-form class, which can be easily re-formulated in terms of PEPA conditions is the Boucherie's product-form. Also in this case it has been initially defined for Stochastic Petri Nets [25]. As it is not the topic of this thesis we cite it informally. Roughly speaking, the condition for the product-form requires that if two agents compete for the same resource, i.e., they must synchronize on it, when one agent is in a state which requires that resource the other agent is blocked, i.e., cannot change its state.

4.5.5 RCAT, ERCAT, MARCAT

Probably the most interesting result for MPA product-forms is the Reversed Compound Agent Theorem (RCAT) [59] and its extension [61, 58]. The importance of this theorem is twofold. From a theoretical point of view, it has been shown that it includes other product-form model classes such as Boucherie's one [61], G-Networks [59, 60], Jackson product-form QNs [59]. Moreover, as original contribution we show in Chapter 5 of this thesis that it includes also Coleman, Henderson et al. product-form. Let us introduce now the main theorems used in the following. Roughly speaking, RCAT derives the steady state probabilities of two interacting components, say P and Q, by the analysis of the reversed processes of two components R and S obtained by replacing in P and Q the occurrence of the passive action type transitions with rates that can be algorithmically calculated. As first let us define how a PEPA component can be reversed *structurally*.

We consider just the cooperation and the prefix combinators. Reversing the arrows in the CTMC of a PEPA model is trivial. In fact, it suffices to reverse the derivation as follows:

$$\begin{aligned} A \stackrel{def}{=} (\alpha, \lambda).P \quad &\Longrightarrow \quad \overline{P} \stackrel{def}{=} (\overline{\alpha}, \overline{\lambda}).\overline{A} \\ A \stackrel{def}{=} Q \underset{L}{\bowtie} R \quad &\Longrightarrow \quad \overline{A} \stackrel{def}{=} \overline{Q} \underset{\overline{L}}{\bowtie} \overline{R}, \end{aligned}$$

where $\overline{A}, \overline{P}, \overline{Q}, \overline{R}$ represent the reversed agents, $\overline{\lambda}$ the rate of the reverse action (usually to be determined), L is the set of action types involved in the synchronization, and $\overline{L} = \{\overline{\alpha} | \alpha \in L\}$ is the set of action types involved in the synchronization of the reversed components. Of course, the problem consists in finding the rates of the reversed actions. Consider a simple agent, i.e., an agent without cooperation. Determining the rates of the reversed process of a single agent has the same complexity of finding its steady state probability distribution. From the reversed CTMC it is straightforward deriving the reversed action rates paying attention to the case of multiple actions of a component having the same derivate. In this case, the total reverse rate is distributed amongst the reversed arcs in proportion to the corresponding forward transition rates.

As this situation will often occur in the following chapters, we show an example and then give the general formula. Suppose that an agent P is defined by the following rules:

$$\begin{aligned} &\vdots \\ &P \stackrel{def}{=} (\alpha, \lambda_1).P' + (\beta, \lambda_2).P' \\ &\vdots \end{aligned}$$

In PEPA this is a common situation. In fact, even if both action types α and β cause a transition from P to P', in cooperation with another agent they can synchronize with different actions. Consider for example the following definition:

$Q \stackrel{def}{=} (\alpha, \top).Q + (\beta, \top).Q'$ and the cooperation $S \stackrel{def}{=} P \underset{\alpha,\beta}{\bowtie} Q$. If Q synchronizes on action type α then we have a transition from $(P;Q)$ to (P',Q), if Q synchronizes with β then we have a transition from $(P;Q)$ to (P',Q').

However, if we study the CTMC $\mathcal{C}_P$ of P in isolation following the rules defined in [68] we have a transition from P to P' with rate $\lambda_1 + \lambda_2$. Suppose that we can reverse $\mathcal{C}_P$ and let $\overline{\lambda}$ be the rate of the reversed transition from P' to P. In order to obtain the definition of the reversed agent we have to determine $\overline{\lambda_1}$ and $\overline{\lambda_2}$. As they are proportional to the forward rates, we have: $\overline{\lambda_1} = \overline{\lambda}\lambda_1/(\lambda_1 + \lambda_2)$ and similarly for $\overline{\lambda_2}$.

In general we use the following definition:

Definition 5 (Reversed actions of multiple actions [59]) *The reversed actions of multiple actions (a_i, λ_i) for $1 \leq i \leq n$ that an agent P can perform, which lead to the same derivate Q, are respectively:*

$$(\overline{a_i}, (\lambda_i/\lambda)\overline{\lambda}),$$

where λ is the sum of the forward rates $\lambda = \sum_{i=1}^{n} \lambda_i$ and $\overline{\lambda}$ is the reversed rate of the (composite) transition in the CTMC with rate λ corresponding to all the arcs between P and Q.

Finally, we recall that RCAT-based theorems only deal with cooperations where an action is active (i.e. with a specified rate) and the other is passive ($\top$).

Let us consider a cooperation $A \stackrel{def}{=} P \underset{L}{\bowtie} Q$. Let $\mathcal{P}_P(L)$ denote the set of action types of L passive with respect to P and $\mathcal{A}_P(L)$ the set of action types of L active with respect to P. Hence, $\mathcal{P}_P(L) \cup \mathcal{A}_P(L) = \mathcal{P}_Q(L) \cup \mathcal{A}_Q(L) = L$. An action type a is enabled in a component if it can carry out an activity with type a.

Theorem 3 (RCAT [59]) *Let us assume that $P \underset{L}{\bowtie} Q$ has an irreducible derivation graph. If the following conditions hold:*

1. *every passive action in $\mathcal{P}_P(L)$ or $\mathcal{P}_Q(L)$ is always enabled in P or in Q (i.e. enabled in all the states of the transition graph),*
2. *every reversed action of an active action type in $\mathcal{A}_P(L)$ or $\mathcal{A}_Q(L)$ is always enabled in $\overline{P}$ or $\overline{Q}$,*
3. *every occurrence of a reversed action of an active action type in $\mathcal{A}_P(L)$ ($\mathcal{A}_Q(L)$) has the same rate in $\overline{P}$ ($\overline{Q}$),*

then the reversed agent $\overline{P \underset{L}{\bowtie} Q}$ has the following derivation graph:

$$\overline{R}\{(\overline{\alpha}, \overline{p_\alpha}) \leftarrow (\overline{\alpha}, \top) | \alpha \in \mathcal{A}_P(L)\} \underset{\overline{L}}{\bowtie} \overline{S}\{(\overline{\alpha}, \overline{q_\alpha}) \leftarrow (\overline{\alpha}, \top) | \alpha \in \mathcal{A}_Q(L)\},$$

where:

- *the arrow* $\leftarrow$ *stands for a syntactical substitution of the left hand side part with the right hand side part in the component definition*
- $R = P\{\top_a \leftarrow x_\alpha | \alpha \in \mathcal{P}_P(L)\}$ *and* $S = Q\{\top_a \leftarrow x_\alpha | \alpha \in \mathcal{P}_Q(L)\}$ *where* $\{x_\alpha\}$ *are the solutions (for* $\top_\alpha$*) of the equations:*

$$\top_\alpha = \overline{q_\alpha} \qquad \alpha \in \mathcal{P}_P(L) \tag{4.1}$$
$$\top_\alpha = \overline{p_\alpha} \qquad \alpha \in \mathcal{P}_Q(L) \tag{4.2}$$

and $\overline{p_\alpha}$ *and* $\overline{q_\alpha}$ *are the symbolic rates of action type* $\overline{\alpha}$ *in* $\overline{P}$ *and* $\overline{Q}$*.*

For comments and examples of applications of this theorem we refer the original paper [59], in this section we just show some simple applications. It can be worthwhile recalling that even if RCAT does not explicitally state that $P \bowtie Q$ is in product-form, this can be easily derived by the reversed process definition [59].

Example 4 (Simple tandem of exponential queues) *This is a very simple example that introduces the RCAT approach. We consider two exponential queues* P *and* Q *with service rates* μ_1 *and* μ_2*. Customers arrive to queue* P *according to a Poisson process with rate* λ*. After being served, the customers leave queue* P *and enter queue* Q*. After being served by* Q *the customers leave the system. The model is depicted in Figure 4.1. We can describe this model* S *by the following PEPA*

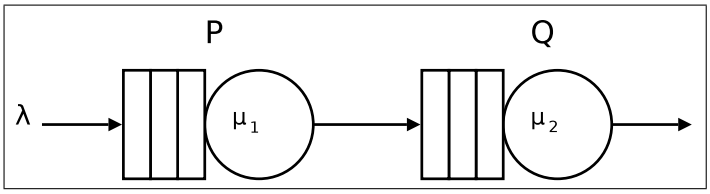

Figure 4.1: Tandem of exponential queues.

definitions:

$$\begin{aligned}
P_0 &\stackrel{\text{def}}{=} (a, \lambda).P_1 \\
P_n &\stackrel{\text{def}}{=} (a, \lambda).P_{n+1} + (e_p, \mu_1).P_{n-1} \qquad \textit{for } n > 0 \\
Q_0 &\stackrel{\text{def}}{=} (e_p, \top).Q_1 \\
Q_n &\stackrel{\text{def}}{=} (e_p, \top).Q_{n+1} + (e_q, \mu_2).Q_{n-1} \qquad \textit{for } n > 0 \\
S &\stackrel{\text{def}}{=} P_0 \underset{e_p}{\bowtie} Q_0,
\end{aligned}$$

Let us assume that the system is stable. We can immediately check RCAT structural conditions. In fact the synchronization occurs on action type e_p *that is passive and always enabled in* Q*, i.e., in each state of the derivation graph of* Q *there is*

an outgoing transition labeled by e_p. Moreover, every instance of P can be reached through an active transition of type e_p (i.e. the reversed action type $\overline{e_p}$ corresponding to e_p is always enabled in $\overline{P}$). We have to determine a solution for x_{e_p}. As the CTMC of P is a birth and death process we straightforwardly obtain $x_{e_p} = \lambda$ and it is constant. Hence the model is in product-form. The steady state solution for P, i.e., the probability of observing the component P_n is $\pi_P(n) \propto (\lambda/\mu_1)^n$. In Q we replace the unknown rate $\top$ of the passive action e_p with the reversed rate of the corresponding active action in P, i.e. λ, obtaining again a birth and death process. Therefore the steady state solution is:

$$\pi(n,m) \propto \pi_P(n)\pi_Q(m) = (\lambda/\mu_1)^n(\lambda/\mu_2)^m.$$

Note 1 *It is worthwhile pointing out an important aspect of the application of RCAT for the analysis of non-PEPA models. Let us consider the tandem of exponential queues of Example 4. We could model the same process using the following PEPA definitions:*

$$\begin{aligned}
P_0 &\stackrel{\text{def}}{=} (a,\lambda).P_1 \\
P_n &\stackrel{\text{def}}{=} (a,\lambda).P_{n+1} + (e_p,\top).P_{n-1} \qquad \textit{for } n > 0 \\
Q_0 &\stackrel{\text{def}}{=} (e_p,\mu_1).Q_1 \\
Q_n &\stackrel{\text{def}}{=} (e_p,\mu_1).Q_{n+1} + (e_q,\mu_2).Q_{n-1} \qquad \textit{for } n > 0 \\
S &\stackrel{\text{def}}{=} P_0 \underset{e_p}{\bowtie} Q_0.
\end{aligned}$$

Obviously, the CTMC underlying these definitions is exactly the same than that underlying the definitions of Example 4. However, in this case, RCAT cannot be applied because the structural conditions are not satisfied, e.g. passive action e_p is not enabled in P_0. Moreover, if one tries to apply RCAT regardless to its structural conditions, the result is not correct. In fact $\pi_Q(m)$ expression would be proportional to $(\mu_1/\mu_2)^m$ instead of $(\lambda/\mu_2)^m$.

In conclusion, if we aim to study the product-form solution of a stochastic model that is not defined in terms of PEPA formalism, we cannot limit our analysis to the underlying CTMC but we also have to give a suitable PEPA definition that allows for the application of RCAT.

Example 5 (Two queues with feedback) *This example aims to show how Definition 5 can be used to solve product-form models. We consider a model consisting of two exponential queues P and Q with service rates μ_1 and μ_2, respectively. The customers arrive according to a Poisson process to P, then after being served they enter into queue Q. After the job completion in Q they can either leave the system with probability p or go back to P with probability $1-p$. Figure 4.2 illustrates the model. We can describe this model S by the following PEPA definitions:*

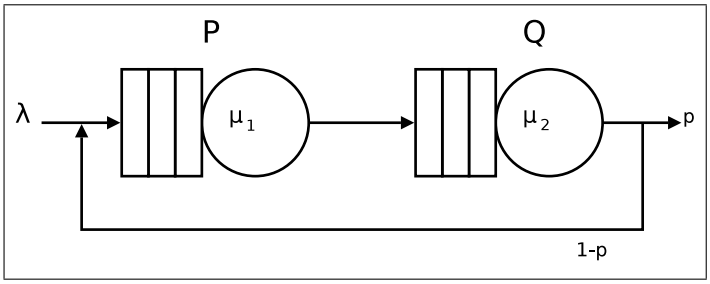

Figure 4.2: Simple network of exponential queues with feedback.

$$
\begin{aligned}
P_0 &\stackrel{\text{def}}{=} (a,\lambda).P_1 + (f_q,\top).P_1 \\
P_n &\stackrel{\text{def}}{=} (a,\lambda).P_{n+1} + (f_q,\top).P_{n+1} + (e_p,\mu_1).P_{n-1} \quad \textit{for } n>0 \\
Q_0 &\stackrel{\text{def}}{=} (e_p,\top).Q_1 \\
Q_n &\stackrel{\text{def}}{=} (e_p,\top).Q_{n+1} + (e_q,\mu_2 p).Q_{n-1} + (f_q,\mu_2(1-p)).Q_{n-1} \quad \textit{for } n>0 \\
S &\stackrel{\text{def}}{=} P_0 \underset{e_p}{\bowtie} Q_0,
\end{aligned}
$$

Note that we have two possible transitions from P_n to P_{n+1}, $n \geq 0$. One is labeled with f_q and the other with a. Similarly in Q the transition due to a job completion is splitted into two transitions labeled by e_q and e_q that denote that a customer exits the system or goes back to P respectively. Figure 4.3 shows the processes of P and Q.

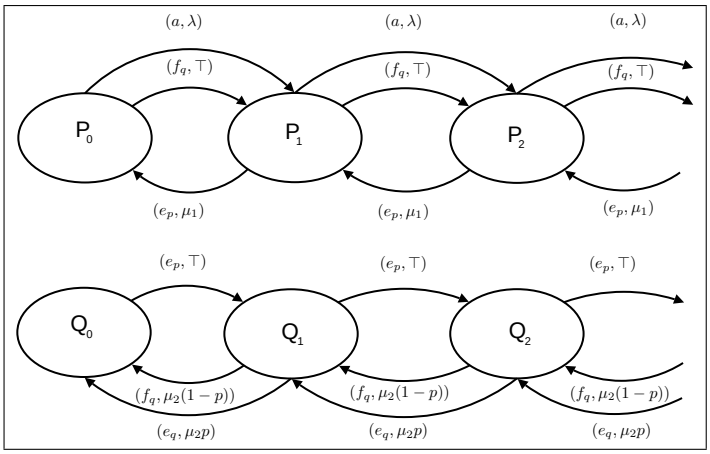

Figure 4.3: Processes associated with the queueing system of Example 5.

If we derive the CTMC of P (Q) we have a birth and death process whose birth rate is $\lambda + x_{f_q}$ (x_{e_p}) and the death rate is μ_1 (μ_2). Note that the structural conditions of RCAT are trivially satisfied. In order to obtain x_{e_p}, i.e., the (constant) reversed rate of the active actions labelled by e_p, we observe that, in the CTMC, it is the

reversed rate of the death transitions: $x_{e_p} = \lambda + x_{f_q}$. *Note that the reversed rate of the death rate in the CTMC of* Q *is* not x_{f_q} *because the forward rate is the sum of the transitions corresponding to action types* e_q *and* f_q. *Therefore applying Definition 5 we obtain* $x_{f_p} = x_{e_p}(1-p)$. *Now we have to solve the traffic equation systems:*

$$\begin{cases} x_{f_p} = x_{e_p}(1-p) \\ x_{e_p} = \lambda + x_{f_q}, \end{cases}$$

that gives $x_{e_p} = \lambda/p$ *and* $x_{f_q} = \lambda + \lambda/p$. *Then, the joint steady state proability of observing* n *customers in* P *and* m *in* Q *is given by:*

$$\pi(n,m) \propto \left(\frac{\lambda}{p\mu_1}\right)^n \left(\frac{\lambda}{p\mu_2}\right)^m$$

ERCAT is a theorem introduced in [61] that generalizes RCAT by relaxing its structural conditions. The requirement of having the passive actions always enabled both in the forward and reversed agents guarantees that the total flow out of the forward and the reversed cooperation is the same for each state. We can avoid these structural requests by introducing an appropriate balance on the flow in and out of the states. The goal is being able to decide whether a cooperation of two agents P and Q is in product-form by the analysis of the agents in isolation. Before recalling the theorem we need to introduce some additional notation, as presented in [61]. Suppose we have two agents P and Q and we want to study the steady state probabilities of $P \underset{L}{\bowtie} Q$. Then we define the following subset of action types in L, where A stands for the agent P or Q:

- $\mathcal{P}_A(L)$: denotes the subset that are passive in A,
- $\mathcal{A}_A(L) = L \setminus \mathcal{P}_A(L)$: denotes the subset that are active in A,
- $\mathcal{P}_A^{i\rightarrow}$ denotes the subset that are passive in A and correspond to transitions out of state i in the Markov process of A,
- $\mathcal{P}_A^{i\leftarrow}$ denotes the subset that are passive in A and correspond to transitions into state i in the Markov process of A,
- $\mathcal{A}_A^{i\rightarrow}$ denotes the subset that are active in A and correspond to transitions out of state i in the Markov process of A,
- $\mathcal{A}_A^{i\leftarrow}$ denotes the subset that are active in A and correspond to transitions into state i in the Markov process of A,
- $\mathcal{P}^{(i,j)\rightarrow} = \mathcal{P}_P^{i\rightarrow} \cup \mathcal{P}_Q^{j\rightarrow}$ and $\mathcal{A}^{(i,j)\rightarrow} = \mathcal{A}_P^{i\rightarrow} \cup \mathcal{A}_Q^{j\rightarrow}$,
- $\mathcal{P}^{(i,j)\leftarrow} = \mathcal{P}_P^{i\leftarrow} \cup \mathcal{P}_Q^{j\leftarrow}$ and $\mathcal{A}^{(i,j)\leftarrow} = \mathcal{A}_P^{i\leftarrow} \cup \mathcal{A}_Q^{j\leftarrow}$,

- $\alpha_a^{(i,j)}$ denotes the instantaneous transition rate out of joint state (i,j) in the Markov process of $P \underset{L}{\bowtie} Q$ corresponding to active action type $a \in L$,
- $\overline{\beta_a^{(i,j)}}$ denotes the instantaneous transition rate out of state (i,j) in the reversed Markov process of $P \underset{L}{\bowtie} Q$ corresponding to passive action type $a \in L$.

Theorem 4 (ERCAT [61]) *Suppose that the cooperation $P \underset{L}{\bowtie} Q$ has a derivation graph with an irreducible subgraph G. Given that every occurrence of a reversed action of an active action type in $\mathcal{A}_P(L)$ (respectively $\mathcal{A}_Q(L)$) has the same rate in $\overline{P}$ (respectively $\overline{Q}$), the reversed subgraph $\overline{G}$ is defined by the derivation graph of the reversed agent $\overline{P \underset{L}{\bowtie} Q} =$*

$$\overline{R}\{(\overline{a},\overline{p_a}) \leftarrow (\overline{a},\top) | a \in \mathcal{A}_P(L)\} \underset{\overline{L}}{\bowtie} \overline{S}\{(\overline{a},\overline{q_a}) \leftarrow (\overline{a},\top) | a \in \mathcal{A}_Q(L)\},$$

where

$$\begin{aligned} R &= P\{\top_a \leftarrow x_a | a \in \mathcal{P}_P(L)\} \\ S &= Q\{\top_a \leftarrow x_a | a \in \mathcal{P}_Q(L)\}, \end{aligned}$$

x_a are the solutions (for $\top_a$) of the equations:

$$\begin{aligned} \top_a &= \overline{q_a} \quad a \in \mathcal{A}_Q(L) \\ \top_a &= \overline{p_s} \quad a \in \mathcal{A}_P(L), \end{aligned} \tag{4.3}$$

and $\overline{p_a}$ (respectively $\overline{q_a}$) is the symbolic rate of action type $\overline{a}$ in $\overline{P}$ (respectively $\overline{Q}$), provided that the underlying Markov chain is ergodic (has a steady state) and:

$$\sum_{a \in \mathcal{P}^{(i,j)\rightarrow}} x_a - \sum_{a \in \mathcal{A}^{(i,j)\leftarrow}} x_a = \sum_{a \in \mathcal{P}^{(i,j)\leftarrow} \setminus \mathcal{A}^{(i,j)\leftarrow}} \overline{\beta_a^{(i,j)}} - \sum_{a \in \mathcal{A}^{(i,j)\rightarrow} \setminus \mathcal{P}^{(i,j)\rightarrow}} \alpha_a^{(i,j)}. \tag{4.4}$$

Example 6 *In this example we use ERCAT to study a Boucherie's product-form. In [61] the author proves that any Boucherie's product-form can be studied by ERCAT, however, for the sake of simplicity, we work on a simple model, i.e., that illustrated by Example 1 in Chapter 1. We have two processes, P_1 and P_2, that alternate three states a_i, b_i, c_i for $i = 1, 2$. The two processes compete for a resource when they are in state c_i. However, Boucherie's product-form requires a stricter blocking mechanism, i.e., when one of the two processes is in state c_i the other one is always blocked. Note that, even if we consider that the transition rates among the sates are identical, the model can be reformulated in order to overcome this limitation. The behavior of P_1 in PEPA can be described as:*

$$\begin{aligned} a_1 &\overset{\text{def}}{=} (a,\lambda_1).b_1 + (b,\top).a_1 \\ b_1 &\overset{\text{def}}{=} (a,\lambda_1).c_1 + (b,\top).b_1 \\ c_1 &\overset{\text{def}}{=} (a,\lambda_1).a_1, \end{aligned}$$

and P_2*:*

$$\begin{aligned} a_2 &\stackrel{\text{def}}{=} (b, \lambda_2).b_2 + (a, \top).a_2 \\ b_2 &\stackrel{\text{def}}{=} (b, \lambda_2).c_2 + (a, \top).b_2 \\ c_2 &\stackrel{\text{def}}{=} (b, \lambda_2).a_2. \end{aligned}$$

The whole system is $S = a_1 \underset{a,b}{\bowtie} a_2$ *(see Figure 4.4). Note that RCAT structural conditions do not hold since passive actions* a *and* b *are not enabled in states* c_2 *and* c_1 *respectively. Therefore, we apply ERCAT.*

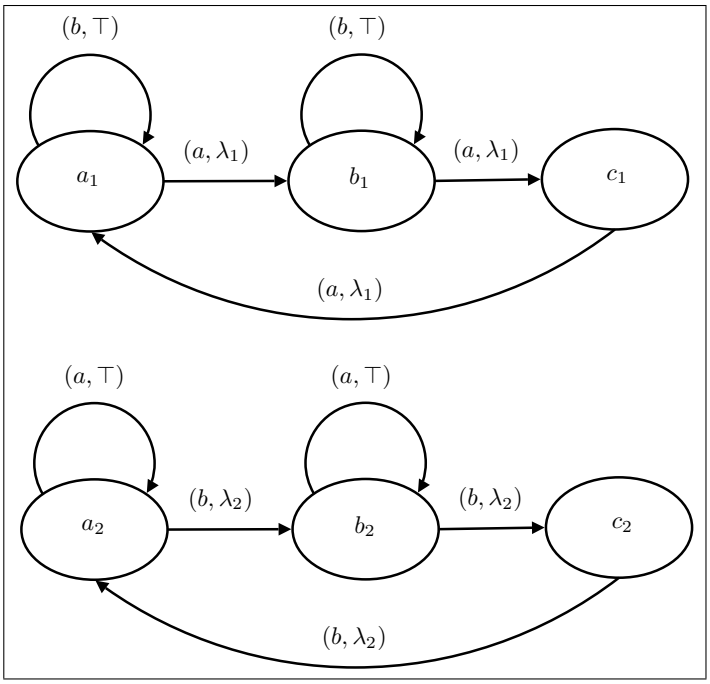

Figure 4.4: Processes associated with the model of Example 6.

Note that states a_i *and* b_i *have the same input and output transition labels and rates, therefore we can check the ERCAT conditions (4.4) for the pairs of states* (α_1, α_2) *(where* $\alpha_i = a_i, b_i$ *for* $i = 1, 2$*),* (α_1, c_2) *(we can derive* (c_1, α_2) *by symmetry) and finally* (c_1, c_2)*.*

- (α_1, α_2)*. In this case we have that:*

$$\begin{aligned} \mathcal{P}^{(\alpha_1,\alpha_2)\rightarrow} &= \{a, b\} \\ \mathcal{A}^{(\alpha_1,\alpha_2)\leftarrow} &= \{a, b\} \\ \mathcal{P}^{(\alpha_1,\alpha_2)\leftarrow} &= \{a, b\} \\ \mathcal{A}^{(\alpha_1,\alpha_2)\rightarrow} &= \{a, b\}, \end{aligned}$$

that leads to: $x_a + x_b - x_a - x_b = 0$*, that is trivially satisfied.*

- (α_1, c_2). *In this case we have that:*

$$\begin{aligned}\mathcal{P}^{(\alpha_1,c_2)\rightarrow} &= \{b\}\\ \mathcal{A}^{(\alpha_1,c_2)\leftarrow} &= \{a,b\}\\ \mathcal{P}^{(\alpha_1,c_2)\leftarrow} &= \{b\}\\ \mathcal{A}^{(\alpha_1,c_2)\rightarrow} &= \{a,b\},\end{aligned}$$

  that leads to: $x_b - x_a - x_b = -\alpha_a(\alpha_1, c_2)$. *Since* $\alpha_a(\alpha_1, c_2) = \lambda_1$, *we have* $x_a = \lambda_1$. *In the same way we derive* $x_b = \lambda_2$.

- (c_1, c_2). *In this case we have that:*

$$\begin{aligned}\mathcal{P}^{(c_1,c_2)\rightarrow} &= \emptyset\\ \mathcal{A}^{(c_1,c_2)\leftarrow} &= \{a,b\}\\ \mathcal{P}^{(C_1,c_2)\leftarrow} &= \emptyset\\ \mathcal{A}^{(c_1,c_2)\rightarrow} &= \{a,b\},\end{aligned}$$

  that leads to: $-x_a - x_b = -\alpha_a(c_1, c_2) - \alpha_b(c_1, c_2)$ *that is satisfied for* $x_a = \lambda_1$ *and* $x_b = \lambda_2$.

Finally, we have to derive x_a *and* x_b. *By Kolmogorov's criteria the sum of the outgoing rates in the forward and in the reversed processes is the same, therefore we have immediately that the reversed rate of* λ_1 *(*λ_2*) is* $x_a = \lambda_1$ *(*$x_b = \lambda_2$*), that is coherent with ERCAT conditions.*

Therefore, we have product-form and its expression is the same given by Boucherie's analysis.

ERCAT theorem has been generalized in [62] in order to deal with multiple-agent, pairwise cooperations. In this section we just introduce the formal definition of this kind of cooperation and, for the sake of brevity, omit to state the whole theorem that can be seen as a straightforward generalization ERCAT. Let us define $\bowtie_{L}^{n}{}_{k=1} P_k$, with $n \geq 2$, where $L = \cup_{k=1}^{n} L_k$ and $L_k = \mathcal{P}_k \cup \mathcal{A}_k$ is the set of synchronizing action types that occur in agent P_k. Agents cooperate pairwise. Therefore, we define the semantic of multi-agent cooperation as:

$$\bowtie_{L}^{n}{}_{k=1} P_k = \Big(\cdots \Big(\Big(P_1 \bowtie_{M_2} P_2 \Big) \bowtie_{M_3} P_3 \Big) \bowtie_{M_4} \cdots \bowtie_{M_{n-1}} P_{n-1} \Big) \bowtie_{M_n} P_n,$$

where $M_k = L_k \cap \Big(\cup_{k=1}^{k-1} L_j \Big)$. Roughly speaking, the pairwise condition of the multi-agent cooperation states that if we have a synchronized transition then it involves just a pair of agents, but, in general, an agent can synchronize with any number of agents. Intuitively, MARCAT generalizes ERCAT by defining a unique system of equations to solve in order to determine the reversed rates of the active action types. This makes the computation of the product-form solution more efficient.

For example, let us consider a Jackson QN. The analysis by ERCAT requires to study the joint product-form process of just two nodes. Then, the analyzer should consider this joint process and combine it with the process corresponding to another node, and so on. Note that in a Jackson QN the synchronization at a job copletion time involves just two nodes. Then, one can study the whole net simultaneously by deriving the set of equations for the reversed rates that is proven to be equivalent to the traffic equations of the corresponding QN [62].

4.6 Conclusions

In this chapter we informally reviewed the PEPA formalism [68] and introduced the main results about product-form PEPA models. The main strength of PEPA formalism is its compositionality, i.e., a modeler can define an agent behavior in isolation and then combine it with other agents by the cooperation $\bowtie$. For example one can describe the behavior of a web server W and a web client C and then obtain an evaluation of the performance indices depending on the number of clients. Instantiating a set of clients is syntactically easy, because it suffices to use the $\bowtie$ operator appropriately.

Among the various analysis that can be done on a PEPA model, we are mainly interested in the analysis of the steady state probability distributions, for those models that admit it. In this context, as seen for other formalisms in the previous chapters, the generation of the infinitesimal generator $\mathbf{Q}$ of the CTMC, and the solution of the global balance equations system can become unfeasible even for relatively small models. Product-forms partially overcome this problem because they allow the analyzer to derive the steady state probabilities through the steady state probabilities of the single agents. Of course, this can be done under appropriate conditions. Several results have been presented in order to characterize product-form PEPA modes. We mainly focused on the most recent ones, i.e., the Reverse Compound Agent Theorem (RCAT) and its extensions (ERCAT, MARCAT). It defines some structural and behavioral conditions that can be decided by the analysis of the forward and reversed CTMC of the cooperating agents. It has been shown that RCAT can identify the Jackson and Gordon-Newell queueing networks product-forms [59], a subclass of BCMP queueing networks [58], a subclass of product-form G-networks [60] as well as new product-forms [61]. It is worthwhile pointing out that it is not RCAT purpose to give an efficient algorithm to obtain the steady state probabilities of single agents, or to solve the system of equations (4.1). In fact it assumes these to be known and then it derives the product-form of the composed agent.

The main strength of RCAT-based theorems is that they inherit the compositional power and the high expressivity of the PEPA language. For example, in the following chapter we use this result in the context of product-form SPN, and in Chapter 6 in the context of GSPN.

II

Contributions

5

A new glance on product-form SPNs using RCAT results

5.1 Introduction

This chapter illustrates some new results on product-form Stochastic Petri Nets (SPNs). This is a joint work with P. G. Harrison (Imperial College, London).

In Chapter 3 we illustrated several results on stochastic Petri Nets (SPNs) in product-form. In particular we reviewed the results of Coleman, Henderson et al. given in [63, 39] (for the sake of brevity, in this chapter we refer to this model class as CH-SPN). In Chapter 4 we reviewed RCAT-based theorems and pointed out that they can be applied to other models than PEPA if the stochastic process of these models can be conveniently expressed in terms of pairwise compositions of PEPA agents. In this chapter we use this idea and apply ERCAT [61] and MARCAT [62] results in order to study CH-SPNs. By this approach we establish a relation between the product-form stochastic processes associated with CH-SPNs and the stochastic processes that can be studied by RCAT-based theorems.

The main contributions of this work can be summarized as follows:

- we define a subclass of CH-SPNs, named CHC-SPN, where the transitions of the SPN have constant firing rates and all the arc weights are 1, i.e., batch token movements are not allowed,
- we define a CHC-SPN *building block*, i.e., a CHC-SPN pattern in which any CHC-SPN can be decomposed,
- we study the building block by ERCAT deriving the conditions for the product-form solution and its form,
- we prove that any CHC-SPN $\mathcal{S}$ can be decomposed into interconnected building blocks and that the steady state probabilities of $\mathcal{S}$ are in product-form by using MARCAT,
- we define an algorithm that identifies the building blocks in a CHC-SPN,

- we analyze the compositional and hierarchical properties of this new approach to product-form SPNs.

Note that even if CHC-SPNs have strong restrictions compared to CH-SPNs they can still model complex behaviors such as the fork and join constructs. In the section dedicated to the conclusions we discuss how these limitations can be overcome.

There are several practical consequences of these results. First of all, we have a new interpretation of CH-SPN product-form theorem. In particular the condition on the matrix rank (3.9) is now expressed in terms of ERCAT conditions, therefore its motivation is no more purely algebraical. Moreover, the CHC-SPN product-form modularity is enhanced. In fact, a CHC-SPN can be composed with models expressed in terms of other formalisms whose stochastic processes satisfy RCAT conditions maintaining the product-form property. For example, it is possible to combine a CHC-SPN with a product-form G-network [50] obtaining a product-form solution because product-form G-networks can be studied by RCAT [60]. Note that when a building block $\mathcal{B}$ is composed with a CHC-SPN $\mathcal{S}$ the modeler can derive the solution of the new model $\mathcal{S}'$ using the steady state probabilities known for $\mathcal{S}$ and $\mathcal{B}$. In Coleman, Henderson et al. approach this operation would require a new analysis of $\mathcal{S}'$.

5.2 The building block

The purposes of this section are twofold. First, we show how it is possible to use RCAT-based theorems (see Chapter 4 for a quick review) to study SPNs and second we introduce a central concept for the following results, i.e., the structure and the analysis of the *building block*. The building block plays a central role in this work because, as we show in Section 5.3, every CHC-SPN can be partitioned into several cooperating building blocks. Then, we use the analysis of the building blocks to decide whether the whole model is in product-form and, if this is the case, to compute its steady-state probabilities.

5.2.1 An introductory example

For the sake of clarity, before proving the general case we study a special simple case called *basic building block model* (BBB). Let us consider the SPN model illustrated in Figure 5.1. According to the notation illustrated in Chapter 3, χ_y ($\chi_{y'}$) denotes the intrinsic firing rate of transition T_y (T'_y). We assume that for every state $\mathbf{m}$, the potential function ψ and ϕ are defined as follows: $\phi(\mathbf{m}) = \psi(\mathbf{m}) = 1$. So the firing rate of transition T_y is marking independent:

$$w(T_y, \mathbf{m}) = \chi_y,$$

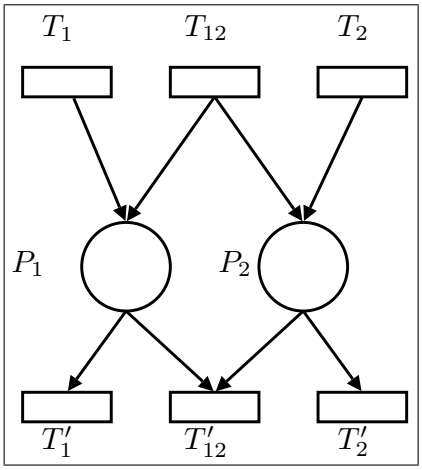

Figure 5.1: A basilar building block model (BBB).

and similarly for T'_y.

In the study of a building block we use these conventions: transitions T_y are always enabled, we call them input transitions and denote the set of all T_y by $\mathcal{T}_I$. Transitions T'_y are transitions with null output vector, we call them output transitions and denote the set of all T'_y by $\mathcal{T}_O$. The subscript y in T_y (T'_y) is a set containing the indices of the output places (input places) for the input (output) transition T_y (T'_y). Usually, we write y as subscript without parenthesis. Moreover, in order to make the following formulas more familiar to those who are comfortable with queueing theory, we set $\chi_y = \lambda_y$ (the arrival rates) and $\chi'_y = \mu_y$ (the service rates).

In order to model the BBB in PEPA we identify two agents that correspond to the two places of the SPN model. The state of the agent is given by the number of tokens in the place. Then, using a PEPA semantic, we can write:

$$
\begin{array}{rcll}
P_0^1 & \stackrel{def}{=} & \lambda_1.P_1^1 + (t_{12}, \lambda_{12}).P_1^1 & \\
P_n^1 & \stackrel{def}{=} & \mu_1.P_{n-1}^1 + (t'_{12}, \top'_{12}).P_{n-1}^1 & \\
 & + & \lambda_1.P_{n+1}^1 + (t_{12}, \lambda_{12}).P_{n+1}^1 & n > 0 \\
P_0^2 & \stackrel{def}{=} & \lambda_2.P_1^2 + (t_{12}, \top_{12}).P_1^2 & \\
P_n^2 & \stackrel{def}{=} & \mu_2.P_{n-1}^2 + (t'_{12}, \mu_{12}).P_{n-1}^2 & \\
 & + & \lambda_1.P_{n+1}^1 + (t_{12}, \top_{12}).P_{n+1}^1 & n > 0 \\
S & \stackrel{def}{=} & P_0^1 \underset{L}{\bowtie} P_0^2, &
\end{array}
$$

where $L \stackrel{def}{=} \{t_{12}, t'_{12}\}$, S is the agent that models BBB. Figure 5.2 gives an intuitive representation of the definitions of P_n^1 and P_n^2. Note that P_n^1 controls the synchronized arrivals (t_{12} is active in P_n^1) while P_n^2 controls the synchronized departures (t'_{12} is active in P_n^2).

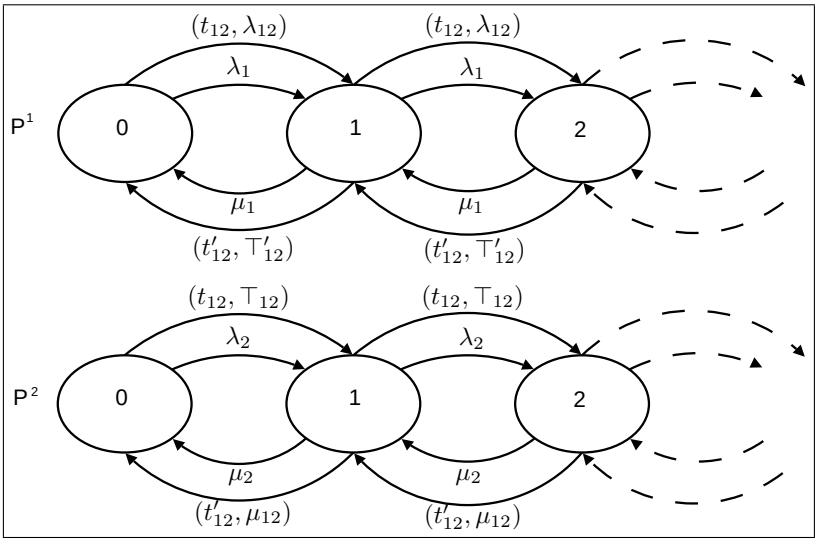

Figure 5.2: Graphical description of P_n^1 and P_n^2 of BBB.

We can note that the structural conditions of RCAT are not satisfied because the action with type t'_{12} is not enabled in every derivative of P^1. This is because a synchronized departure is not possible if place P^1 is empty. Therefore, we have to check the conditions to apply ERCAT.

ERCAT application First of all, we study the necessary conditions (4.4). Even if these conditions require to study all the pairs (P_m^1, P_n^2) with $m, n \geq 0$ we can reduce this test to only four cases. In fact, it is easy to note that the outgoing and incoming transitions and rates for all the states P_k^1 (P_k^2) with $k > 0$ are the same. Therefore, we have to analyze the pairs $(0,0)$, $(0,k)$, (k,k), $(k,0)$:

- $(0,0)$. For this pair we have the following sets:

$$\mathcal{P}^{(0,0)\rightarrow} = \{t_{12}\} \qquad \mathcal{A}^{(0,0)\leftarrow} = \{t'_{12}\}$$
$$\mathcal{P}^{(0,0)\leftarrow} \setminus \mathcal{A}^{(0,0)\leftarrow} = \emptyset \qquad \mathcal{A}^{(0,0)\rightarrow} \setminus \mathcal{P}^{(0,0)\rightarrow} = \emptyset.$$

  In order to apply ERCAT, condition (4.4) must be satisfied. Therefore, we derive the condition $x_{12} = x'_{12}$.

- $(0,k)$. For this pair we have the following sets:

$$\mathcal{P}^{(0,k)\rightarrow} = \{t_{12}\} \quad \mathcal{A}^{(0,k)\leftarrow} = \{t'_{12}\}$$
$$\mathcal{P}^{(0,k)\leftarrow} \setminus \mathcal{A}^{(0,k)\leftarrow} = \{t_{12}, t'_{12}\} \setminus \{t'_{12}\} = \{t_{12}\}$$
$$\mathcal{A}^{(0,k)\rightarrow} \setminus \mathcal{P}^{(0,k)\rightarrow} = \{t_{12}, t'_{12}\} \setminus \{t_{12}\} = \{t'_{12}\}.$$

  In order to satisfy condition (4.4), we have that $x_{12} - x'_{12} = \overline{\beta_{12}(0k)} - \alpha_{12}(0k)$. Both this condition and the one derived for pair $(0,k)$ must hold, then we can write $\alpha_{12}(0k) = \overline{\beta_{12}(0k)}$, where $\alpha_{12}(0k) = \mu_{12}$.

- $(k, 0)$ and (k, k). For these pairs the ERCAT conditions (4.4) are trivially satisfied. In fact, every passive action and every reversed action corresponding to an active action, are always enabled (therefore for these pairs RCAT structural conditions are satisfied).

Therefore, ERCAT conditions are:

$$\begin{cases} x_{12} = x'_{12} \\ \overline{\beta_{12}(0k)} = \alpha_{12}(0k) = \mu_{12}. \end{cases} \tag{5.1}$$

Let us write the system of equations (4.3) for this model using Definition 5 (of Chapter 4).

$$\begin{cases} x_{12} = \frac{\lambda_{12}}{\lambda_{12}+\lambda_1}(\mu_1 + x'_{12}) \\ x'_{12} = \frac{\mu_{12}}{\mu_{12}+\mu_2}(\lambda_2 + x_{12}) \end{cases} . \tag{5.2}$$

Let us assume $x_{12} = x'_{12}$, we obtain:

$$\begin{cases} x_{12} = \frac{\lambda_{12}\mu_1}{\lambda_1} \\ x'_{12} = \frac{\mu_{12}\lambda_2}{\mu_2} \end{cases} ,$$

that gives the condition:

$$\lambda_1\lambda_2\mu_{12} = \lambda_{12}\mu_1\mu_2. \tag{5.3}$$

Now we still have to check the second condition of (5.1):

$$\overline{\beta_{12}(0k)} = \frac{x_{12}}{x_{12} + \lambda_2}(\mu_2 + \mu_{12}) = \mu_{12},$$

that can be verified by easy calculations.

Under condition (5.3) the steady state probability distribution $\pi(m_1, m_2)$ is in product-form by ERCAT. By replacing $\top_{12}$ with x_{12} and $\top'_{12}$ with x'_{12} the stochastic processes associated with P^1 and P^2 are simple birth and death processes therefore, after some calculations, we can write:

$$\pi(m_1, m_2) \propto \left(\frac{\lambda_1}{\mu_1}\right)^{m_1}\left(\frac{\lambda_2}{\mu_2}\right)^{m_2}. \tag{5.4}$$

Coleman, Henderson et al. theorem application. In this paragraph we repeat the analysis of BBB model in order to compare the product-form conditions required by ERCAT and those required by Theorem 2 of Chapter 3. We can note that the structural conditions of Theorem 2 are not immediately satisfied because the input transitions have the same input vector (the null vector). Therefore, we fuse the transitions into one transition called T as shown in Figure 5.3. Different linestyles represent different output vectors of the same transition T.

We have that transition T has three output vectors: $O_1(T) = (1, 0)$ with probability λ_1/λ, $O_2(T) = (0, 1)$ with probability λ_2/λ and $O_3(T) = (1, 1)$ with probability

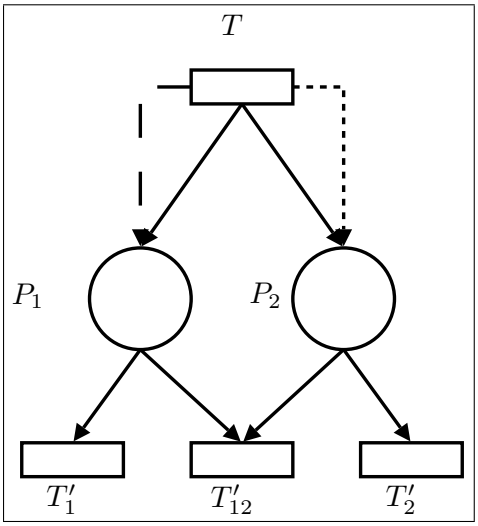

Figure 5.3: BBB after melting the input transitions.

λ_{12}/λ, where $\lambda = \lambda_1 + \lambda_2 + \lambda_{12}$ is the firing rate of T. Therefore, the traffic equation system (3.6) becomes:

$$\begin{cases} \lambda f = \mu_1 f_1 + \mu_2 f_2 + \mu_{12} f_{12} \\ \mu_1 f_1 = \lambda f \frac{\lambda_1}{\lambda} \\ \mu_2 f_2 = \lambda f \frac{\lambda_2}{\lambda} \\ \mu_{12} f_{12} = \lambda f \frac{\lambda_{12}}{\lambda} \end{cases}$$

That gives $f = 1$, $f_1 = \lambda_1/\mu_1$, $f_2 = \lambda_2/\mu_2$, $f_{12} = \lambda_{12}/\mu_{12}$.

From these f values we can derive vector $\mathbf{C}$ (3.7):

$$\mathbf{C} = \begin{bmatrix} \log \frac{\mu_1}{\lambda_1} \\ \log \frac{\mu_2}{\lambda_2} \\ \log \frac{\mu_{12}}{\lambda_{12}} \\ \log \frac{\lambda_1}{\mu_1} \\ \log \frac{\lambda_2}{\mu_2} \\ \log \frac{\lambda_{12}}{\mu_{12}} \end{bmatrix}.$$

In the following we use c_i to denote the i-th component of vector $\mathbf{C}$. The incidence matrix $\mathbf{A}$ is:

$$\mathbf{A} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \\ -1 & 0 \\ 0 & -1 \\ -1 & -1 \end{bmatrix},$$

and we have $\text{rank}(\mathbf{A}) = 2$. Matrix $[\mathbf{A}|\mathbf{C}]$ is the matrix:

$$[\mathbf{A}|\mathbf{C}] = \begin{bmatrix} 1 & 0 & c_1 \\ 0 & 1 & c_2 \\ 1 & 1 & c_3 \\ -1 & 0 & c_4 \\ 0 & -1 & c_5 \\ -1 & -1 & c_6 \end{bmatrix}.$$

We have that $\text{rank}([\mathbf{A}|\mathbf{C}]) = \text{rank}(\mathbf{A}) = 2$ if and only if $c_1 + c_2 = c_3$, i.e.:

$$\frac{\mu_1}{\lambda_1}\frac{\mu_2}{\lambda_2} = \frac{\mu}{\lambda} \implies \mu_1\mu_2\lambda = \lambda_1\lambda_2\mu.$$

Under this condition SPN BBB is in product-form by Henderson, Coleman et al. theorem. Now we can derive the product-form expression by solving system (3.10), that gives:

$$\begin{cases} -\log y_1 = c_1 \implies y_1 = \frac{\lambda_1}{\mu_1} \\ -\log y_2 = c_2 \implies y_2 = \frac{\lambda_2}{\mu_2} \end{cases}.$$

This gives the product-form solution:

$$\pi(m_1, m_2) \propto \left(\frac{\lambda_1}{\mu_1}\right)^{m_1}\left(\frac{\lambda_2}{\mu_2}\right)^{m_2}.$$

Asymmetric BBB Let us consider model BBB shown in Figure 5.1 and suppose that both transitions T_1 and T_1' are missing. In this case the definition of the PEPA agents is different but the analysis of conditions (4.4) does not change. In particular condition $x_{12} = x'_{12}$ is trivially satisfied, so:

$$x_{12} = x'_{12} = \frac{\mu_{12}\lambda_2}{\mu_2}$$

is unconditionally satisfied. It is also easy to check that $\overline{\beta_{12}(0k)} = \mu_{12}$ unconditionally. The product-form is then given by:

$$\pi(m_1, m_2) \propto \left(\frac{\mu_2\lambda_{12}}{\lambda_2\mu_{12}}\right)^{m_1}\left(\frac{\lambda_2}{\mu_2}\right)^{m_2}.$$

It is possible to apply again Theorem 2 and verify that the conditions and the steady state probability function π are the same.

Some notes. This simple example has shown the different approaches of the well-know technique for solving product-form SPNs, i.e., based on Theorem 2, and the one we discuss in this thesis based on ERCAT (Theorem 4). Note that both ERCAT and Theorem 2 give the same condition for the product-form and, obviously, the

same steady state probabilities. However, informally, ERCAT needs the condition in order to balance the flows in the forward and reversed stochastic processes of the model, while Coleman and Henderson approach seems to be more algebraical.

In particular, it is worthwhile pointing out that the conditions that arise from the application of ERCAT have clear motivations based on the first Kolmogorov's criteria (see the proof of ERCAT for more details [61]). In fact, we know that for each state of a CTMC, the sum of the outgoing rates in the forward process is equal to the sum of the outgoing rates in the reversed process. As a consequence ERCAT condition (4.4) is required (while in RCAT it is always satisfied).

Another aspect we want to point out is the following. If we consider the symmetric BBB condition (5.3) then we can explicit λ_1/μ_1 obtaining the base of the first factor of the product-form solution for the asymmetric case. We show in the following sections that this result can be generalized.

Finally, a last peculiarity of this example is that, as far as we know, it is the first case of application of ERCAT in which all the sets specified in 4.4 are not empty, therefore showing that the theorem defined in order to consider all the possible cases has not only a theoretical relevance but also practical applications.

5.2.2 Analysis of the building blocks

A building block consists of a set of places $P_1, \dots P_N$, a set of input transitions $\mathcal{T}_I$ whose input vectors are null, and a set of output transitions $\mathcal{T}_O$ whose output vectors are null. We say that a building block is *complete* if for every non-empty subset y of $\{1, \dots, N\}$ there exists an input transition $T_y \in \mathcal{T}_I$ such that the output vector has 1 values in position i if and only if $i \in y$ and all the other components are 0. Symmetrically there must exist an output transition T'_y whose input vector coincides with the output vector of T_y. Therefore, a complete building block with N places has $2^N - 1$ input transitions and $2^N - 1$ output transitions. It should be clear that this is a generalization of model BBB presented before. The main result presented in this section is the following lemma, that states the conditions for (and the expression of) the product-form solution for a building block:

Lemma 1 *Consider an SPN S consisting of N places $\{P_1, \dots, P_N\}$ and a set of transitions $\mathcal{T} = \mathcal{T}_I \cup \mathcal{T}_O$ where $\mathcal{T}_I = \{T_y | y \in \mathcal{P}\{1, \dots, N\} \setminus \emptyset\}$ and $\mathcal{T}_O = \{T'_y | y \in \mathcal{P}\{1, \dots, N\} \setminus \emptyset\}$. The i-th component of the output (input) vector of T_y (T'_y) is 1 if $i \in y$, 0 otherwise. The firing rates of transitions $T_y \in \mathcal{T}_I$ ($T'_y \in \mathcal{T}_O$) are denoted by λ_y (μ_y). Assuming that S has a steady state distribution, then it has the following product-form solution:*

$$\pi(\mathbf{m}) \propto \prod_{i=1}^{N} \left(\frac{\lambda_i}{\mu_i}\right)^{m_i} \tag{5.5}$$

provided that the following set of conditions holds:

$$\forall y \in \mathcal{P}\{1, \dots, N\} \setminus \emptyset, \lambda_y \prod_{i \in y} \mu_i = \mu_y \prod_{i \in y} \lambda_i. \tag{5.6}$$

As first we simplify a little the notation. In the following, letters i and j take values in $\{1,\dots,N\}$ while y is a set belonging to the set of the parts of $\{1,\dots,N\}$ unless differently specified. We write λ_i instead of $\lambda_{\{i\}}$, and similarly for μ and x. If we want to add an index to a set we will just write y,i instead of $y\cup\{i\}$ and similarly for the minus operator we write $y-i$ instead of $y\setminus\{i\}$.

In order to simplify the proof of Lemma 1 we introduce the following simple lemma.

Lemma 2 *Conditions (5.6) holds if and only for each y such that $|y|>2$, and for each $i\in y$ we can write:*

$$\lambda_y\mu_{y-i}\mu_i=\mu_y\lambda_{y-i}\lambda_i \tag{5.7}$$

Proof of Lemma 2

The case $|y|=2$ is trivial because condition (5.6) is always true.

- (5.6) $\Rightarrow$ (5.7) Let $|y|>2$, then we can rewrite Equation (5.6) as:

$$\lambda_y\mu_i\prod_{\substack{j\in y\\ j\neq i}}\mu_j=\mu_y\lambda_i\prod_{\substack{j\in y\\ j\neq i}}\lambda_j.$$

  By hypothesis:

$$\frac{\lambda_{y-i}}{\mu_{y-i}}=\frac{\prod_{\substack{j\in y\\ j\neq i}}\lambda_j}{\prod_{\substack{j\in y\\ j\neq i}}\mu_j},$$

  we obtain (5.7).

- (5.7) $\Rightarrow$ (5.6) We can prove the Lemma by induction on $|y|$. The base case $|y|=2$ is trivial, so let us consider $|y|>2$. By Formula (5.7) we can write:

$$\lambda_y\mu_{y-i}\mu_i=\mu_y\lambda_{y-i}\lambda_i.$$

  By inductive hypothesis we have that:

$$\frac{\lambda_{y-i}}{\mu_{y-i}}=\frac{\prod_{\substack{j\in y\\ j\neq i}}\lambda_j}{\prod_{\substack{j\in y\\ j\neq i}}\mu_j},$$

  that proves the second verse of the double implication. This completes the proof. ♠

Introduction to the proof of Lemma 1

The proof of Lemma 1 can result complicated. It is done by induction on the number of places of the building block. In order to help the intuition, in this paragraph we show how we can study a building block consisting of 3 places basing the results on the analysis of a building block with 2 places (BBB). Readers who are confortable with ERCAT applications can skip this paragraph.

Let us consider a complete BB with 3 places, P_1, P_2 and P_3. In this case the set of input and output transitions are:

$$\mathcal{T}_I = \{T_1, T_2, T_3, T_{12}, T_{13}, T_{23}, T_{123}\}$$
$$\mathcal{T}_O = \{T'_1, T'_2, T'_3, T'_{12}, T'_{13}, T'_{23}, T'_{123}\}$$

Let us describe such a building block by a PEPA component S defined as a synchronization of a model S' associated with places P_1 and P_2 (a BBB) and the a model P^3 associated with place P_3:

$$S \stackrel{def}{=} S'_{00} \underset{t_{123},t_{13},t_{23},t'_{123},t'_{13},t'_{23}}{\bowtie} P^3_0.$$

Basically, S' and P^3 synchronize on every input and output transition that includes at least one place in S' and one in P_3. Figure 5.4 informally illustrates how S' and P^3 interact.

Note that structural conditions of RCAT are not satisfied, therefore we apply ERCAT.

Now we check ERCAT condition (4.4) for every pair that forms the joint process state. Our aim is to derive some conditions on the reversed rates $x_{y,3}$ and $x'_{y,3}$ where $y = \{1\}, \{2\}, \{1,2\}$. Let K, J be a positive integers. If we consider the joint state $(00, 0)$ then we have:

$$\begin{aligned}
\mathcal{P}^{(00,0)\rightarrow} &= \{t_{13}, t_{23}, t_{123}\} \\
\mathcal{A}^{(00,0)\leftarrow} &= \{t'_{13}, t'_{23}, t'_{123}\} \\
\mathcal{P}^{(00,0)\leftarrow} &= \{t'_{13}, t'_{23}, t'_{123}\} \\
\mathcal{A}^{(00,0)\rightarrow} &= \{t_{13}, t_{23}, t_{123}\}.
\end{aligned}$$

Then the condition on $x_{y,3}$ and $x'_{y,3}$ is:

$$x_{123} + x_{23} + x_{13} = x'_{123} + x'_{23} + x'_{13}$$

Let us consider the joint state $(0K, 0)$, then we have:

$$\begin{aligned}
\mathcal{P}^{(0K,0)\rightarrow} &= \{t_{13}, t_{23}, t_{123}, t'_{23}\} \\
\mathcal{A}^{(0K,0)\leftarrow} &= \{t'_{13}, t'_{23}, t'_{123}, t_{23}\} \\
\mathcal{P}^{(0K,0)\leftarrow} &= \{t'_{13}, t'_{23}, t'_{123}\} \\
\mathcal{A}^{(0K,0)\rightarrow} &= \{t_{13}, t_{23}, t_{123}\}.
\end{aligned}$$

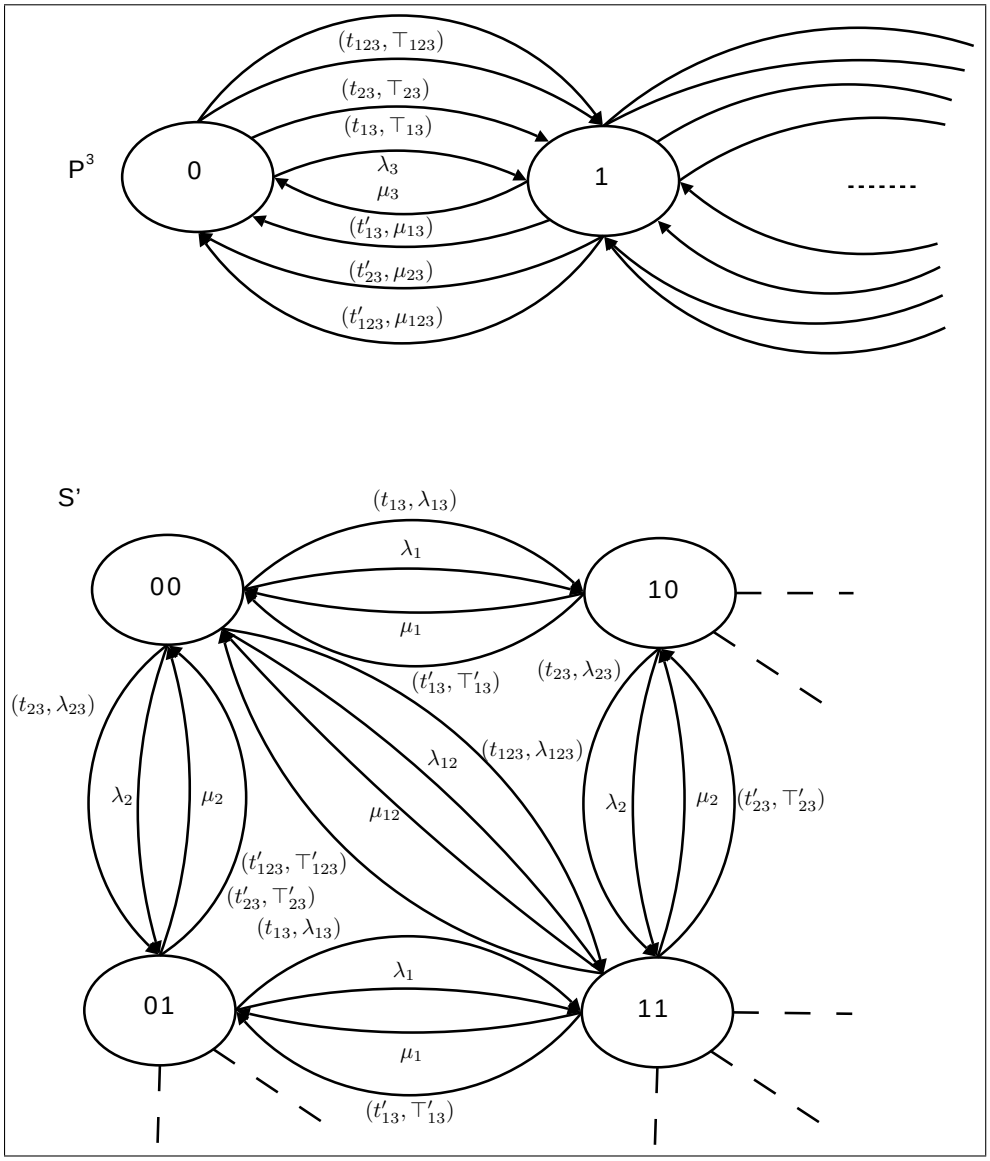

Figure 5.4: Interactions of a BBB with another PEPA agent in order to model a building block with 3 places.

Hence, we derive:

$$x_{123} + x_{23} + x_{13} + x'_{23} = x'_{123} + x'_{23} + x'_{13} + x_{23},$$

that, combined with the first condition, straightforwardly gives $x_{23} = x'_{23}$. Using exactly the same technique we derive the condition $x_y = x'_y$, with $y = \{1,3\}, \{2,3\}, \{1,2,3\}$. Assuming this condition, the following calculations result relatively simple.

Let us consider the joint state $(0K, J)$. Then we have:

$$\begin{aligned}
\mathcal{P}^{(0K,J)\rightarrow} &= \{t_{13}, t_{23}, t_{123}, t'_{23}\} \\
\mathcal{A}^{(0K,J)\leftarrow} &= \{t'_{13}, t'_{23}, t'_{123}, t_{23}\} \\
\mathcal{P}^{(0K,J)\leftarrow} &= \{t'_{13}, t'_{23}, t'_{123}, t_{13}, t_{23}, t_{123}\} \\
\mathcal{A}^{(0K,J)\rightarrow} &= \{t_{13}, t_{23}, t_{123}, t'_{13}, t'_{23}, t'_{123}\}.
\end{aligned}$$

Hence, we derive:

$$x'_{23} - x_{23} = \overline{\beta_{13}(0K, J)} + \overline{\beta_{123}(0K, J)} - \alpha_{13}(0K, J) - \alpha_{123}(0K, J). \tag{5.8}$$

In the same way, from the analysis of joint state $(K0, J)$ we obtain:

$$x'_{13} - x_{13} = \overline{\beta_{23}(K0, J)} + \overline{\beta_{123}(K0, J)} - \alpha_{23}(0K, J) - \alpha_{123}(0K, J). \tag{5.9}$$

Note that, by assumptions, $x_{23} = x'_{23}$ and $x_{13} = x'_{13}$.

The analysis of agent P^3 basically is the analysis of a birth and death process. The formula for x'_{13} is:

$$x'_{13} = \frac{\mu_{13}}{\mu_3 + \mu_{13} + \mu_{23} + \mu_{123}}(\lambda_3 + x_{13} + x_{23} + x_{123}).$$

Note that the condition $x_{y,3} = x'_{y,3}$, for $y = \{1, 3\}, \{2, 3\}, \{1, 2, 3\}$, allows us to derive the solutions, i.e:

$$x'_{y,3} = x_{y,3} = \frac{\lambda_3}{\mu_3}\mu_{y,3}. \tag{5.10}$$

Now we have to verify that Equation (5.10) satisfies conditions (5.8) and (5.9). We have that:

$$\overline{\beta_{13}(K0, J)} = \frac{x_{13}}{x_{23} + x_{123} + \lambda_3}(\mu_{13} + \mu_{23} + \mu_{123} + \mu_3) = \mu_{13} = \alpha_{13}(0K, J),$$

and similarly $\overline{\beta_{23}(K0, J)} = \mu_{23}$ and $\overline{\beta_{123}(K0, J)} = \mu_{123}$. Therefore, conditions (5.8) and (5.9) are verified.

We assume the hypothesis (5.6), that can be rewritten as:

$$\begin{cases}
\mu_{123}\lambda_1\lambda_2\lambda_3 = \lambda_{123}\mu_1\mu_2\mu_3 \\
\mu_{12}\lambda_1\lambda_2 = \lambda_{12}\mu_1\mu_2 \\
\mu_{13}\lambda_1\lambda_3 = \lambda_{13}\mu_1\mu_3 \\
\mu_{23}\lambda_2\lambda_3 = \lambda_{23}\mu_2\mu_3
\end{cases} \tag{5.11}$$

In order to study the BB modeled by S' we have to prove that these conditions imply:

$$(\lambda_{12} + \lambda_{123})(\mu_1 + x_{13})(\mu_2 + x_{23}) = (\mu_{12} + x_{123})(\lambda_1 + \lambda_{13})(\lambda_2 + \lambda_{23}).$$

The proof is purely algebraical and we omit it because we give a proof for the general case in the following paragraph. The BB identified by S' has the following product-form solution:

$$\pi'(\mathbf{m}') \propto \left(\frac{\lambda_1 + \lambda_{13}}{\mu_1 + x_{13}}\right)^{m_1} \left(\frac{\lambda_2 + \lambda_{23}}{\mu_2 + x_{23}}\right)^{m_2},$$

where $\mathbf{m}' = (m_1, m_2)$, $x_{i3} = (\lambda_3/\mu_3)\mu_{i3}$. Model S is in product-form and its solution is given by:

$$\pi(\mathbf{m}) \propto \pi'(\mathbf{m}') \left(\frac{\lambda_{N+1}}{\mu_{N+1}}\right)^{m_3},$$

where $\mathbf{m} = (m_1, m_2, m_3)$. Using Hypothesis (5.11) we can derive that:

$$\pi(\mathbf{m}) \propto \prod_{i=1}^{3} \left(\frac{\lambda_1}{\mu_i}\right)^{m_i}.$$

Proof of Lemma 1

The proof is inductive on N, the number of places. The case $N = 1$ is a simple exponential queue, and the case $N = 2$ has already been studied because it is model BBB. Let us consider a SPN S with $N + 1$ places. We remove from the SPN place P_{N+1} and all the arcs incoming to or outgoing from P_{N+1} obtaining the model called S'. From the point of view of S', this operation causes transitions $T_{y,N+1}$ ($T'_{y,N+1}$) to have the same behavior (i.e. the same input and output vectors) of T_y (T'_y) with $y \in \mathcal{P}\{1, \ldots, N\} \setminus \emptyset$. However T_y does not synchronize with P_{N+1} while $T_{y,N+1}$ does. Let us describe the interactions between S' and P_{N+1} in PEPA. We use small letter t to denote the action type associated with transition T, and the PEPA component P_n^{N+1} represents the state with n tokens in P_{N+1}.

Process P_n^{N+1} can be described in PEPA as follows:

$$\begin{aligned} P_0^{N+1} &\stackrel{def}{=} \lambda_{N+1}.P_1^{N+1} + (t_{y,N+1}, \top_{y,N+1}).P_1^{N+1} \\ P_n^{N+1} &\stackrel{def}{=} \lambda_{N+1}.P_{n+1}^{N+1} + (t_{y,N+1}, \top_{y,N+1}).P_{n+1}^{N+1} \\ &\quad + \mu_{N+1}.P_{n-1}^{N+1} + (t'_{y,N+1}, \mu_y).P_{n-1}^{N+1}, \end{aligned}$$

where $n > 0$ and $y \in \mathcal{P}\{1, \ldots, N\} \setminus \emptyset$.

ERCAT structural conditions. In this part of the proof we study the ERCAT conditions for the product-form of the cooperation

$$S' \underset{\{\forall y \neq \emptyset, t_{y,N+1}, t'_{y,N+1}\}}{\bowtie} P_0^{N+1}.$$

We use an analysis that is completely similar to the one illustrated for model BBB.

Let $\mathbf{m}$ denote a derivate of S' and we define $\mathcal{O}(\mathbf{m}) = \{y : t'_y \text{is enabled in } \mathbf{m}\}$. The derivatives of P^{N+1} are clustered in two classes, 0 is P_0^{N+1} while K is a general P_K^{N+1} with $K > 0$.

For a joint state $(\mathbf{m}, 0)$ it is possible to show that the ERCAT conditions are:

$$\sum_{(y,N+1)\in\mathcal{O}(\mathbf{m})} x'_{y,N+1} + \sum_{y\in\mathcal{P}\{1,\ldots,N\}\setminus\emptyset} x_{y,N+1} = \sum_{(y,N+1)\in\mathcal{O}(\mathbf{m})} x_{y,N+1} + \sum_{y\in\mathcal{P}\{1,\ldots,N\}\setminus\emptyset} x'_{y,N+1} \tag{5.12}$$

while from a joint state $(\mathbf{m}, K)$ the ERCAT conditions are:

$$\sum_{(y,N+1)\notin\mathcal{O}(\mathbf{m})} \overline{\beta_{y,N+1}(\mathbf{m}, K)} = \sum_{(y,N+1)\notin\mathcal{O}(\mathbf{m})} \alpha_{y,N+1}(\mathbf{m}, K). \tag{5.13}$$

If we consider derivate $\mathbf{m}$ such that $\mathcal{O}(\mathbf{m}) = \emptyset$ we can write (5.12) as:

$$\sum_{y\in\mathcal{P}\{1,\ldots,N\}\setminus\emptyset} x_{y,N+1} = \sum_{y\in\mathcal{P}\{1,\ldots,N\}\setminus\emptyset} x'_{y,N+1},$$

This, combined with (5.12) written for $\mathbf{m}$ such that $\mathcal{O}(\mathbf{m}) = \{i\}$ with $1 \leq i \leq N$ gives the conditions:

$$\forall y, \quad x_{y,N+1} = x'_{y,N+1} \tag{5.14}$$

where $y \in \mathcal{P}\{1, \ldots, N\} \setminus \emptyset$.

Analysis of agent P^{N+1} and solution for $x_{y,N+1}$ and $x'_{y,N+1}$. Let us assume that conditions (5.14) are satisfied. Figure 5.5 illustrates agent P^{N+1} graphically, where labels ys have to be intended as all the existing synchronized transitions from S' and y', y'' as two examples of these transitions. ERCAT gives us a method

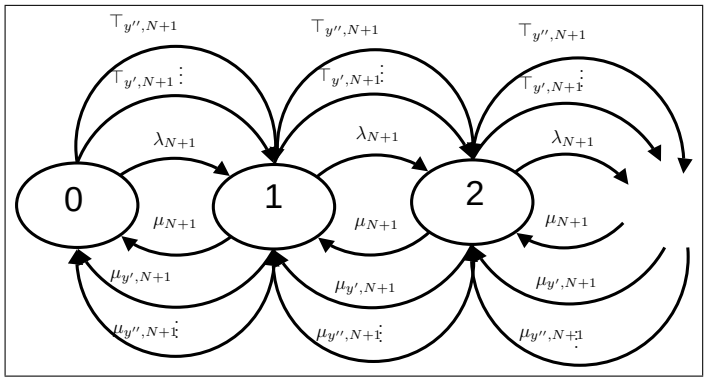

Figure 5.5: Description of agent P^{N+1}.

to replace $\top$s with opportune rates $x_{y,N+1}$s as shown by Figure 5.6. As we have that $x_{y,N+1} = x'_{y,N+1}$ we can determine $x'_{y,N+1}$s rates (and then $x_{y,N+1}$s) by the analysis of P^{N+1} in isolation. In order to achieve this, we must obtain the CTMC

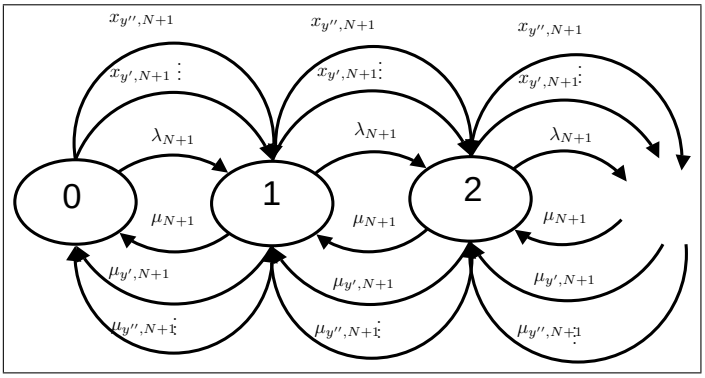

Figure 5.6: Description of agent P^{N+1} after replacing $\top$ with x.

associated with P^{N+1} and, once it is reversed, determine the rates associated with the transitions with type $t'_{y,N+1}$ using Definition 5. If these reversed rates are constant we have the solution for $x'_{y,N+1}$s. The CTMC associated with the process of Figure 5.6 is a birth and death process with constant birth and death rates. The total birth rate is $\sum_y x'_{y,N+1} + \lambda_{N+1}$ and the total death rate is $\sum_y \mu_{y,N+1} + \mu_{N+1}$, where $y \in \mathcal{P}\{1,\dots,N\} \setminus \emptyset$. Then we can obtain $x_{y,N+1}$:

$$x'_{y,N+1} = \frac{\mu_{y,N+1}}{\mu_{N+1} + \sum_y \mu_{y,N+1}} \Big(\sum_y x'_{y,N+1} + \lambda_{N+1} \Big), \tag{5.15}$$

therefore, we can note that $x'_{y,N+1} = k\mu_{y,N+1}$ with $k > 0$ constant. This gives:

$$k - k\frac{\sum_y \mu_{y,N+1}}{\mu_{N+1} + \sum_y \mu_{y,N+1}} = \frac{\lambda_{N+1}}{\mu_{N+1} + \sum_y \mu_{y,N+1}},$$

that implies:

$$x'_{y,N+1} = x_{y,N+1} = \frac{\lambda_{N+1}}{\mu_{N+1}} \mu_{y,N+1}. \tag{5.16}$$

We can show that Formula (5.16) satisfies condition (5.13), indeed using again Definition 5, we have:

$$\overline{\beta_{y,N+1}(\mathbf{m}, K)} = \frac{x_{y,N+1}}{\sum_y x_{y,N+1} + \lambda_{N+1}} \Big(\sum_y \mu_{y,N+1} + \mu_{N+1} \Big) = \mu_{y,N+1},$$

that is independent of state $\mathbf{m}$. So we have proved that:

1. $x_{y,N+1} = x'_{y,N+1} \iff x'_{y,N+1} = x_{y,N+1} = \frac{\lambda_{N+1}}{\mu_{N+1}} \mu_{y,N+1}$,
2. $x_{y,N+1} = x'_{y,N+1} \implies$ condition (5.13).

Analysis of agent S'. If we derive the CTMC of S' we have that each possible state transition is caused by two possible actions. One is the internal action of S', t_y (t'_y), and the other is the synchronizing action $t_{y,N+1}$ ($t'_{y,N+1}$) (see Figure 5.7). So, in the CTMC, the total rate of each transition is $\mu_y + x'_{y,N+1}$ where $x'_{y,N+1}$ is the rate obtained by the application of ERCAT theorem, or $\lambda_y + \lambda_{y,N+1}$. Therefore, by the inductive hypothesis we have the following product-form solution of S':

$$\pi'(\mathbf{m}) \propto \prod_{i=1}^{N} \left(\frac{\lambda_i + \lambda_{i,N+1}}{\mu_i + x_{i,N+1}} \right)^{m_i}. \tag{5.17}$$

if the following condition holds:

$$\forall y \in \mathcal{P}\{1,\ldots,N\} \setminus \emptyset, (\lambda_y + \lambda_{y,N+1}) \prod_{j \in y} (\mu_j + x_{j,N+1}) = (\mu_y + x_{y,N+1}) \prod_{j \in y} (\lambda_j + \lambda_{j,N+1}). \tag{5.18}$$

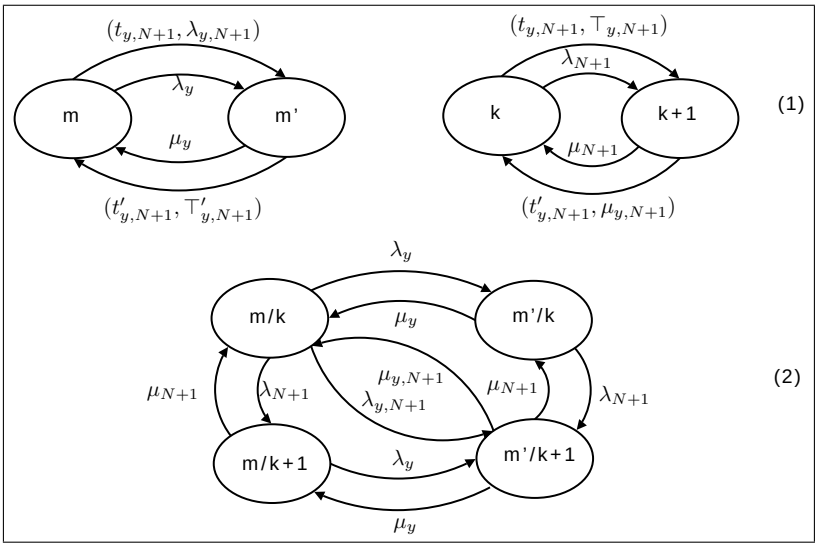

Figure 5.7: Intuition of the possible transitions between two states in the CTMC of S (2) and the way they are modeled in PEPA (1).

(5.6) $\Longrightarrow$ (5.5). This is the main part of the proof. In order to help the intuition, Figure 5.8 illustrates the steps we are going to do in order to reach our goal.

We assume that (5.6) holds for the BB with $N+1$ places, and we prove (5.18), i.e., Condition (5.6) holds for S'. If $|y| = 1$ Equation (5.18) is an identity. By Lemma 2 we can prove that for any set y, $|y| > 1$, and label $i \in y$ the following

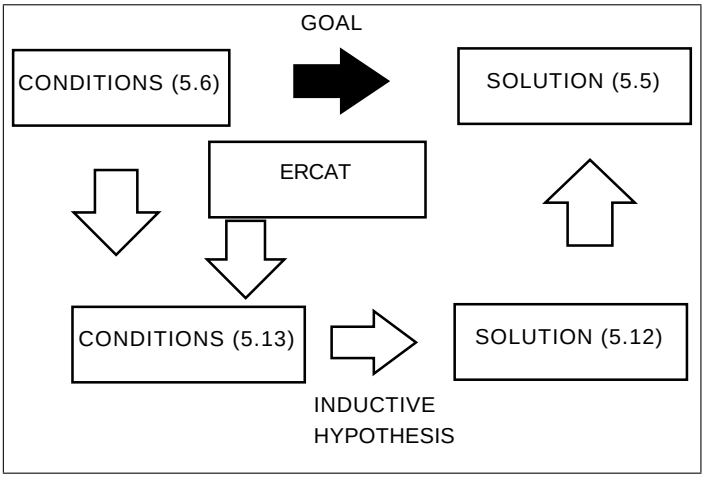

Figure 5.8: Schema of the proof of Lemma 1.

relation holds:

$$(\lambda_y + \lambda_{y,N+1})(\mu_{y-i} + x_{y-i,N+1})(\mu_i + x_{i,N+1}) \\ = (\mu_y + x_{y,N+1})(\lambda_{y-i} + \lambda_{y-i,N+1})(\lambda_i + \lambda_{i,N+1}).$$

As $x_{y,N+1} = (\lambda_{N+1}/\mu_{N+1})\mu_{y,N+1}$, we can rewrite:

$$(\lambda_y + \lambda_{y,N+1})(\mu_{y-i} + \frac{\lambda_{N+1}}{\mu_{N+1}}\mu_{y-i,N+1})(\mu_i + \frac{\lambda_{N+1}}{\mu_{N+1}}\mu_{i,N+1}) \\ = (\mu_y + \frac{\lambda_{N+1}}{\mu_{N+1}}\mu_{y,N+1})(\lambda_{y-i} + \lambda_{y-i,N+1})(\lambda_i + \lambda_{i,N+1}).$$

By hypothesis, Condition (5.6) holds, then $\mu_y = \lambda_y \prod_{j\in y} \mu_j / \prod_{j\in y} \lambda_j$ and similarly we can rewrite $\mu_{y,N+1}$, hence, the second hand side becomes:

$$\prod_{j\in y} \frac{\mu_j}{\lambda_j}(\lambda_y + \frac{\lambda_{N+1}}{\mu_{N+1}}\frac{\mu_{N+1}}{\lambda_{N+1}}\lambda_{y,N+1})(\lambda_{y-i} + \lambda_{y-i,N+1})(\lambda_i + \lambda_{i,N+1}).$$

After some simplifications, we have to prove:

$$(\mu_{y-i} + \frac{\lambda_{N+1}}{\mu_{N+1}}\mu_{y-i,N+1})(\mu_i + \frac{\lambda_{N+1}}{\mu_{N+1}}\mu_{i,N+1}) = \prod_{j\in y} \frac{\mu_j}{\lambda_j}(\lambda_{y-i} + \lambda_{y-i,N+1})(\lambda_i + \lambda_{i,N+1}).$$

By rewriting μ_{y-i} and $\mu_{y-i,N+1}$, the left hand side becomes:

$$\frac{\lambda_i}{\mu_i}\prod_{j\in y} \frac{\mu_j}{\lambda_j}(\lambda_{y-i} + \lambda_{y-i,N+1})(\mu_i + \frac{\lambda_{N+1}}{\mu_{N+1}}\mu_{i,N+1}).$$

After some simplifications, eventually we obtain:

$$\lambda_i + \frac{\lambda_i\lambda_{N+1}}{\mu_i\mu_{N+1}}\mu_{i,N+1} = \lambda_i + \lambda_{i,N+1},$$

that is satisfied by hypothesis.

As conditions (5.18) are verified, by inductive hypothesis we conclude that the steady state probabilities of S' are given by (5.17). Let us verify that the solutions for $x_{y,N+1}$ expressed by (5.16) are correct, i.e., the reversed rates of the active actions of types $t_{y,N+1}$ are constant.

In S' we have that $x_{y,N+1} = \overline{\lambda_{y,N+1}}$, i.e, the reversed rate of the firing rates of transition T_y where $y \in \mathcal{P}\{1,\ldots,N\} \setminus \emptyset$. As we know the steady state solution of S' by the inductive hypothesis (5.17), we can easily obtain the reversed rates:

$$\pi(\mathbf{m})(\lambda_{y,N+1} + \lambda_y) = \pi(\mathbf{m}')\overline{(\lambda_{y,N+1} + \lambda_y)},$$

where $\mathbf{m}'$ is the state reached from $\mathbf{m}$ after the firing of transition $T_{y,N+1}$. We obtain:

$$x_{y,N+1} = \prod_{i \in y} \left(\frac{\mu_i + x_{i,N+1}}{\lambda_i + \lambda_{i,N+1}}\right) \lambda_{y,N+1},$$

that is independent of state $\mathbf{m}$ (as required) and that can be proved to be coherent with solutions (5.16) using hypothesis (5.6). By ERCAT, the product-form solution for $S' \bowtie P^{N+1}$ is:

$$\pi(\mathbf{m}) \propto \prod_{i=1}^{N} \left(\frac{\mu_{N+1}(\lambda_i + \lambda_{i,N+1})}{\mu_{N+1}\mu_i + \lambda_{N+1}\mu_{i,N+1}}\right)^{m_i} \left(\frac{\sum_y [(\lambda_{N+1}/\mu_{N+1})\mu_{y,N+1}] + \lambda_{N+1}}{\sum_y \mu_{y,N+1} + \mu_{N+1}}\right)^{m_{N+1}},$$

that can be shown to be equal to (5.5) using (5.16) and (5.6). In particular, it can be proved that

$$\frac{\mu_{N+1}(\lambda_i + \lambda_{i,N+1})}{\mu_{N+1}\mu_i + \lambda_{N+1}\mu_{i,N+1}} = \frac{\lambda_i}{\mu_i},$$

and that:

$$\frac{\sum_y [(\lambda_{N+1}/\mu_{N+1})\mu_{y,N+1}] + \lambda_{N+1}}{\sum_y \mu_{y,N+1} + \mu_{N+1}} = \frac{\lambda_{N+1}}{\mu_{N+1}}.$$

The left hand side of the last equality can be rewritten as:

$$\frac{\sum_y x_{y,N+1} + \lambda_{N+1}}{\sum_y \mu_{y,N+1} + \mu_{N+1}},$$

that by (5.15) becomes:

$$\frac{x_{y,N+1}}{\mu_{y,N+1}} = \frac{\lambda_{N+1}}{\mu_{N+1}} \frac{\mu_{y,N+1}}{\mu_{y,N+1}} = \frac{\lambda_{N+1}}{\mu_{N+1}}.$$

This concludes the proof. ♠

Note that we have not proved that conditions (5.6) are necessary for the product-form solution of the model, but we can show that they are necessary in order to

apply ERCAT. In fact conditions (5.12) and (5.13) are necessary for ERCAT. Let us calculate the reversed rates $\overline{\lambda_{y,N+1}}$. We first consider singleton $y = \{i\}$, then:

$$x_{i,N+1} = \frac{\mu_i \lambda_{i,N+1}}{\lambda_i},$$

that, compared to (5.16) gives $\mu_i \mu_{N+1} \lambda_{i,N+1} = \lambda_i \lambda_{N+1} \mu_{i,N+1}$. If $|y| \geq 2$ we have:

$$x_{y,N+1} = \frac{\mu_y \lambda_{y,N+1}}{\lambda_y},$$

that, compared to (5.16) gives $\mu_y \lambda_{y,N+1} \mu_{N+1} = \lambda_y \lambda_{N+1} \mu_{y,N+1}$. It is easy to prove that these conditions together with the previous ones are equivalent to (5.6) (for example by induction on $|y|$).

5.2.3 Comparison between Lemma 1 and Coleman, Henderson et al. approach

In this subsection we review the analysis of the building block using Theorem 2. In particular we analyze an incomplete building block, i.e., a building block in which there is at least one set $y \in \mathcal{P}\{1, \ldots, N\} \setminus \emptyset$ for which transitions $T_y \notin \mathcal{T}_I$ and $T'_y \notin \mathcal{T}_O$. This analysis can be done by ERCAT in a analogue way of what we have shown in the proof of Lemma 1 but, as only some algebraic steps change, we prefer to see the same problem from a different point of view. The case of incomplete (asymmetric) building block for $N = 2$ places has already been studied using ERCAT.

Lemma 3 *Consider an SPN S consisting of N places $\{P_1, \ldots, P_N\}$ and a set of transitions $\mathcal{T} = \mathcal{T}_I \cup \mathcal{T}_O$ where $\mathcal{T}_I = \{T_y | y \in \mathcal{Y}\}$ and $\mathcal{T}_O = \{T'_y | y \in \mathcal{Y}\}$, where $\mathcal{Y} \subseteq \{1, \ldots, N\} \setminus \emptyset$. The i-th component of the output (input) vector of T_y (T'_y) is 1 if $i \in y$, 0 otherwise. The firing rates of transitions $T_y \in \mathcal{T}_I$ ($T'_y \in \mathcal{T}_O$) are denoted by λ_y (μ_y). Assuming that S admits a steady state, then it is in product-form if:*

$$\forall y \in \mathcal{P}\{1, \ldots, N\} \setminus \emptyset : |y| > 1, \lambda_y \prod_{i \in y} \mu_i = \mu_y \prod_{i \in y} \lambda_i. \tag{5.19}$$

The product-form solution is:

$$\pi(\mathbf{m}) \propto \prod_{i=1}^{N} \left(\frac{\lambda_i}{\mu_i}\right)^{m_i}, \tag{5.20}$$

where in case of $y = \{i\} \notin \mathcal{Y}$ with $i \in \{1, \ldots, N\}$, ratio λ_i/μ_i is obtained by a condition related to a transition y such that $i \in y$.

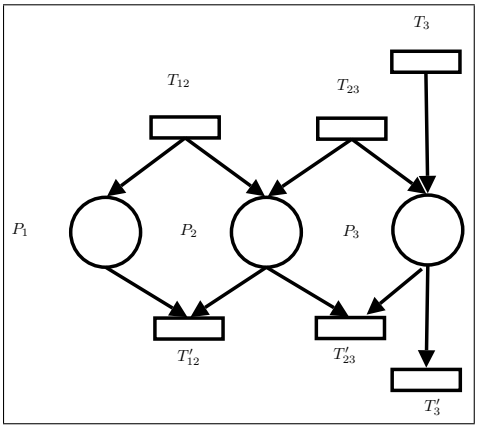

Figure 5.9: Example of incomplete building block.

Example 7 *Suppose we aim to study the incomplete building block of Figure 5.9. Using Lemma 3 we have the following conditions for the product-form:*

$$\begin{cases} \lambda_{12}\mu_1\mu_2 = \mu_{12}\lambda_1\lambda_2 \\ \lambda_{23}\mu_2\mu_3 = \mu_{23}\lambda_2\lambda_3 \end{cases}$$

In this example transitions T_1, T'_1, T_2, T'_2 are missing, so the conditions above are satisfied. Therefore, we can explicit the ratios λ_1/μ_1 and λ_2/μ_2 obtaining:

$$\begin{cases} \frac{\lambda_2}{\mu_2} = \frac{\lambda_{23}}{\mu_{23}}\frac{\mu_3}{\lambda_3} \\ \frac{\lambda_1}{\mu_1} = \frac{\lambda_{12}}{\mu_{12}}\frac{\mu_2}{\lambda_2} = \frac{\lambda_{12}}{\mu_{12}}\frac{\lambda_3}{\mu_3}\frac{\mu_{23}}{\lambda_{23}} \end{cases}$$

Therefore, the steady state probabilities are given by:

$$\pi(m_1, m_2, m_3) \propto \left(\frac{\lambda_{12}\lambda_3\mu_{23}}{\mu_{12}\mu_3\lambda_{23}}\right)^{m_1}\left(\frac{\lambda_{23}\mu_3}{\mu_{23}\lambda_3}\right)^{m_2}\left(\frac{\lambda_3}{\mu_3}\right)^{m_3}.$$

As explained before, we prove Lemma 3 using Theorem 2 in order to clarify the relations between our approach and the well-known one.

Proof of Lemma 3

In order to satisfy Theorem 2, we fuse all the input transitions into one named T with rate $\lambda = \sum_{y\in\mathcal{Y}} \lambda_y$, and use the probabilistic output vectors in order to preserve the model behavior, i.e., transition T has the output vector of T_y with probability λ_y/λ. Then, the traffic equations (3.6) become:

$$\begin{cases} \lambda f = \sum_{y\in\mathcal{Y}} \mu_y f'_y \\ \forall y \in \mathcal{Y} \quad \mu_y f'_y = \lambda f \frac{\lambda_y}{\lambda} = f\lambda_y. \end{cases}$$

A possible solution is given by $f = 1$ and $f'_y = \lambda_y/\mu_y$.

Vector $\mathbf{C}$ (3.7) has the first $|\mathcal{Y}|$ rows whose components are $c_y = \log(\mu_y/\lambda_y)$ and the last $|\mathcal{Y}|$ ones whose components are $c'_y = \log(\lambda_y/\mu_y)$.

Let us determine the incidence matrix. The matrix has $2|\mathcal{Y}|$ rows and N columns. Let $\mathbf{e}_i$ be a N-dimension row vector in which only component $e_i = 1$ and $e_j = 0$ for $j \neq i$. Then the row $\mathbf{r}_y$ of the incidence matrix $\mathbf{A}$ is given by $\sum_{i \in y} \mathbf{e}_i$ for the first $|\mathcal{Y}|$ rows, and $\mathbf{r}'_y = -\sum_{i \in y} \mathbf{e}_i$ for the last $|\mathcal{Y}|$ rows.

Now suppose that row $\mathbf{r}_y$ can be written as sum of L other rows of the first half $\mathbf{A}$: $\mathbf{r}_y = \mathbf{r}_{y_1} + \ldots + \mathbf{r}_{y_L}$. Then, the rank condition (3.9) becomes:

$$\sum_{i=1}^{L} c_{y_i} = c_y,$$

that can be written as:

$$\prod_{i=1}^{L} \frac{\mu_{y_i}}{\lambda_{y_i}} = \frac{\mu_y}{\lambda_y} \implies \lambda_y \prod_{i=1}^{L} \mu_{y_i} = \mu_y \prod_{i=1}^{L} \lambda_{y_i}.$$

This is Lemma 3 condition. The product-form solution is given by (3.10) and can been easily seen that it is coherent with the thesis of the lemma. ♠.

5.3 The composition of the building blocks

In this section and in the following ones we implicitly assume that the models we are considering admit a steady state. Although the proof of Lemma 1 by using ERCAT is complex, we now take advantage from this approach and show how the lemma can be used in order to study complex product-form SPN models. As the building blocks are in product-form by ERCAT we can observe that:

1. the reversed rates of the reversed actions corresponding to output transition firings are constant,

2. an input transition is always enabled,

3. each state of the building block can be reached by the firing of any output transition.

Then we can use RCAT (Theorem 3) to combine two building blocks (i.e. we do not need to check the conditions of ERCAT theorem). Considering only pairwise compositions of several building blocks for each action type, we can apply MARCAT in order to solve the model in one step.

5.3.1 CHC-SPN

In this part of the section we characterize a subset of the models considered by Coleman, Henderson et al. in their works [63, 39] and presented here in Chapter 3. In the following definition we refer to the notation for SPNs introduced in Chapter 3.

Definition 6 (CHC-SPN) *A Coleman, Henderson et al. SPN with constant firing rates (CHC-SPN) is a net with the following structural conditions:*

1. *Each arc has weight* 1,
2. *All the exponential transition rates are constant and are denoted by* χ_i *where* i *is the index of the corresponding transition* T_i*:* $w(T_i, \mathbf{m}) = \chi_i$,
3. *For each input vector* $\mathbf{I}(T_i)$ *there exists a transition* T_j *with an output vector* $\mathbf{O}_b(T_j)$ *such that* $\mathbf{I}(T_i) = \mathbf{O}_b(T_j)$.

Note that every CHC-SPN satisfies the structural conditions of the product-form model class required by Theorem 2.

CHC-SPNs exhibit a useful property, i.e., they can be decomposed into building blocks. Let us consider the following relation between places: $P_i \sim P_j$ if there exists at least a transition T such that $(T, P_j) \in \mathcal{A}$ and $(T, P_i) \in \mathcal{A}$, where according with the notation defined in Chapter 3, $\mathcal{A}$ is the set of arcs. Note that relation $\sim$ is symmetric and reflexive but it is not transitive as shown by the counterexample of figure 5.10, where $P_1 \sim P_2$, $P_2 \sim P_3$ but $P_1 \sim P_3$ is not true. Therefore, if we want

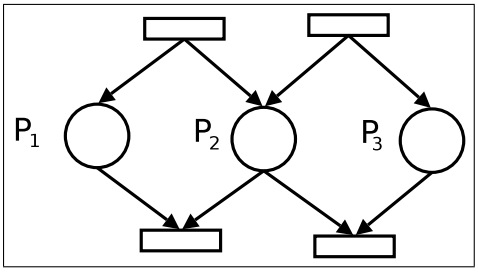

Figure 5.10: Example of relations among transitions.

to identify a relation which induces a partition of the places in the net, we need to extend $\sim$ with transitivity. Hence, we have that $P_i \sim' P_j$ if $P_i \sim P_j$ or if there exists $P_k \in \mathcal{P}$ such that $P_i \sim P_k$ and $P_k \sim' P_j$. Now relation $\sim'$ is an equivalence relation. As a consequence, each set of the partition $\sim'$ (an equivalence class) identifies the places of a building block as required. As a consequence of the definition of this

relation, we can use a new approach to study SPN models in product-form. If we have a CHC-SPN we can partition it into building blocks, and then apply MARCAT [62] to identify the product-form solution.

The steps of the analysis are the following:

1. We identify the building blocks in the net,
2. We study each building block considering as unknown the rates of the input transitions (i.e. x_i is the rate of the input transition T_i),
3. For each building block we determine the reversed rates of the output transitions and set these rates as x_i. This originates a linear system of equations to be solved.

Recall that the reversed rate of a transition T'_y of a building block can be easily obtained by its steady state solution using Kolmogorov's criteria:

$$\pi(\mathbf{m})\mu_y = \pi(\mathbf{m}')\overline{\mu_y},$$

where $\mathbf{m}'$ is a state reachable by $\mathbf{m}$ through the firing of T'_y. Note that $\overline{\mu_y}$ is independent of the choice of $\mathbf{m}$:

$$\overline{\mu_y} = \frac{\pi(\mathbf{m})}{\pi(\mathbf{m}')}\mu_y. \tag{5.21}$$

5.3.2 Other compositions

When a CHC-SPN and a building block satisfy RCAT conditions (for the latter one given by Lemmas 1 and 3) we can compose them with other ones with the same properties maintaining the product-form solution. We introduce some of these models expressed by GSPN in the following chapters, but one can even use a completely different formalism. For example, CHC-SPNs can be composed with product-form G-Networks as the product-form solution for this model class has been analyzed by RCAT in [60]. In [58] the author introduces several models in product-form that can be analyzed by RCAT and each of these can be composed with CHC-SPN in product-form originating other product-form models. Note that, as corollary of these results, we have that a combination of CHC SPNs and exponential queueing centers are in product-form, i.e., we straightforwardly prove the product-form results for queueing Petri nets (without batch customer movements) presented in [18, 19] and without solving the set of global balance equations. We point out again what we have said in Note 1, i.e., when we say that a stochastic model fulfils RCAT conditions, we mean that it is possible to define a convenient PEPA definition of its underlying stochastic process that satisfies the conditions of Theorem 3. Unfortunately, the problem of mapping the underlying process into a PEPA definition has not a unique solution, therefore an automatic general method seems hard to define.

In the following, we interpret RCAT conditions for CHC-SPNs by the analysis of a convenient PEPA representation of the underlying process.

We show in the next chapters some example of hybrid modeling in product-form.

5.3.3 A first example

In this section, we analyze one of the examples illustrated in [39] so that one can compare our method with that presented in the original paper. The SPN structure is shown in Figure 5.11. It consists of 3 building blocks shown in Figure 5.12.

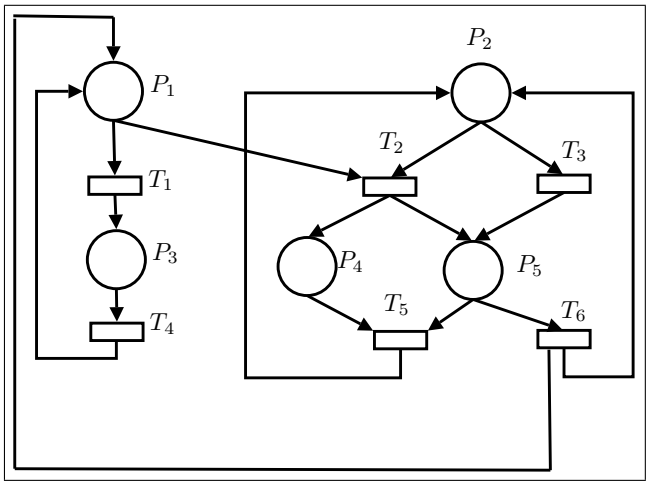

Figure 5.11: CHC-SPN of the example of Section 5.3.3.

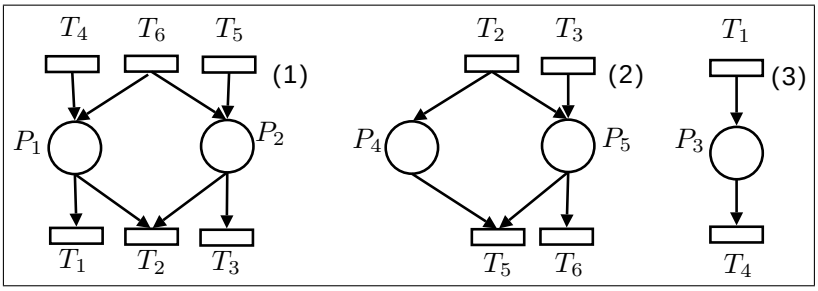

Figure 5.12: Decomposition in building blocks of the CHC-SPN of the example presented in Section 5.3.3.

We apply MARCAT.

Block 1 (Figure 5.12 (1)). The product-form condition is $x_6\chi_1\chi_3 = \chi_2 x_4 x_5$ and the steady state solution has the form:

$$\left(\frac{x_4}{\chi_1}\right)^{m_1}\left(\frac{x_5}{\chi_3}\right)^{m_2},$$

the reversed rates are $x_1 = x_4$ and $x_3 = x_5$.

Block 2 (Figure 5.12 (2)). The block is unconditionally in product-form. The steady state solution has the form:

$$\left(\frac{\chi_6 x_2}{x_3 \chi_5}\right)^{m_4} \left(\frac{x_3}{\chi_6}\right)^{m_5},$$

the reversed rates are $x_5 = x_2$ and $x_6 = x_3$.

Block 3 (Figure 5.12 (3)). The block is unconditionally in product-form. The steady state solution has the form $(x_1/\chi_4)^{m_3}$, and $x_4 = x_1$.

The condition of block 1 is satisfied if $x_4 = \chi_1\chi_3/\chi_2$. In order to reproduce the results of [39] it suffices to set $x_6 = \chi_3$:

$$\pi(\mathbf{m}) \propto \left(\frac{\chi_3}{\chi_2}\right)^{m_1} \left(\frac{\chi_1\chi_3}{\chi_2\chi_4}\right)^{m_3} \left(\frac{\chi_6}{\chi_3}\right)^{m_4} \left(\frac{\chi_3}{\chi_6}\right)^{m_5}.$$

5.3.4 A second example

In this section, we analyze one of the examples illustrated in [56]. The SPN structure is shown in Figure 5.13. This example is significant because it deals with transitions with the same input vectors and transitions with the same output vectors. In the original paper it is studied using Theorem 2. We can identify the building blocks of

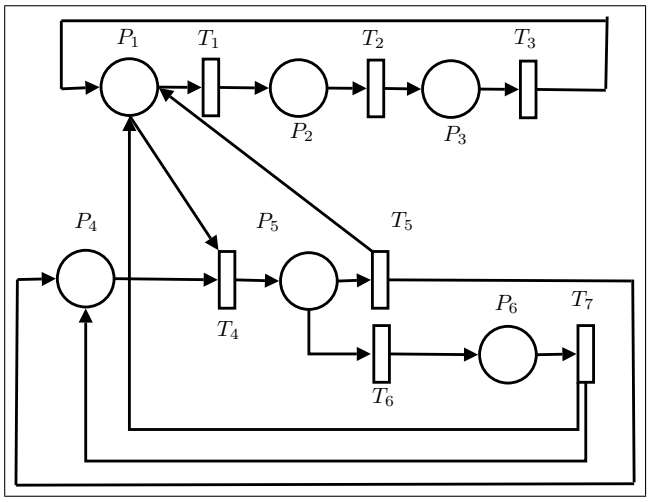

Figure 5.13: CHC-SPN of the example of Section 5.3.4.

Figure 5.14. We can simply focus on building blocks 1 and 4. Building block 1 is an incomplete building block, and is unconditionally in product-form. Its product-form is $(x_3/\chi_1)^{m_1}[(x_5\chi_1)/(\chi_4 x_3)]^{m_4}$. The other building blocks are simple exponential queues. We study block 4 because of the presence of T_5 and T_6. Obviously, in this case we have $x_5 + x_6 = x_4$, however, we have to establish the rate ratio for x_5 and

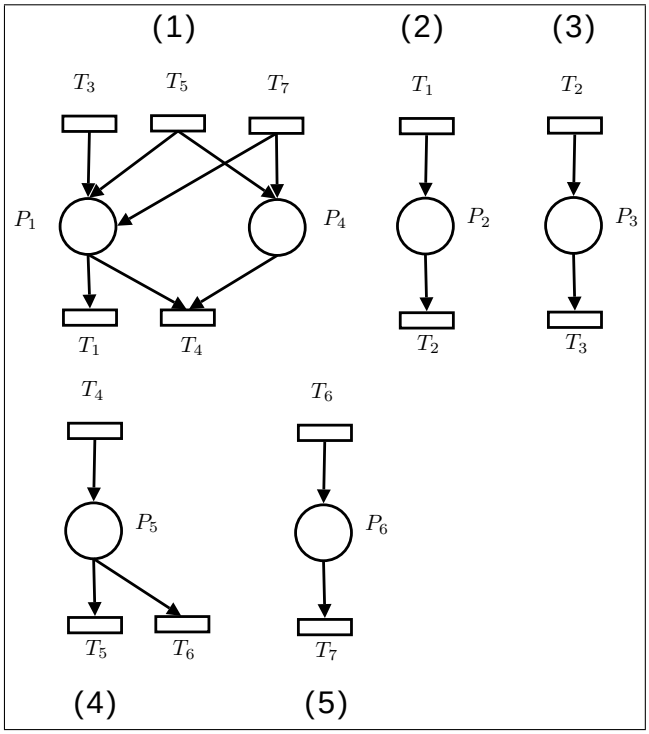

Figure 5.14: Decomposition in building blocks of the CHC-SPN of the example presented in Section 5.3.4.

x_6. We use Definition 5 that gives $x_6 = \chi_6/(\chi_6 + \chi_5)x_4$ and similarly we have for x_5. The system of equations for x_i is:

$$\begin{cases} x_1 = x_2 = x_3 \\ x_6 = x_7 \\ x_4 = x_5 + x_7 = x_5 + x_6 \\ x_6 = \frac{\chi_6}{\chi_6+\chi_5} x_4. \end{cases}$$

The linear system is underdetermined, so we have a solution for $x_1 = x_2 = x_3 = \chi_1$ and $x_4 = \chi_4$. The product-form of the model is given by:

$$\pi(\mathbf{m}) \propto \left(\frac{\chi_1}{\chi_2}\right)^{m_2} \left(\frac{\chi_1}{\chi_3}\right)^{m_3} \left(\frac{\chi_4}{\chi_5 + \chi_6}\right)^{m_5} \left(\frac{\chi_4\chi_6}{(\chi_6 + \chi_5)\chi_7}\right)^{m_6}.$$

5.3.5 Modular and hierarchical composition of CHC-SPNs: another example

Let us consider a CHC-SPN in which we can identify a set of transitions $\mathcal{T}_I \subseteq \mathcal{T}$ whose input vectors are the null vector, and the symmetrical set of transitions $\mathcal{T}_O$ whose output vectors are the null vector. In this case we can interpret the SPN

as an open model, where the input transitions model the arrivals to the system, and the output transitions model the departures. In this section we study product-form properties of a composition of CHC-SPNs. By composition of CHC-SPNs we mean a model in which a subset of the input transitions of a block becomes output transitions of the other block. Basically, the firing of an output transition corresponds to the firing of the corresponding input transition.

The conditions that the composed CHC-SPNs must satisfy in order to have a product-form solution are derived from ERCAT condition, therefore we analyze the model from the PEPA point of view (see Note 1 of Chapter 4). Suppose that we have two SPN models S_1 and S_2 and that we want to connect them as follows: output transition of S_1 $T_o \in \mathcal{T}_O^{(S_1)}$ has to be connected with transition $T_i \in \mathcal{T}_I^{(S_2)}$. Recall that, following our conventions, the action type t_o (t_i) in the PEPA model is associated with the firing of T_o (T_i) for the components that correspond to the state of the SPN in which T_o (T_i) is enabled. Let us call the PEPA definitions of S_1 and S_2: S_1^* and S_2^* respectively. Then we replace transition (t_i, χ_i) in S_2^* as follows:

$$S_2^{**} \stackrel{def}{=} S_2^*\{(t_i, \chi_i) \leftarrow (t_o, \top_o)\},$$

and the composed model S^* can be defined as follows:

$$S^* = S_2^{**} \underset{\{t_o\}}{\bowtie} S_1^*.$$

It is also possible to connect more output transitions of S_1 to the same input transition of S_2 (if one wants to connect more than one input transition to the same output transitions it suffices to split the output transitions). Suppose one wants to connect output transitions T_{o1} and $T_{o2} \in \mathcal{T}_O^{(S_1)}$ with $T_I \in \mathcal{T}_I^{(S_2)}$. Then S_2^{**} does not change, but we define S_1^{**} as:

$$S_1^{**} \stackrel{def}{=} S_1^*\{(t_{o1}, \chi_{o1}) \leftarrow (t_o, \chi_{o1}),\ (t_{o2}, \chi_{o2}) \leftarrow (t_o, \chi_{o2})\},$$

that is the actions corresponding to the firing of the output transitions have the same type t_o. In general, if we want to connect a set of output transitions in S_1 with a set of input transitions in S_2 in the PEPA definition of S we have more than one action type. Before applying RCAT we have to check the following conditions:

- The passive actions are enabled in every derivative of S_2^*. This is obvious because the input transitions have null input vectors by hypothesis.

- The active actions in the reversed process of a CHC-SPN have to be always enabled in every derivative of S_1. In the forward process this means that for every state $\mathbf{m}$ of the SPN there has to exist a state $\mathbf{m}'$ such that $\mathbf{m}$ is reachable from $\mathbf{m}'$ through the firing of every output transition.

  This condition can be decided structurally using the minimal closed support T-invariants [109]. Let X be a minimal T-invariant, and $||X||$ its support (see Chapter 3), then X is a minimal closed support T-invariant if for every transition $T \in ||X||$ we have that:

– there is a transition in $||X||$ whose input vector is the output vector of T,
– there is a transition in $||X||$ whose output vector is the input vector of T.

Let T_o be an output transition, then if T_o is covered by a minimal closed support T-invariant, every reachable state $\mathbf{m}$ can be reached through the firing of T_o. The proof is trivial and is based on the definition of the firing sequence that from a general reachable state $\mathbf{m}$ takes the net to a state $\mathbf{m}'$ such that $\mathbf{m}$ is reachable from $\mathbf{m}'$ through the firing of T_o. Let us assume that X_{T_o} is a minimal closed support T-invariant covering output transition T_o, then there exists an input transitions $T_i \in ||X_{T_o}||$. Let $\mathbf{m}$ be a generic reachable state of the net. Since the input transitions are always enabled, T_i can fire and it takes the net to state $\mathbf{m}'$. By hypothesis, there exists a transition $T_1 \in ||X_{T_o}||$ such that the input vector of T_1 is the output vector of T_i, therefore T_1 is enabled by $\mathbf{m}_1$. If $T_1 = T_o$ then $\mathbf{m}_1 \xrightarrow{T_o} \mathbf{m}$ and this concludes the proof, otherwise we consider transition $T_2 \in ||X_{T_o}||$, $T_2 \neq T_1$, such that its input vector is the output vector of T_1. We can iterate this until transition T_n whose output vector is the input vector of T_o fires. Since X_{T_o} is a minimal support T-invariant, when transition T_n fires, T_o is the only transition of $||X_{T_o}||$ that has not fired. The described firing sequence is uniquely determined [109].

Note that in [109] the authors prove that the traffic equations of the routing processes of a CH-SPN admit the solutions only if every transition is covered by a minimal support closed T-invariant.

- The reversed rates of output transitions have to be constant for each transition. This is true thanks to the analysis of the building blocks made by ERCAT and then of the net by RCAT.

Therefore, if we have a set of CHC-SPNs that admit a product-form solution by RCAT, then we can compose them obtaining a new model that can be efficiently studied by RCAT.

Note that this approach enhances the modularity of that proposed by Coleman, Henderson et al. Moreover, our approach allows for a hierarchical modeling technique, in fact, a product-form CH-SPN still satisfies RCAT theorem, therefore it can be used as block for further modelings.

The following paragraph shows an example.

Example Let us consider the model depicted in Figure 5.15. The figure also shows the decomposition into building blocks. The building block with P_1 and P_2 is an incomplete basic building block, therefore it is in product-form unconditionally. The product-form is:

$$\left(\frac{x_8 + \chi_1}{\chi_5 + \chi_4}\frac{\chi_3}{\chi_2}\right)^{m_1}\left(\frac{\chi_2}{\chi_3}\right)^{m_2}.$$

The reversed rates are $x_5 = (x_8 + \chi_1)/(\chi_4 + \chi_5)\chi_5$ and $x_4 = (x_8 + \chi_1)/(\chi_4 + \chi_5)\chi_4$. The other blocks are simple blocks: $x_3 = \chi_2$, $x_6 = x_5 = x_8$, $x_4 = x_7$. The solution of the equations for x_i is $x_6 = x_5 = x_8 = \frac{\chi_1\chi_5}{\chi_4}$, $x_4 = x_7 = \chi_1$. Therefore, the steady state solution for the model is:

$$\pi_1(\mathbf{m}) \propto \left(\frac{\chi_1\chi_3}{\chi_4\chi_2}\right)^{m_1}\left(\frac{\chi_2}{\chi_3}\right)^{m_2}\left(\frac{\chi_1\chi_5}{\chi_4\chi_6}\right)^{m_3}\left(\frac{\chi_1\chi_5}{\chi_4\chi_8}\right)^{m_4}\left(\frac{\chi_1}{\chi_7}\right)^{m_5}. \tag{5.22}$$

The input transition set is $\mathcal{T}_I^{(1)} = \{T_1, T_2\}$ and the output transition set is $\mathcal{T}_O^{(1)} = \{T_7, T_3\}$. Note that minimal closed support T-invariants covering T_3 and T_7 are $(0,1,1,0,0,0,0,0)^T$ and $(1,0,0,1,0,0,1,0)^T$, respectively:

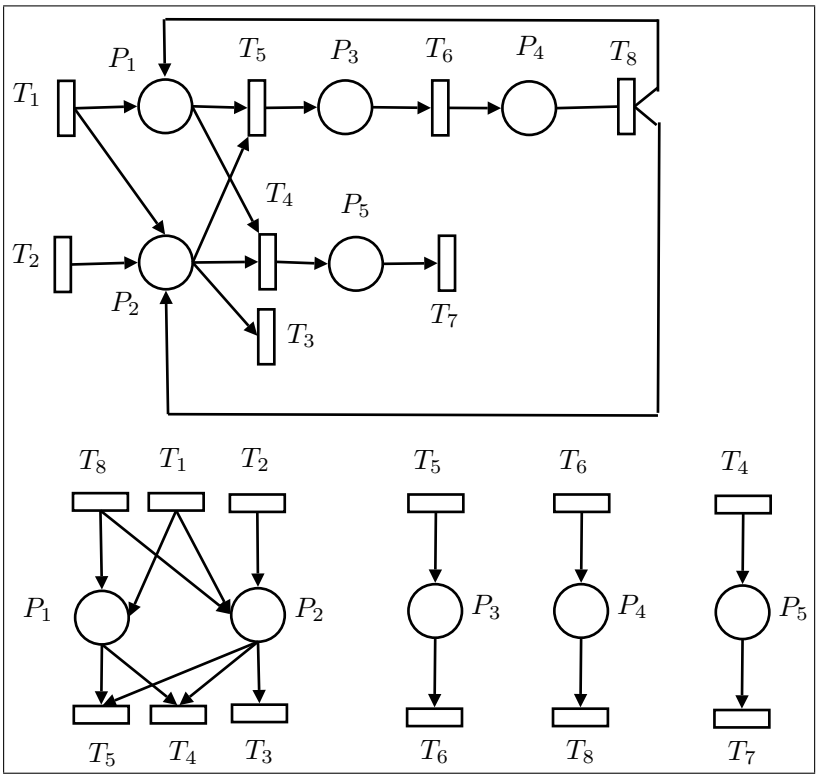

Figure 5.15: BLOCK1: CHC-SPN model with input/output transitions. Net and building blocks.

Now we consider the open CHC-model depicted in Figure 5.16 in which the input transition set is $\mathcal{T}_I^{(2)} = \{T_1, T_2\}$ and the output transition set is $\mathcal{T}_O^{(2)} = \{T_6, T_8\}$. In this case there are 4 building blocks, 3 of which are trivial. The block with places P_1, P_2, P_3 is an incomplete building block and is unconditionally in product-form. Its steady state probabilities can be expressed as:

$$\left(\frac{\chi_1\chi_4}{\chi_3 x_7}\right)^{m_1}\left(\frac{x_7}{\chi_4}\right)^{m_2}\left(\frac{\chi_2\chi_4}{\chi_5 x_7}\right)^{m_3}.$$

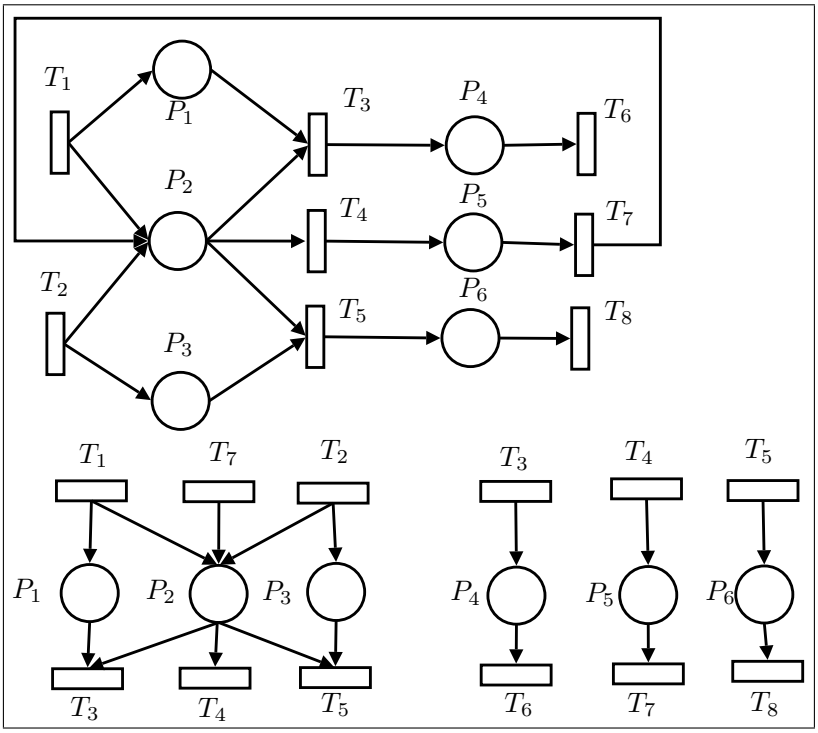

Figure 5.16: BLOCK2: CHC-SPN model with input/output transitions. Net and building blocks.

The reversed rates are given by $x_3 = \lambda_1$, $x_4 = x_7$ and $x_5 = \chi_2$. The analysis of the building block with P_5 gives the condition $x_7 = x_4$. We can choose $x_4 = x_7 = \chi_4$. Therefore, the product-form solution can be obtained straightforwardly:

$$\pi_2(\mathbf{m}) \propto \left(\frac{\chi_1}{\chi_3}\right)^{m_1} \left(\frac{\chi_2}{\chi_5}\right)^{m_3} \left(\frac{\chi_1}{\chi_6}\right)^{m_4} \left(\frac{\chi_4}{\chi_7}\right)^{m_5} \left(\frac{\chi_2}{\chi_8}\right)^{m_6}.$$

Now we want to compose models BLOCK1 and BLOCK2 such that a firing of output transition T_3 in BLOCK1 corresponds to a firing of an input transition T_1 in BLOCK2. Similarly, we do for T_7 and T_2. We use the notation $\chi_i^{(n)}$ with $n = 1, 2$ in order to distinguish the firing rates for model $n = 1$ and $n = 2$ where it could be ambiguous. In the expression of the steady state probability function π_2 we replace χ_1 and χ_2 by x_3 and x_7, respectively. Then we calculate the reversed rates x_3 and x_7 from model BLOCK1:

$$x_3 = \chi_2^{(1)}, x_7 = \chi_1^{(1)}.$$

Therefore, we can easily obtain the steady state solution for the composed model in product-form.

The solution of models BLOCK1 and BLOCK2 using Theorem 2 is given in Appendix B.

Note that:

- The composition can have feedback. In fact, RCAT just requires the reversed rates to be constant.
- Something similar to the probabilistic routing of the queueing networks can be modeled. One can do this by replacing an output transition with a set of transitions whose rates are such that their sum gives the original output transition rate.
- A model obtained by composition is itself a model that satisfies RCAT so it can be used in an hierarchical modeling.

5.3.6 The algorithm to identify the building blocks in an SPN

In this section we present a simple algorithm which identifies the building blocks in the SPN. The input of the algorithm is a CHC-SPN. The output is a partition of the set of the places, and each class of the partition contains the places of a building block. The algorithm is shown by Algorithm 1.

The correctness of the algorithm is based on the fact that the relation among the places determined by the building blocks is an equivalence relation. Therefore, sets $\mathcal{P}_i$ calculated by the algorithm are such that: $\cup_i \mathcal{P}_i = \mathcal{P}$ and $\mathcal{P}_i \cap \mathcal{P}_j = \emptyset$ if $i \neq j$.

The complexity is $\mathcal{O}(|\mathcal{P}| \cdot |\mathcal{A}|)$.

5.4 Conclusions

In this chapter we have presented a new technique to analyze SPN in product-form. We have restricted our analysis to a subset of SPN models that we have called CHC-SPN. This model class is characterized by the following structural properties:

- The transitions have constant firing rates,
- Every input vector of a transition has a correspondent output vector and vice versa,
- Every arc has multiplicity 1.

The analysis is based on the application of some theorems formulated for PEPA models, i.e., RCAT, ERCAT and MARCAT [59, 61, 62]. The main idea is that we can represent a CHC-SPN by the cooperation of PEPA agents, and then determine the product-form conditions and expression by those theorems. First we have studied some special CHC-SPNs called building blocks and then we have proved that every CHC-SPN can be expressed as a composition of building blocks. Building blocks

can be thought as the bricks of a general CHC-SPN. Note that it is not the case that every CHC-SPN is in product-form because the product-form conditions may depend on the transition firing rates.

In particular, we have shown that the composition of building blocks and of CHC-SPNs in product-form can be studied by the RCAT based approach. In this sense the technique presented in this chapter is modular and hierarchical.

The flexibility of low-level theorems such as those based on RCAT allows us to analyze hybrid models in product-form, i.e., we can compose a CHC-SPN in product-form with other models in a product-form based on RCAT. In literature RCAT has been successfully used to study product-forms that go from Jackson queueing networks [73, 59] to BCMP queueing networks [17, 58], G-networks [50, 60] and others [61]. For example, we can study a product-form queueing Petri net [19] without batch token movements without verifying the set of global balance equations.

It is worthwhile comparing our product-form SPN class with that of Coleman, Henderson et al. [63, 39]. As we pointed out before, the structural conditions of our CHC-SPN are more restrictive than those required by the model class studied in [63, 39] (CH-SPN). In fact, in the latter class, batch token movements are allowed as well as a sort of marking dependent firing rates (see Chapter 3 for a quick review). However, the results of Coleman, Henderson et al. are based on the analysis of the whole stochastic process of the considered SPN and they obtain an algebraic condition that arises from a matrix rank involving the transition rates. Our approach is more modular. Note that even if we have proved that for the building blocks the product-form criteria are the same when using ERCAT and Theorem 2, we cannot state the same for a composition of the building blocks. In other words, if we consider a CHC-SPN we cannot be sure that the conditions for the product-form required by our technique are identical to those required by Theorem 2. Further research efforts will be devoted to deeply explore this point.

We think that one of the contributions of this work is pointing out a proof for product-form SPNs based on a compositional analysis. Informally speaking, the techniques to prove a product-form based on the global balance equations verification tell us whether a model is in product-form or not, but do not say anything about the reasons that explain why it is or not. From this point of view, RCAT based theorems are more expressive. Historically, after the definition of the BCMP queueing network model class [17] several research efforts have been devoted to explore why some queueing disciplines originate a product-form model and others do not [94, 32, 98]. We think that this attention should be devoted to product-form SPNs as well.

In our opinion further research should study the following aspects:

- Using ERCAT to deal with building blocks with batch token movements. In this case the analysis of the building blocks cannot be based on the analysis of simple birth and death precesses. In particular, in our opinion, it is not obvious how to solve the system of equations generated by ERCAT conditions

in the general case.

- Extending the approach in order to deal with state dependent firing rates. This could be seen as a consequence of a generalization of RCAT theorem.

Input: SPN structure: $\mathcal{P}, \mathcal{A}, \mathcal{T}$
/* Input: places, arcs and transitions */
Output: Building blocks: $\mathcal{P}_i$
/* P_i is a subset of $\mathcal{P}$ such that $\mathcal{P}_i \cap \mathcal{P}_j = \emptyset$ with $i \neq j$ */
/* $\bigcup_i \mathcal{P}_i = \mathcal{P}$ */
/* The places in $\mathcal{P}_i$ belongs to the same building block i */
/* Identify the building blocks in a SPN */
foreach $t \in \mathcal{T}$ **do**
    /* $i-1$ denotes the number of classes identified */
    $i := 1$;
    /* Consider every output arc */
    $\mathcal{P}_t = \{P \in \mathcal{P}$ such that $(t, P) \in \mathcal{A}\}$;
    if *exists* $P \in \mathcal{P}_t$ *and* $j > 0$ *such that* $P \in \mathcal{P}_j$ **then**
        $\mathcal{P}_j := \mathcal{P}_j \cup \mathcal{P}_t$;
    end
    else
        /* Create a new class */
        $\mathcal{P}_i := \mathcal{P}_t$;
        $i := i + 1$;
    end
end

Algorithm 1: Algorithm which identifies the building blocks of a CHC-SPN.

6

Representing BCMP queueing centers by GSPN models

6.1 Introduction

In this chapter and in the following one we focus on the relations between two classes of product-form stochastic models: multiple-class BCMP QNs and a class of product-form Generalized Stochastic Petri Nets (GSPN). These results have been published in [15, 16, 14].

We have reviewed the BCMP product-form theorem in Chapter 2 and the various classes of product-form models based on GSPNs in Chapter 3. Although the product-form results are defined for different formalisms, it is important to consider that product-form is a property that is strictly related with the underlying CTMCs of the models. It is a good practice interpreting the conditions for the product-form of the stochastic process of a given model in terms of properties of the formalism used to express that model. However, if one aims to study the relations among product-form model classes it can be useful to investigate and characterize the corresponding CTMCs.

The starting point of this and the following chapter is the definition of a GSPN model for each BCMP service center type. Then, we define an equivalence relation and prove that every BCMP queueing center is equivalent to the associated GSPN model if considered in isolation. In particular, the equivalence terms are interesting because they preserve the average performance indices. This means that, under an appropriate interpretation of the state of the GSPN model (e.g. associating to every reachable marking the number of customers that are present in the corresponding queueing station), the average response time, the throughput and the mean number of customers for each class, are the same than those that can be derived by the analysis of the corresponding queueing system.

In the next chapter we study the compositional properties of these models and we show that they can be composed in order to preserve the BCMP stationary product-form solution. In order to achieve this result we explore a property that characterizes the CTMC of a BCMP queueing center: Muntz's $M \Rightarrow M$, that has been described in Chapter 2. Using the same property we also derive a result

concerning the characterization of service centers with probabilistic disciplines in QNs with BCMP product-form solutions. In particular, we derive a constraint on the possible choices for probabilistic queueing disciplines in multiple-class service centers that leads to a product-form stochastic process based on $M \Rightarrow M$.

Finally, we analyze the behavior of the defined GSPN models with non-BCMP models and we state sufficient conditions for a product-form solution.

It is worthwhile noting that the results are not obvious because multiclass QNs are *not* equivalent to SPN state machines (i.e. models whose transitions have at most one input and one output arc) since the queueing discipline influences the steady state probabilities and then also the conditions for the product-form solution. In fact, if we consider the stochastic process that underlies a BCMP queueing center we can easily show that it is not isomorphic to the stochastic process associated with the corresponding GSPN model. Therefore, the equivalence definition is not immediate and must be proved. In particular, we prove that:

- the aggregated stationary distribution for a the BCMP station as calculated in [17] is identical to an appropriately aggregated stationary distribution of the corresponding GSPN model,
- the composition of the introduced GSPN models admits a product-form solution.

Figure 6.1 illustrates by Venn diagrams the relations between the introduced GSPN models and the well-known classes of product-form models reviewed in Chapter 3. In particular the multiple-class BCMP QNs are mapped into a class of product-form GSPN models that is not included in any of the already well-known product-form GSPN models [39, 63, 83, 84, 6].

6.1.1 Motivations

The motivations of this work are twofold. First, let us consider the differences between the semantic of QN and GSPN models. The former formalism consists of a high-level definition (such as the description of a queueing discipline) and the definition of the associated Markov process is not automatic. On the other hand, GSPNs are defined at a lower level of abstraction, i.e., their semantic is entirely defined by a bipartite graph and a set of functions. Moreover, the Markov process (or the semi-Markov process) associated with a GSPN is uniquely defined. Hence, having a collection of GSPN models that, under a set of conditions, behave like the BCMP service centers, can be helpful in order to describe a stochastic model by a hybrid formalism that can be studied by a product-form analyzer. In Section 7.5 we present a simple example in order to clarify this point by using a model that combines GSPN and QN. Note that this approach is different from that used in [18, 19]. Another reason to explore the relations between the product-form of BCMP QNs and GSPN is theoretical and it goes in the direction of offering an

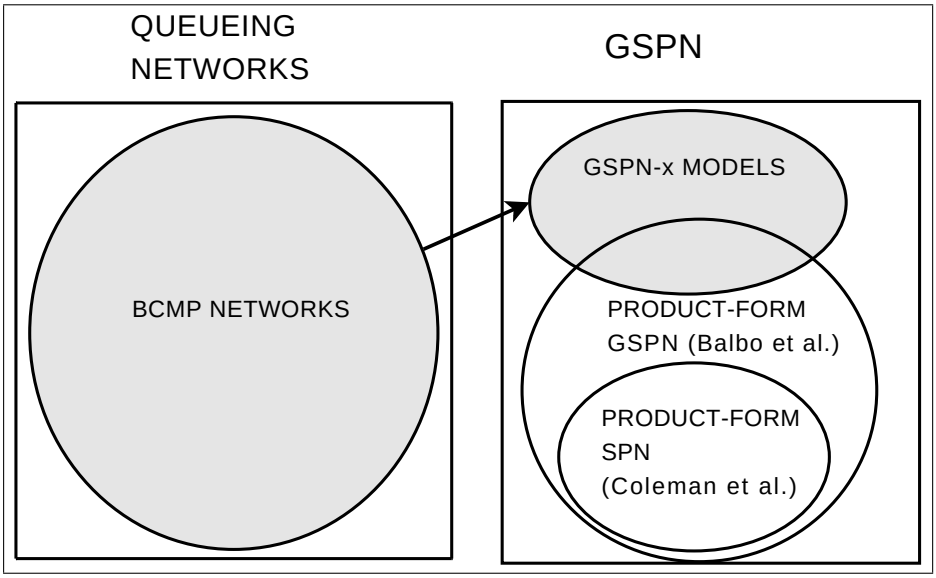

Figure 6.1: Venn diagrams illustrating relations between various product-form solutions.

original contribution to the effort of developing a unique theory for product-form interacting Markov processes.

6.1.2 Comments to bibliography

As first we think that it is important to recall that product-form is a property of the interacting CTMCs so it is interesting to study the relations among these classes of product-form processes. For example, a well-known result partially proved in [39] and explored deeply in [9] states that Coleman and Henderson product-form includes the SPNs called state machines which are the result of the mapping of Jackson and Gordon-Newell QNs [73, 54] (which can be considered single-class open and closed BCMP QNs with exponential servers) onto SPNs. This means that the same stochastic process has a well-known product-form solution both in the QN domain and the SPN domain. However, note that the problem becomes more complex when considering QNs with multiple classes of customers, different scheduling disciplines and Coxian service time distributions. All these factors are needed to determine whether a product-form solution exists and to identify its definition. On the other hand, standard GSPN models (i.e., those described in Chapter 3 without colors and without the possibility of specifying the discipline of the transitions or different distributions for the transition delays) do not implement any of these high-level features (even if one can consider classes of extended models based on SPNs, such as Queueing Petri Nets [18]). Hence, the conditions defined in the QN domain cannot be immediately reformulated in the GSPN domain. In [121] the authors introduce

a comparison between QN models and SPN models based on the representation of multiclass features by colored Petri nets. However, the differences between different scheduling disciplines are not analyzed. Balbo et al. in [5] combine GSPNs and product-form QNs by replacing subsystems in a low-level model with their flow equivalents models. Still little attention is devoted to the scheduling disciplines. In [7] the authors observe how they can map each service station of a BCMP QN into a complex GSPN. The GSPN model depends on the scheduling disciplines and it has an infinite number of places and transitions for the FCFS and LCFSPR stations but it is *not* in Coleman, Henderson et al. or Boucherie's product-form. They also give a finite and remarkably compact representation which satisfies the product-form conditions for SPN shown in [63, 39], but that representation does not distinguish different queuing disciplines by mapping everything to the PS discipline. Thus, it is not possible to define product-form conditions on the GSPN that are equivalent to those defined for the corresponding BCMP queueing stations. Roughly speaking, we can say that none of the product-form defined on Stochastic Petri Nets catches the product-form property of the stochastic process underlying the BCMP QNs.

6.1.3 Contribution

In this chapter we introduce a class of GSPN models that have a product-form solution, but that do not belong to any of the classes identified by Boucherie or Coleman, Henderson et al. In the next we prove that a combination of these models is in product-form. In order to achieve this result, we use two properties defined on the Markov process associated with the model. One is the $M \Rightarrow M$ property introduced by Muntz [94] and the other is introduced by Harrison in the PEPA domain [59, 61]. However, in general, the conditions expressed on the CTMCs cannot be straightforwardly interpreted in terms of general structural conditions of the GSPN.

In this chapter we define a set of GSPN models that are associated with every BCMP station type. Their stochastic processes are *not* isomorphic to those underlying the corresponding BCMP queueing centers, but we prove that the stationary probability distributions of the former ones are identical to appropriate aggregations (precisely defined in the following sections) of the stationary probability distributions of the latter ones. The GSPN models have a finite structure, thus they can be used and easily integrated in existing software tools which deal with GSPNs with state-dependent transition firing rates and immediate transition weights.

The definition of the GSPN models is based on the idea of mapping the deterministic queueing disciplines FCFS and LCFSPR used in BCMP theorem into probabilistic queueing disciplines with special features intended to represent the preemption and Coxian service times. The equivalence terms between RANDOM and FCFS queuing disciplines are well-known, however we do not just reformulate them in the GSPN domain [1], but we also show how *real* multiple servers can be represented by GSPN (hence not using one server with a load dependent service time). In

our opinion this leads to a more intuitive model of the system with multiple servers. Finally, we prove that in the GSPN model with probabilistic discipline, only the uniform choice between the queued customers leads to a product-form solution based on $M \Rightarrow M$. To the best of our knowledge, the GSPN model for LCFSPR is totally new.

In the next chapter we study the interaction among the defined GSPN models and other product-form GSPN blocks which satisfy RCAT theorem conditions. The conditions are derived from the analysis of the reversed processes of the GSPN models equivalent to the BCMP queueing stations. These results extend the application fields of these GSPN models. For example, we can combine a GSPN model equivalent to a FCFS BCMP station with single server with a G-Network with negative customers that satisfies RCAT conditions [60].

6.2 Representing BCMP stations by GSPN models

In this section we introduce the main results of our work, i.e., we present the four GSPN models which are shown to be equivalent to the corresponding BCMP stations in terms of performance indices. In this section we use $\mathbf{e}_i$ to denote a vector with all zero components but the i-th, that is equal to 1. Vector $\mathbf{e}_i$ has the same dimension of the GSPN marking that is being described or analyzed.

6.2.1 FCFS discipline

We define a GSPN model that represents a R-multiclass M/M/k/FCFS queue. Then we prove that the GSPN model is equivalent to the queuing system in terms of a given aggregation of the steady state probabilities. The structure of this section is organized in the following way:

1. The following Definition 7 formally introduces the GSPN-EXP model.

2. Lemma 4 states a closed form solution for the steady state probabilities of GSPN-EXP model by considering the set of reachable markings:

$$\mathbf{m} = (m_1, \ldots, m_{2R+1}).$$

3. Then we introduce a state aggregation by defining the aggregate state of the GSPN model such that a state differers from another just for the number of customers of one or more classes and not for being in queue or in service. Theorem 5 provides the closed form solution for model GSPN-EXP in terms of aggregated stationary probabilities.

4. Corollary 1 states that the GSPN-EXP model is equivalent to the M/M/k FCFS multiclass queueing system in terms of the aggregated stationary probabilities.

Given a M/M/K/FCFS station let us define the model called *GSPN-EXP*. We recall that: R is the number of classes, K the number of servers, μ is the class-independent and load-independent service rate for a single customer and λ_r the arrival rate for class r customer, $1 \leq r \leq R$.

Definition 7 (GSPN-EXP) *According to GSPN definition given in Chapter 3 let us define:*

- $\mathcal{P} = \mathcal{P}_q \cup \mathcal{P}_s \cup \{P_{2R+1}\}$ *with* $\mathcal{P}_q = \{P_1, \ldots, P_R\}$ *and* $\mathcal{P}_s = \{P_{R+1}, \ldots, P_{2R}\}$,
- $\mathcal{T} = \mathcal{T}_w \cup \mathcal{T}_q$ *where* $\mathcal{T}_q = \{t_1, \ldots, t_R\}$ *and* $\mathcal{T}_w = \{T_{R+1}, \ldots, T_{2R}\}$,
- *function* Π *is defined as follows:*

$$\Pi(\tilde{t_i}) = \begin{cases} 0 & \text{if } R+1 \leq i \leq 2R \\ 1 & \text{if } 1 \leq i \leq R \end{cases},$$

- *input and output vectors for transition* t_i, $1 \leq i \leq R$: $\mathbf{I}(t_i) = \mathbf{e_i} + \mathbf{e_{2R+i}}$ *and* $\mathbf{O}(t_i) = \mathbf{e_{R+i}}$. *Input and output vector for transition* T_{R+i}: $\mathbf{I}(T_{R+i}) = \mathbf{e_{R+i}}$ *and* $\mathbf{O}(T_{R+i}) = \mathbf{e_{2R+1}}$,
- $\mathbf{H}(t_i) = (0, \ldots, 0)$ *for all* $t_i \in \mathcal{T}$,
- $w(T_{R+i}, \mathbf{m}) = m_{R+i}\mu$ *for* $1 \leq i \leq R$ *and* $w(t_i, \mathbf{m}) = m_i$ *for* $1 \leq i \leq R$,
- $\mathbf{m_0} = (0, \ldots, 0, K)$.

Tokens arrive to places P_i, $1 \leq i \leq R$ *according to Poisson stochastic processes.*

Figure 6.2 illustrates the graphical representation of GSPN-EXP model where $t_1, \ldots, t_R$ are immediate transitions and $T_{R+1}, \ldots, T_{2R}$ are exponential transitions.

Note that the model defines a uniform probability among the tokens in places $P_1, \ldots, P_R$ to enter in place $P_{R+1}, \ldots, P_{2R}$. From a queueing system point of view, this means that every customer in the queue has the same probability to enter in service when a server becomes available. We prove in Section 7.3 that this is the only possible choice in order to have a product-form model based on $M \Rightarrow M$ property. Hence, let $\mathbf{m}$ be a valid vanishing state of the GSPN-EXP, and let $\mathcal{T}_a \subseteq \mathcal{T}_q$ be the set of immediate transitions enabled by $\mathbf{m}$, then the probability of firing of $t_i \in \mathcal{T}_a$ can be written as:

$$p(t_i, \mathbf{m}) = p_i(\mathbf{m}) = \frac{m_i}{\sum_{j \in \{j | t_j \in \mathcal{T}_a\}} m_j} \tag{6.1}$$

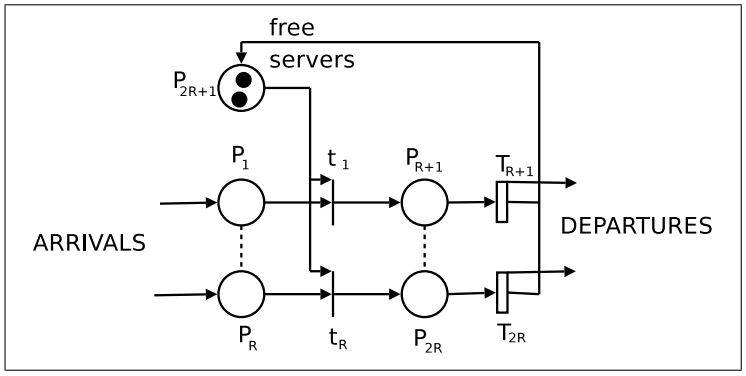

Figure 6.2: Graphical representation of GSPN-EXP model

Intuition of the model behavior. Let us review the model and give an interpretation in the queueing theory field. The system can be associated with a FCFS BCMP exponential station with R classes of customers. Tokens representing the customers arrive to places $P_1, \ldots, P_R$ with rates $\lambda_1, \ldots, \lambda_R$. The tokens in P_{2R+1} represent the free servers, i.e, in the initial state $m_{2R+1} = K$. The tokens in $P_{R+1}, \ldots, P_{2R}$ represent the customers in service. When a customer of class r arrives to P_r and a server is free, the customer immediately enters in P_{R+r} and a token is removed from P_{2R+1} through the firing of t_r. If $m_{2R+1} = 0$ at a class r customer arrival, then it has to wait in the queue represented by P_r. At job completions (firing of timed transitions T_{R+r}) there is a customer departure and a server becomes free in P_{2R+1}. It is easy to see that the GSPN-EXP model does not represent the customer arrival order, therefore it cannot implement a real FCFS discipline. In fact, when there are more than one customer in queue belonging to different classes, immediate transitions t_x are in conflict. According to GSPN semantic, the conflict is resolved probabilistically by assigning to each immediate transition a weight w. In our case the weight assigned to transition t_r is proportional to the number of class r customers waiting in P_r. Note that the weights of timed transitions represent the transition rates. The transition weight of T_r is μm_{R+r} where μ is the class independent service rate.

The following lemma gives a closed formula for the steady state probabilities of model GSPN-EXP.

Lemma 4 *Let* $\mathbf{m} = (m_1, \ldots, m_{2R+1})$ *be a reachable tangible state of the GSPN-EXP. Then if the stability condition holds, the stationary state probability can be written as follows:*

$$\pi(\mathbf{m}) = \pi_0 \prod_{i=1}^{R} \lambda_i^{m_i+m_{R+i}} \frac{(\sum_{i=R+1}^{2R} m_i)!}{\prod_{i=R+1}^{2R} m_i!} \cdot \frac{(\sum_{i=1}^{R} m_i)!}{\prod_{i=1}^{R} m_i!} \prod_{j=1}^{\sum_{i=1}^{2R} m_i} \frac{1}{\mu(j)}. \qquad (6.2)$$

where π_0 is a normalizing constant, $\mu(j) = x(j)\mu = \min(j, K)\mu$ following the BCMP conventions.

The proof is purely algebraic and is illustrated in Section A.1. It is based on verifying the set of the global balance equations.

The following theorem states the terms of the equivalence between a multiclass FCFS exponential queueing station and a GSPN-EXP model. It basically defines an aggregation of states on the GSPN-EXP model CTMC and on the FCFS station CTMC, then it states that the steady state probability function is the same in the two cases. The two aggregations are not exact lumps of the CTMCs, therefore an algebraic proof of the equivalence is required. From a modelling point of view, the aggregation is meaningful because in both cases the aggregated state can be interpreted as the number of customers (tokens) for every class either in service or in queue. As a consequence of this equivalence we can say that GSPN-EXP model and FCFS BCMP station have the same performance indices.

Theorem 5 *Consider the model GSPN-EXP and let $n_i = m_i + m_{R+i}$, $1 \leq i \leq R$ and $\mathbf{n} = (n_1 \ldots, n_R)$ be an aggregated state. Let $\pi_a(\mathbf{n})$ be the steady state probability of n_i for $i = 1, \ldots, R$. Then we can write:*

$$\pi_a(\mathbf{n}) = \pi_0 \frac{(\sum_{i=1}^{R} n_i)!}{\prod_{i=1}^{R} n_i!} \prod_{i=1}^{R} \lambda_i^{n_i} \prod_{i=1}^{\sum_{i=1}^{R} n_i} \frac{1}{\mu(i)} \quad \forall \mathbf{n} \in \mathbb{N}^R. \tag{6.3}$$

The proof is based on a convolution formula for binomial coefficients and can be found in Section A.2.

Corollary 1 *The M/M/k queuing system with FCFS discipline, R customer classes, arrival rates λ_i, $1 \leq i \leq R$, single server rate μ and steady state probability $\pi'(\mathbf{n})$ is equivalent to the GSPN-EXP in terms of steady state probability, i.e., $\pi_a(\mathbf{n}) = \pi'(\mathbf{n})$ for all $\mathbf{n} \in \mathbb{N}^R$ where $\pi_a(\mathbf{n})$ is the aggregated probability of GSPN given by formula (6.3).*

Proof 1 *It follows immediately from Equation (2.12) and Theorem 5.*

Note that it can be shown by trivial counterexamples (see Example 8) that GSPN-EXP does not have the steady state distribution (2.10) when the service rate is class dependent. Therefore, an automatic tool can decide if the model has a product-form solution by checking the firing rates of the timed transitions.

Example 8 *Consider a M/M/1/PS queue with $R = 2$ classes of customers with average service time $1/\mu_1$ and $1/\mu_2$. From queueing theory we can write the steady state probability function as follows:*

$$\pi'(n_1, n_2) = \pi'_0 \lambda_1^{n_1} \lambda_2^{n_2} \frac{(n_1 + n_2)}{n_1! n_2!} \frac{1}{\mu_1^{n_1} \mu_2^{n_2}}.$$

Suppose we aim to represent this queue by model GSPN-EXP with some changes, i.e., $w(T_3, \mathbf{m}) = m_3\mu_1$ *and* $w(T_4, \mathbf{m}) = m_4\mu_2$ *in the model of Figure 6.3. Let us determine the effective arrival rate to reachable tangible state* $\mathbf{m} = (0, m_2, 1, 0, 0)$*, with* $m_2 > 0$*. The adjacent states are* $\mathbf{m}_1 = (0, m_2 - 1, 1, 0, 0)$*,* $\mathbf{m}_2 = (1, m_2, 1, 0, 0)$ *and* $\mathbf{m}_3 = (1, m_2, 0, 1, 0)$*. Thus the effective arrival rate to state* $\mathbf{m}$ *is:*

$$\pi(\mathbf{m})\Big[\frac{1}{\lambda_2}\frac{m_2}{m_2}\mu_2\lambda_2 + \lambda_1(m_2+1)\frac{1}{\mu_1}\mu_1\frac{1}{m_2+1} + \lambda_2(m_2+1)\frac{1}{\mu_2}\mu_2\frac{1}{m_2+1}\Big] = \pi(\mathbf{m})[\mu_2 + \lambda_1 + \lambda_2].$$

The effective departure rate from state $\mathbf{m}$ *is clearly* $\pi(\mathbf{m})(\mu_1 + \lambda_1 + \lambda_2)$*, i.e., the GBEs are verified if* $\mu_1 = \mu_2$*. Therefore, we observe that model GSPN-EXP shares the same condition for the product-form of FCFS queueing centers.*

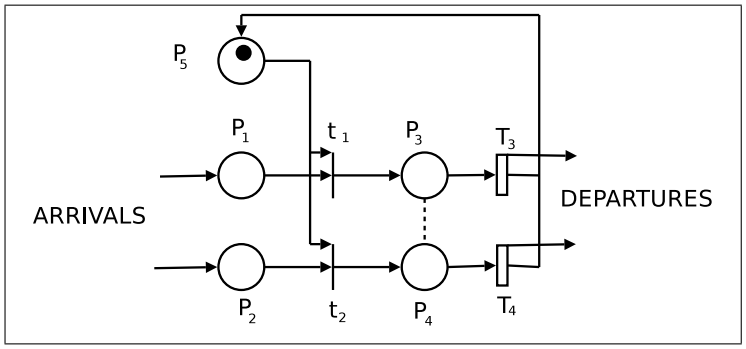

Figure 6.3: Model used in Example 8

GSPN-EXP can as well simulate a single server FCFS service station with a BCMP-like load dependent service rate as proved in the technical report [11]. The net structure complexity is linear on R, the number of customer classes.

6.2.2 LCFSPR discipline

In this section we introduce a GSPN which can be considered equivalent, for the steady state probability distribution, to a multiclass M/COX/K queue with LCFSPR scheduling discipline. We provide a model for this queueing system whose structure is finite and depends only on the number of classes of customers and stages of the Coxian service time, i.e., not on the number of servers. The structure of this section is similar to the previous one, but the proof of the aggregation results is more complex because it is divided in two steps. First, we prove that Coxian time distributions can be *replaced* by exponential ones with appropriate means without changing the performance measures, and then we aggregate the states such that two aggregated states are distinguished only by the number of customers for each class (and not for being in queue or in service).

Given a M/M/K/LCFSPR station let us define the model called *GSPN-COX*. We recall that: R is the number of classes, K the number of servers, $\mu_{r,\ell}$ is the class-dependent service rate at stage ℓ of the Coxian random variable, λ_r the arrival rate for class r customer, and L_r is the number of stages for class r customers, with $1 \leq r \leq R$ and $1 \leq \ell \leq L_r$.

Definition 8 (GSPN-COX) *According to the definition given in Section 3.3:*

- *Places.* $\mathcal{P} = \{P_{r,l}, P_{R+r,l}, P_{r,0} : \ 1 \leq r \leq R, 1 \leq l \leq L_r\} \cup \{P_0, P_{2R+1}\}$
- *Transitions.* $\mathcal{T} = \mathcal{T}_s \cup \mathcal{T}_p \cup \mathcal{T}'$, *where* $\mathcal{T}_s = \{ts_{r,l} : \ 1 \leq r \leq R, 0 \leq l \leq L_r\}$, $\mathcal{T}_p = \{tp_{r,l} : \ 1 \leq r \leq R, 1 \leq l \leq L_r\}$, $\mathcal{T}' = \{T_{r,l} : \ 1 \leq r \leq R, 0 \leq l \leq L_r\}$. *Function* Π *is defined as follows:*

$$\Pi(\tilde{t}) = \begin{cases} 1 & \text{if } \tilde{t} \in \mathcal{T}_p \cup \mathcal{T}_s \\ 0 & \text{if } \tilde{t} \in \mathcal{T}' \end{cases}$$

- *Arcs. Let* $tp_{r,l} \in \mathcal{T}_p$, *then* $\mathbf{I}(tp_{r,l}) = \mathbf{e_{r,l}} + \mathbf{e_0}$, $\mathbf{H}(tp_{r,l}) = \mathbf{e_{2R+1}}$ *and* $\mathbf{O}(tp_{r,l}) = \mathbf{e_{2R+1}} + \mathbf{e_{R+r,l}} + \mathbf{e_0}$. *Let* $ts_{r,l} \in \mathcal{T}_s$ *and* $l > 0$, *then* $\mathbf{I}(ts_{r,l}) = \mathbf{e_{R+r,l}} + \mathbf{e_{2R+1}}$, $\mathbf{H}(ts_{r,l}) = \mathbf{e_0}$ *and* $\mathbf{H}(ts_{r,l}) = \mathbf{e_{r,l}}$. *Let* $ts_{r,0} \in \mathcal{T}_s$ *then* $\mathbf{I}(ts_{r,0}) = \mathbf{e_{r,0}} + \mathbf{e_0}$, $\mathbf{H}(ts_{r,0}) = \mathbf{0}$ *and* $\mathbf{O}(ts_{r,0}) = \mathbf{e_{r,1}}$. *Let* $\mathbf{T}_{r,l} \in \mathcal{T}'$ *then* $\mathbf{I}(T_{r,l}) = \mathbf{e_{r,l}}$, $\mathbf{H}(T_{r,l}) = 0$. *If* $l < L_r$ *then the output vector is probabilistic:* $\mathbf{O}_1(T_{r,l}) = \mathbf{e_{2R+1}}$ *with probability* $b_{r,l}$ *and* $\mathbf{O}_2(T_{r,l}) = \mathbf{e_{r,l+1}}$ *with probability* $a_{r,l}$ *with* $a_{r,l} + b_{r,l} = 1$. *If* $l = L_r$ *the output vector is deterministic,* $\mathbf{O}(T_{r,L_r}) = \mathbf{e_{2R+1}}$.
- *Weights. Let* $1 \leq r \leq R$ *and* $l > 1$ *then* $w(ts_{r,l}) = m_{R+r,l]}$, $w(ts_{r,0}) = 1$, $w(tp_{r,l}) = m_{r,l}$ *and* $w(T_{r,l}) = \mu_{r,l} m_{r,l}$.
- *Initial marking.* $\mathbf{m_0} = (0, \ldots, 0, K)$.

Tokens arrive to places $P_{r,0}$ *for* $r = 1, \ldots, R$ *and* P_0.

Note that, given a marking $\mathbf{m}$, $m_0 = 1$ if and only if $m_{r,0} = 1$ for a r, and that if $m_0 > 0$ then $\mathbf{m}$ is a vanishing marking, with $1 \leq r \leq R$. Figure 6.4 illustrates a graphical model for $R = 2$, $L_1 = 3$, $L_2 = 2$. Exponential transitions $T_{0,1}$ and $T_{0,2}$ are introduced to show the arrival behaviors. The dotted lines and the solid lines represent alternative firing modes while grey lines are used for sake of clarity. Finally, transitions tp have an incoming arc from P_0 and an outgoing art to P_0, we represent this by an arc with a double arrow.

Intuition of the model behavior. We can give the following interpretation to the model: place P_5 has as many tokens as the free servers, place $P_{r,\ell}$, $1 \leq r \leq 2$, has as many tokens as the number of customers being served at stage ℓ of class r. The tokens in place $P_{R+r,\ell}$, $1 \leq r \leq 2$ represent the number of customers in the queue

of class r and stage ℓ. When a customer of class r arrives to the system a token is temporarely (i.e. it originates a vanishing state) stored in places $P_{r,0}$ and P_0. Immediate transition $t_{s,0}$ puts the arrived token in service at stage 1, the customer which must be preempted is chosen probabilistically by the transitions tps. When a customer leaves the system by the firing of a timed transition, a preempted customer from places $P_{R+r,\ell}$ is chosen probabilistically by a transition tss and it resumes the service.

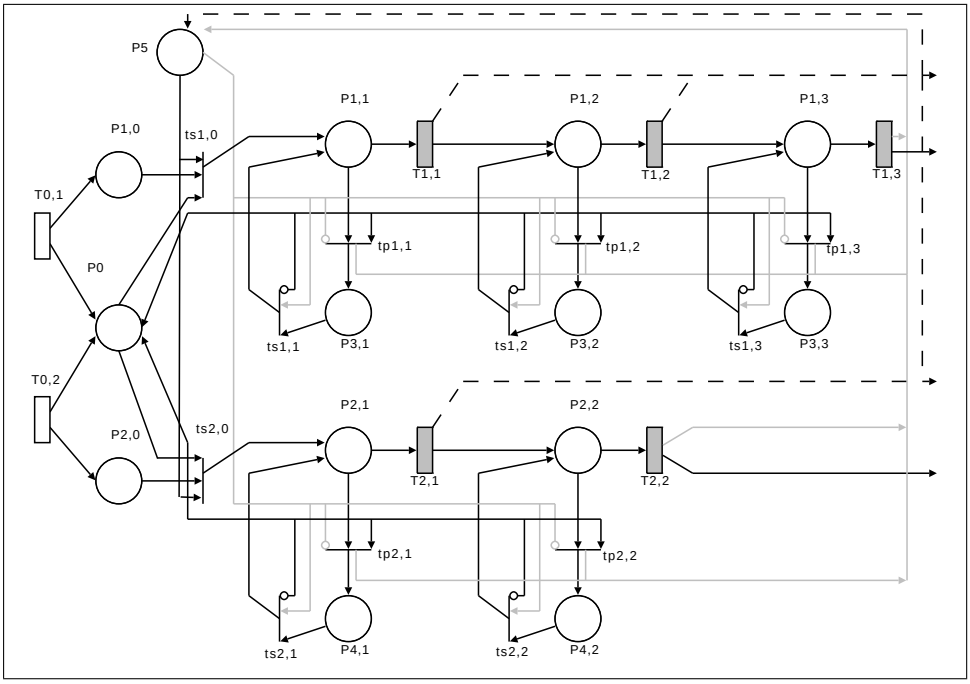

Figure 6.4: GSPN-COX for $R = 2$ classes, $L_1 = 3$ and $L_2 = 2$ stages.

In the following, we first introduce a lemma which gives the stationary probabilities for model GSPN-COX under Poisson arrivals. Then we aggregate the states giving the marginal probabilities in closed form and the final corollary states how the model GSPN-COX can be considered equivalent to a M/COX/k/LCFSPR station.

Lemma 5 *Let* $\mathbf{m}$ *be a tangible state of model GSPN-COX. Then the steady state*

probability of $\mathbf{m}$ *is given by:*

$$\pi(\mathbf{m}) = \pi_0 \Big[\prod_{r=1}^{R} \lambda_r^{m_r+m_{R+r}} \Big] \cdot \Big[\frac{(\sum_{r=1}^{R} m_r)!}{\prod_{r=1}^{R}\prod_{l=1}^{L_r} m_{r,l}!} \frac{(\sum_{r=1}^{R} m_{R+r})!}{\prod_{r=1}^{R}\prod_{l=1}^{L_r} m_{R+r,l}!} \Big] \cdot \Big[\prod_{r=1}^{R}\prod_{l=1}^{L_R} \Big(\frac{A_{r,l}}{\mu_{r,l}}\Big)^{m_{R+r,l}+m_{r,l}} \Big] \Big[\prod_{a=1}^{m} \frac{1}{\min\{n,k\}} \Big], \quad (6.4)$$

where $m_r = \sum_{l=1}^{L_r} m_{r,l}$, $m_{R+r} = \sum_{l=1}^{L_r} m_{R+r,l}$, $m = \sum_{r=1}^{R} m_r + m_{R+r}$ *and* π_0 *is the normalizing constant.*

The proof is based on verifying that formula (6.4) satisfies the set of GBEs of the model and is illustrated in Section A.3.

The following lemma states a closed formula for an exact aggregation of states. The basic idea is to obtain an aggregated state which does not represent the stage at which a token is, but it just distinguishes between a token being in service and being in queue. The actual relevance of the lemma is to show a result which is equivalent to the well-known result of the BCMP theorem, that is, the marginal distribution for a Coxian service time LCFSPR station is equal to that of the same station with an exponential service time distribution with the same mean. In other words, by Lemmas 5 and 6 we have derived the steady state probabilities for two levels of details of the state representation: a detailed state, where the stage of services are represented, and a more compact one, where the customers are distinguished only by the class and by the fact that they are in the queue on being served. In practice, this is helpful because often a simplified version of model GSPN-COX can be used, i.e., with just an exponential stage for each class. However, the detailed state represents more information, and the generalized GSPN-COX model has a theoretical relevance because it allows us to prove the insensitivity property for this probabilistic queueing discipline that, to the best of our knowledge, has never been studied before. In other words, a modeler can decide to use a GSPN-COX model with multiple stages to give a detailed description of the system, but as a consequence of Lemma 6 he/she can decide to perform an analysis that is not interested in distinguishing the customers for each stage of service, using a simplified model with just one service stage for class.

Lemma 6 *Let us define the aggregate tangible state as follows:*

$$\mathbf{n} = (n_1, \ldots, n_R, n_{R+1}, \ldots, n_{2R}, n_{2R+1}),$$

where $n_r = m_r = \sum_{l=1}^{L_r} m_{r,l}$ *and* $n_{R+r} = m_{R+r} = \sum_{l=1}^{L_r} m_{R+r,l}$ *and* $n_{2R+1} = m_{R+1}$, *with* $1 \le r \le R$ *and* $\mathbf{m}$ *is a tangible state of model GSPN-COX. The steady state*

probabilities for GSPN-COX can be written as follows:

$$\pi_A(\mathbf{n}) = \pi_0 \prod_{r=1}^{R} \lambda_r^{n_r+n_{R+r}} \frac{(\sum_{r=1}^{R} n_r)!}{\prod_{r=1}^{R} n_r!} \frac{(\sum_{r=1}^{R} n_{R+r})!}{\prod_{r=1}^{R} n_{R+r}!} \cdot \prod_{r=1}^{R} \left(\frac{1}{\mu_r}\right)^{n_r+n_{R+r}} \prod_{a=1}^{n} \frac{1}{\min(n,k)}, \quad (6.5)$$

where $1/\mu_r$ *is the mean service time, i.e.,* $\sum_{l=1}^{L_r} A_{r,l}/\mu_{r,l}$, $n = \sum_{r=1}^{R} n_r = m$.

The proof of this lemma is given in Section A.4.

The following theorem states the equivalence of the BCMP LCFSPR marginal probabilities and the GSPN-COX ones, that is, the fundamental result of this section.

Theorem 6 *Consider the model GSPN-COX and let* $\mathbf{m}$ *be a tangible state. Let the aggregated state* $\mathbf{u}$ *be defined as follows:* $u_r = \sum_{l=1}^{L_r} m_{r,l} + \sum_{l=1}^{L_r} m_{R+r,l}$ *for* $r = 1, \ldots, R$, *i.e.,* u_r *represents the number of customers of class* r *in the system, either in service or in queue. Then we can write:*

$$\pi_A(\mathbf{u}) = \pi_0 \frac{u!}{\prod_{r=1}^{R} u_r!} \prod_{r=1}^{R} \left[\lambda_r^{u_r} \left(\frac{1}{\mu_r}\right)^{u_r}\right] \prod_{a=1}^{u} \frac{1}{\min\{K,a\}}, \quad (6.6)$$

where $u = \sum_{r=1}^{R} u_r$.

The proof is based on a convolution for multinomial coefficient and is very similar to that of Theorem 6.

Note on the model CTMCs. It is worthwhile considering the case that the relations between the GSPN-COX and GSPN-EXP models and the corresponding BCMP stations can be deduced by ax exact lumping of states in the CTMCs. This approach would be appreciable because it would imply a stronger equivalence than the one given here. For example, the product-form of combinations of this class of models would be a straightforward consequence. However, it is easy to show that the CTMC of GSPN-EXP cannot be obtained by an exact lumping of the CTMC of FCFS BCMP queueing station, and the same holds for GSPN-COX and LCFSPR.

6.2.3 IS and PS disciplines

In the BCMP theorem, the state of PS and IS scheduling disciplines is defined as a finite vector with fixed size [17]. In fact, for both disciplines the state represents the

number of customers in the station for each class and stage of service. Therefore, the arrival order in the queue does not influences the steady state probabilities of the corresponding CTMC.

In the following we give the definition of a GSPN model equivalent to a PS station and immediately after we explain the differences with a IS station.

Given the M/M/K/PS models let us define the model called *GSPN-PS*. We recall that: R is the number of classes, K is the number of servers, $\mu_{r,\ell}$ is the load-independent service rate for class r customers at stage ℓ, and λ_r the arrival rate for class r customers, $1 \leq r \leq R$. $a_{r,\ell}$ denotes the probability that a class r customer goes to stage $\ell+1$ after being server at stage ℓ, and $b_{r,\ell} = 1 - a_{r,\ell}$ is the probability of leaving the system after stage ℓ.

Definition 9 (GSPN-PS) *According to the GSPN definition given in Chapter 3 let us define:*

- $\mathcal{P} = \{P_{r,\ell} : \ 1 \leq r \leq R, 1 \leq \ell \leq L_r\}$,
- $\mathcal{T} = \{T_{r,\ell} : \ 1 \leq r \leq R, 1 \leq \ell \leq L_r\}$,
- *function* Π *is defined as* $\Pi(\tilde{t}) = 0$ *for all* $\tilde{t} \in \mathcal{T}$,
- *input and output vectors for transition* $T_{r,\ell}$ *are* $\mathbf{I}(T_r, \ell) = \mathbf{e}_{r,\ell}$, $O_1 T_r, \ell = \mathbf{0}$ *with* $1 \leq \ell \leq L_r$, *and probability* $b_{r,\ell}$. $\mathbf{O}_2(T_r, \ell) = \mathbf{e}_{r,\ell+1}$ *with* $1 \leq \ell < L_r$, *with probability* $a_{r,\ell}$.
- $\mathbf{H}(T) = \mathbf{0}$ *for all* $T \in \mathcal{T}$,
- $w(T_{r,\ell}, \mathbf{m}) = m_{r,\ell} / [\sum_{s=1}^{R} \sum_{i=1}^{L_r} m_{s,i}] \mu_{r,\ell}$ *for* $1 \leq r \leq R$ *and* $1 \leq \ell \leq L_r$.
- $\mathbf{m_0} = (0, \ldots, 0)$.

Tokens arrive to places $P_{r,1}$, $1 \leq r \leq R$ *according to independent Poisson stochastic processes.*

In order to obtain an IS station it suffices to change the definition of function w as follows:

$$w(T_{r,\ell}) = m_{r,\ell} \mu_{r,\ell}$$

Figure 6.5 illustrates the model for two classes with $L_1 = 3$ and $L_2 = 2$.

Intuition of the model behavior GSPN-PS and GSPN-IS models differ from GSPN-EXP and GSPN-COX for several reasons. First of all, in GSPN-EXP and GSPN-COX models the number of servers was represented by the number of tokens in the initial state in a specific place. In the GSPN-PS model the number of servers influences the weight function w of the models. The idea is that transition $T_{r,\ell}$ rates is proportional to the number of tokens in $P_{r,\ell}$. The total computation rate is shared

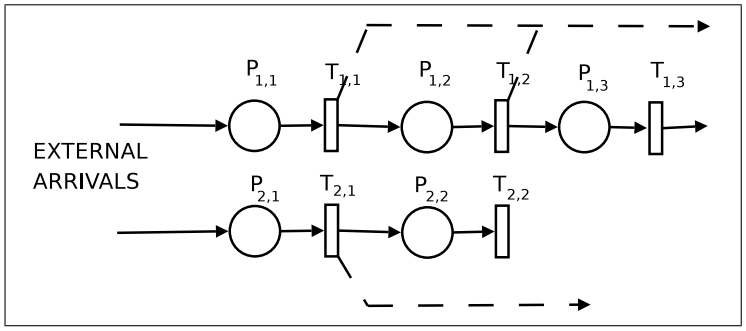

Figure 6.5: PS and IS model example for two classes of customers, with $L_1 = 3$ and $L_2 = 2$ stages of service.

by all the customers in the model, i.e., $1/\sum_{r=1}^{R}\sum_{\ell=1}^{L_r} m_{r,\ell}\mu_{r,\ell}$. For GSPN-IS model the transition firing rates for each token are independent of the total number of customers in the station.

Note 2 *The CTMCs underlying GSPN-IS and GSPN-PS models are isomorphic to those defined in BCMP theorem [17] for the IS and PS scheduling disciplines respectively. Therefore it is obvious that the stationary probabilities are the same and that they yield the same properties of the corresponding queueing models.*

We can say that the equivalence between GSPN-PS, GSPN-IS and PS, IS BCMP scheduling disciplines is stronger than that introduced for GSPN-EXP and FCFS, GSPN-COX and LCFSPR. In fact in the former cases there is an isomorphism between the CTMCs while in the latter ones there is an equivalence of the steady state probabilities under specific aggregations.

6.3 Conclusions

In this chapter we have defined a GSPN model for each queueing center type defined in the BCMP theorem [17]. Then, we have defined an equivalence relation that preserves the performance indices and we have proved that the BCMP stations are equivalent to the corresponding GSPN models under class independent Poisson arrivals.

In this chapter we have mainly given a theoretical result that is functional to the following chapter. In order to clarify the importance and the practical relevance of these proposed models, we need to study their compositional properties. We address this problem in the following chapter, and we show that the equivalence holds even under some non-Poisson arrival processes.

However, it is interesting to note that the defined probabilistic queueing disciplines are suitable to replace the corresponding deterministic ones if the modeler is

interested in the analysis of the average performance indices (of course the response time distributions for a probabilistic queueing discipline and the corresponding deterministic one cannot be the same).

7

Composition of GSPN models equivalent to BCMP stations

7.1 Introduction

In this chapter we study the compositional properties of the GSPN models introduced in Chapter 6. In particular, we are interested in determining the conditions under which a model consisting of a set of sub-models, some of which are GSPN-EXP, GSPN-COX, GSPN-IS or GSPN-PS, has a product-form solution. We can distinguish three types of possible compositions:

BCMP-like composition: In this case the GSPN blocks called GSPN-EXP, GSPN-COX, GSPN-IS and GSPN-PS are used to obtain a model that is equivalent in terms of average performance indices to a BCMP queueing network (QN). The product-form property of the BCMP QNs is maintained in the equivalent GSPN model. This technique can be useful for large models with complex load-dependent service rates, i.e., models that do not satisfy the conditions required by the algorithms defined to study the BCMP QNs [27, 114, 29, 34, 93, 101, 40, 26, 107, 41]. In fact, in these cases simulation can be required due to numerical problems or algorithm preconditions not satisfied. GSPN simulators are widely available therefore it can be useful to map a BCMP QN into a GSPN.

Extended BCMP-like composition: After the introduction of product-form QNs [73, 54, 17], several research efforts have been devoted to characterize queueing stations that can be embedded within a product-form network originating maintaining the product-form property. Since product-form is a property related to the CTMC of the cooperating models, some authors have worked in this direction (see for example [76]), while others focused on the high level properties of the QN stations that originate product-form solutions (for example [32]). Muntz's property $M \Rightarrow M$ can be seen from both these points of view (see Chapter 2). In fact, it requires to analyze the CTMC of a station in isolation in order to decide whether it can be composed with other $M \Rightarrow M$ models originating a product-form model, but it also involves typical high-level

notions of queueing theory such as *customers*, *customer arrivals* or *customer departures*. The main strength of this property is that it identifies a set of queueing stations with a general (non structural) property that can be composed in order to obtain product-form queueing networks. For example, Le Boudec used this result in order to prove that a new queueing discipline (with practical applications) was in product-form by $M \Rightarrow M$ [86], and a similar approach has been used by Afshari et al. in [1]. However, it is not easy to take advantage from this generalization of BCMP QNs in software tools. Ideally the modeler would like a tool that allows the specification of a queueing discipline and then it should be able to decide whether the discipline satisfies the $M \Rightarrow M$ property or not. There are two problems in this sense: the first one is that the queueing disciplines are usually described at high level, therefore it can be hard to automatically define the underlying stochastic process. The second one is that deciding the $M \Rightarrow M$ property requires the analysis of the whole stochastic process from its steady-state solution, that can be unfeasible in case of large of infinite state spaces. Therefore, the production of a tool that allows a product-form analysis with arbitrary queueing disciplines added to the BCMP ones appears to be a really difficult task. We partially overcome the first problem by using GSPNs as formal language to describe queueing disciplines or stations. We show that the GSPN models defined in Chapter 6 satisfy the $M \Rightarrow M$ property, therefore they can be combined with other GSPN models, even not equivalent to BCMP stations, obtaining a product-form model. For example, a practical consequence is that we can represent the Le-Boudec MSCCC station [86] using a GSPN model, combine it with GSPN-EXP, GSPN-COX, GSPN-IS and GSPN-PS and study or simulate the resulting model. In the following sections we analyze how these models have to be combined.

Non BCMP-like composition: BCMP product-form QNs require the solution of a linear system, called system of the traffic equations. In the last few years a class of product-form models with non-linear traffic equations has been defined. One of the first results in this direction is the definition of the product-form for G-Networks [50]. In his talk at SIGMETRICS 2008 E. Gelenbe claims that:

> There can be product-form solutions (a) without local balance, (b) without one step transitions, (c) without quasi-reversibility, (d) with non-linear traffic equation.

However, the proves based on the verification of the set of global balance equations of the CTMC underlying the model are difficult to handle. On one hand they often require a lot of calculations and on the other hand they just do not give information on what causes (or prevents) the product-form solution. RCAT-based theorems [59, 61, 62] base the identification of the product-form

of a model on the analysis of the reversed processes of the interacting submodels. These theorems, reviewed in Chapter 4, have been successfully used to prove several classes of product-forms. In particular, they can be used to study Jackson QN product-form [59] and also models with non-linear traffic equation systems [60]. In this chapter we reconsider the GSPN models introduced in Chapter 6 and show that they can be composed using RCAT under some restrictions. This result extends their field of application. We show how a modeler can take advantage of this result in order to study hybrid models (i.e. models that can be defined by different formalisms) in product-form.

The chapter is structured as follows. Section 7.2 studies the composition of GSPN-EXP, GSPN-COX, GSPN-IS and GSPN-PS models by $M \Rightarrow M$, Section 7.4 studies the composition of the same models using RCAT. We need both these analysis because we show that it is not the case that one class of models includes the other, even if there is a non-empty intersection. Within Section 7.4 we also show how it is possible to use RCAT results to define new product-form GSPNs. Section 7.3 considers a generic probabilistic queueing discipline with exponential service time and without preemption. It investigates the conditions on the probability distribution of a customer to obtain the service and we prove that only the uniform distribution characterizes a model that satisfies the $M \Rightarrow M$ property. Finally, Sections 7.5 and the following ones show some examples of hybrid modeling. The examples are structured as follows: the first one shows a modeling of a real system that is in product-form, the second one shows a hybrid model in product-form using G-queues and originates a linear system of traffic equations, the third one illustrates a hybrid model in product-form with a non-linear system of traffic equations.

7.2 $M \Rightarrow M$ property on GSPN models

Markov implies Markov property has been introduced by Muntz in [94] with the aim of characterizing the queueing disciplines whose composition has a product-form solution. In particular, every BCMP queueing discipline satisfies the $M \Rightarrow M$ property. In order to prove that the compositions of GSPN-EXP, GSPN-COX, GSPN-IS, GSPN-PS models have product-form solutions, we use the $M \Rightarrow M$ property. However, this property is defined in terms of queueing theory notions, such as customers, service completion etc. In this section we simply reformulate the $M \Rightarrow M$ property so that it can be used in the GSPN analysis. The main idea is based on the work of Melamed [90] which generalizes [94]. Let us consider an ergodic CTMC with state space Γ and a collection of sets of traffic transitions denoted by $\Theta_1, \ldots, \Theta_B$ where $\Theta_i \subseteq \Gamma \times \Gamma$ and $\Theta_i \neq \emptyset$. Let us define $K_i(t)$ as the process which counts the number of transitions $(\alpha, \beta) \in \Theta_i$ up to t. Let $m_i(\gamma)$ be defined as $m_i(\gamma) = \sum_{\eta \in \Theta_i(\cdot,\gamma)} \pi(\eta) q(\eta, \gamma)$, where $\Theta_i(\cdot, \gamma) = \{\beta | (\beta, \gamma) \in \Theta_i\}$, $q(\eta, \gamma)$ is the transition rate between states η and γ and $\pi(\eta)$ is the stationary probability of state

η. Intuitively $m_i(\gamma)$ is the total flux to state $\gamma \in \Gamma$ limited to the transitions in Θ_i. We define $m_i = \sum_{\gamma \in \Gamma} m_i(\gamma)$.

Example 9 *In order to clarify these definitions we introduce the following example. Let us consider a BCMP FCFS station with $R = 2$ classes of customers. The state of the station is its occupancy vector, i.e., a vector whose i-th component is the label of the class of the i-th oldest customer in the station. The size $N(t)$ of the vector is not fixed and the customer in service is in position $N(t)$. Figure 7.1 illustrates part of the CTMC, where λ_1, λ_2 are the arrival rates of class 1 and class 2 customers and similarly for the service rates μ_1 and μ_2. Suppose that Θ_1 and Θ_2 identify the*

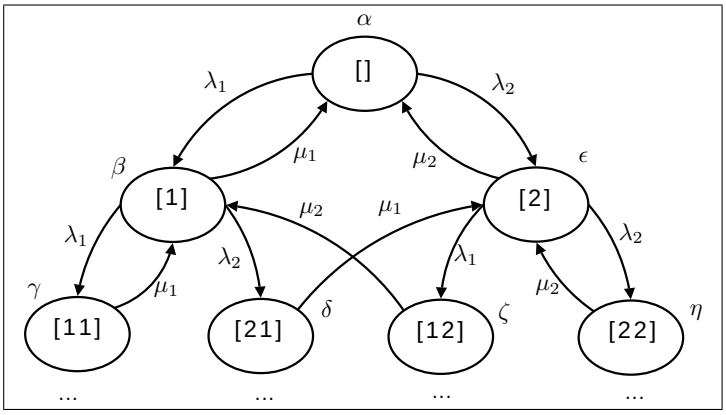

Figure 7.1: Part of the stochastic process of a 2 classes FCFS BCMP station.

traffic processes of class 1 and 2 job completions. Then we can write:

$$\Theta_1 = \{(\beta, \alpha), (\gamma, \beta), (\delta, \epsilon), \ldots\}$$
$$\Theta_2 = \{(\epsilon, \alpha), (\eta, \epsilon), (\zeta, \beta), \ldots\}$$

Let us consider state β, then we have that:

$$\Theta_1(\cdot, \beta) = \{\gamma\}$$
$$\Theta_2(\cdot, \beta) = \{\zeta\}$$

Intuitively, $m_1(\beta) = \pi(\gamma)\mu_1$ is the rate of the expected traffic count due to transitions into state β, while $m_1 = \sum_{\sigma \in \Gamma} m_1(\sigma)$ is the total rate of expected traffic count. Note that, since by hypothesis the queueing system admits a steady state, we have $m_i = \lambda_i$ for $i = 1, 2$.

Melamed's result [90] states that $K_i(t)$ are mutually independent Poisson processes if and only if the following equation holds:

$$\forall \gamma \in \Gamma, \quad \sum_{i=1}^{B} m_i(\gamma) = \pi(\gamma) \sum_{i=1}^{B} m_i, \tag{7.1}$$

that is proved to be equivalent to:

$$\forall \gamma \in \Gamma, \quad \forall i = 1, \ldots, B \quad m_i(\gamma) = \pi(\gamma) m_i \tag{7.2}$$

The similarity between Equation (7.2) and Munt'z Equation (2.16) is clearly visible by noting that, for a queueing center in equilibrium, we have that $m_i = \lambda_i$.

We now apply Melamed's results to our case. Let us consider the model GSPN-EXP with R classes of customers. We can define R traffic processes as follows:

$$\Theta_r = \{(\mathbf{m}', \mathbf{m}) | |\mathbf{m}'|_r = |\mathbf{m}|_r + 1\}, r = 1, \ldots, R, \tag{7.3}$$

where $|\mathbf{m}|_i = m_r + m_{R+r}$. Intuitively, Θ_r is the set of transitions that cause a reduction of the number of class r customers, i.e., a class r customer departure (as we know we deal with work-conserving disciplines).

In this case, in order to prove that $K_i(t)$ are independent Poisson processes, we have to prove that:

$$\forall \gamma \in \Gamma, \quad \sum_{\eta \in \Theta_i(\cdot, \gamma)} \pi(\eta) q(\eta, \gamma) = \lambda_i \pi(\gamma). \tag{7.4}$$

In the following section we prove that this condition holds for GSPN-EXP, GSPN-COX, GSPN-IS, GSPN-PS. As observed in [90] this property is equivalent to $M \Rightarrow M$ therefore it ensures that a BCMP-like composition of the introduced GSPN models has a closed-form steady state probability function.

7.2.1 Composing GSPN models by $M \Rightarrow M$

In this section we show that GSPN-EXP, GSPN-COX, GSPN-IS and GSPN-PS models satisfies the $M \Rightarrow M$ property in the sense specified in Section 7.2 and that they can be composed in a BCMP-like way maintaining a product-form solution. Moreover, they can be combined with other non-BCMP stations which fulfil the same property maintaining the product-form solution.

The following lemma plays a central role because it basically states that a network consisting of GSPN-EXP models or other GSPN models satisfying the $M \Rightarrow M$ property has a product-form solution. We state the lemma on GSPNs using the intuition of queueing theory, i.e, we consider R as the number of classes instead of the number of places, and similarly for the notion of customer, etc. This should improve the readability of the results.

Lemma 7 *Given a GSPN-EXP model with R classes, if the arrivals are class independent Poisson processes then the departures are class independent Poisson processes too.*

The proof of the lemma is based on verifying the property expressed by Equation (7.4) and it is illustrated in Section A.5.

Similarly, we can state a lemma on the $M \Rightarrow M$ property and GSPN-COX models. For GSPN-EXP model we showed the proof of $M \Rightarrow M$ with details and separated from the one that verifies the GBEs. However, the two proves can be fused into one, because the definition of the traffic process is straightforward and one can prove the GBEs with partial balances that correspond to the $M \Rightarrow M$ condition (7.4). This is the approach we use for GSPN-COX model. In fact, the proof of the following lemma is embedded in the proof of Lemma 5.

Lemma 8 *Given a GSPN-COX model with R classes, if the arrivals are class independent Poisson processes then the departures are class independent Poisson processes too.*

GSPN-IS and GSPN-PS models fulfil the $M \Rightarrow M$ property. The proof is trivial by Note 2. In fact, since the CTMCs of the GSPN-IS and GSPN-PS models are isomorphic to those of the corresponding BCMP stations, and recalling that the latter ones satisfy the $M \Rightarrow M$ property [94], we immediately conclude that $M \Rightarrow M$ holds also for GSPN-IS and GSPN-PS models.

7.2.2 Analysis of a hybrid model with an extended BCMP-like product-form

In this section we use the results illustrated in this chapter to analyze a hybrid QN/GSPN model in product-form using the $M \Rightarrow M$ property.

Model description We study a system that consists of a communication line with two channels, and three identical servers. The customers are clustered into three classes and two chains:

- Class A customers arrive from outside to the communication line *COM*. After being transmitted they are served by a set of three identical servers *SER* and then they leave the system.
- Class B customers arrive from outside to the communication line *COM*. After being transmitted they join class A.
- Class C customers are internal customers and their number is fixed. Their behavior is the following: they iterate a phase in which they perform a computation *DEL*, then they pass through the communication line *COM* and finally they are served by the servers *SER*. Once these passages are done, the customers return to the phase *DEL*.

The communication line follows the FCFS discipline, but cannot serve two customers of the same class simultaneously. Customers belonging to class A and B have the same stochastic behavior once they reach the servers *SER*. Class C customers are interior control processes and their number in our example is fixed: $K = 5$. Server *SER* queueing discipline is FCFS.

Note that a usual BCMP QN exact analyzer cannot analyze this system because of the presence of the multiple line communication channel. Moreover, in these tools it is usually not possible to define new queueing disciplines because of their high-level description.

Representing the model by GSPN In order to represent the model by a product-form GSPN we need the following further assumptions:

- The service time distributions of the channel *COM* and servers *SER* must be exponentially distributed and class independent.
- The service time distribution for the *DEL* phase of the internal processes can have any distribution with rational Laplace transform. In order to keep the example simple we assume (but it is not necessary) that it is exponentially distributed as well.
- The external arrival processes are independent Poisson processes with constant rates λ_A and λ_B for class A and class B customers respectively.

The behavior of the communication line is exactly modeled by a MSCCC station defined by Le Boudec in [86]. Although it is not a BCMP discipline type, Le Boudec proved that it satisfies the $M \Rightarrow M$ property, therefore it is in product-form when combined with other $M \Rightarrow M$ stations.

Therefore, we can represent an equivalent system with GSPN by using the results presented here as shown by Figure 7.2. The set of three servers with FCFS scheduling discipline can be represented by a GSPN-EXP service center, and finally the *DEL* phase of the internal processes are represented by a simple GSPN-IS model.

Model analysis As all the three blocks satisfy the $M \Rightarrow M$ property, we can state that the whole system, under stability assumption, has a product-form stationary probability distribution. The arrival rates of class A and class B customers to station *SER* is $e_{2A} = \lambda_A + \lambda_B$, the arrival rates to the stations that form the closed chain can be set to 1. Then, the steady state solution is given by the normalized product of the steady state solutions of the three GSPN sub-models considered in isolation.

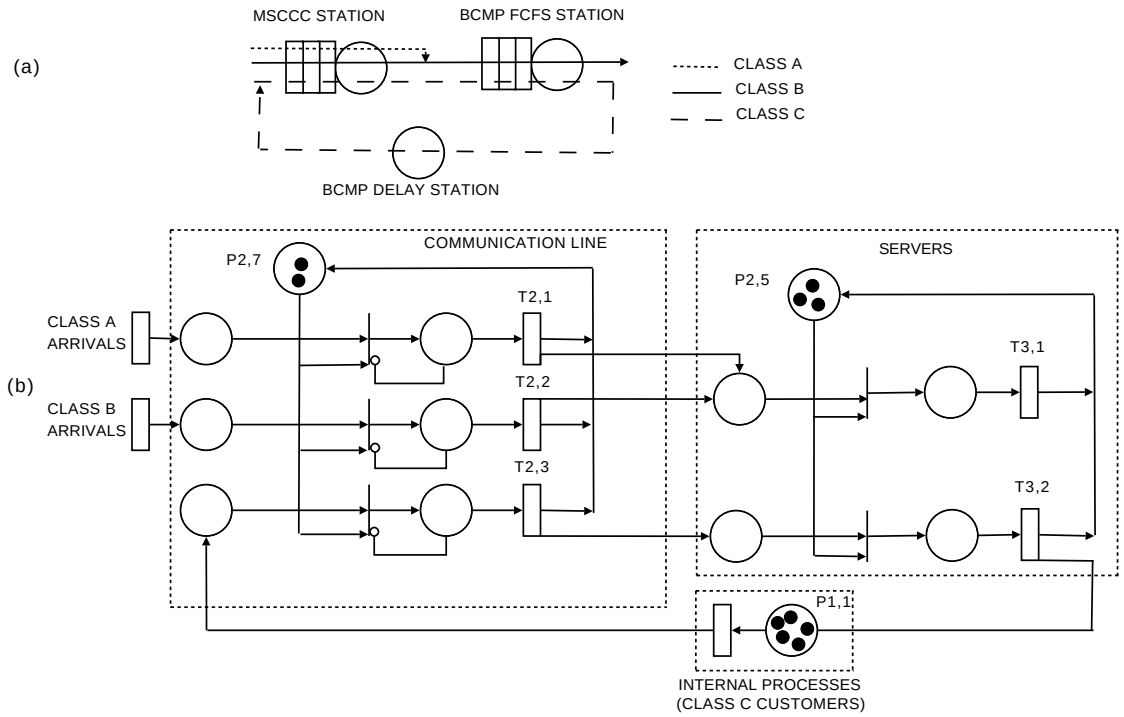

Figure 7.2: Example of product-form GSPN obtained by hybrid modeling.

7.3 On the characterization of probabilistic queueing disciplines with a single server

In this section we study model GSPN-EXP under the restriction of a single server and a general weight function for the immediate transitions, i.e., Definition 7 of Section 6.2.1 changes to:

- The initial state. Instead of $\mathbf{m}_0 = (0, \ldots, 0, K)$ we consider the initial state $\mathbf{m}_0 = (0, \ldots, 0, 1)$.

- The weight of immediate transitions. Instead of $w(t_r, \mathbf{m}) = m_r$ $(1 \leq r \leq R)$ we allow more general weights: $w(t_r, \mathbf{m}) = w_r(\mathbf{m})$.

Let us name this model GSPN-EXP-1. Note that model GSPN-EXP with a single server has a CTMC which is isomorphic to the CTMC of a random queueing discipline with multiple classes of customers and it is known to satisfy the $M \Rightarrow M$ property if the probability of getting a free server is uniform among the customers in queue [1]. Are there other possible choices, but the uniform, that lead to a $M \Rightarrow M$ product-form? In this section we prove that the only possible distribution that gives

a $M \Rightarrow M$ product-form is the uniform, i.e.,

$$\Pr\{t_r \text{ fires}\} = \frac{m_r}{\sum_{i=1}^{R} m_i} \iff M \Rightarrow M \text{ property holds} \tag{7.5}$$

where $r = 1, \ldots, R$. Lemma 7 proves the $\Longrightarrow$ verse of the implication, in this section we prove the opposite verse.

Although we are stating a negative result, we think that it is interesting from a theoretical point of view. In fact, informally speaking, it binds the $M \Rightarrow M$ property for random queueing disciplines to an idea of fairness of accessing the server among the customers.

The importance of this result does not concern only with GSPN models but also with the QN models and specifically with the definition of probabilistic queueing disciplines that exhibit BCMP-like product-form solutions. In other words, we study the conditions under which a multiclass station with a probabilistic queueing discipline without preemption and with an exponential server has a $M \Rightarrow M$ based product-form solution. Is it possible to define a random choice between the classes such that the probability of choosing a class r customer is proportional to the class arrival rate? Or is proportional with the square of the number of customers of the same class in the queue? By the result we are going to present, the answer to all these questions is negative.

The proof is rather technical so we introduce a simplification of the notation in order to make it more readable. We assume $w_s(\mathbf{m}) = 0$ if (and only if) $m_s = 0$, and we denote by $\mathbf{m}^{(r)}$ a state $\mathbf{m}$ with $m_{R+r} = 1$, i.e., with a class r customer in service, and by $\mathbf{m}_0$ the state with $m_{2R+1} = 1$, that is no customer is at the station. We aim to prove that the only choice for function w_r which leads to a $M \Rightarrow M$ product-form is the uniform one.

In this section we state the main theorem and we prove it in Appendix A, Section A.6.

Theorem 7 *Le us consider a GSPN-EXP-1 model. Let* $\mathbf{m}$ *be a generic tangible state of model GSPN-EXP-1,* $A = \{r : m_r > 0, 1 \leq r \leq R\}$. $M \Rightarrow M$ *property holds if and only if the following condition holds:*

- *For every* A *such that* $|A| \geq 2$, $w_r(\mathbf{m}) = f(\mathbf{m})m_r$ *for* $r \in A$, *with* f *an arbitrary positive function.*

In this case the stationary distribution is given by Formula 6.2 with $K = 1$.

Note that the theorem formally states what we have introduced before. Given a tangible marking $\mathbf{m}$, set A is the set of the immediate transitions that are enabled after a job completion (i.e., after the firing of a timed transition). If $|A| > 1$ then there is a conflict among the immediate transitions. Recall that, in this case, the probability of firing of a transition is proportional to its weight. The theorem states that the transition weights must be in the form $w_r(\mathbf{m}) = f(\mathbf{m})m_r$. Note that function f only depends on the current state and not on the transition. Therefore, it can be set $f(\mathbf{m}) = 1$ without loss of generality for obvious algebraic reasons.

7.4 RCAT composition

In Section 7.2 we have shown how we can interpret the $M \Rightarrow M$ property in the GSPN formalism context. This leads to the definition of a class of GSPN models that can be combined in a QN-like manner such that the solution of the whole model is in product-form. In this section we apply a similar approach using RCAT. The section is organized as follows. At first, we interpret RCAT theorem in the GSPN formalism context with special attention for models GSPN-EXP and GSPN-COX. Then we prove that under a set of structural conditions these models satisfy RCAT conditions. This means that the models can be composed with other models satisfying RCAT conditions including product-form G-Networks [50] or Coleman, Henderson et al. SPNs in product-form [63, 39] with the restrictions presented in Chapter 5. The last part of this section illustrates some examples of applications of this result.

7.4.1 A comparison between $M \Rightarrow M$ and RCAT conditions

In this section we compare the condition on the reversed transition rates of RCAT (Theorem 3) and the condition for the $M \Rightarrow M$ property (2.16). For the sake of clarity we reason for a composition of queueing systems, although the conclusions can be extended to the general case. Let us consider a multiclass queueing system without blocking, i.e., it can always accept customer arrivals, and let us suppose that each state of the queueing system can be reached by a customer departure transition of any served class. Le ξ be a generic state of the queue and, following Munt'z notation, let $\mathcal{S}_r^+(\xi)$ be the set of states with one customer of class r more than state ξ from which ξ is reachable. We are sure, by hypothesis, that $\mathcal{S}_r^+(\xi)$ is not empty. Each transition from a state $\xi' \in \mathcal{S}_r^+(\xi)$ to ξ corresponds to a class r customer departure. From RCAT point of view this means that:

$$x_r = \frac{\pi(\xi')}{\pi(\xi)} q(\xi', \xi)$$

where x_r is the reversed rate. x_r must be independent of $\xi' \in \mathcal{S}_r^+$ and ξ. The condition required by $M \Rightarrow M$ is different because it states that the processes corresponding to the customer departures must be class independent Poisson processes. Algebraically this is expressed by Equation (2.16). Note that, in the reversed process, this equation means that the sum of the reversed rates $q^R(\xi, \xi')$, $\xi' \in S_r^+(\xi)$ must be equal to the arrival rate of class r customer to state ξ.

It is worthwhile pointing out when RCAT condition and $M \Rightarrow M$ conditions coincide. We introduce the following observation:

Observation 1 *A multiclass queueing model that satisfies the following conditions:*

- *Constant arrival rates for each class* λ_r,

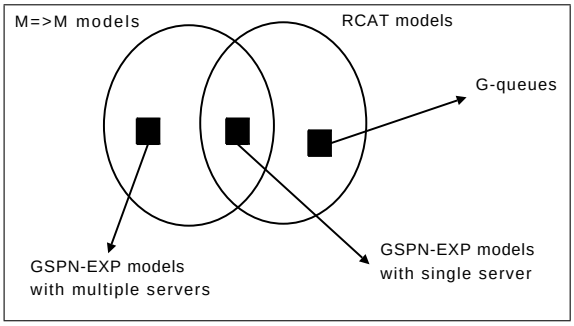

Figure 7.3: Relation between $M \Rightarrow M$ models and RCAT models.

- *For every state* ξ *we have that* $|\mathcal{S}_r^+| = 1$,
- *For every state* ξ *there exists a state* ξ'' *reachable due to a class* r *customer arrival,*
- $M \Rightarrow M$ *property is satisfied for this model,*

also satisfies the conditions of RCAT theorem.

Note that by the considerations formulated in this section we cannot say that one of RCAT or $M \Rightarrow M$ is more general than the other. $M \Rightarrow M$ can easily deal with variable arrival rates, but it requires the sum of the reversed rates from any state ξ to $\xi' \in \mathcal{S}_r^+(\xi)$ to be exactly equal to the arrival rate of class r to ξ. On the other hand RCAT just asks the reversed rate of every transition from ξ to $\xi' \in \mathcal{S}_r^+(\xi)$ to be constant (so not related to the arrival rate). In the following we show that models GSPN-EXP and GSPN-COX can be used in an RCAT composition under some restrictions. Moreover, recall that there are several models that do not fulfil $M \Rightarrow M$ but satisfy RCAT conditions, e.g. the G-queue. Therefore, we have shown examples of models that satisfy $M \Rightarrow M$ but not RCAT conditions, models that satisfy both of them, and models that satisfy RCAT conditions but not $M \Rightarrow M$. This situation is depicted by Figure 7.3.

7.4.2 RCAT for GSPN models

In this part of the section we give an interpretation of RCAT conditions for what concerns models GSPN-COX and GSPN-EXP. The approach can be extended to a general GSPN model although special attention should be devoted to represent transitions with multiple synchronization. In Chapter 5 we have shown a possible technique to deal with this case. Roughly speaking we can interpret the three RCAT conditions on a queueing station as follows:

1. There is no blocking for the customer arrivals in any state of the station. This means that the customer can be destroyed, served, or put in queue, but the process which generated the arrival cannot be blocked.

2. For each customer class r and reachable state ξ, there is at least another reachable state ξ' such that ξ is reachable from ξ' by a class r job completion.

3. The reversed rates of the transitions corresponding to a job completion are constant for each customer class.

In order to apply RCAT to a GSPN model we need to give an interpretation of the model using PEPA. A GSPN with immediate transitions generates a semi-Markov process that can be reduced to a CTMC. Let us consider the CTMC generated by a GSPN model such as GSPN-EXP. In a SPN model every state transition in the CTMC corresponds to the firing of a timed transition and the state transition rate in the CTMC is equal to the transition rate of the SPN. This is not straightforwardly true in a GSPN model. In fact, even if it is still true that a state transition in the CTMC corresponds to the firing of a timed transition in the GSPN, the presence of immediate transitions can change the rates.

Example 10 *Let us consider a model GSPN-EXP with $R = 2$, rate μ and $\mathbf{m}_0 = (0, 0, 0, 0, 1)$, i.e., there is just a single server. We consider state $(m_1, m_2, 0, 1, 0)$ with $m_1, m_2 > 0$. A job completion event corresponds to the firing of transition T_4 and transition T_4 has rate $m_4\mu = \mu$. However the state transitions in the CTMC due to the firing of T_4 have different rates. Figure 7.4 illustrates the reason by showing the passage from the semi-Markov process to the CTMC. Note that this example is simple because the the immediate transitions can be easily reduced by embedding their probabilistic behaviors in the timed transition firing rates.*

In order to study how to apply RCAT to a GSPN, we need to model the underlying CTMC of the GSPN in PEPA and then we can interpret RCAT conditions. As any CTMC can be described by PEPA using just the prefix operator and the parallel composition as noted in [68] we can for sure describe in PEPA the CTMC of a GSPN model in isolation. Each transition in the CTMC of the GSPN model correspond to a PEPA activity. The activity rate is the state transition rate, and the activity type is a type corresponding to the transition in the GSPN:

Example 11 *Let us consider again the model of Example 10. Suppose that $\mathbf{m} = (m_1, m_2, 0, 1, 0)$, $\mathbf{m}' = (m_1 - 1, m_2, 1, 0, 0)$ and $m'' = (m_1, m_2 - 1, 0, 1, 0)$. Then the PEPA description of the transitions depicted in Figure 7.4 is:*

$$\mathbf{m} = (\frac{m_1}{m_1 + m_2}\mu, T_4).\mathbf{m}' + (\frac{m_2}{m_2 + m_1}\mu, T_4).\mathbf{m}'.$$

Note that, in general, the fusion of timed and immediate transitions is not trivial. In particular, cycles of immediate transitions can be difficult to analyze [89, 88, 74], however it is out of the scope of this thesis reviewing the techniques to solve this problem, and we limit our analysis to the models GSPN-x.

In the following we refer to the reversed rate of a transition T of the GSPN without deriving the CTMC and the PEPA description of the model as it is a purely formal operation.

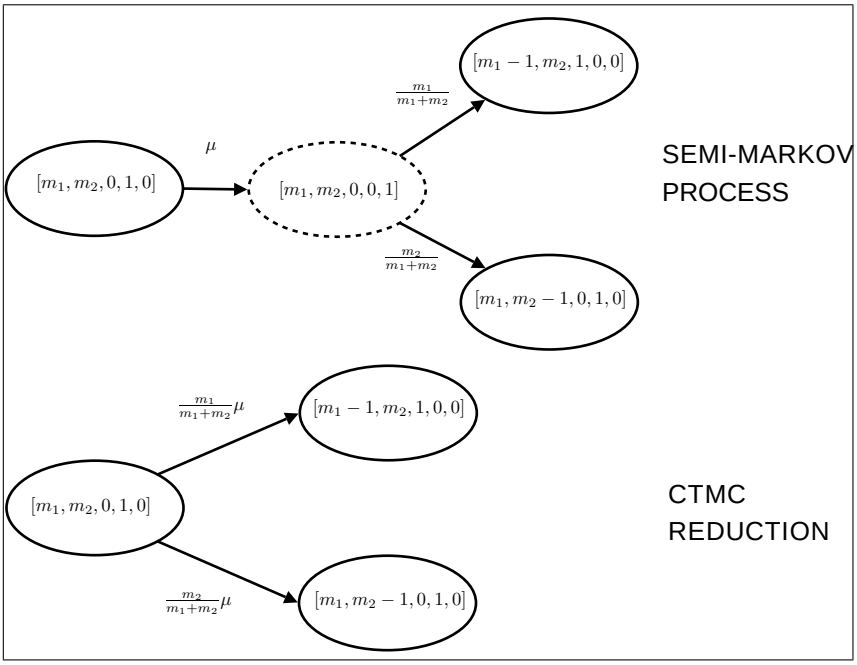

Figure 7.4: State transitions in the semi-Markov process and in the corresponding CTMC for GSPN-EXP model of Example 10. Dotted line is used for vanishing states.

Now, if we consider any of the GSPN-x models defined in Chapter 6, it is easy to informally verify that conditions 1 and 2 are satisfied. Condition 3 has to be checked algebraically.

7.4.3 Theoretical results

In this section we show that in the reversed CTMC underlying the models GSPN-EXP and GSPN-COX the transitions associated with a class r customer departure in the forward process have constant rates in the reversed process under the assumption of load-independent single server. This result, combined with RCAT [59] illustrated in Chapter 4, allows one to analyze GSPN models with a compositional approach and then derive new product-form solutions, i.e., closed form stationary probability functions which are not identified by previous product-form model classes.

Analysis of GSPN-EXP model The analysis of GSPN-EXP model is relatively simple. The only condition required to have constant rate reversed departure transitions is that there must be a single server. Formally we introduce the following Lemma.

Lemma 9 *Given model GSPN-EXP with a single server, i.e.,* $m_{2R+1} = 1$ *in the initial state, then:*

1. *The reversed rates of the CTMC transitions corresponding to customer departures have constant rates.*

2. *Let* $\mathbf{m}$ *be an arbitrary state, then there is always a possible transition to a state* $\mathbf{m}'$ *due to a class* r *customer arrival, for any* $r = 1, \ldots, R$.

3. *Every state of the CTMC generated by GSPN-EXP model is reachable due to a class* r *customer departure transition, for any* $r = 1, \ldots, R$.

Proof. First of all we identify the transitions corresponding to a customer departure. They clearly are the transitions which reduce the total number of tokens in $P_1, \ldots, P_{2R}$.

Let us prove the first proposition of the lemma. We have to distinguish two cases. Suppose that there is at least one token in one of the places $P_1, \ldots, P_R$ (intuitively at least one customer in the queue). The firing of a transition $T_r \in \{T_1, \ldots, T_R\}$ causes a transition from state $\mathbf{m}$ to state $\mathbf{m}'$. Clearly $m'_r + m'_{R+r} = m_r + m_{R+r} - 1$ that we can interpret as a departure of a customer of class r, $1 \leq r \leq R$. In particular, we can write $\mathbf{m}' = \mathbf{m} - \mathbf{e}_{R+r} + \mathbf{e}_{R+s} - \mathbf{e}_s$, where s denotes the class of the customer that obtains the server. The forward rate of this transition is $\mu m_s / \sum_{j=1}^{R} m_i$. Then we can calculate the reversed rate using the steady state probability distribution given by formula (6.2):

$$q'(\mathbf{m}', \mathbf{m}) = \frac{\pi(\mathbf{m})}{\pi(\mathbf{m}')} q(\mathbf{m}, \mathbf{m}') = \frac{\sum_{i=1}^{R} m_i}{m_s} \frac{m_s}{m_r} \lambda_r \frac{1}{\mu} \mu \frac{m_r}{\sum_{i=1}^{R} m_i} = \lambda_r.$$

Let us consider the case of $m_s = 0$ for all $s = 1, \ldots, R$, intuitively, just a customer in the station. In this case the only possible transition associated with a customer departure is from $\mathbf{m} = \mathbf{e}_{R+r}$ to $\mathbf{m}' = \mathbf{e}_{2R+1}$ with rate μ. The computation of the reversed rate straightforwardly gives λ_r. Therefore, the reversed rate for class r customer departure is constant.

The second and third parts of the lemma are trivial. Second part: given an arbitrary state $\mathbf{m}$ there can always be an arrival of a customer that enters in the queue. Third part: given an arbitrary state $\mathbf{m}$ we can identify a state $\mathbf{m}''$ such that there is a customer of class r in service, for any $r = 1, \ldots, R$, and all the customers in $\mathbf{m}$ are in the queue. The departure transition of the class r customer can lead the system to state $\mathbf{m}$. ♠.

Note that for single class FCFS queueing stations it is a well-known result that the reversed rates of the departure transitions are constant and equal to the arrival rates (usually denoted by λ). However, here we deal with multiclass stations and the queueing discipline of GSPN-EXP is *not* FCFS but random.

Note that we can look at this result from the point of view of Observation 1. GSPN-EXP models with a single server are such that given a tangible state $\xi = \mathbf{m}$, the set $\mathcal{S}_r^+(\xi)$ has cardinality 1. A counterexample can be shown of a GSPN-EXP model with multiple servers not satisfying RCAT conditions.

Analysis of GSPN-COX model The case of Coxian service time distribution requires more attention because in the original definition of the Coxian service time a customer can leave the system from any service stage. We show that a sufficient condition to have constant reversed rates of the customer departure transitions is that a customer can leave the system only from the last stage of service, i.e., we set $a_{r,\ell} = 1$ for $\ell = 1, \ldots, L_r - 1$ and $b_{r,L_r} = 1$. Note that this is a strong restriction because the service time distribution of the station becomes a generalized Erlang distribution.

Lemma 10 *Given model GSPN-COX with a single server, i.e., $m_{R+1} = 1$ in the initial state, then*

1. *the reversed rates of the CTMC transitions corresponding to the customer departures have constant rates if $a_{r,\ell} = 1$ for $\ell = 1, \ldots, L_r - 1$ and $a_{r,L_r} = 0$.*

2. *Let $\mathbf{m}$ be an arbitrary state, then there is always a possible transition to a state $\mathbf{m}'$ due to a class r customer arrival, for any $r = 1, \ldots, R$.*

3. *Every state of the CTMC generated by GSPN-COX model is reachable due to a class r customer departure transition, for any $r = 1, \ldots, R$.*

Proof. We just sketch the proof of the first part, and explain the reason of the conditions on $a_{r,\ell}$ parameters. The second and third propositions are proven as for Lemma 9. Let us consider a GSPN-COX model with a single server and $0 < a_{r,\ell} \leq 1$ for $r = 1, \ldots, R$ and $\ell = 1, \ldots, L_r - 1$. Consider the case of $m_{R+r,\ell} > 0$ for some r and ℓ, intuitively there is at least a customer preempted (in queue). By the steady state distribution, we can derive the reversed transition rate $\mathbf{m}' \to \mathbf{m}$ assuming that $m_{r,\ell} = 1$, similarly to what we have done in the previous proof, we obtain:

$$q'(\mathbf{m}', \mathbf{m}) = \lambda_r A_{r,\ell} b_{r,\ell}.$$

Note that $q'(\mathbf{m}', \mathbf{m})$ depends on $A_{r,\ell} b_{r,\ell}$. This explains the conditions required by the Lemma on the values of $a_{r,\ell}$. In fact, if the $a_{r,\ell} = 1$ for $\ell = 1, \ldots, L_r - 1$ then the departures are possible only for $\ell = L_r$, thus we obtain: $q'(\mathbf{m}', \mathbf{m}) = \lambda_r$, because $A_{r,L_r} = 1$ and $b_{r,L_r} = (1 - a_{r,L_r}) = 1$. Finally, we observe that the value of the expression $\lambda_r A_{r,\ell} b_{r,\ell}$ can be state independent even for some values of $a_{r,\ell}$ and $b_{r,\ell}$ of a Coxian distribution. However we have preferred to state a stricter but just structural product-form condition. ♠

Roughly speaking, Lemma 9 and Lemma 10 state that GSPN-EXP and GSPN-COX model can either be fed by or feed other RCAT-compatible blocks under the specified restrictions.

7.4.4 New product-form GSPN models with RCAT

This part of the chapter illustrates how it is possible to generate new product-form GSPN models using RCAT. Our approach may seem a little peculiar. In fact, we start from a model that it is *not* in product-form and we try to modify it in order to allow the application of RCAT. In this way the resulting model can be seen as a product-form approximation of the original model. By this technique several product-form models can be generated in a relatively simple way. It is out of the purpose of this thesis the analysis of how good such approximation can be, however we think that an analysis in this direction could give interesting results.

It is worthwhile pointing out that this approach is possible because RCAT proves the product-form of a model without analyzing the global balance equations, therefore we have a high-level point of view of the product-form.

Let us consider a system with N servers. When a customer arrives to the system from the outside (according to a Poisson process) it immediately enters in service if there is an available server, and after an exponentially distributed service time it leaves the system. However, differently from normal queueing systems, the server does not become immediately available after the job completion but, it can be used again after an exponentially distributed random delay. If a customer arrives to the system and there are not free servers, then it waits in queue with a RANDOM discipline. We use the SPN model of Figure 7.5 to represent this system.

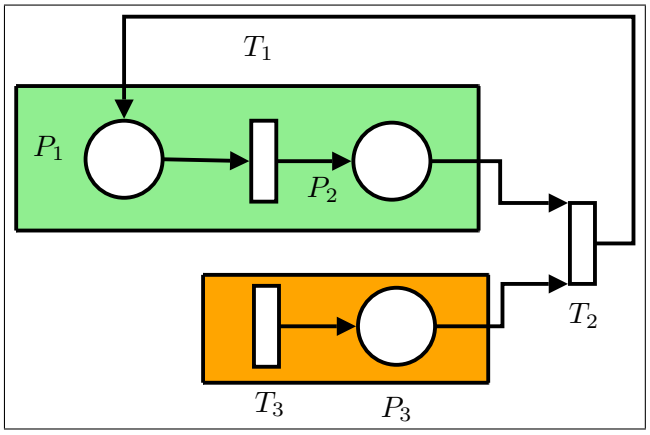

Figure 7.5: Example of SPN not in product-form.

In the model of Figure 7.5 (let us call it IDEAL) transition T_3 has rate $\mu_3 = \lambda$ and represents the external arrivals, transition T_2 has rate μ_2 and represents the service rate. The firing of T_2 frees a server, however the corresponding token is placed in P_1. The firing of transition T_1 (with rate μ_1) makes the server available again.

SPN model IDEAL is not in product-form by Theorem 2 in fact it is not true that for every input vector there is a corresponding output vector (for example there is not a transition whose output vector is the null vector). Moreover, model IDEAL is not in Boucherie product-form.

We can see the model consisting of two cooperating processes. One process represents the availability of the server, and the other process represents the number of customers in the station. Even if the application of RCAT we are going to do can be performed quickly, we show it step by step. Let us formulate the PEPA description of the two processes. P_i is an agent that represents the availability of i servers. Therefore, we obtain:

$$\begin{aligned} P_0 &\stackrel{def}{=} \mu_1.P_1 \\ P_i &\stackrel{def}{=} \mu_1.P_{i+1} + (t_2, \mu_2).P_{i-1} \quad \text{for } i = 1, \ldots N-1 \\ P_N &\stackrel{def}{=} (t_2, \mu_2).P_{N-1}, \end{aligned}$$

where N is the total number of servers. Let us consider the PEPA definition Q_j of the other process:

$$\begin{aligned} Q_0 &\stackrel{def}{=} \lambda.Q_1 \\ Q_j &\stackrel{def}{=} \lambda.Q_{j+1} + (t_2, \top).Q_{j-1} \quad j > 0 \end{aligned}$$

Figure 7.6 shows a graphical representation of P_i and Q_j. RCAT cannot be applied

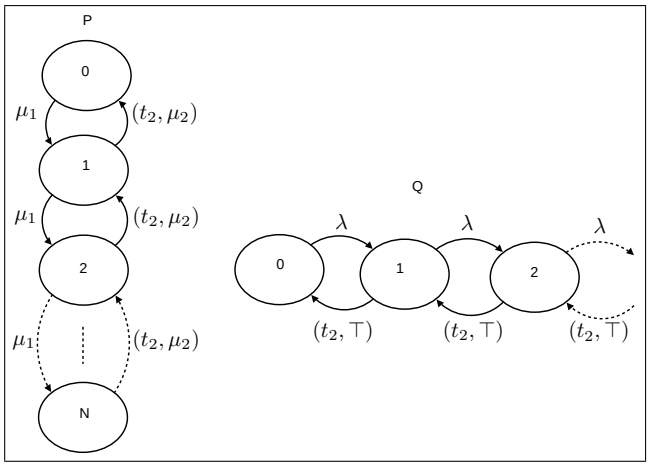

Figure 7.6: Representation of IDEAL model by cooperating processes.

because:

- In the reversed process of P there is not any outgoing arc of active type t_2 from state P_N.
- In the forward process of Q there is not any outgoing arc of passive type t_2 from state Q_0.

In order study model IDEAL we need to modify the structure of the processes obtaining model SIMPLE. Of course the resulting model and analysis are different from the original ones, but we are going to show that there are some minor changes. We call P' the modified model of P, Q' the modified model of Q (see Figure 7.7):

$$\begin{aligned}
P'_0 &\stackrel{def}{=} \mu_1.P'_1 \\
P'_i &\stackrel{def}{=} \mu_1.P'_{i+1} + (t_2,\mu_2).P'_{i-1} \quad \text{for } i = 1,\dots N-1 \\
P'_N &\stackrel{def}{=} (t_2,\mu_2).P'_{N-1} + (t_2,\mu_1).P'_N \\
Q'_0 &\stackrel{def}{=} \lambda.Q'_1 + (t_2,\top).Q'_0 \\
Q'_j &\stackrel{def}{=} \lambda.Q'_{j+1} + (t_2,\top).Q'_{j-1} \quad j > 0
\end{aligned}$$

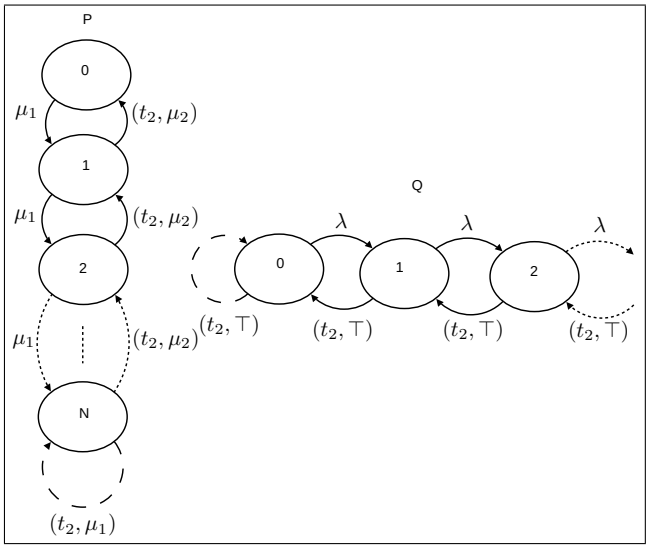

Figure 7.7: Representation of model SIMPLE by cooperating processes.

As we can see the self-loop on P_N has rate μ_1 and not μ_2 as one could expect. This is because we set the rate such that the reversed rate of the actions of type t_2 in P is constant. In our case this reversed rates is μ_1 and the reversed rate of a selfloop is the rate in the forward process. Therefore, we have two birth and death processes in product-form and the steady state solution is given by:

$$\pi(i,j) \propto \left(\frac{\mu_1}{\mu_2}\right)^i \left(\frac{\lambda}{\mu_1}\right)^j.$$

However, our analysis cannot be considered finished. In fact, it is worthwhile understanding how we have modified model IDEAL in terms of SPN representation and system behavior in order to obtain model SIMPLE. Indeed, we have introduced the following undesired behavior:

- if all the servers are available there can be a customer that is served and the resource becomes immediately available by going straightforwardly to place P_2 without passing through P_1. The service rate is μ_1 instead of μ_2;
- if there are not any waiting customers then an available server can become unavailable. From the system point of view, this can be interpreted as a sort of energy-save politic.

Note that the unwanted behavior happens only when all the server are available or when there is not any customer in the center. So a limited part of the joint process of the model results modified.

A GSPN model corresponding to model SIMPLE is depicted in Figure 7.8. T_4 and T_5 are the new transitions and we use inhibitor arcs.

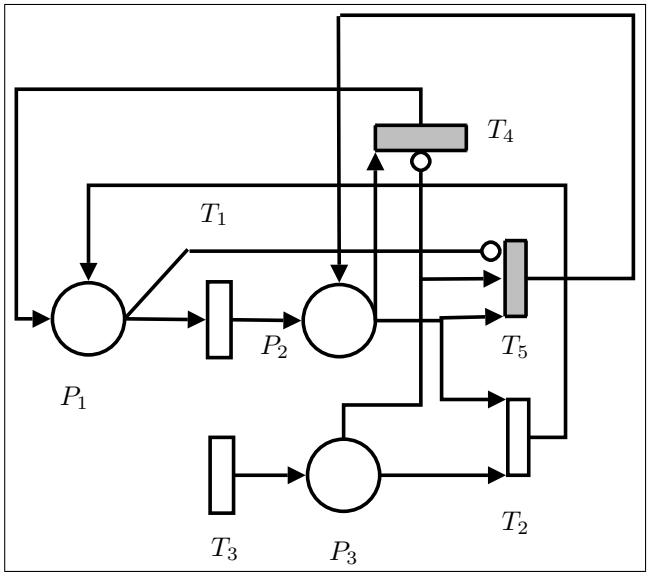

Figure 7.8: Example of SPN in product-form. Model SIMPLE.

Note that the RCAT analysis is still possible if we set the rate of T_2 to $m_2\mu_2$. In this case all the available servers work even if there are less customers than servers. Note that the reversed rate of the actions labeled by t_2 is still state independent and equal to μ_1. The steady state solution becomes:

$$\pi(i,j) \propto \left(\frac{\mu_1}{\mu_2}\right)^i \frac{1}{i!} \left(\frac{\lambda}{\mu_1}\right)^j$$

If we want to combine model SIMPLE with other sub-models, we can assume T_3 to be a transition whose firing is governed by another sub-model. This can be done because transition T_3 of model SIMPLE is always enabled, therefore it can be seen as a passive action in the corresponding process. This means that model SIMPLE can receive tokens from other models that satisfies RCAT. The state transitions that are associated with a customer departure are those corresponding to the firing of transitions T_5 and T_2. Let us consider a generic state $\mathbf{m} = (m_1, m_2, m_3)$ with $m_3 > 0$ and $m_1 \neq 0$. The reversed rate for a job completion $\overline{\mu_2}$ can be obtained as:

$$\pi(\mathbf{m})\mu_2 = \pi(\mathbf{m} - \mathbf{e}_3 - \mathbf{e}_2 + \mathbf{e}_1)\overline{\mu_2},$$

where, as always $\mathbf{e}_i$ is a vector whose component are all 0 but the i-th that is 1. Then, we can write:

$$\overline{\mu_2} = \frac{\mu_1}{\mu_2}\frac{\lambda}{\mu_1}\mu_2 = \lambda.$$

If we consider state $\mathbf{m} = (0, m_2, m_3)$ then transition T_5 is enabled. Transition T_5 changes the state from $\mathbf{m}$ to $\mathbf{m} - \mathbf{e}_3$ with rate μ_1. In a similar way we obtain $\overline{\mu_5}$:

$$\overline{\mu_5} = \frac{\lambda}{\mu_1}\mu_1 = \lambda.$$

Note that the reversed rates are independent of state $\mathbf{m}$ and $\overline{\mu_5} = \overline{\mu_2} = \lambda$ hence the model can feed other RCAT models as shown by the following example.

Example 12 (New product-form GSPNs) *Let us consider the GSPN of Figure 7.9. The model consists of a composition of model SIMPLE and a simple place. From a high level point of view we can see the model as depicted by Figure 7.10.*

Sub-model SIMPLE is in product-form by RCAT if the following conditions on the transition rates hold:

$$\begin{cases} \chi_6 = \chi_2 \\ \chi_5 = \chi_1 \end{cases},$$

while the other sub-model is unconditionally in product-form. In Section B.2.1 we derive the product-form solution and verify it by substitution in the GBEs.

7.5 A first example of hybrid modeling

In this section we apply the previous theoretical results to study a simple system consisting of a FCFS multiclass service center and a communication channel obtaining an example of a hybrid model with GSPN and QN sub-models.

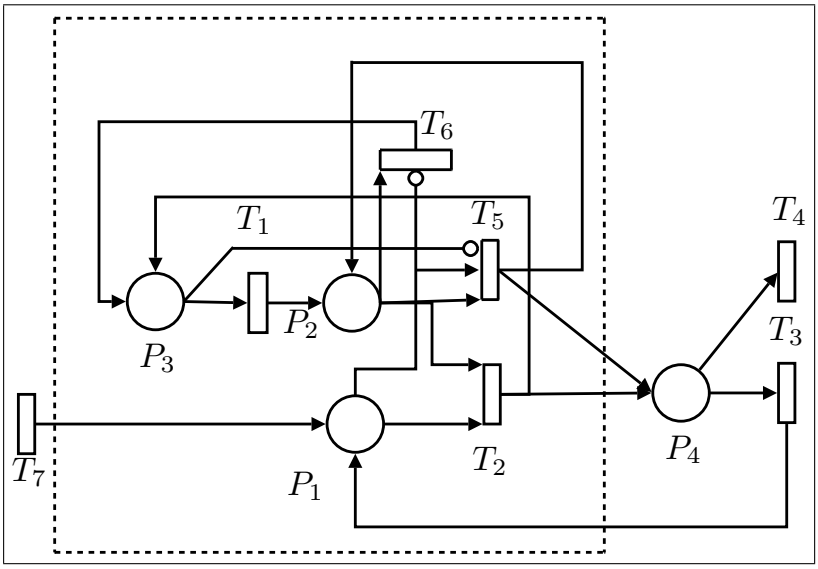

Figure 7.9: Example of GSPN in product-form under some conditions on the transition rates.

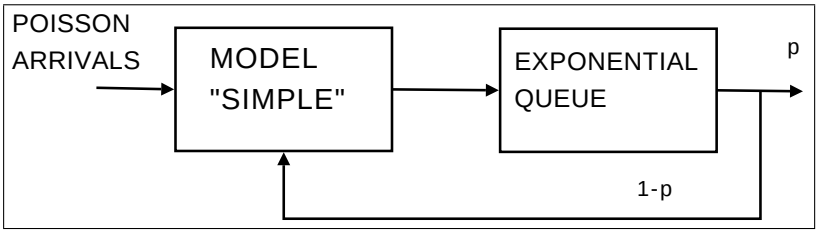

Figure 7.10: High level interpretation of the model of Figure 7.9.

System description The system consists of two classes of customers. There is a fixed number K_1 of class 1 customers. They alternate a thinking phase (TP) and a service phase (SP). In the SP they require a service from a server S. Class 2 customers arrive from the outside and they ask a service to S. Before leaving the system they have to pass through a communication channel. The channel transmission protocol consists of 3 phases and only one customer can occupy a phase at a given time. Therefore, at most 3 customers can stay simultaneously in the channel. Customers pass from a phase to the next one simultaneously. If there is a customer arrival when the first protocol phase is busy then a collision occurs and the first phase becomes available but both the customers (the one which generated the collision and the one occupying the first phase) are lost. We can say that the first protocol phase is the *vulnerable* phase. Figure 7.12 illustrates a high level representation of the system.

Assumptions In order to obtain an exact analytical solution we have to introduce some assumptions. The service time of S must be exponentially distributed and class independent with rate μ. Class 2 arrivals to the system occur according to a

Poisson process with parameter λ. For what concerns the communication channel we assume that the protocol phase transitions occur every exponentially distributed random time with rate μ_p. The thinking time for class 1 customers can be very general, e.g. Coxian distributed. For the sake of simplicity we assume hereafter that it is exponentially distributed with rate μ_t.

GSPN representation Server S representation is given by a GSPN-EXP model with $R = 2$. The communication channel is depicted in Figure 7.11. The tokens representing the customers arrive to place P_1. P_2, P_3, P_4 represent phases 1, 2, 3 of the protocol, respectively. Immediate transition t_1 models the collision in the first protocol phase. Immediate transition t_2 models the entering of a customer in the channel in phase 1. The timed transitions, that are labelled with the number of the input places to enhance the readability of the picture, model the customers movements across the protocols phases. For every timed transition the firing rate is $w(T) = \mu_p$ and for every immediate transition $w(t) = 1$ (as there are not conflicts between t_1 and t_2). The transitions filled in grey are the ones whose firings correspond to a service completion by the channel. The TP for class 1 customers is modelled by a couple place/transition in standard way of state machines (see [74] for example).

Steady state analysis In this paragraph we obtain the steady state solution for our model under stability assumptions. We have studied the GSPN-EXP under Poisson arrivals. We proved that the system fulfils the $M \Rightarrow M$ property therefore the composition with feedback with the block representing the class 1 customers thinking phase is in product-form. Note that because of the feedback the arrival process to server S is *not* a Poisson process, but the product-form still holds [94].

Let us consider the communication channel. Figure 7.13 shows the underlying process. A state is described by a sequence of three bits. The one on the left denotes if the first phase of the protocol is busy (1) or free (0) and similarly the other ones for phases 2 and 3. Symbol $\top$ denotes a transition caused by a class 2 customer arrival from the server S block. In the following we keep the convention of denoting by $\mathbf{m}$ the state of the whole system. The state of block i and its steady state probability distribution is denoted by $\mathbf{m}^{(i)} = (m_1^{(i)}, m_2^{(i)}, \ldots)$ and $\pi^{(i)}$ respectively, where the subscripts of the vector components correspond to the labels of the places.

In order to obtain the steady state solution we can apply RCAT because the reversed rate of a class 2 customer departure transitions from the server is constant and we proved in Section 7.4.3 that it is equal to λ. Moreover, in every state of the channel a customer arrival is allowed. Therefore, we can replace in the process of Figure 7.13 the $\top$ symbol by the value λ obtaining in few steps:

$$\pi^{(2)}(m_2^{(2)}, m_3^{(2)}, m_4^{(2)}) \propto \left(\frac{\lambda}{\lambda + \mu_p}\right)^{\sum_{k=2}^{4} m_k^{(2)}} . \tag{7.6}$$

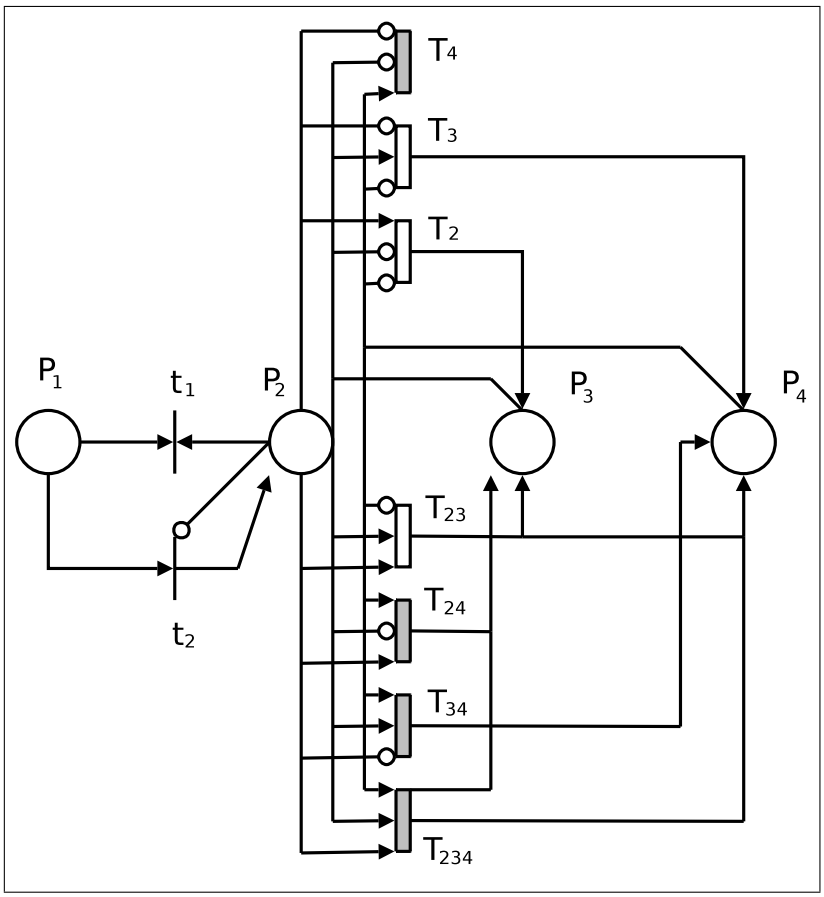

Figure 7.11: GSPN model of a communication channel with a 3 phases protocol. Phase 1 is vulnerable to collisions

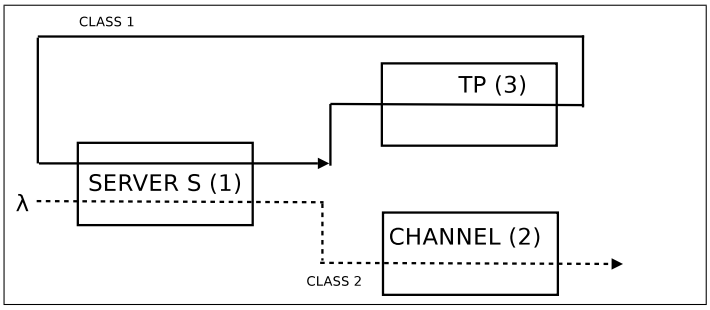

Figure 7.12: Scheme of the interacting GSPN blocks for the model described in Section 7.5

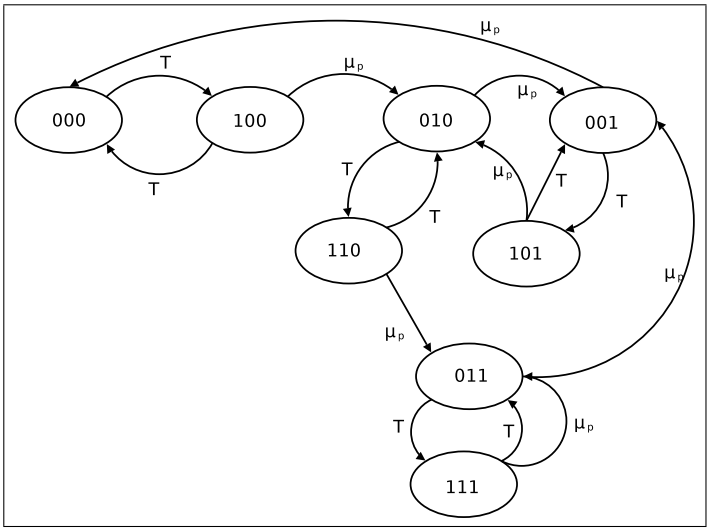

Figure 7.13: Description of the stochastic process associated with the communication channel.

The steady state solution for block 1, the server, is given by Formula (6.2) with $R = 2$, $\lambda_1 = \lambda$ and $\lambda_2 = 1$, that is the relative visit ratio between blocks 1 and 3. Hence we obtain:

$$\pi^{(1)}(\mathbf{m}^{(1)}) \propto \lambda_1^{m_1^{(1)}+m_3^{(1)}} \cdot \frac{(m_1^{(1)}+m_2^{(1)})!}{m_1^{(1)}!m_2^{(1)}!} \cdot \left(\frac{1}{\mu}\right)^{\sum_{i=1}^{4} m_i^{(1)}}.$$

Block 3 behaves like a single class $M/M/\infty$ station.

As we have product-form we can state that the stationary distribution can be expressed by:

$$\pi(\mathbf{m}) = \frac{1}{G}\pi^{(1)}(\mathbf{m}^{(1)})\pi^{(2)}(\mathbf{m}^{(2)})\pi^{(3)}(\mathbf{m}^{(3)}), \tag{7.7}$$

where G is the normalizing constant.

7.6 An example of hybrid models in product-form with a G-queue

This example is more theoretical than the previous one. It does not aim to represent a real system but to show how flexible the approach we have presented in this chapter can be. We consider a model consisting of three blocks defined by three different formalism: a GSPN-EXP model, a CHC-SPN (see Chapter 5 for a formal definition) and a G-queue. Before describing the model we present a brief introduction to the G-networks because we have not reviewed this formalism before in this thesis. Then we

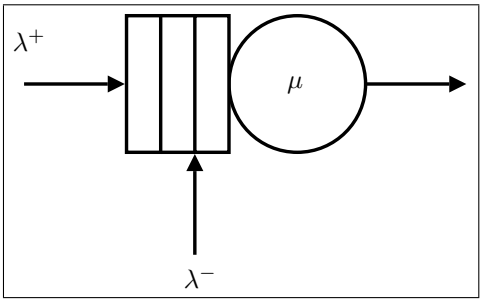

Figure 7.14: Simple G-queue with positive and negative customers.

describe the composition of the three models and analyze the steady state solution of the whole system provided that it exists.

7.6.1 Description of a general G-Queue

In its simplest definition, a single G-queue has one class of customers. Customers can be either positive or negative. The G-queue treats the positive customers exactly like a normal exponential queue. When a negative customer arrives to the G-queue we can have two possibilities:

- The queue is empty. In this case the negative customer arrival does not change the state of the queue.
- The queue has a positive number of customers. In this case the negative customer removes one customer form the queue.

We assume Poisson arrivals for positive and negative customers with rates λ^+ and λ^- respectively. The service time is exponentially distributed with rate μ. Figure 7.14 illustrates a simple G-queue.

We can see a G-queue as a normal exponential queue with service rate $\lambda^- + \mu$, therefore the stationary probability $\pi(n)$ of observing n customers in the queue is given by:

$$\pi(n) \propto \left(\frac{\lambda^+}{\lambda^- + \mu}\right)^n.$$

G-queues can be combined in order to obtain G-networks [50]. A customer that leaves a G-queue can leave the system or enter in another G-queue as a positive or a negative customer. The routing is probabilistic with fixed probabilities. It can be proved that under stability conditions the steady state solution of a G-network is in product-form.

G-networks can be studied using RCAT [60]. In fact:

- There can always be a positive or a negative customer arrival;

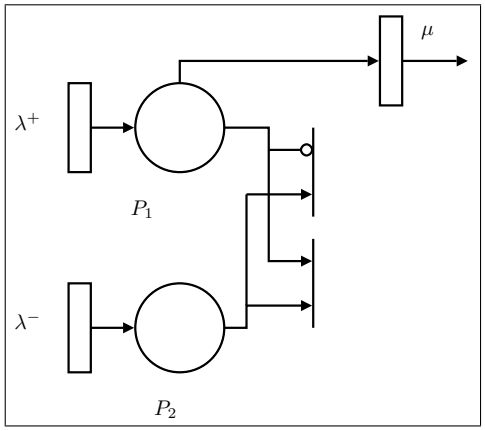

Figure 7.15: GSPN model of a simple G-queue with positive and negative customers.

- Every state can be reached by a customer departure (i.e., the reversed transitions corresponding to forward active action types are enabled in every state);
- The reversed rates of the transitions corresponding to customer departures have constant rates.

The latter point can be easily proved using Kolmogorv's criteria. Let n_1 be a state of the G-queue with $n_1 > 0$. Then the rate to state $n_2 = n_1 - 1$ is given by the sum of the rate of a service completion (μ) and the rate of negative customer arrivals λ^-. Therefore, we have:

$$\pi(n_1)(\mu + \lambda^-) = \pi(n_2)\overline{\alpha},$$

that gives an easy expression for the reversed transition rate $\overline{\alpha}$:

$$\overline{\alpha} = \lambda^+.$$

The reversed rate of the departure transitions is then obtained by Definition 5:

$$\overline{\mu} = \frac{\lambda^+ \mu}{\lambda^- + \mu}.$$

7.6.2 The model description

We now describe the hybrid model we are going to study in the following. We first give a graphical representation of the whole model, and then we describe in detail each of the three composing blocks.

The hybrid model is depicted by Figure 7.16. The model parameters are the two arrival rates λ_1 and λ_2, the probability p of a customer to get back to BLOCK 1 as well as all the parameters of the sub-models BLOCK 1, BLOCK 2 and BLOCK 3 that we are describing in the following paragraphs.

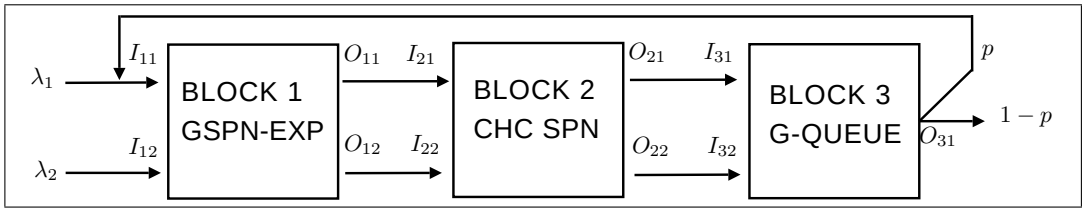

Figure 7.16: Hybrid model studied in Section 7.6

BLOCK 1 description. This sub-model is model GSPN-EXP with a single server as described in Section 6.2.1, and $R = 2$. Transition T_2 (T_7 in Figure 7.17) has rate $\lambda_{12} = \lambda_2$. Note that arrivals to place P_1 (P_{11}) come from the outside and also from the internal feedback (i.e., from point O_{31} if Figure 7.16). We can model this by setting the transition rate of T_1 (T_6) equals to $\lambda_{11} = \lambda_1$ and the feedback flow entering directly to P_1 (P_{11}). We call μ_1 the service rate.

BLOCK 2 description. This sub-model is the CHC-SPN studied in Section 5.3.5 depicted in Figure 5.15. According to the original labels of Figure 5.15, transition T_1 fires when a customer goes from O_{11} to I_{21} and transition T_2 fires when a customer goes from O_{12} to I_{22}. In Figure 7.17 the labels of places and transitions preserve the same order of Figure 5.15 but a 2 is added. Therefore, the rate of transition T_{2i} is χ_{2i} for $i = 3, \ldots, 8$. The output denoted by O_{21} is controlled by the firing of transition T_7 (T_{27}) while the output denoted by O_{22} is controlled by transition T_3 (T_{23}).

BLOCK 3 description. This sub-model is a G-queue. Positive customers arrive from I_{31} and negative customers arrive from I_{32}. After a job completion customers depart from O_{31} and go back to BLOCK 1 with probability p and exit the system with probability $1 - p$. We denote the service rate by μ_3.

Note that the model of Figure 7.16 has a feedback, therefore the arrivals to I_{11} are, in general, not according to a Poisson process.

Figure 7.17 shows a possible mapping of the hybrid model of Figure 7.16 into a GSPN in product-form. Transitions T_6 and T_7 model the external arrivals and have rates λ_1 and λ_2 respectively. Immediate transitions t_4 and t_5 are used to model the probabilistic output from the G-queue. In this case the use of immediate transitions could be replaced by using two timed transitions with rate $\mu_3(1 - p)$ and $\mu_3 p$.

7.6.3 The model analysis

By the results presented in this thesis we know that:

- Model GSPN-EXP with a single server satisfies RCAT conditions with a single exponential server and constant rate arrivals,

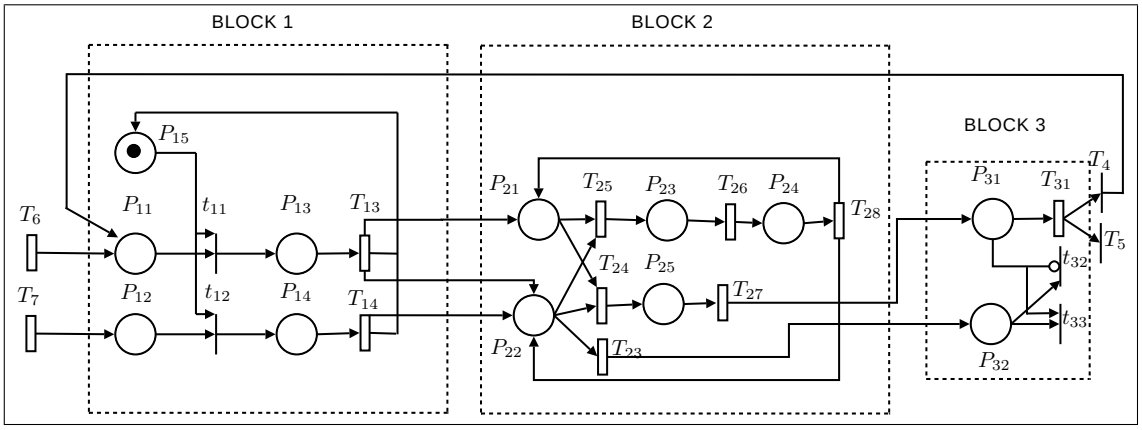

Figure 7.17: GSPN model equivalent to the hybrid model studied in Section 7.6

- The CHC-SPN of BLOCK 2 satisfies RCAT conditions,

moreover from [60] we know that the G-queue satisfies RCAT conditions. Therefore, we can do an RCAT based analysis of the whole system. The unknown parameters are:

- x_a: the transition rate corresponding to an arrival to I_{21},
- x_b: the transition rate corresponding to an arrival to I_{22},
- x_c: the transition rate corresponding to an arrival to I_{31},
- x_d: the transition rate corresponding to an arrival to I_{32},
- x_e: the transition rate corresponding to an arrival to I_{11} due to the feedback. The total transition rate due to any arrival to I_{11} is $x_e + \lambda_1$.

In the analysis of GSPN-EXP model we proved that the reversed rate corresponding to a class r customer departure is λ_r, therefore we have $x_a = \lambda_1 + x_e$ and $x_b = \lambda_2$. Let us consider the CHC-SPN of BLOCK2. We already determined the reversed rates of the transition rates due to the firings of T_3 and T_7 in Section 5.3.5, so we have $x_c = x_a$ and $x_d = x_b$.

From the analysis of the G-queue we obtain an equation for x_e. Note that in this case when the queue passes from state with k customers to a state with $k-1$ customers there are three possible transitions combined:

- an arrival of a negative customer,
- a job completion and the customer leaves the system,
- a job completion and the customer goes back to I_{11}.

According to the introduced notation we have that $\lambda^+ = x_c$ and $\lambda^- = x_d$, therefore we have $x_e = (x_c\mu_3 p)/(\mu_3 + x_d)$. Now we have to solve the following system of equations:

$$\begin{cases} x_a = \lambda_1 + x_e \\ x_b = \lambda_2 \\ x_c = x_a \\ x_d = x_b \\ x_e = \frac{x_c\mu_3 p}{\mu_3 + x_d} \end{cases}$$

The solution for this system is:

$$\begin{cases} x_a = x_c = \frac{\lambda_1(\mu_3+\lambda_2)}{(1-p)\mu_3+\lambda_2} \\ x_b = x_d = \lambda_2 \\ x_e = \frac{\lambda_1\mu p}{(1-p)\mu+\lambda_2}. \end{cases}$$

Under stability assumptions, we can say that the product-form solution π is:

$$\pi(\mathbf{S}) \propto g_1(S_1)g_2(S_2)g_3(S_3),$$

where $\mathbf{S} = (S_1, S_2, S_3)$ is the model state, S_i the state of BLOCK i model, $i = 1, 2, 3$. Functions g_i are defined as follows. For BLOCK 1, S_1 is the GSPN state $\mathbf{m}_1 = (m_{11}, m_{12}, m_{13}, m_{14}, m_{15})$, and the unnormalized steady state probabilities are given by (6.2).

$$g_1(\mathbf{m}_1) = \left(\frac{\lambda_1 + x_e}{\mu_1}\right)^{m_{11}+m_{13}} \left(\frac{\lambda_2}{\mu_1}\right)^{m_{12}+m_{14}} \frac{(m_{11} + m_{12})!}{m_{11}!m_{12}!}$$

For BLOCK 2, state S_2 is vector $\mathbf{m}_2 = (m_{21}, m_{22}, m_{23}, m_{24}, m_{25})$ we have that the unnormalized steady state probabilities are given by (5.22):

$$g_2(\mathbf{m}_2) = \left(\frac{x_a\chi_{23}}{\chi_{24}\lambda_2}\right)^{m_{21}} \left(\frac{\lambda_2}{\chi_{23}}\right)^{m_{22}} \left(\frac{x_a\chi_{25}}{\chi_{24}\chi_{26}}\right)^{m_3} \left(\frac{\lambda_1\chi_{25}}{\chi_{24}\chi_{28}}\right)^{m_{24}} \left(\frac{x_a}{\chi_{27}}\right)^{m_{25}}.$$

State S_3 of BLOCK 3 is just a natural number n, therefore we have:

$$g_3(n) = \left(\frac{x_a}{\lambda_2 + \mu_3}\right)^n = \left(\frac{\lambda_1}{(1-p)\mu_3 + \lambda_2}\right)^n.$$

7.7 An example of hybrid models with non-linear traffic equations

In the introduction of this chapter we said that the combination of models based on RCAT could originate a system of non-linear traffic equations. However the examples analyzed until now have shown just models with linear traffic equations.

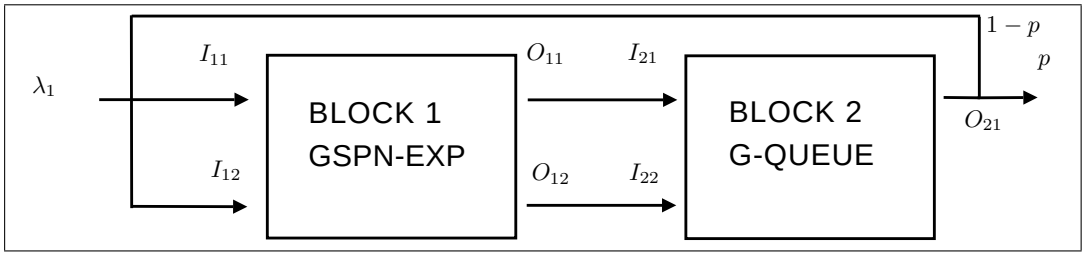

Figure 7.18: Hybrid model studied in Section 7.7

It is clear that dealing with non-linear systems is in general more difficult than solving linear ones. However it is out of the purpose of this thesis analyzing efficient algorithms for solving this class of traffic equations. The purpose of this example is only to show a model with non-linear traffic equations, therefore we study a really simple case.

7.7.1 The model description

We now describe the hybrid model we are going to study in the following. As done before, we first give a graphical representation of the whole model, and then we describe in detail each of the two composing blocks.

The hybrid model is depicted by Figure 7.18. The model parameters are the arrival rate λ_1, the probability p of a customer to exit the system after being served by BLOCK 2 as well as all the parameters of the sub-models BLOCK 1 and BLOCK 2.

BLOCK 1 description. This sub-model is model GSPN-EXP with a single server as described in Section 7.4.3, and $R = 2$. Transition T_2 firing is governed by the customer flow from O_{21} to I_{12}, while T_1 has rate $\lambda_{11} = \lambda_1$. We call μ_1 the service rate.

BLOCK 2 description. This sub-model is a G-queue. Positive customers arrive from I_{21} and negative customers arrive from I_{22}. After a job completion customers depart from O_{21} and go back to BLOCK 1 (I_{21}) with probability $1-p$ and exit the system with probability p. We denote the service rate by μ_2.

Note that the model of Figure 7.18 has a feedback.

7.7.2 The model analysis

The unknown parameters of this model are:

- x_a: the transition rate corresponding to an arrival to I_{21},

- x_b: the transition rate corresponding to an arrival to I_{12},
- x_c: the transition rate corresponding to an arrival to I_{22},

In the analysis of GSPN-EXP model we proved that the reversed rate corresponding to a class r customer departure is λ_r, therefore we have $x_a = \lambda_1$ and $x_c = x_b$. By the analysis of the reversed process of the simple G-queue, we have that:

$$x_b = \frac{x_a \mu_2 (1-p)}{x_c + \mu_2}.$$

From this last equation we have:

$$x_b = \frac{\lambda_1 \mu_2 (1-p)}{x_b + \mu_2} \Rightarrow (x_b + \mu_2) x_b = \lambda_1 \mu_2 (1-p).$$

Note that the equation is not linear for x_b. Hence we have:

$$x_b^2 - \mu_2 x_b - \lambda_1 \mu_2 (1-p) = 0,$$

that gives the solution:

$$x_b = \frac{-\mu_2 + \sqrt{\mu_2^2 + 4\lambda_1 \mu_2 (1-p)}}{2} = \epsilon.$$

Then, under stability, the steady state solution of the model depicted in Figure 7.18 is in product-form and is given by the normalized product of the steady state solution of model GSPN-EXP where the transition rate for T_2 is ϵ and the steady state solution for the G-queue where $\lambda^+ = \lambda_1$ and $\lambda^- = \epsilon$.

7.8 Conclusions

In this chapter we have discussed the following results:

- Models GSPN-EXP, GSPN-COX, GSPN-PS and GSPN-IS satisfy the $M \Rightarrow M$ property, therefore they can be combined in a BCMP-like way. The resulting model can include non-BCMP stations if they fulfil the $M \Rightarrow M$ property. The resulting model is in product-form.
- We have proved that, given a queueing station with a probabilistic queueing discipline, class independent exponential service time and without preemption, the $M \Rightarrow M$ property holds if and only if the every customer in the queue has the same probability to enter in service after a job completion.

- Models GSPN-EXP, GSPN-COX can be combined using RCAT. Note that for what concerns models GSPN-PS and GSPN-IS, their CTMCs are isomorphic to the CTMCs of the equivalent BCMP stations, therefore an RCAT based analysis can be found in [58]. By this result we have shown a hybrid modeling technique that originates models whose product-form solution can be decided and studied by RCAT.

- We have shown how it is possible to derive new product-form GSPN models using RCAT. In our opinion this approach is really promising for the definition of new product-form GSPN models with practical applications.

8

An algorithm to transform BCMP QNs into GSPNs

8.1 Introduction

In this chapter we define an algorithm that, given a BCMP queueing network, may calculate an equivalent GSPN with a finite structure. The equivalence terms are those explained in Chapter 6, i.e., we can say that the steady state probabilities of the BCMP network and of the corresponding GSPN model are identical under appropriate aggregations. These aggregations are defined such that the average performance indices are preserved. Another property that is preserved by the GSPN model is that it is in product-form by $M \Rightarrow M$ property. These results have been published in [13].

The main idea of the algorithm is simple. We translate each service center of the BCMP QN into a GSPN using the models presented in Chapter 6. Then, we compose these models using immediate transitions used to represent the probabilistic routing. The resulting GSPN is in product-form by the results presented in Chapter 7. We aim to define an algorithm that can be easily extended in order to deal with non-BCMP stations that fulfil the $M \Rightarrow M$ property, e.g., the MSCCC station of Le Boudec [86]. Moreover, the compositionality and the possibility of hierarchical compositions should be allowed. To this aim we mark some places of the GSPN blocks with special labels:

- The arrival places (or input places). These places are the output vector of the transition modeling the arrivals.
- The departure places (or output places). These places are used to store the tokens representing customers that have completed their service in the station.

We call the set of input places the *input interface* of the model, and the set of output places the *output interface* of the model. Hence, the algorithm becomes very modular. In fact, one can define an arbitrary service center that satisfies the $M \Rightarrow M$ property and then it can be connected with the rest of the network just knowing its input and output interfaces.

8.2 Algorithm definition

We shall now define the algorithm that converts a BCMP QN with multiple classes of customers into a product-form GSPN. Note that, in order to keep the notation simple, we just deal with networks with multiple chains but just one class for chain. Therefore, in the first part of the chapter chain and class become synonymous. Let Ω be the set of queueing stations of the BCMP network. In the algorithm we use the following syntactical conventions for the input that is represented by the set of parameters of the QN, according to the definition introduced in the section on product-form stochastic models:
- $\mathbf{P}$ is the routing matrix.
- $\Omega = \{c_1, \ldots c_N\}$ is the set of queueing stations, and c_i is a record with the following fields: $c_i.\mu^{(c)}$ is the single server service rate, $c_i.K$ is the number of servers, $c_i.type$ is a description of the station type. For FCFS stations, we use $c_i.\mu$ to point out that class-dependent service rate is not allowed.
- If station i has a Coxian service time distribution, then we use the following notation that reflects the influence of customers of class r: $c_i.L_r$ is the number of stages of the random variable, $c_i.\mu_\ell^{(r)}$ the rate of stage ℓ, $c_i.a_\ell^{(r)}(\ell < L_r)$ the probability that a customer goes to stage $\ell+1$ after being served at stage ℓ, and by $c_i.b_\ell^{(r)}$ the probability of leaving the Coxian service after being served at stage ℓ.
- $\lambda = (\lambda_1, \ldots, \lambda_R)$ is the vector of the arrival rates for chain r, $1 \leq r \leq R$. If chain r is closed then $\lambda_r = 0$. Vector $\mathbf{K} = (K_1, \ldots, K_R)$ components denote the number of customers for closed chains. $K_r = 0$ for open chains.

Let us describe the output syntactical conventions that is the definition of the GSPN equivalent to the given QN.
- $\mathcal{P}, \mathcal{T}$ are the sets of places and transitions, respectively. Each element of $\mathcal{P}$ or $\mathcal{T}$ can be labeled by a superscript (e.g. $P^I_{r,\ell,i}$ is labeled by an I). Subscript letters denote some variables defined in the algorithm. In particular letter r denotes the customer chain/class, ℓ the stage of a Coxian random variable, i, j the corresponding service center number. For example $P^S_{r,\ell,i}$ is a place defined in the i-th service center translation, corresponding to the ℓ-th stage of the r class Coxian service time. Timed transitions use capital T. Labels I and O play a special role for places, as P^I_r represents the input-place for class r customers, and P^O_r the output place. Later in this section we show an example.
- $\mathbf{m}$ is a net state and $\mathbf{M}$ is the initial state. Vector $\mathbf{m}$ consists of components whose names are derived from the corresponding place names. For example m^S_i is the number of initial tokens in place P^S_i.
- The arcs are specified in terms of input, output, and inhibition functions as defined above in the section on product-form stochastic models. Transition priorities can be either 0 or 1 and they are determined by function Π introduced above.
- The arc weights are defined by function $w(t, \mathbf{m})$ for each timed transition t and state $\mathbf{m}$. For brevity we write just $w(t)$. As arc weights can be state dependent, a symbolic function must be assigned $w(t)$. In order to point out this, we use the

assignment symbol ← instead of the usual :=.
- Function $d(t, j)$ defines the probability of the output vector $O_j(t, P_j)$ for a transition t and a place P_j, as described above in the section on product-form stochastic models.

Before introducing the algorithm, it is worthwhile illustrating some notes on the translation approach. The algorithm first translates every QN station into a GSPN (sub)model. Then it combines these GSPNs obtained by the first step by connecting them through a set of immediate transitions that model the QN routing. In order to simplify the definition of the new combined GSPN in product-form, we use a standard name for input/output places for each GSPN (sub)model corresponding to a station type. This can be thought as an input and output interface of each GSPN submodel that simplifies their composition (see Figure 8.1). Although this can be a complication in the net structure, as a set of reducible immediate transitions could be generated, the modularity of our algorithm results really enhanced. In fact, let

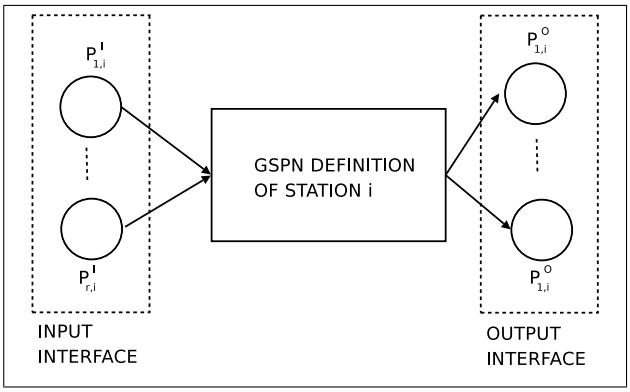

Figure 8.1: Modularity of station equivalent GSPN blocks

us consider station i and suppose that $p_{ij}^{(r)} > 0$ and $p_{ik}^{(r)} > 0$. Using input and output interfaces we can represent this probabilistic routing without caring about the queueing discipline of stations i and j as illustrated in Figure 8.2. Note that the use of probabilistic arcs to implement the routing is not really necessary. In fact, a set of immediate transitions whose weights are proportional to the routing probabilities can be used, even if we think that the readability of the model gets worse. However, this solution should be used in most of the existing tools at least to describe the incidence matrix and allow the structural analysis. Finally we point out that a job completion can enable together the internal immediate transitions of the station that generated that event, and the immediate transitions that model the routing. However these two classes of transitions are not in conflict, therefore the behavior of the net is not ill-defined. For the sake of precision one can decide to give different priorities to the routing immediate transitions and those internal to the station.

The main structure of the algorithm is simple and is shown by Algorithm 2. The main cycle of the algorithm considers each service center of the QN and executes

Input: BCMP QN: $\Omega, \mathbf{P}, \lambda, \mathbf{K}$
Output: GSPN $\mathcal{T}, \mathcal{P}, w, H, I, O, d, \mathbf{M}$
`/* Initialization */`
$\mathbf{M} := \mathbf{0}$; $\mathcal{P} := \emptyset$; $\mathcal{T} := \emptyset$;
graph$(d) := \emptyset$; graph$(H) := \emptyset$; graph$(I) := \emptyset$; graph$(O) := \emptyset$; graph$(w) := \emptyset$;
`/* Transform every service center */`
foreach $c_i \in \Omega$ **do**
  switch $c_i.type$ **do**
    case *FCFS*
      FCFSBlock;
    end
    case *LCFSPR*
      LCFSPRBlock;
    end
    case *IS*
      ISBlock;
    end
    case *PS*
      PSBlock;
    end
  end
end
`/* Model the routing */`
ROUTINGBlock;
`/* Model arrivals and closed chains population */`
CHAINSBlock;

Algorithm 2: Main program

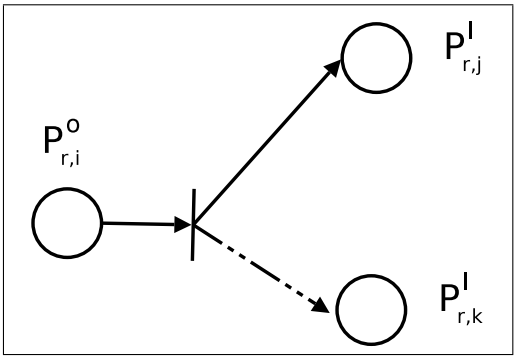

Figure 8.2: Modelling the QN probabilistic routing. In this example the output vector of transition $t^Z_{r,i}$ is determined probabilistically and $O_0(t^Z_{r,i}, P^I_{r,j}) = 1$, $O_0(t^Z_{r,i}, P^I_{r,k}) = 1$ and $d(t^Z_{r,i}, 0) = p^{(r)}_{ij}$, $d(t^Z_{r,i}, 1) = p^{(r)}_{ik}$.

the appropriate code block. Finally, the queueing network routing is modelled by *ROUTINGBlock*. Instructions graph$(g) := \emptyset$, where g is a function, are used to initialize the function definitions to the empty set, i.e. their domain is initially empty. The labels are slightly modified with respect to those introduce in the theoretical analysis of Chapter 6, this should enhance the algorithm readability.

FCFSBlock is defined by Algorithm 3. It generates the FCFS-equivalent GSPN block described in the previous section. P^S_i is the place for the free servers, $P^I_{r,i}$ the place for queued customers of class r (and also the input place), $P^S_{r,i}$ the place for customers being served. Place $P^O_{r,i}$ receives the class r customers after job completion. Transition $t_{r,i}$ puts in service a class r customer and $T_{r,i}$ models the service time.

Let us illustrate the *LCFSPRBlock*. In order to clarify the notation, we recall that i denotes the considered station, ℓ the Coxian service stage, r the customer class, label Q denotes the queue and label S denotes the service. The transformation algorithm for the *LCFSPRBlock* is illustrated by Algorithm 4 and 5 that is splitted in two parts for typographical reasons. Place $P^S_{r,\ell,i}$ contains the tokens representing class r customers in service at stage ℓ while place $P^Q_{r,\ell,i}$ contains the preempted ones. Transition $t^P_{r,\ell,i}$ implements the preemption if there is an arrived customer $(I(t^P_{r,\ell,i}, P^T_i) := 1)$, there is at least a class r customer in stage ℓ $(I(t^P_{r,\ell,i}, P^S_{r,\ell,i}) := 1)$, there are no free servers $(H(t^P_{r,\ell,i}, P^S_i) := 1)$. Transition $t^R_{r,\ell,i}$ implements the customer resume. Place $P^T_{r,i}$ stores the class r just arrived customers that will get the service immediately.

The *ISBlock* is simple and is illustrated by Algorithm 6. *PSBlock* is similar to *ISBlock* so the same Algorithm 6 applies, except for the definition of function w. In fact in PS stations there is a limited number of servers, hence the servers speed must be partitioned among all the customers in the station. The the weight w of

/* Add a place for the free servers */
$\mathcal{P} := \mathcal{P} \cup \{P_i^S\}$;
foreach $r \in c_i.\mathcal{R}$ **do**
  /* Add 2 places, an immediate transition and a timed transition for each class */
  $\mathcal{P} := \mathcal{P} \cup \{P_{r,i}^I, P_{r,i}^S, P_{r,i}^O\}$;
  $\mathcal{T} := \mathcal{T} \cup \{t_{r,i}, T_{r,i}\}$;
  /* Input functions of immediate transition */
  $I(t_{r,i}, P_i^S) := 1$;
  $I(t_{r,i}, P_{r,i}^I) := 1$;
  $I(T_{r,i}, P_{r,i}^S) := 1$;
  /* Set immediate transitions */
  $O(t_{r,i}, P_{r,i}^S) := 1$;
  $O(T_{r,i}, P^O(r,i) := 1$;
  $O(T_{r,i}, P_i^S) := 1$;
  $w(t_{r,i}) \leftarrow m_{r,i}^A$;
  /* Set timed transition rates */
  $w(T_{f,r,i}) \leftarrow m_{r,i}^S * c_i.\mu$;
  /* Transition priority */
  $\Pi(t_{r,i}) := 1$;
  $\Pi(T_{r,i}) := 0$;
  $M_i^S := c_i.K$;
end

Algorithm 3: FCFSBlock

```
/* Add a place for the free servers                                   */
P := P ∪ {P_i^S};
P := P ∪ {P_i^T};
foreach i ∈ c_i.R do
    /* Set up the arrival places and transitions                      */
    P := P ∪ {P_{r,i}^I, P_{r,i}^T};
    T := T ∪ {t_{r,i}^I};
    I(t_{r,i}^I, P_{r,i}^I) := 1;
    O(t_{r,i}^I, P_{r,i}^T) := 1;
    O(t_{r,i}^I, P_i^T) := 1;
    w(t_{r,i}^I) := 1; Π(t_{r,i}^I) := 1;
    /* Set up the output places                                       */
    P := P ∪ {P_{r,i}^O};
    /* Add the other needed places and transitions for each class
       */
    for ℓ := 1 to c_i.L_r do
        P := P ∪ {P_{r,ℓ,i}^Q, P_{r,ℓ,i}^S};
        /* Add transitions which model the service time               */
        T := T ∪ {T_{r,ℓ,i}};
        w(T_{r,ℓ,i}) ← c_i.μ_ℓ^(r) * m_{r,ℓ,i}^S;
        Π(T_{r,ℓ,i}) := 0;
        I(T_{r,ℓ,i}, P_{r,ℓ,i}^S) := 1;
        if ℓ ≠ c_i.L_r then
            O_0(T_{r,ℓ,i}, P_{r,ℓ+1,i}^S) := 1;
            d(T_{r,ℓ,i}, 0) := c_i.a_ℓ^(r);
        end
        O_1(T_{r,ℓ,i}, P_i^S) := 1;
        O_1(T_{r,ℓ,i}, P_i^O) := 1;
        d(T_{r,ℓ,i}, 1) := c_i.b_ℓ^(r);
        /* Add transitions modelling the preemption (label P)        */
        T := T ∪ {t_{r,ℓ,i}^P}; I(t_{r,ℓ,i}^P, P_{r,ℓ,i}^S) := 1;
        I(t_{r,ℓ,i}^P, P_i^T) := 1; H(t_{r,ℓ,i}^P, P_i^S) := 1;
        O(t_{r,ℓ,i}^P, P_{r,ℓ,i}^Q) := 1; O(t_{r,ℓ,i}^P, P_i^S) := 1;;
        w(t_{r,ℓ,i}^P) ← m_{r,ℓ,i}^S Π(t_{r,ℓ,i}^P) := 1;
        /* Add transitions modelling the resume (label R)             */
        T := T ∪ {t_{r,ℓ,i}^R}; I(t_{r,ℓ,i}^R, P_{r,ℓ,i}^Q) := 1;
        I(t_{r,ℓ,i}^R, P_i^S) := 1; H(t_{r,ℓ,i}^R, P_i^T) := 1;
        O(t_{r,ℓ,i}^R, P_{r,ℓ,i}^S) := 1; w(t_{r,ℓ,i}^R) ← m_{r,ℓ,i}^Q;
        Π(t_{r,ℓ,i}^R) := 1;
    end
end
```

Algorithm 4: LCFSPRBlock (Part 1)

transition $T_{r,\ell,u}$ for PS station is defined as follows:

$$w(T_{r,\ell,i}) \leftarrow \frac{\min\left(\sum_{t\in c_i.\mathcal{R}} \sum_{u:=1}^{c_i.L_r} m_{t,u,i}, c_i.K\right)}{\sum_{t\in c_i.\mathcal{R}} \sum_{u:=1}^{c_i.L_r} m_{t,u,i}} * c_i.\mu_\ell^{(r)} * m_{r,\ell,i}.$$

Place $P_{r,\ell,i}$ contains the class r customers at stage ℓ of station i. Transition $T_{r,\ell,i}$ models the stage ℓ service time and its output vector is probabilistic according to the Coxian random variable parameters.

In the *ROUTINGBlock* we define a set of transitions t^Z, where $t^Z_{r,i}$ models the probabilistic routing for class r customers after being served by station i. The main idea has been introduced at the beginning of this section. The external arrivals are modelled by appropriate timed transition that are always enabled. In order to model a chain population it suffices to set the initial marking $M^I_{r,i}$ for an arbitrary service center i equals to the chain population. This work is done by *CHAINSBlock* illustrated by Algorithm 8.

8.3 Supported extensions

The proposed algorithm that transforms BCMP QNs into GSPNs can support the extensions of the introduced class of BCMP QNs. In this section, for sake of brevity we just cite some extensions that can be easily supported by the algorithm with small changes.

State dependent service rate. BCMP theorem defines several extensions of the product-form solution to include state dependent service rates. We can represent all the extensions whose service rates depend only on the state of the stations (i.e., we exclude the service rates depending on the state of a subnet of the QN).

Multiple chain and multiple class. In this work we have not considered the case of customer class switching. This has been done just to keep the notation simple. In fact by introducing some easy changes to the algorithm, with a more complex state notation we can also model multiple classes and multiple chains BCMP QNs.

Other service station queueing disciplines. Some extensions of BCMP theorem have been defined to allow different queueing disciplines that lead to $M \Rightarrow M$ product-form. If a GSPN model can represent such disciplines, then the proposed transformation algorithm from QN to GSPN can be easily modified in order to include these new station types. In fact it suffices to define a station type label and extend the *switch* construct of *Main Program* to include that new type of station. Then the model definition must provide an input interface and an output interface as described in the previous section.

/* Add transitions modelling the customers entering in stage of service 1 */
$\mathcal{T} = \mathcal{T} \cup \{t^R_{r,0,i}\}$; $I(t^R_{r,0,i}, P^S_i) := 1$;
$I(t^R_{r,0,i}, P^T_i) := 1$; $I(t^R_{r,0,i}, P^T_{r,i}) := 1$;
$O(t^R_{r,0,i}, P^S_{r,1,i}) := 1$; $w(t^R_{r,0,i}) := 1$;
$\Pi(t^R_{r,0,i}) := 1$;
$M^S_i := 0$;

Algorithm 5: LCFSPRBlock (Part 2)

/* Set the places set */
foreach $r \in c_i.\mathcal{R}$ **do**
  $\mathcal{P} := \mathcal{P} \cup \{P^O_{r,i}\}$;
  for $\ell := 1$ *to* $c_i.L_r$ **do**
    /* Add place for stage ℓ of class r customers */
    $\mathcal{P} := \mathcal{P} \cup \{P_{r,\ell,i}\}$;
    /* Add transitions modelling service time */
    $\mathcal{T} := \mathcal{T} \cup \{T_{r,\ell,i}\}$;
    $w(T_{r,\ell,i}) \leftarrow m_{r,\ell,i} * c_i.\mu^{(r)}_{\ell,i}$;
    $I(T_{r,\ell,i}, P_{r,\ell,i}) := 1$;
    $O_0(T_{r,\ell,i}, P_{r,\ell+1,i}) := 1$;
    $d(T_{r,\ell,i}, 0) := c_i.a_{r,\ell}$;
    $O_1(T_{r,\ell,i}, P^O_{r,i}) := 1$;
    $d(T_{r,\ell,i}, 1) := c_i.b_{r,\ell}$;
  end
  Let $P^I_{r,i}$ be an alias for $P_{r,\ell,1}$;
end

Algorithm 6: ISBlock

```
/* model the QN routing by GSPN                                    */
foreach c_i ∈ Ω do
    foreach r ∈ c_i.R do
        /* Model internal routing                                  */
        T := T ∪ {t^Z_{r,i}};
        I(t^Z_{r,i}, P^O_{r,i}) := 1;
        w(t^Z_{r,i}) := 1;Π(t^Z_{r,i}) := 1;
        f := 0;
        foreach c_j ∈ Ω do
            if p^(r)_{i,j} > 0 then
                f := f + 1;
                O_f(t^Z_{r,i}, P^I_{r,j}) := 1;
                d(t^Z_{r,i}, f) := p^(r)_{i,j};
            end
        end
        /* model QN departures                                     */
        if p^(r)_{i,0} > 0 then
            d(t^Z_{r,i}, f + 1) := p^(r)_{i,0};
        end
    end
end
```

Algorithm 7: ROUTINGBlock

8.4 Example

In this section we sketch an example of application of the proposed algorithm, by considering also its extensions. We apply the algorithm to the queueing network illustrated in Figure 8.3 (a). It is a QN with three classes of customers clustered in two chains (classes A and B, class C) and there is a class switching. Classes A and B form an open chain while class C a closed one. Note that the QN has product-form solution, but it is *not* a BCMP QN because of the presence of a MSCCC station, i.e., a queueing discipline not considered by BCMP theorem. MSCCC discipline follows a multiple servers RANDOM discipline, but cannot serve two customers of the same class simultaneously. It is described in [86] and it is proved and it holds the $M \Rightarrow M$ property. Customers of class A and B have the same stochastic behavior once they reach the servers, and they leave the system at the end of the service. Class C customers can be thought as representing a set of interior control processes whose number is given, an is denoted in this case by $K = 5$. In order to simplify the system model we assume that all the service times are exponentially distributed. We assume that station 2 has 2 servers and station 3 has 3 servers.

By applying the proposed algorithm the three service centers can be translated into GSPN models that are eventually composed and connected according to the routing matrix, as described in the previous section. Then we obtain the overall GSPN equivalent to the given QN, as showed in Figure 8.3 (b). The parameters of the GSPN are completely defined by the various steps of the algorithm.

As the three blocks satisfy $M \Rightarrow M$ property, we can state that the whole system has a product-form stationary probabilities function. Then the derived GSPN can be analyzed by product-form solution or by simulation.

Note that in this example we have showed how it is possible to deal with class switching and no-BCMP queueing disciplines.

8.5 Conclusions

- to represent the concept of class of a place. Note that this does not necessarily require the idea of color, as defined in Colored Petri Net extension.
- to identify whether a model satisfies the $M \Rightarrow M$ property. An open problem is the definition of an automatic efficient algorithm to decide this condition.
- to represent the stationary state probability expression of the model in isolation for each GSPN model. In fact, although we know that a GSPN model satisfying the $M \Rightarrow M$ property has a product-form solution, only if the explicit expression of the product-form is known we can obtain the stationary state probabilities for the whole net. For some models although it is known that they fulfil the $M \Rightarrow M$ property, the computation of their steady state solution can be computationally inefficient. For example the MSCCC model satisfies $M \Rightarrow M$ property, but as far as we know there are not algorithms to efficiently derive the average performance

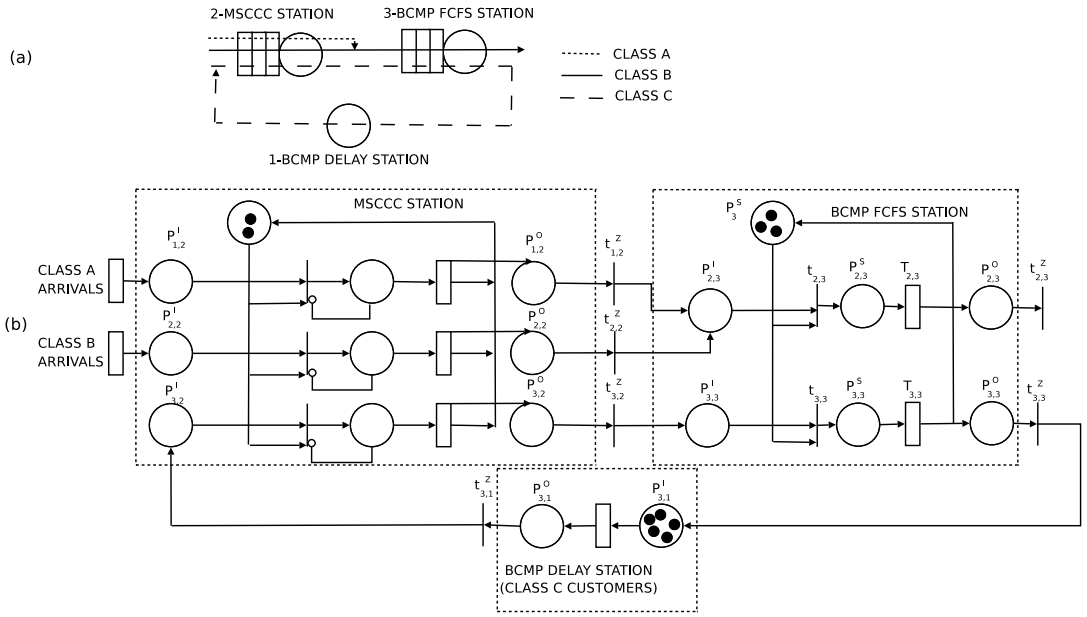

Figure 8.3: (a) System modelled by a no-BCMP queueing network. (b) System modelled by a product-form GSPN.

indices, and the computation of its steady state solution involve the definition of a recursive function. In these cases the models can be still studied by simulation, and the theoretical results guarantee that the performance indices are the same of the original hybrid model.

```
for r := 1 to R do
    if λ_r > 0 then
        /* Open chain                                          */
        T := T ∪ {T_{r,0}};
        w(T_{r,0}) := λ_r;
        f := 0;
        foreach p^{(r)}_{0,j} > 0 do
            O_f(T_{r,0}, P^I_{r,j}) := 1;
            d(T_{r,0}, f) := p^{(r)}_{0,j};
            f := f + 1;
        end
    end
    else
        /* Closed chain                                        */
        Choose an arbitrary i such that P^I_{r,i} exists;
        M^I_{r,i} := K_r;
    end
end
```

Algorithm 8: CHAINSBlock

Conclusions

This thesis is mainly focused on stochastic models in product-form. Our research has had two main guidelines: one is the comparison among different product-form formalisms, and the other is the definition of a framework that allows a hybrid modeling of various types of formalisms presenting product-form solutions.

Comparing product-form results and the goal of defining a unique framework for studying product-form models are not new topics. Several works in literature addressed these problems (e.g. [7, 60, 58]) or proposed the translation of some theoretical results obtained for a formalism in terms of a different formalism (e. g. [110, 57, 70]).

We point out that the originality of the theoretical results presented in this thesis is twofold. First, we have used two properties to analyze most of the well-known product-form models expressed by different formalisms, i.e., Muntz's $M \Rightarrow M$ [94] and *RCAT* [59, 61]. We have shown that the well-known product-form SPN model class defined in [63, 39] without batch token movements and with constant transition rates belongs to the class of models whose stochastic process can be expressed by RCAT in product-form. Second, we have shown the relation between the class of models that fulfil the $M \Rightarrow M$ property and the class of models whose product-form can be studied by RCAT. In particular, we derive a condition under which a model belongs to both the classes and we observe than none of the two classes of models includes the other one. We have presented some examples of stochastic models that can be studied by one of these two results (RCAT or $M \Rightarrow M$) but not by the other, or by both. Figure C.4 illustrates the relations between the model class satisfying the $M \Rightarrow M$ property and that satisfying RCAT conditions, and examples of models belonging to theses classes.

From a practical point of view, we have taken advantage from these theoretical results in order to define a possible framework for a hybrid modeling formalism in which product-forms can be identified. The main ideas can be summarized as follows:

- The product-form analysis is based on the $M \Rightarrow M$ property or RCAT theorem (or both when possible).

- We map every formalism that is used in the hybrid model into an appropriate GSPN. The choice of GSPN as the base formalism has at least two motivations. First, GSPNs (with inhibitor arcs) are Turing-complete and they are very expressive. For example, there exist GSPN models with a finite structure that model systems with infinite state space. This can be difficult to do in practice with other formalisms such as PEPA (that requires a recursive definition, or

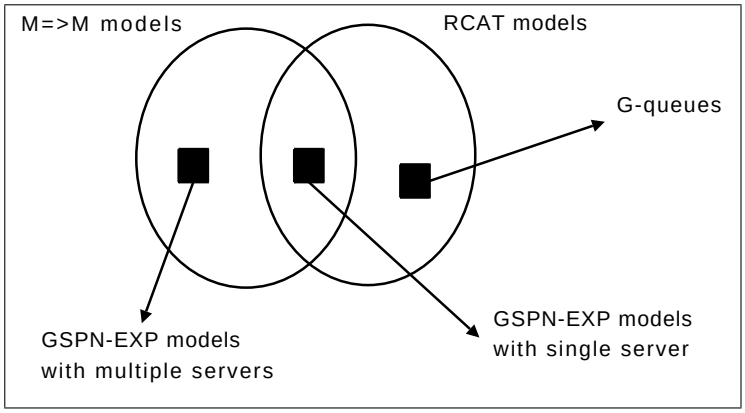

Figure C.4: Relation between $M \Rightarrow M$ models and RCAT models.

a truncation of the derivatives). Moreover, GSPNs have important structural properties and there are several tools for structural analysis, simulation, or exact analysis (e.g. GreatSPN [35], TimeNET [124], PIPE [23]).

We have addressed the problem of representing multiclass product-form queueing networks by product-form GSPNs. This is not a trivial problem because if single class queueing networks are known to be equivalent to state machine SPNs this is not true for multiclass ones. In fact, BCMP theorem [17] proved that for multiclass QNs the queueing disciplines affect the performance measures and the product-form conditions of the queueing model. We have defined a set of GSPN models that represent the BCMP queueing disciplines satisfying the following properties:

- The models are defined using standard GSPN formalism, without the need of extensions such as colors or the definition of queueing disciplines for the tokens.
- The models have a finite structure, i.e., they can be actually used in existing GSPN tools.
- The GSPN models preserve the average performance indices of the associated queueing system with the considered queueing discipline.
- The GSPN models can be composed by the $M \Rightarrow M$ property originating BCMP-like product-form GSPNs.
- The defined models do not belong to any of the class of well-known product-form for GSPNs.

Figure C.5 graphically illustrates the relations among various product-form models that can be derived by these results. The label GSPN-x identifies the set of GSPN models defined in Chapter 6 where x stands for EXP, COX, IS, PS. In the figure

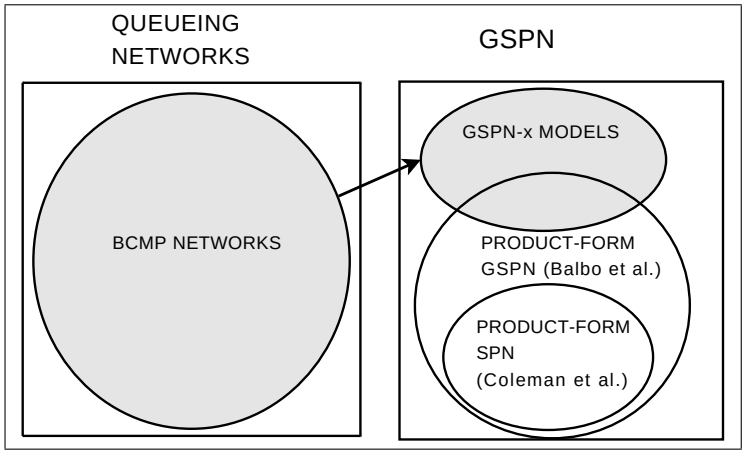

Figure C.5: Relations among (G)SPNs product-form stochastic models.

we show that the BCMP queueing networks are mapped into GSPN models that do not belong to any of the well-known product-form model classes.

Defining new product-form GSPNs using $M \Rightarrow M$ is not an easy task, however we have shown that GSPNs can be used to model some BCMP extensions satisfying $M \Rightarrow M$ studied in other works [1, 86]. In this context we have proved a new result on the product-form of probabilistic queueing disciplines. If we consider a probabilistic queueing discipline with exponential distributed service time, without preemption and class independent Poisson arrivals, we have proved that the following two sentences are equivalent:

- The station satisfies $M \Rightarrow M$ property.
- The choice of the customer in the queue to be served after a job completion is done according to a uniform distribution.

In other words, after a job completion every customer in the queue must have the same probability to enter the service in order to guarantee that the station fulfils the $M \Rightarrow M$ property.

The analysis of GSPNs based on RCAT has lead to several results. The first one, as we mentioned above, has been the definition of a relation between Coleman, Henderson et al. product-form SPN [63, 39] and RCAT models as illustrated by Figure C.6. This has allowed us to prove the product-form for a class of SPNs without verifying the set of global balance equations. Moreover, RCAT enhances the compositional property of the product-form models, i.e., we have shown that two product-form SPNs based on RCAT can be composed obtaining a new product-form model and that the new model still satisfies RCAT conditions.

Then, we have applied RCAT to study a set of GSPN models (with some restrictions) equivalent to the BCMP station types presented in Chapter 6. RCAT

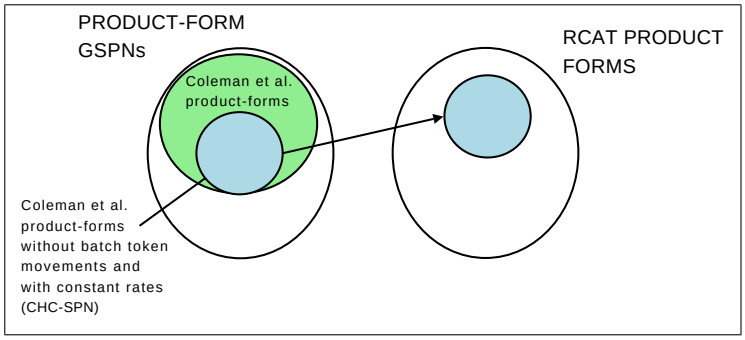

Figure C.6: Relations among (G)SPNs product-form stochastic models and RCAT.

composition for product-form models is very versatile. In fact, several product-form models have been proved to satisfy RCAT analysis (e.g., G-networks with several extensions [60]). RCAT is a powerful framework to study product-form models. We have shown that we can apply RCAT in order to derive new product-form GSPNs. This can be done by following two approaches. First, we can consider a specific model that arises from the analysis of a real system, e.g., in our example we proved a product-form solution for an interaction between two GSPN models, one representing a queueing station and the other one representing a communication channel with collisions. Another approach is considering a model that is known to be *not* in product-form and modifying its behavior in order to meet the RCAT conditions. In this latter case the modeler should carefully study the effects of the changes on the desired performance indices.

Using these results we have presented several examples of hybrid modeling that combines various stochastic models with product-form. The formalisms that are combined are queueing stations, G-queues, SPNs and GSPNs. The product-form is proved without the analysis of the global balance equations of the composed model. Each of these models has been mapped into a GSPN and hence we have been able to formally specify the semantic of the composed model.

Chapter 8 presents an algorithm that transforms a BCMP queueing network into a GSPN using the models presented in Chapter 6. The algorithm is based on a modular approach. In other words, every queueing discipline is transformed into a parameterizable GSPN block that has an input and an output interface. Such block interfaces make the modeling of the routing among the GSPN blocks independent of the internal definition of the block. As a consequence one can define GSPN blocks that are different from those defined in Chapter 6 and then embed them within a BCMP queueing network as long as the new block fulfils the $M \Rightarrow M$ property. The idea of using interfaces and modules perfectly matches with the recent developments of languages defined to describe Petri net models, such as the Petri Net Markup Language (PNML) [122].

Future works. Starting from the results presented in this thesis, further research can be done in several directions:

- We have applied RCAT to a subclass of Coleman, Henderson et al. stochastic Petri nets (CH-SPNs). It is likely that an extension of RCAT can be defined in order to deal with state dependent firing rates. In fact, for CH-SPN the transition firing rates have to be in the form (3.5) and the product-form solution is given by Equation (3.8). Note that function ϕ affects the steady state solution only as a multiplicative factor. In practice, one ignores the load dependent factor when studying the product-form conditions. As function ϕ does not affect the product-form conditions, we think that it is reasonable that RCAT could be extended in order to include this class of state dependent transition rates. Another possible extension of the SPN class to which RCAT can be applied is considering batch movements of tokens.

- Another promising research line is defining sufficient structural conditions for (G)SPNs that lead to RCAT or $M \Rightarrow M$ product-form models. This is interesting because it would be the base to develop a very general hybrid product-form analyzer that uses GSPNs as underlying formalism. In fact, by the results presented in this thesis, a modeler has to know that he is composing submodels that satisfy $M \Rightarrow M$ property or RCAT. We think that an interesting improvement of the proposed approach is to embed this information in the XML-based definition of the submodels. Using this information and by the inspection of the routing the analyzer can decide if the whole model is in product-form.

- We think that a special attention should be devoted to clearly identify and distinguish between the conditions for a product-form solution and the conditions required to define appropriate and possibly efficient algorithms to derive the performance measures of a product-form model. In fact, it is worthwhile recalling that product-form models define and allow for an efficient computation of the unnormalized steady state probabilities. However, this does not necessary leads to efficient solution algorithms. In particular, it can be necessary to build the whole set of reachable states of the model to derive the normalizing constant and a set of performance measures. This is a major limitation of the product-form SPN class, for which efficient solution algorithms such as MVA [111] and Convolution [39] can be applied only under further conditions. However, even in the BCMP queueing networks, that have been studied for many years and for which several solution algorithms have been defined, there are some models that, although in product-form, cannot be efficiently solved analytically. This especially happens for models with nodes with some state dependent service rate functions, such as those that depend on the state of a subset of places of the network. This problem is also evident for those product-form models that require the solution of nonlinear traffic equation systems, such as the G-Networks. We think that a strong effort should

be devoted to the research of algorithms that extend the class of product-form models that can be efficiently studied for practical purposes.

A

Proves of lemmas and theorems

In this section we present the proves of theorems and lemmas enunciated in this thesis. Hereafter, for the sake of helping the intuition, we use some of the queueing network concepts for the GSPN analysis. For example, we can say *waiting customers* to denote the tokens in places $P_1, \dots P_R$ for the GSPN-EXP model and similarly we use *customers being served* or *customer arrivals* or *free servers*. From our point of view this description should make the following proves easier to read. In the following we denote by $\mathbf{e}_i$ a vector whose components are all equal to 0 but the i-th which is 1.

A.1 Proof of Lemma 4

The proof is based on verifying that Equation 6.2 satisfies the GBEs of the stochastic process defined by GSPN-EXP model. As this stochastic process has infinite states we cluster them in classes that are characterized by four cases.

Case 1) Suppose that all the servers are busy and at least one customer is waiting in queue, so $m_{2R+1} = 0$ and $m_i > 0$ for some $i = 1, \dots, R$. Consider state $\mathbf{m} = (m_1, \dots, m_R, m_{R+1}, \dots, m_{2R}, 0)$. Clearly we have that $\sum_{i=R+1}^{2R} m_i = K$, that is, the number of servers. Consider the tangible markings from which $\mathbf{m}$ is reachable possibly through the firing of sequences of immediate and timed transitions and the corresponding transition rates:

- $\mathcal{M}_\alpha = \{\mathbf{m}' = \mathbf{m} - \mathbf{e}_i | m_i > 0\}$ with rate λ_i
- $\mathcal{M}_\beta = \{\mathbf{m}' = \mathbf{m} + \mathbf{e}_i | m_{R+i} > 0\}$ with rate $(m_{R+i})\mu p_i(\mathbf{m}')$
- $\mathcal{M}_\gamma = \{\mathbf{m}' = \mathbf{m} + \mathbf{e}_i - \mathbf{e}_{R+i} + \mathbf{e}_{R+j} | m_{R+i} > 0, i \neq j\}$ with rate $(m_{R+j} + 1)\mu p_i(\mathbf{m}')$.

The set of markings reachable from state $\mathbf{m}$ and the correspondent transition rates can be classified as follows:

- $\mathcal{M}_a = \{\mathbf{m}' = \mathbf{m} + \mathbf{e}_i\}$ with rate λ_i

- $\mathcal{M}_b = \{\mathbf{m}' = \mathbf{m} - \mathbf{e}_i | m_{R+i} > 0, m_i > 0\}$ with rate $(m_{R+i})\mu p_i(\mathbf{m})$

- $\mathcal{M}_c = \{\mathbf{m}' = \mathbf{m} - \mathbf{e}_i + \mathbf{e}_{R+i} - \mathbf{e}_{R+j} | m_{R+j} > 0, m_i > 0, i \neq j\}$ with rate $(m_{R+j})\mu p_i(\mathbf{m})$.

We have to prove that Equation (6.2) satisfies the GBEs. The proof algebraically verifies the following relation:

$$\begin{aligned} & \sum_{\mathbf{m}' \in M_\alpha} \pi(\mathbf{m}')\xi(\mathbf{m}', \mathbf{m}) = \pi(\mathbf{m}) \sum_{\mathbf{m}' \in \mathcal{M}_b \cup \mathcal{M}_c} \xi(\mathbf{m}, \mathbf{m}') \\ \wedge \quad & \sum_{\mathbf{m}' \in \mathcal{M}_\beta \cup \mathcal{M}_\gamma} \pi(\mathbf{m}')\xi(\mathbf{m}', \mathbf{m}) = \pi(\mathbf{m}) \sum_{\mathbf{m}' \in \mathcal{M}_\alpha} \xi(\mathbf{m}, \mathbf{m}') \\ \Longrightarrow \quad & \sum_{\mathbf{m}' \in \mathcal{M}_\alpha \cup \mathcal{M}_\beta \cup \mathcal{M}_\gamma} \pi(\mathbf{m}')\xi(\mathbf{m}', \mathbf{m}) = \pi(\mathbf{m}) \sum_{\mathbf{m}' \in \mathcal{M}_a \cup \mathcal{M}_b \cup \mathcal{M}_c} \xi(\mathbf{m}, \mathbf{m}'), \end{aligned}$$

where $\xi(\mathbf{a}, \mathbf{b})$ represents the transition rate from marking $\mathbf{a}$ to marking $\mathbf{b}$. The effective leaving rate from state $\mathbf{m}$ due to a job completion is:

$$\pi(\mathbf{m}) \Big[\sum_{m_{j+k} > 0} (m_{j+k})\mu \Big] = \pi(\mathbf{m})(K\mu).$$

The effective arrival rate to state $\mathbf{m}$ due to arrivals to the system is:

$$\begin{aligned} \sum_{\mathbf{m}' \in \mathcal{M}_\alpha} \pi(\mathbf{m}')\xi(\mathbf{m}', \mathbf{m}) &= \pi(\mathbf{m}) \Big[\sum_{i \in X} \Big(\frac{1}{\lambda_i} \frac{m_i}{\sum_{j=1}^{R} m_j} \mu \Big(\sum_{j=1}^{2R} m_j \Big) \lambda_i \Big) \Big] \\ &= \pi(\mathbf{m})(K\mu) \Big[\sum_{i \in X} \frac{m_i}{\sum_{j=1}^{R} m_j} \Big] = \pi(\mathbf{m})(K\mu), \end{aligned}$$

where $X = \{i | m_i > 0, 1 \leq i \leq R\}$ is the set of the indices corresponding to the states in which there is at least one customer in queue. Recalling that, by hypothesis, $\sum_{i=1}^{2R} m_i \geq K$ then:

$$\mu \Big(\sum_{i=1}^{2R} m_i \Big) = K\mu.$$

Consider $\mathbf{m}$ and set $Y = \{j | m_{R+j} > 0, 1 \leq j \leq R\}$. The effective leaving rate from state $\mathbf{m}$ due to customer arrivals is $\pi(\mathbf{m}) \sum_{i=1}^{R} \lambda_i$. The effective arrival rate to state

m is:

$$\begin{aligned}
& \pi(\mathbf{m})\Big[\sum_{j\in Y}\lambda_j\frac{(\sum_{i=1}^{R}m_i)+1}{m_j+1}\frac{1}{\mu((\sum_{i=1}^{2R}m_i)+1)}m_{R+j}\mu\frac{m_j+1}{\sum_{i=1}^{R}m_i+1} \\
+ & \sum_{j=1}^{R}\sum_{\substack{i\in Y\\ i\neq j}}\lambda_j\frac{m_{R+i}}{m_{R+j}+1}\frac{(\sum_{g=1}^{R}m_g)+1}{m_i+1}\frac{1}{\mu((\sum_{g=1}^{2R}m_g)+1)}(m_{R+j}+1)\mu \\
& \cdot\frac{m_i+1}{(\sum_{g=1}^{R}m_g)+1}\Big] \\
= & \pi(\mathbf{m})\Big[\sum_{j\in Y}\lambda_j\frac{1}{K\mu}m_{R+j}\mu+\sum_{j=1}^{R}\sum_{\substack{i\in Y\\ i\neq j}}\lambda_j m_{R+i}\frac{1}{K\mu}\mu\Big] \\
= & \pi(\mathbf{m})\Big[\sum_{j\in Y}\lambda_j\frac{m_{R+j}}{K}+\sum_{j=1}^{R}\sum_{\substack{i\in Y\\ i\neq j}}\lambda_j\frac{m_{R+i}}{K}\Big] \\
= & \pi(\mathbf{m})\Big[\sum_{i=1}^{R}\lambda_i\Big].
\end{aligned}$$

Case 2) Consider now the case in which all the servers are busy, but places $P_1,\ldots,P_R$ are empty. Consider a generic state $\mathbf{m}=(0,\ldots,0,m_{R+1},\ldots,m_{2R},0)$, i.e., all the servers are busy, but the queue is empty. The markings from which **m** is reachable are classified as:

- $\mathcal{M}_\alpha=\{\mathbf{m}'=\mathbf{m}-\mathbf{e}_{R+i}+\mathbf{e}_{2R+1}|m_{R+i}>0$ with rate λ_i
- $\mathcal{M}_\beta=\{\mathbf{m}'=\mathbf{m}+\mathbf{e}_i|m_{R+i}>0\}$ with rate $(m_{R+i})\mu p_i(\mathbf{m}')$
- $\mathcal{M}_\gamma=\{\mathbf{m}'=\mathbf{m}+\mathbf{e}_i-\mathbf{e}_{R+i}+\mathbf{e}_{R+j}|m_{R+i}>0, j\neq i\}$ with rate $(m_{R+j}+1)\mu p_i(\mathbf{m}')$

The markings reachable from **m** and their effective rates are exactly those described for case 1). Let Y be the set defined in case 1). Let us prove that the effective arrival rate to state **m** from marking in $\mathcal{M}_\alpha$ is equal to the effective leaving rate from state **m** due to a job completion which is $\pi(\mathbf{m})(K\mu)$.

$$\pi(\mathbf{m})\Big[\sum_{i\in Y}(\frac{1}{\lambda_i}\frac{m_{R+i}}{K}\mu(K)\lambda_i)\Big]=\pi(\mathbf{m})\Big[\sum_{i\in Y}m_{R+i}\mu\Big]=\pi(\mathbf{m})(K\mu).$$

Let us prove that the effective arrival rate from the markings in $\mathbf{M}_\beta \cup \mathcal{M}_\gamma$ is equal to the effective leaving rate from $\mathbf{m}$.

$$\pi(\mathbf{m})\Big[\sum_{j\in Y}(\lambda_j \frac{1}{\mu(K+1)} m_{R+j}\mu) + \sum_{j=1}^{R}\sum_{\substack{i\in Y\\ i\neq j}} \lambda_j \frac{m_{R+i}}{m_{R+j}+1}\frac{1}{\mu(K+1)}(m_{R+j}+1)\mu\Big]$$
$$= \pi(\mathbf{m})\Big[\sum_{j\in Y}\lambda_j\frac{m_{R+j}}{K} + \sum_{j=1}^{R}\sum_{\substack{i\in Y\\ i\neq j}}\lambda_j\frac{m_{R+i}}{K}\Big] = \pi(\mathbf{m})\Big[\sum_{j=1}^{R}\lambda_j\Big].$$

Case 3) Assume that there are not tokens in places $P_1, \ldots, P_R$ and that at least one server is free and at least one is busy, (that is $0 < m_{2R+1} < K$). The effective leaving rate from $\mathbf{m}$ is simply $\pi(\mathbf{m})[\mu \sum_{j=1}^{R} m_{R+j} + sum_{j=1}^{R}\lambda_j]$. The states from which $\mathbf{m}$ is reachable and the corresponding rates are:

- $\mathcal{M}_\alpha = \{\mathbf{m}' = \mathbf{m} - \mathbf{e}_{R+i} + \mathbf{e}_{2R+1} | m_{R+i} > 0\}$ with rate λ_i
- $\mathcal{M}_\beta = \{\mathbf{m}' = \mathbf{m} + \mathbf{e}_{R+i} - \mathbf{e}_{2R+1}\}$ with rate $(m_{R+i}+1)\mu$

Let Y be defined as in case 1). We now prove that the effective arrival rate to state $\mathbf{m}$ is equal to the effective leaving rate from state $\mathbf{m}$.

$$\begin{aligned}
& \pi(\mathbf{m})\Big[\sum_{i\in Y}\frac{1}{\lambda_i}\frac{m_{R+i}}{\sum_{j=1}^{R} m_{R+j}}\mu(\sum_{j=1}^{R} m_{R+j})\lambda_i \\
+ & \sum_{i=1}^{R}\lambda_i\frac{\sum_{j=1}^{R} m_{R+j}+1}{m_{R+i}+1}\frac{1}{\mu(\sum_{j=1}^{R} m_{R+j}+1)}(m_{R+i}+1)\mu\Big] \\
= & \pi(\mathbf{m})\Big[\sum_{i\in Y}\frac{1}{\lambda_i}\frac{m_{R+i}}{\sum_{j=1}^{R} m_{R+j}}(\sum_{j=1}^{R} m_{R+j})\mu\lambda_i \\
+ & \sum_{i=1}^{R}\lambda_i\frac{\sum_{j=1}^{R} m_{R+j}+1}{m_{R+i}+1}\frac{1}{(\sum_{j=1}^{R} m_{R+j}+1)\mu}(m_{R+i}+1)\mu\Big] \\
= & \pi(\mathbf{m})\Big[(\sum_{j=1}^{R} m_{R+j})\mu + \sum_{i=1}^{R}\lambda_i\Big].
\end{aligned}$$

Case 4) Assume that the system is empty, i.e., $m_{2R+1} = K$. This case is trivial. In fact the effective leaving rate from $\mathbf{m}$ is $\pi(\mathbf{m})[\sum_{i=1}^{R}\lambda_i]$. The effective arrival rate to $\mathbf{m}$ can just be due to a job completion and it is easy to show that is equal to the leaving rate. ♠

A.2 Proof of Theorem 5

In order to derive Equation (6.3) we prove that:

$$\pi_a(\mathbf{n}) = \sum_{\substack{\mathbf{m}|m_i+m_{R+i}=n_i \\ 1\le i\le R}} \pi(\mathbf{m}), \tag{A.1}$$

for $\mathbf{n} \in \mathbb{N}^R$ and $\mathbf{m}$ in the reachability set of model GSPN-EXP. Consider the two following cases: case 1) $\sum_{i=1}^{R} n_i > K$ and case 2) $\sum_{i=1}^{R} n_i < K$.

Case 1) $\sum_{i=1}^{R} n_i \ge K$. Consider any possible combination of j_i with $1 \le i \le r$ and $0 \le j_i \le n_i$. Then, by Formula (6.2), the right-hand side of Equation (A.1) can be written as:

$$\begin{aligned}
&\sum_{\substack{j_1+\ldots,+j_R=K \\ j_i\le n_i}} \pi(n_1-j_1, n_2-j_2, \ldots, n_R-j_R, j_1, \ldots, j_R, 0) \\
= \;& \pi_0 \prod_{j=1}^{R} \lambda_j^{n_j} \prod_{j=1}^{\sum_{i=1}^{R} n_j} \frac{1}{\mu(j)} \sum_{\substack{j_1+\ldots+j_r=K \\ j_i\le n_i}} \frac{K!}{\prod_{i=1}^{R} j_i!} \frac{(\sum_{i=1}^{R} n_i - K)!}{\prod_{i=1}^{R}(n_i-j_i)!} \\
= \;& \pi_0 \prod_{j=1}^{R} \lambda_j^{n_j} \prod_{j=1}^{\sum_{i=1}^{R} n_j} \frac{1}{\mu(j)} \frac{(\sum_{i=1}^{R} n_i - K)!}{\prod_{i=1}^{R} n_i!} K! \sum_{\substack{j_1+\ldots+j_R=K \\ j_i\le n_i}} \frac{\prod_{i=1}^{R} n_i!}{\prod_{i=1}^{R}(n_i-j_i)!} \frac{1}{\prod_{i=1}^{R} j_i!} \\
= \;& \pi_0 \prod_{j=1}^{R} \lambda_j^{n_j} \prod_{j=1}^{\sum_{i=1}^{R} n_i} \frac{1}{\mu(j)} \frac{(\sum_{i=1}^{R} n_i - K)!}{\prod_{i=1}^{R} n_i} K! \sum_{\substack{j_1+\ldots+j_R=K \\ j_i\le n_i}} \prod_{i=1}^{R} \binom{n_i}{j_i},
\end{aligned}$$

where the last sum is a Vandermonde convolution, thus we can write:

$$\begin{aligned}
& \pi_0 \prod_{j=1}^{R} \lambda_j^{n_j} \prod_{j=1}^{\sum_{i=1}^{R} n_i} \frac{1}{\mu(j)} \frac{(\sum_{i=1}^{R} n_i - K)!}{\prod_{i=1}^{R} n_i!} K! \binom{\sum_{i=1}^{R} n_i}{K} \\
= \;& \pi_0 \prod_{j=1}^{R} \lambda_j^{n_j} \prod_{j=1}^{\sum_{i=1}^{R} n_i} \frac{1}{\mu(j)} \frac{(\sum_{i=1}^{R} n_i)!}{\prod_{i=1}^{R} n_i!},
\end{aligned}$$

that is Formula (6.3).

Case 2) $\sum_{i=1}^{R} n_i < K$, that corresponds to the behavior of the queueing system where all the customers are being served and in GSPN-EXP every place P_i with $1 \le i \le R$ is empty. Note that $n_i = m_{R+i}$ so, by Equation (4) we can write:

$$\pi(0, \ldots, 0, m_{R+1}, \ldots, m_{2R}, l) = \pi_0 \prod_{i=1}^{R} \lambda_i^{n_i} \prod_{i=1}^{\sum_{i=1}^{R} n_i} \frac{1}{\mu(i)} \frac{(\sum_{i=1}^{R} n_i)!}{\prod_{i=1}^{R} n_i!},$$

that yields Formula (6.3) as required. ♠

A.3 Proof of Lemma 5

The proof is based on verifying that formula (6.4) satisfies the set of global balance equations (GBEs) of the Markov process associated with the model. We work by cases: first we consider 1) the case of $m_{R+1} = 0$ and $m_{R+r,l} > 0$ for some $r = 1, \ldots, R$ and $1 \leq l \leq L_r$, then 2)the case $m_{R+1} = 0$ and $m_{R+r,l} = 0$ for all $r = 1, \ldots, R$ and $1 \leq l \leq L_r$, then 3) the case $1 \leq m_{R+1} < k$, and finally 4) the case $m_{R+1} = k$. The $M \Rightarrow M$ property, (that is Equation (7.4)) is immediately verified by the chosen partial balance.

Case 1) Consider tangible state $\mathbf{m}$ where $m_{R+1} = 0$ and $m_{R+r,l} > 0$ for some $r = 1, \ldots, R$. State $\mathbf{m}$ can be reached from the following set of states:

- $\mathcal{A} = \{\mathbf{m}' : \mathbf{m}' = \mathbf{m} + \mathbf{e_{r,l-1}} - \mathbf{e_{r,l}},\ m_{r,l} > 0,\ 1 \leq r \leq R,\ 1 < l \leq L_r\}$ with rate $\mu_{r,l-1} a_{r,l-1}(m_{r,l-1} + 1)$.
- $\mathcal{B} = \{\mathbf{m}' : \mathbf{m}' = \mathbf{m} - \mathbf{e_{s,1}} + \mathbf{e_{r,l}} - \mathbf{e_{R+r,l}},\ 1 \leq r, s \leq R,\ m_{s,1} > 0,\ m_{R+r,l} > 0\}$ with rate $\lambda_s(m_{r,l} + 1)/(\sum_{r'=1}^{R} \sum_{l'=1}^{L_{r'}} m_{r',l'})$.
- $\mathcal{C} = \{\mathbf{m}' = \mathbf{m} + \mathbf{e_{s,\ell}} + \mathbf{e_{R+r,l}} - \mathbf{e_{r,l}},\ 1 \leq r, s \leq R, 1 \leq l \leq L_r, m_{r,l} > 0$ with rate $\mu_{s,m} b_{s,m}(m_{s,m} + 1)(m_{R+r,l} + 1)/(1 + \sum_{r'=1}^{R} \sum_{l'=1}^{L_{r'}} m_{R+r',l'})$.

The leaving rate from state $\mathbf{m}$ is:

$$\sum_{r=1}^{R} \lambda_r + \sum_{r=1}^{R} \sum_{l=1}^{L_r} m_{r,l} \mu_{r,l}, \tag{A.2}$$

so we have to prove that:

$$\sum_{\mathbf{m}' \in \mathcal{A} \cup \mathcal{B} \cup \mathcal{C}} \pi(\mathbf{m}') \xi(\mathbf{m}' \to \mathbf{m}) = \pi(\mathbf{m}) \Big[\sum_{r=1}^{R} \lambda_r + \sum_{r=1}^{R} \sum_{l=1}^{L_r} m_{r,l} \mu_{r,l} \Big], \tag{A.3}$$

where $\xi(\mathbf{m}' \to \mathbf{m})$ denotes the transition rate from state $\mathbf{m}'$ to state $\mathbf{m}$. In the following we write $\pi(\mathbf{m}')$ as product of $\pi(\mathbf{m})$ and an opportune factor.

We verify the GBEs considering the effective arrival rates from states belonging to different sets $\mathcal{A}, \mathcal{B}, \mathcal{C}$ separately. Let $\mathbf{m}' \in \mathcal{A}$, thus $m_{r,l} > 0$ and $l > 1$, then we can write the effective arrival rate to state $\mathbf{m}$ as follows:

$$\pi(\mathbf{m}) \Big[\frac{m_{r,l}}{1 + m_{r,l-1}} \frac{\mu_{r,l}}{A_{r,l}} \frac{A_{r,l-1}}{\mu_{r,l-1}} \Big] \mu_{r,l-1} a_{r,l-1}(1 + m_{r,l-1}) = \pi(\mathbf{m})[m_{r,l} \mu_{r,l}]. \tag{A.4}$$

Let $\mathbf{m}' \in \mathcal{B}$ thus $m_{s,1} > 0$, and let us define $Y = \{(r,l)|1 \leq r \leq R, 1 \leq l \leq L_r, m_{R+e,l} > 0\}$. Then we can write the effective arrival rate to state $\mathbf{m}$ as follows:

$$\begin{aligned}
&\pi(\mathbf{m})\Big[\sum_{(r,l)\in Y}\frac{1}{\lambda_s}\frac{m_{s,1}}{1+m_{r,l}}\frac{m_{R+r,l}}{\sum_{r'=1}^{R}\sum_{l'=L_{r'}}m_{R+r',l'}}\Big]\Big[\frac{\mu_{s,1}}{A_{s,1}}k\lambda_s\frac{1+m_{r,l}}{\sum_{r'=1}^{R}\sum_{l'=1}^{L_{r'}}m_{r',l'}}\Big]\\
&=\pi(\mathbf{m})\Big[m_{s,1}\mu_{s,1}\frac{1}{\sum_{r'=1}^{R}\sum_{l'=1}^{L_{r'}}m_{R+r',l'}}\sum_{(r,l)\in Y}m_{R+r}\Big]\\
&=\pi(\mathbf{m})[m_{s,1}\mu_{s,1}]. \qquad (A.5)
\end{aligned}$$

Consider now $\mathbf{m}' \in \mathcal{C}$. The effective arrival rate to state $\mathbf{m}$ can be written as follows:

$$\begin{aligned}
&\pi(\mathbf{m})\Big[\sum_{\ell=1}^{L_s}\sum_{(r,l)\in Y}\lambda_s\frac{m_{r,l}}{m_{s,\ell}+1}\frac{1+\sum_{r'=1}^{R}\sum_{l'=1}^{L_{r'}}m_{R+r',l'}}{1+m_{R+r,l}}\\
&\frac{A_{s,\ell}}{\mu_{s,\ell}}\frac{1}{k}\mu_{s,\ell}b_{s,\ell}(m_{s,\ell}+1)\frac{1+m_{R+r,l}}{1+\sum_{r'=1}^{R}\sum_{l'=1}^{L_{r'}}m_{R+r',l'}}\Big]\\
&=\pi(\mathbf{m})[\lambda_s\sum_{\ell=1}^{L_s}A_{s,\ell}b_{s,\ell}\sum_{(r,l)\in Y}\frac{m_{r,l}}{k}\Big]\\
&=\pi(\mathbf{m})\lambda_s. \qquad (A.6)
\end{aligned}$$

Note that summing over all the possible $\mathbf{m}' \in \mathcal{A} \cup \mathcal{B} \cup \mathcal{C}$ equations (A.4), (A.5), (A.6) we obtain the total arrival rate to state $\mathbf{m}$ which equates the effective leaving rate from state $\mathbf{m}$ given by expression (A.2).

Case 2) Consider the tangible state $\mathbf{m}$ where $m_{R+1} = 0$ and $m_{R+r,l} = 0$ for all $r = 1\ldots, R$ and $1 \leq l \leq L_r$. The only difference with respect to case (a) is that set $\mathcal{B}$ has to be redefined as: $\mathcal{B} = \{\mathbf{m}'|\mathbf{m}' = \mathbf{m} - \mathbf{e_{s,1}} + \mathbf{e_{R+1}}, 1 \leq r, s \leq R, m_{s,1} > 0\}$. Hence the effective arrival rate to state $\mathbf{m}$ from a state in $\mathcal{B}$ can be written as follows:

$$\pi(\mathbf{m})[\frac{1}{\lambda_s}\frac{m_{s,1}}{\sum_{r'=1}^{R}\sum_{l'=1}^{L_{r'}}m_{r',l'}}\mu_{s,1}k\lambda_s] = \pi(\mathbf{m})[\mu_{s,1}m_{s,1}]. \qquad (A.7)$$

Noting that the right hand side of equation (A.7) is equal to the right hand side of equation (A.5) and summing over all the possible states $\mathbf{m}'$ we verify the GBEs.

Case 3) Consider the tangible state $\mathbf{m}$ where $1 \leq \sum_{r=1}^{R}\sum_{l=1}^{L_r}m_{r,l} < k$ thus we have $\sum_{r=1}^{R}\sum_{l=1}^{L_r}m_{R+r,l} = 0$. We partition the set of states from which state $\mathbf{m}$ can be reached as follows:

- $\mathcal{A}$ is defined as done in case (a).
- $\mathcal{B} = \{\mathbf{m}'|\mathbf{m}' = \mathbf{m} - \mathbf{e_{s,1}}, 1 \leq r, s \leq R, m_{s,1} > 0\}$ with rate λ_s.

- $\mathcal{C} = \{\mathbf{m}' | \mathbf{m}' = \mathbf{m} + \mathbf{e_{s,\ell}} - \mathbf{e_{R+1}}, 1 \leq s \leq R, 1 \leq \ell \leq L_s\}$ with rate $\mu_{s,\ell} b_{s,\ell}(m_{s,\ell} + 1)$.

The calculations for states in $\mathcal{A} \cup \mathcal{B}$ are the same as those of the previous case. Consider $\mathbf{m}' \in \mathcal{C}$, the effective arrival rate to $\mathbf{m}$ can be written as follows:

$$\pi(\mathbf{m}) \Big[\sum_{\ell=1}^{L_s} \lambda_s \frac{1 + \sum_{r'=1}^{R} \sum_{l'=1}^{L_{r'}} m_{r',l'}}{1 + m_{s,\ell}} \frac{A_{s,\ell}}{\mu_{s,\ell}} \mu_{s,\ell} b_{s,\ell} \cdot (m_{s,\ell} + 1) \frac{1}{1 + \sum_{r'=1}^{R} \sum_{l'=1}^{L_{r'}} m_{r',l'}} \Big] = \pi(\mathbf{m}) \lambda_s. \quad \text{(A.8)}$$

Noting that the right hand side of equation (A.8) is equal to the right hand side of equation (A.6) and summing over all $\mathbf{m}' \in \mathcal{A} \cup \mathcal{B} \cup \mathcal{C}$ we verify the the GBEs.

Case 4) . The proof is trivial. ♠

A.4 Proof of Lemma 6

In order to prove Lemma 6 we calculate the aggregation as follows. Given $\mathbf{n}$, the aggregation is obtained in $2R$ steps. We first sum over $\mathbf{m}$ such that $m_1 = n_1$, that is, we aggregate class 1 customers in service, obtaining an intermediate state $\mathbf{m}^{\alpha_1}$, where $\pi^{\alpha_1}(\mathbf{m}^{\alpha_1})$ is given by the sum:

$$\pi^{\alpha_1}(\mathbf{m}^{\alpha_1}) = \sum_{\mathbf{m}: m_1 = n_1} \pi(\mathbf{m}). \quad \text{(A.9)}$$

The subsequent step aggregates class 1 customers in the queue, obtaining the intermediate state $\mathbf{m}^{\beta_1}$, whose stationary probability is given by:

$$\pi^{\beta_1}(\mathbf{m}^{\beta_1}) = \sum_{\mathbf{m}^{\alpha_1}: m^{\alpha_1}_{R+1} = n_{R+1}} \pi^{\alpha_1}(\mathbf{m}^{\alpha_1}). \quad \text{(A.10)}$$

Proceeding with the aggregation for all classes $1 \ldots R$, we have that: $\mathbf{m}^{\beta_R} = \mathbf{n}$ and $\pi^{\beta_R}(\mathbf{m}^{\beta_R}) = \pi_a(\mathbf{n})$.

In order to simplify the notation, in the following we write $\sum_{\alpha_r}$ and $\sum_{\beta_r}$ to denote the sums which give intermediate states $\mathbf{m}^{\alpha_r}$ and $\mathbf{m}^{\beta_r}$ respectively.

Hence we can simplify what we have to prove as follows:

$$\sum_{\beta_R} \sum_{\alpha_R} \cdots \sum_{\beta_1} \sum_{\alpha_1} \Big[\frac{(\sum_{r=1}^{R} m_r)!}{\prod_{r=1}^{R} \prod_{\ell=1}^{L_r} m_{r,\ell}!} \frac{(\sum_{r=1}^{R} m_{R+r})!}{\prod_{r=1}^{R} \prod_{\ell=1}^{L_r} m_{R+r,\ell}!} \Big] \cdot \Big[\prod_{r=1}^{R} \prod_{\ell=1}^{L_R} \Big(\frac{A_{r,\ell}}{\mu_{r,\ell}} \Big)^{m_{R+r,\ell} + m_{r,\ell}} \Big]$$
$$= \frac{(\sum_{r=1}^{R} n_r)!}{\prod_{r=1}^{R} n_r!} \frac{(\sum_{r=1}^{R} n_{R+r})!}{\prod_{r=1}^{R} n_{R+r}!} \prod_{r=1}^{R} \Big(\frac{1}{\mu_r} \Big)^{n_r + n_{R+r}}. \quad \text{(A.11)}$$

Let us consider the inner sum of the left hand side of equation (A.11):

$$\sum_{\alpha_1}\Big[\frac{(\sum_{r=1}^{R}m_r)!}{\prod_{r=1}^{R}\prod_{\ell=1}^{L_r}m_{r,\ell}!}\frac{(\sum_{r=1}^{R}m_{R+r})!}{\prod_{r=1}^{R}\prod_{\ell=1}^{L_R}m_{R+r,\ell}!}\Big[\prod_{r=1}^{R}\prod_{\ell=1}^{L_r}\Big(\frac{A_{r,\ell}}{\mu_{r,\ell}}\Big)^{m_{R+r,\ell}+m_{r,\ell}}\Big]$$

$$=\frac{(\sum_{r=1}^{R}m_r)!}{\prod_{r=2}^{R}\prod_{\ell=1}^{L_r}m_{r,\ell}!}\frac{(\sum_{r=1}^{R}m_{R+r})!}{\prod_{r=1}^{R}\prod_{\ell=1}^{L_R}m_{R+r,\ell}!}\prod_{\ell=1}^{L_1}\Big(\frac{A_{1,l}}{\mu_{1,\ell}}\Big)^{m_{R+1,\ell}}\prod_{r=2}^{R}\prod_{\ell=1}^{L_r}\Big(\frac{A_{r,\ell}}{\mu_{r,\ell}}\Big)^{m_{R+r,\ell}+m_{r,\ell}}$$

$$\cdot\sum_{\alpha_1}\Big[\frac{1}{\prod_{\ell=1}^{L_1}m_{1,\ell}!}\prod_{\ell=1}^{L_1}\Big(\frac{A_{1,\ell}}{\mu_{1,\ell}}\Big)^{m_{1,\ell}}\Big]$$

$$=\frac{(\sum_{r=1}^{R}m_r)!}{\prod_{r=2}^{R}\prod_{\ell=1}^{L_r}m_{r,\ell}!}\frac{(\sum_{r=1}^{R}m_{R+r})!}{\prod_{r=1}^{R}\prod_{\ell=1}^{L_R}m_{R+r,\ell}!}\prod_{\ell=1}^{L_1}\Big(\frac{A_{1,l}}{\mu_{1,\ell}}\Big)^{m_{R+1,\ell}}\prod_{r=2}^{R}\prod_{\ell=1}^{L_r}\Big(\frac{A_{r,\ell}}{\mu_{r,\ell}}\Big)^{m_{R+r,\ell}+m_{r,\ell}}\frac{1}{m_1!}$$

$$\cdot\sum_{\alpha_1}\Big[\binom{m_1}{m_{1,1}\ldots m_{1,L_1}}\prod_{\ell=1}^{L_1}\Big(\frac{A_{1,\ell}}{\mu_{1,\ell}}\Big)^{m_{1,\ell}}\Big]$$

that by applying the binomial theorem, and noting that $\sum_{\ell=1}^{L_1}(\frac{A_{1,\ell}}{\mu_{1,\ell}})$ is the mean of the Coxian distributed service time, i.e., $1/\mu_1$, the previous result can be rewritten as follows:

$$\begin{aligned}&=\Big[\frac{(\sum_{r=1}^{R}m_r)!}{m_1!\prod_{r=2}^{R}\prod_{\ell=1}^{L_r}m_{r,\ell}!}\frac{(\sum_{r=1}^{R}m_{R+r})!}{\prod_{r=1}^{R}\prod_{\ell=1}^{L_R}m_{R+r,\ell}!}\Big]\\&\cdot\Big[\prod_{r=2}^{R}\prod_{\ell=1}^{L_r}\Big(\frac{A_{r,\ell}}{\mu_{r,\ell}}\Big)^{m_{R+r,\ell}+m_{r,\ell}}\prod_{\ell=1}^{L_1}\Big(\frac{A_{1,\ell}}{\mu_{1,\ell}}\Big)^{m_{R+1,\ell}}\\&\cdot\Big(\frac{1}{\mu_1}\Big)^{m_1}\Big]\end{aligned}\tag{A.12}$$

Summing expression (A.12) to obtain $\mathbf{m}^{\beta_1}$, by similar calculations, gives:

$$\begin{aligned}&\Big[\frac{(\sum_{r=1}^{R}m_r)!}{m_1!\prod_{r=2}^{R}\prod_{\ell=1}^{L_r}m_{r,\ell}!}\frac{(\sum_{r=1}^{R}m_{R+r})!}{m_{R+1}!\prod_{r=2}^{R}\prod_{\ell=1}^{L_R}m_{R+r,\ell}!}\Big]\\&\cdot\Big[\prod_{r=2}^{R}\prod_{\ell=1}^{L_r}\Big(\frac{A_{r,\ell}}{\mu_{r,\ell}}\Big)^{m_{R+r,\ell}+m_{r,\ell}}\Big(\frac{1}{\mu_1}\Big)^{m_1+m_{R+1}}\Big]\end{aligned}\tag{A.13}$$

Expression (A.13) can be similarly summed over α_2 and then β_2, and so on. Noting that $m_r = n_r$ and $m_{R+r} = n_{R+r}$ for $r = 1,\ldots,R$ we have proved equation (A.11) and the lemma. ♠

A.5 Proof of Lemma 7

Let us consider model GSPN-EXP. Let $\Theta_i = \{(\mathbf{m}',\mathbf{m}) : |\mathbf{m}'|_i = |\mathbf{m}|_i + 1\}$ for $i = 1,\ldots,R$. Intuitively $|\mathbf{m}|_i$ is the number of customers of class i for state $\mathbf{m}$.

In order to verify Equation (7.4) let us consider a generic tangible marking $\mathbf{m}$. We consider two cases: 1) $m_{2R+1} = 0$ and 2) $m_{2R+1} > 0$ that correspond to the situation of all the servers busy and at least one server free, respectively.

Case 1) Let $\mathbf{m}$ be a reachable tangible state with $m_{2R+1} = 0$, then:

$$\Theta_i(\cdot, \mathbf{m}) = \{\mathbf{m}' : \mathbf{m}' = \mathbf{m} + \mathbf{e}_{R+i} + \mathbf{e}_j - \mathbf{e}_{R+j}, m_{R+j} > 0, j \neq i\} \\ \cup \{\mathbf{m}' | \mathbf{m}' = \mathbf{m} + \mathbf{e_i}, m_{R+i} > 0\}, 1 \leq i, j \leq R.$$

Let $\mathrm{sgn}(m_i)$ be the indicator function defined as follows: $\mathrm{sgn}(m_i) = 1$ if $m_i > 0$ and $\mathrm{sgn}(m_i) = 0$ otherwise. Let $Y = \{j : m_{R+j} > 0, 1 \leq j \leq R\}$. Then the left hand side of Equation 7.4 can be rewritten as follows:

$$\begin{aligned} \pi(\mathbf{m}) = [&\sum_{\substack{j \in Y \\ j \neq i}} \lambda_i \frac{1 + \sum_{a=1}^{R} m_a}{m_j + 1} \frac{m_{R+j}}{m_{R+i} + 1} 1 K\mu (m_{r+i} + 1)\mu \frac{m_j + 1}{1 + \sum_{a=1}^{R} m_a} \\ + \quad &\mathrm{sgn}(m_i)\lambda_i \frac{1 + \sum_{a=1}^{R} m_a}{m_i + 1} \cdot \frac{1}{K\mu} m_{R+i}\mu \frac{m_i + 1}{1 + \sum_{a=1}^{R} m_a}] \\ = \quad &\pi(\mathbf{m}) \Big[\lambda_i \Big(\sum_{\substack{j \in Y \\ j \neq i}} \frac{m_{R+j}}{K} + \frac{m_{R+i}}{K} \Big)\Big] = \pi(\mathbf{m})\lambda_i, \end{aligned}$$

which gives the right-hand side of Equation (7.4).

Case 2) Let $\mathbf{m}$ be a reachable tangible state with $m_{2R+1} > 0$. Then $\Theta_i(\cdot, \mathbf{m}) = \{\mathbf{m} + \mathbf{e}_{R+i} - \mathbf{e}_{2R+1}\}$. Thus Equation 7.4 holds, in fact:

$$\pi(\mathbf{m}) \Big[\lambda_i \frac{1 + \sum_{a=1}^{R} m_{R+a}}{m_{R+i} + 1} \frac{1}{(1 + \sum_{a=1}^{R} m_{R+a})\mu} (1 + m_{R+i})\mu\Big] = \pi(\mathbf{m})\lambda_i.$$

This proves that the traffic processes associated with Θ_i, $1 \leq i \leq R$, are pointwise independent Poisson processes, i.e., the departure processes for each class of customers are independent Poisson processes under independent Poisson arrivals.

A.6 Proof of Theorem 7

In order to prove Theorem 7 we introduce some definitions related to the GSPN CTMC. Then we prove that the stationary distribution is uniquely determined if it has to satisfy the $M \Rightarrow M$ property.

The following definitions introduce some useful notions which will be used in the following proofs. First of all we call GSPN-EXP-1 a model GSPN-EXP with one server and a general weight function for the immediate transitions:

$$w(t_r, \mathbf{m}) = w_r(\mathbf{m}),$$

where w_r is a function such that $w_r(\mathbf{m}) = 0$ if and only if $m_r = 0$.

Definition 10 *Let* $\mathbf{m}$ *and* $\mathbf{m}'$ *be two tangible states of GSPN-EXP-1' model. We say that* $\mathbf{m} \leq \mathbf{m}'$ *if for all* $r = 1, \ldots, R$ *we have that* $m_r + m_{R+r} \leq m'_r + m'_{R+r}$. *In this case we say that dist*$(\mathbf{m}', \mathbf{m}) = \sum_{r=1}^{2R} m'_r - \sum_{r=1}^{2R} m_r$.

Note that relation $\leq$ induces a partial order of states. Intuitively we use $\mathbf{m} \leq \mathbf{m}'$ if $\mathbf{m}$ has less or the same number of customers of $\mathbf{m}'$ and dist$(\mathbf{m}', \mathbf{m})$ represents the number of customers in $\mathbf{m}'$ more than in $\mathbf{m}$.

Definition 11 (MM-step and MM-path) *Let* $\mathbf{m}$ *be a tangible state. Then we define a MM-step as:*

$$\begin{aligned} (\mathbf{m} + \mathbf{e}_{R+r'} + \mathbf{e}_r - \mathbf{e}_{R+r})^{(r')} &\xrightarrow{r} \mathbf{m}^{(r)} \quad \text{if} \quad \mathbf{m} = \mathbf{m}^{(r)} \\ (\mathbf{m} + \mathbf{e}_{R+r'} - \mathbf{m}_{2R+1})^{(r')} &\xrightarrow{\epsilon} \mathbf{m}_0 \quad \text{if} \quad \mathbf{m} = \mathbf{m}_0, \end{aligned}$$

with $1 \leq r, r' \leq R$. *A MM-path is a sequence of MM-steps.*

Intuitively if we have a step $\mathbf{m}_1^{(r')} \xrightarrow{r} \mathbf{m}_2^{(r)}$, we can say that state $\mathbf{m}_2^{(r)}$ can be reached by $\mathbf{m}_1^{(r')}$ by a class r' job completion, and a selection of a job of class r for the service. An MM-path can be seen as the vector of the labels of the arrows and the initial state.

Example 13 (MM-path example) *Consider a GSPN-EXP-1' model with two classes of customers, and the state* $\mathbf{m}^{(2)} = (2, 1, 0, 1, 0)$, *a possible MM-Path* α *from* $\mathbf{m}^{(2)}$ *to* $\mathbf{m}_0$ *is:*

$$\alpha : (2, 1, 0, 1, 0) \xrightarrow{1} (1, 1, 1, 0, 0) \xrightarrow{2} (1, 0, 0, 1, 0) \xrightarrow{1} (0, 0, 1, 0, 0) \xrightarrow{\epsilon} \mathbf{m}_0.$$

In this case $\alpha = (2, 1, 0, 1, 0), (1, 2, 1, \epsilon)$. *In general, given a state* $\mathbf{m}$, *the number of MM-pathes from* $\mathbf{m}$ *to* $\mathbf{m}_0$ *is given by the multinomial coefficient* $\binom{\sum_{a=1}^{R} m_a}{m_1, \ldots, m_R}$.

Definition 12 (Function Ψ) *Given two states* $\mathbf{m}_A$ *and* $\mathbf{m}_B$ *with* $\mathbf{m}_A < \mathbf{m}_B$ *and an MM-path* α *from* $\mathbf{m}_B$ *to* $\mathbf{m}_A$, *we define the function* Ψ *on* α *as follows:*

$$\Psi(\alpha) = \begin{cases} 1 & \text{if } \alpha = \mathbf{m}_B, () \vee \alpha = \mathbf{m}_B, (\epsilon) \\ \Psi(\alpha) = \frac{\sum_{a=1}^{R} w_a(\mathbf{m}_B)}{w_r(\mathbf{m}_B)} \Psi(\beta) & \text{otherwise}, \end{cases}$$

where β *is the MM-path* α *with the first MM-step removed.*

The following lemmas state that if a GSPN-EXP-1 has a $M \Rightarrow M$ product-form then function Ψ is uniquely determined. In the following we use this result to prove that also the weight functions w_r are in the form $w_r = km_r$ with k an arbitrary positive constant.

Lemma 11 *If model GSPN-EXP-1 satisfies the $M \Rightarrow M$ property, then:*

- *if* $\mathbf{m}_1^{(r)} \xrightarrow{s} \mathbf{m}_2^{(s)}$ *we have that:*

$$\frac{\pi(\mathbf{m}_1^{(r)})}{\pi(\mathbf{m}_2^{(s)})} = \frac{\lambda_r}{\mu} \frac{\sum_{a=1}^{R} w_a(\mathbf{m}_1^{(r)})}{w_s(\mathbf{m}_1^{(r)})}.$$

- *if* $\mathbf{m}_1^{(r)} \xrightarrow{\epsilon} \mathbf{m}_0$ *we have that:*

$$\frac{\pi(\mathbf{m}_1^{(r)})}{\pi(\mathbf{m}_0)} = \frac{\lambda_r}{\mu}.$$

Proof 2 *The proof is trivial after noting that if* $\mathbf{m}_1^{(r)} \xrightarrow{s} \mathbf{m}_2^{(s)}$, *state* $\mathbf{m}_1^{(r)}$ *is the only state with a customer of class r added to state* $\mathbf{m}_2^{(s)}$ *from which* $\mathbf{m}_2^{(s)}$ *is reachable. Writing $M \Rightarrow M$ equation (7.4) for state* $\gamma = \mathbf{m}_2^{(s)}$ *the result follows immediately by noting that* $\gamma^{r+} = \{\mathbf{m}_1^{(r)}\}$ *and* $q_{\eta\to\gamma} = \mu \frac{w_s(\mathbf{m}_1^{(r)})}{\sum_{a=1}^{R} w_a(\mathbf{m}_1^{(r)})}$. *The second relation can be derived in a similar way.*

The following lemma is crucial for the proof. Let α be a MM-path from $\mathbf{m}^{(r)}$ to $\mathbf{m}_0$. It basically states that if a GSPN-EXP-1 model holds $M \Rightarrow M$ property then the values assumed by function Ψ must not depend on the whole path α but only on the first state of the path $\mathbf{m}^{(r)}$.

Lemma 12 *System GSPN-EXP-1 fulfils the $M \Rightarrow M$ property if and only if for all states* $\mathbf{m}^{(r)}$ *and MM-pathes* α *from* $\mathbf{m}^{(r)}$ *to* $\mathbf{m}_0$, *function* $\Psi(\alpha) = \psi(\mathbf{m}^{(r)})$. *In this case the stationary probability for state* $\mathbf{m}^{(r)}$ *is given by:*

$$\pi(\mathbf{m}^{(r)}) = \pi(\mathbf{m}_0) \frac{\prod_{a=1}^{R} \lambda_a^{m_a + m_{R+a}}}{\mu^m} \psi(\mathbf{m}^{(r)}), \tag{A.14}$$

where $m = \sum_{a=1}^{2R} m_a$.

Proof 3 *Suppose the system fulfils the $M \Rightarrow M$ property. Then let us choose an arbitrary MM-path* α *from* $\mathbf{m}^{(r)}$ *to* $\mathbf{m}_0$. *We first prove that:*

$$\pi(\mathbf{m}^{(r)}) = \pi(\mathbf{m}_0) \frac{\prod_{a=1}^{R} \lambda_a^{m_a + m_{R+a}}}{\mu^m} \Psi(\alpha), \tag{A.15}$$

Let us proceed by induction. If $\mathbf{m}^{(r)} \xrightarrow{\epsilon} \mathbf{m}_0$, *then by Lemma 11 and Definition 12, equation (A.15) is verified. Suppose* $\alpha = \mathbf{m}^{(r)}(s, \beta)$ *with* $s \neq \epsilon$. *Then, by Lemma 11 we can write:*

$$\frac{\pi(\mathbf{m}^{(r)})}{\pi\big((\mathbf{m}^{(r)} - \mathbf{e}_{R+r} + \mathbf{r}_{R+s} - \mathbf{m}_s)^{(s)}\big)} = \frac{\lambda_r}{\mu} \frac{\sum_{a=1}^{R} w_a(\mathbf{m}^{(r)})}{w_s(\mathbf{m}^{(r)})}. \tag{A.16}$$

Noting that $\frac{\sum_{a=1}^{R} w_a(\mathbf{m}^{(r)})}{w_s(\mathbf{m}^{(r)})}\Psi(\beta) = \Psi(\alpha)$ *by Definition 12 and by using inductive hypothesis to make* $\pi\big((\mathbf{m}^{(r)} - \mathbf{e}_{R+r} + \mathbf{r}_{R+s} - \mathbf{m}_s)^{(s)}\big)$ *explicit, we prove relation (A.15). In order to prove that* $\Psi(\alpha) = \psi(\mathbf{m}^{(r)})$ *for a function* ψ *which depends only on the state* $\mathbf{m}^{(r)}$ *it suffices to consider the fact that the ratio* $\pi(\mathbf{m}^{(r)})/\pi(\mathbf{m}_0)$ *cannot depend on a path between the states.*

Suppose Equation (A.14) is satisfied and $\Psi(\alpha) = \psi(\mathbf{m}^{(r)})$ *for any state* $\mathbf{m}^{(r)}$ *and MM-path* α *from* $\mathbf{m}^{(r)}$ *and* $\mathbf{m}_0$*. In order to prove Equation (7.4), i.e., the system satisfies* $M \Rightarrow M$*, consider a generic state* $\mathbf{m}_2^{(s)}$ *and* $\mathbf{m}_1^{(r)} = \mathbf{m}_2^{(s)} + \mathbf{e}_s - \mathbf{e}_{R+s} + \mathbf{e}_{R+r}$*. We have to prove that:*

$$\pi(\mathbf{m}_1^{(r)})\mu\frac{w_s(\mathbf{m}_1^{(r)})}{\sum_{a=1}^{R} w_a(\mathbf{m}_1^{(r)})} = \pi(\mathbf{m}_2^{(s)})\lambda_r, \tag{A.17}$$

by the same consideration used to prove Lemma 11. By replacing π *with (A.14) in (A.17), and by Definition 12, we have:*

$$\pi(\mathbf{m}_0)\frac{\prod_{a=1}^{R}\lambda_a^{m_{1,a}+m_{1,R+a}}\lambda_s}{\mu^{m_1+1}}\psi(\mathbf{m}_2^{(s)})\frac{\sum_{a=1}^{R} w_a(\mathbf{m}_1^{(r)})}{w_s(\mathbf{m}_1^{(r)})}\mu\frac{w_s(\mathbf{m}_1^{(r)})}{\sum_{a=1}^{R} w_a(\mathbf{m}_1^{(r)})}$$
$$= \pi(\mathbf{m}_0)\frac{\prod_{a=1}^{R}\lambda_a^{m_{1,a}+m_{1,R+a}}}{\mu^{m_1}}\psi(\mathbf{m}_2^{(s)})\lambda_r.$$

which is clearly an identity.

Corollary 2 *A necessary and sufficient condition for GSPN-EXP-1 to fulfil the* $M \Rightarrow M$ *property is that for any state* $\mathbf{m}$ *there exists a function* ψ *depending only on* $\mathbf{m}$ *such that* $\psi(\mathbf{m}) = \Psi(\alpha)$ *for any MM-Path* α *from* $\mathbf{m}$ *to* $\mathbf{m}_0$*.*

Note that even if Corollary 2 is quite simple, checking its proposition requires a number of operations which grows exponentially with the number of customers in the system. The following theorem avoids this problem by explicitly determining function ψ and by giving strict necessary and sufficient structural conditions for $M \Rightarrow M$.

We are now ready to prove the main result, i.e. Theorem 7, state in Section 7.3.

Proof 4 *Basically we have to prove that function* ψ *is uniquely determined as follows:*

$$\psi(\mathbf{m}^{(t)}) = \frac{(\sum_{a=1}^{R} m_a)!}{\prod_{a=1}^{R} m_a!}. \tag{A.18}$$

In fact, by substitution in (A.15) we obtain (A.14). First of all note that in case of a state $\mathbf{m}^{(r)}$ *with* $|A| = 1$ *we have that* $\psi(\mathbf{m}^{(r)}) = 1$ *follows by the definition of* Ψ*.*

The proof is by induction on dist$(\mathbf{m}^{(t)}, \mathbf{m}_0)$*.*

Base. *Let* $\mathbf{m}^{(t)} = \mathbf{e}_r + \mathbf{e}_s + \mathbf{e}_{R+t}$, *with* $1 \leq r, s, t \leq R$, *and then* $A = \{r, s\}$. *The pathes from* $\mathbf{m}^{(t)}$ *to* $\mathbf{m}_0$ *are the following:*

$$\alpha : \mathbf{m}^{(t)} \xrightarrow{s} \mathbf{e}_{R+s} + \mathbf{e}_r \xrightarrow{s} \mathbf{e}_{R+r} \xrightarrow{\epsilon} \mathbf{m}_0$$
$$\beta : \mathbf{m}^{(t)} \xrightarrow{r} \mathbf{e}_{R+r} + \mathbf{e}_s \xrightarrow{r} \mathbf{e}_{R+s} \xrightarrow{\epsilon} \mathbf{m}_0$$

Hence,

$$\Psi(\alpha) = 1 \cdot 1 \cdot \frac{w_s(\mathbf{m}^{(t)}) + w_r(\mathbf{m}^{(t)})}{w_s(\mathbf{m}^{(t)})}$$

and

$$\Psi(\beta) = 1 \cdot 1 \cdot \frac{w_s(\mathbf{m}^{(t)}) + w_r(\mathbf{m}^{(t)})}{w_r(\mathbf{m}^{(t)})}.$$

By Corollary 2 the system satisfies the $M \Rightarrow M$ *property if and only if* $w_r(\mathbf{m}^{(t)}) = w_s(\mathbf{m}^{(t)})$. *Consequently* $\psi(\mathbf{m}^{(t)}) = 2$.

Induction step. *Consider a generic state* $\mathbf{m}^{(v)}$ *with* $A \subseteq \{1, \ldots, R\}$ *and* $|A| \geq 2$. *Let* $t \in A$. *Then we have the following possible MM-path to* $\mathbf{m}^{(v)}$*:*

$$\alpha : \mathbf{m}^{(v)} \xrightarrow{t} \mathbf{m}'^{(t)} \to \ldots \to \mathbf{m}_0.$$

Let $A' = \{r : m'_r > 0, 1 \leq r \leq R\}$, *and consider the case* $|A'| \geq 2$, *then by inductive hypothesis, we have that:*

$$\psi(\mathbf{m}'^{(t)}) = \frac{(\sum_{a=1}^{R} m'_a)!}{\prod_{a=1}^{R} m'_a} = \frac{(\sum_{a=1}^{R} m_a - 1)!}{\prod_{a=1, a\neq t}^{R} (m_t - 1)!}.$$

Hence, we have that:

$$\Psi(\alpha) = \frac{(\sum_{a=1}^{R} m_a - 1)!}{\prod_{a=1, a\neq t}^{R} (m_t - 1)!} \cdot \frac{\sum_{a=1}^{R} w_a(\mathbf{m}^{(v)})}{w_t(\mathbf{m}^{(v)})}.$$

By Corollary 2 we have that $M \Rightarrow M$ *holds if and only if* Ψ *is independent of the MM-Path. Hence function* Ψ *must be independent of the choice of* $t \in A$. *It is easy to see that this is the case if and only if* $w_t(\mathbf{m}^{(v)}) = f(\mathbf{m}^{(v)}) m_t$. *If* $|A'| = 1$ *we have that* $\Psi(\mathbf{m}'^{(t)}) = 1$ *and the same result holds.*

In other words, when there are customers of two or more classes waiting for the server, function w_r must be $w_r(\mathbf{m}) = m_r$ (the multiplier factor is not really important for obvious algebraic reasons). Actually for the states where customers of just one class r are in queue, the definition of w_r is irrelevant important because it does not influences the system stochastic behavior. So we can conclude that the only reasonable definition for functions w_r is $w_r(\mathbf{m}) = m_r$ for every class r and every state $\mathbf{m}$. This concludes the proof. ♠

B

Solution of examples

B.1 Stochastic Petri net models of Chapter 5

In this section we study the product-form SPN models of Chapter 5 using Theorem 2 introduced in [63, 39].

B.1.1 Solution of model depicted in Figure 5.15

First of all we fuse transitions T_1 and T_2 in a transition named T_{12} with rate $\chi_{12} = \chi_1 + \chi_2$. Transition T_{12} has two possible output vectors $O_1(T_{12}) = (1, 1, 0, 0, 0)$ with probability $\chi_1/(\chi_1 + \chi_2)$ and $O_2(T_{12}) = (0, 1, 0, 0, 0)$ with probability $\chi_2/(\chi_1 + \chi_2)$. Similarly transitions T_4 and T_5 must be fused into one transition T_{45} with rate $\chi_{45} = \chi_4 + \chi_5$. The output vector $O_1(T_{45}) = (0, 0, 0, 0, 1)$ occurs with probability $\chi_4/(\chi_4 + \chi_5)$ while $O_2(T_{45}) = (0, 0, 1, 0, 0)$ with probability $\chi_5/(\chi_4 + \chi_5)$.

The system of traffic equations is:

$$\begin{cases} (\chi_1 + \chi_2) f_{12} = \chi_7 f_7 + \chi_3 f_3 \\ \chi_3 f_3 = \chi_2 f_{12} \\ (\chi_4 + \chi_5) f_{45} = \chi_1 f_{12} + \chi_8 f_8 \\ \chi_6 f_6 = \chi_5 f_{45} \\ \chi_7 f_7 = \chi_4 f_{45} \\ \chi_8 f_8 = \chi_6 f_6 \end{cases},$$

whose solution is:

$$\begin{cases} f_{12} = 1 \\ f_3 = \frac{\chi_2}{\chi_3} \\ f_{45} = \frac{\chi_1}{\chi_4} \\ f_6 = \frac{\chi_5 \chi_1}{\chi_4 \chi_6} \\ f_7 = \frac{\chi_1}{\chi_7} \\ f_8 = \frac{\chi_5 \chi_1}{\chi_4 \chi_8} \end{cases}.$$

Vector $\mathbf{C} = (c_1, \ldots, c_8)^T$ defined in (3.7) is:

$$c_1 = \log\left(\frac{\chi_4}{\chi_1}\right) \; c_2 = \log\left(\frac{\chi_3}{\chi_2}\right) \; c_3 = \log\left(\frac{\chi_2}{\chi_3}\right) c_4 = \log\left(\frac{\chi_7}{\chi_4}\right) \; c_5 = \log\left(\frac{\chi_6}{\chi_5}\right)$$
$$c_6 = \log\left(\frac{\chi_8}{\chi_6}\right) \; c_7 = \log\left(\frac{\chi_1}{\chi_7}\right) \; c_8 = \log\left(\frac{\chi_5}{\chi_8}\right)$$

The incidence matrix $\mathbf{A}$ of the net is:

$$\mathbf{A} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 1 \\ -1 & -1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & -1 \\ 1 & 1 & 0 & -1 & 0 \end{bmatrix}.$$

We have that $\text{rank}(\mathbf{A}) = 5$. In order to have $\text{rank}([\mathbf{A}|\mathbf{C}]) = 5$ we have to check that:

$$\begin{cases} c_2 = -c_3 \\ c_1 = -(c_4 + c_7) \\ c_5 = -(c_6 + c_8) \end{cases},$$

that are trivially verified for any value of χ_i. We now solve the system (3.10) and we obtain:

$$y_1 = \frac{\chi_1\chi_3}{\chi_4\chi_2} \; y_2 = \frac{\chi_2}{\chi_3} \; y_3 = \frac{\chi_1\chi_5}{\chi_4\chi_6} \; y_4 = \frac{\chi_1\chi_5}{\chi_4\chi_8} \; y_5 = \frac{\chi_1}{\chi_7}.$$

The steady state proabilities $\pi_1(\mathbf{m})$ are given by:

$$\pi_1(\mathbf{m}) \propto \left(\frac{\chi_1\chi_3}{\chi_4\chi_2}\right)^{m_1} \left(\frac{\chi_2}{\chi_3}\right)^{m_2} \left(\frac{\chi_1\chi_5}{\chi_4\chi_6}\right)^{m_3} \left(\frac{\chi_1\chi_5}{\chi_4\chi_8}\right)^{m_4} \left(\frac{\chi_1}{\chi_7}\right)^{m_5}.$$

B.1.2 Solution of model depicted in Figure 5.16

First of all we fuse transitions T_1 and T_2 in a transition named T_{12} with rate $\chi_{12} = \chi_1 + \chi_2$. Transition T_{12} has two possible output vectors $O_1(T_{12}) = (1, 1, 0, 0, 0)$ with probability $\chi_1/(\chi_1 + \chi_2)$ and $O_2(T_{12}) = (0, 1, 0, 0, 0)$ with probability $\chi_2/(\chi_1 + \chi_2)$. All the other transitions have different input vectors. The system of traffic equation (3.6) is:

$$\begin{cases} (\chi_1 + \chi_2) f_{12} = \chi_6 f_6 + \chi_8 f_8 \\ \chi_3 f_3 = \chi_1 f_{12} \\ \chi_4 f_4 = \chi_7 f_7 \\ \chi_5 f_5 = \chi_2 f_{12} \\ \chi_6 = \chi_3 f_3 \\ \chi_7 f_7 = \chi_4 f_4 \\ \chi_8 f_8 = \chi_5 f_5 \end{cases},$$

whose solution is:

$$\begin{cases} f_{12} = 1 \\ f_3 = \frac{\chi_1}{\chi_3} \\ f_4 = 1 \\ f_5 = \frac{\chi_2}{\chi_5} \\ f_6 = \frac{\chi_1}{\chi_6} \\ f_7 = \frac{\chi_4}{\chi_7} \\ f_8 = \frac{\chi_2}{\chi_8} \end{cases}.$$

Note that the traffic process consists of two communicating class, i.e., the set of transitions $\{T_{12}, T_3, T_6, T_5, T_8\}$, and the set of transitions $\{T_4, T_7\}$. Vector $\mathbf{C} = (c_1, \ldots, c_8)^T$ is given by:

$$c_1 = \log\left(\frac{\chi_3}{\chi_1}\right) \; c_2 = \log\left(\frac{\chi_5}{\chi_2}\right) \; c_3 = \log\left(\frac{\chi_6}{\chi_3}\right) \; c_4 = \log\left(\frac{\chi_7}{\chi_4}\right) \; c_5 = \log\left(\frac{\chi_8}{\chi_5}\right)$$
$$c_6 = \log\left(\frac{\chi_1}{\chi_6}\right) \; c_7 = \log\left(\frac{\chi_4}{\chi_7}\right) \; c_8 = \log\left(\frac{\chi_2}{\chi_8}\right).$$

The incidence matrix $\mathbf{A}$ is:

$$\mathbf{A} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ -1 & -1 & 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 0 & 1 & 0 \\ 0 & -1 & -1 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix}.$$

We have that $\text{rank}(\mathbf{A}) = 5$, therefore in order to have $\text{rank}([\mathbf{A}|\mathbf{C}]) = 5$ the following conditions are needed:

$$\begin{cases} c_4 = -c_7 \\ c_6 = -(c_1 + c_3) \\ c_2 = -(c_5 + c_8) \end{cases},$$

that are unconditionally satisfied.

One solution of system (3.10) is:

$$y_1 = \frac{\chi_1}{\chi_3} \; y_2 = 1 \; y_3 = \frac{\chi_2}{\chi_5} \; y_4 = \frac{\chi_1}{\chi_6} \; y_5 = \frac{\chi_4}{\chi_7} \; y_6 = \frac{\chi_2}{\chi_8}.$$

Hence, the steady state solution of the model is:

$$\pi(\mathbf{m}) \propto \left(\frac{\chi_1}{\chi_3}\right)^{m_1} \left(\frac{\chi_2}{\chi_5}\right)^{m_3} \left(\frac{\chi_1}{\chi_6}\right)^{m_4} \left(\frac{\chi_4}{\chi_7}\right)^{m_5} \left(\frac{\chi_2}{\chi_8}\right)^{m_6}.$$

B.2 Product-form GSPN models composition of Chapter 7

B.2.1 Analysis of the GSPN shown by Example 12

. In this section we study the GSPN illustrated by Figure 7.9. Recall that:

$$\begin{cases} \chi_6 = \chi_2 \\ \chi_5 = \chi_1 \end{cases},$$

Let us call x_1 the (constant) reversed rate of the departures by the first block, and x_2 the (constant) reversed rates of the departures from the second block. By the analysis of model SIMPLE we have that:

$$\begin{cases} x_1 = \chi_7 + x_2 \frac{\chi_3}{\chi_3+\chi_4} \\ x_2 = \chi_1 \end{cases},$$

that gives:

$$x_1 = x_2 = \chi_7 \frac{\chi_3 + \chi_4}{\chi_4}.$$

$\{P_1, P_2, P_4\}$ is a sufficient place set, in fact $P_3 = K - P_4$ where K is the total number of token in P_3 and P_2 of the initial state (the total number of servers). Therefore the product-form solution is given by:

$$\pi(m_1, m_2, m_4) \propto \left(\frac{\chi_7(\chi_3 + \chi_4)}{\chi_1\chi_4}\right)^{m_1} \left(\frac{\chi_1}{\chi_2}\right)^{m_2} \left(\frac{\chi_7}{\chi_4}\right)^{m_4}.$$

The analysis is completed.

The global balance equations. Since the analysis approach is rather new, as a check we show that the solution satisfies the global balance equations of the stochastic process associated with the GSPN. Therefore the following calculations have not to be considered part of the analysis.

As the model has an infinite state space we cluster the spaces as follows:

- m_1 can be either 0 or $N > 0$,
- m_2 can be 0, ℓ with $1 \leq \ell < K$, or K,
- m_4 can be either 0 or $M > 0$.

The twelve combinations of these cases give all the possible states of the net.

1. $[0,0,0]$: $\pi(0,0,0)[\chi_7+\chi_1] = \pi(0,0,1)\chi_4+\pi(0,1,0)\chi_2 = \pi(0,0,0)[\frac{\chi_7}{\chi_4}\chi_4+\frac{\chi_1}{\chi_2}\chi_2]$.

2. $[0,0,M]$:

$$\pi(0,0,M)[\chi_7+\chi_1+\chi_3+\chi_4] = \pi(0,0,M+1)\chi_4+\pi(0,1,0)\chi_2+\pi(1,1,M-1)\chi_2$$
$$= \pi(0,0,M)[\frac{\chi_7}{\chi_4}\chi_4 + \frac{\chi_1}{\chi_2}\chi_2 + \frac{\chi_7(\chi_3+\chi_4)}{\chi_4\chi_1}\frac{\chi_1}{\chi_2}\frac{\chi_4}{\chi_7}\chi_2].$$

3. $[0,\ell,0]$:

$$\pi(0,\ell,0)[\chi_7+\chi_2+\chi_1] = \pi(0,\ell,1)\chi_4 + \pi(0,\ell-1,0)\chi_1 + \pi(0,\ell+1,0)\chi_2$$
$$= \pi(0,\ell,0)[\frac{\chi_7}{\chi_4}\chi_4 + \frac{\chi_2}{\chi_1}\chi_1 + \frac{\chi_1}{\chi_2}\chi_2].$$

4. $[0,\ell,M]$:

$$\pi(0,\ell,M)(\chi_7+\chi_2+\chi_1+\chi_3+\chi_4) = \pi(0,\ell,M+1)\chi_4$$
$$+\pi(0,\ell-1,M)\chi_1 + \pi(0,\ell+1,0)\chi_2 + \pi(1,\ell+1,M-1)\chi_2$$
$$= \pi(0,\ell,M)[\frac{\chi_7}{\chi_4}\chi_4 + \frac{\chi_2}{\chi_1}\chi_1 + \frac{\chi_1}{\chi_2}\chi_2 + + \frac{\chi_7(\chi_3+\chi_4)}{\chi_4\chi_1}\frac{\chi_1}{\chi_2}\frac{\chi_4}{\chi_7}\chi_2].$$

5. $[0,K,0]$:

$$\pi(0,K,0)[\chi_7+\chi_2] = \pi(0,K-1,0)\chi_1+\pi(0,K,1)\chi_4 = \pi(0,K,0)[\frac{\chi_2}{\chi_1}\chi_1+\frac{\chi_7}{\chi_4}\chi_4].$$

6. $[0,K,M]$:

$$\pi(0,K,M)[\chi_7+\chi_2+\chi_3+\chi_4] = \pi(0,K-1,M)\chi_1 + \pi(0,K,M+1)\chi_4$$
$$+\pi(1,K-1,M-1)\chi_2 = \pi(0,K,M)[\frac{\chi_2}{\chi_1}\chi_1 + \frac{\chi_7}{\chi_4}\chi_4 + \frac{\chi_7(\chi_3+\chi_4)}{\chi_4\chi_1}\frac{\chi_1}{\chi_2}\frac{\chi_4}{\chi_7}\chi_2].$$

7. $[N,0,0]$:

$$\pi(N,0,0)[\chi_7+\chi_1] = \pi(N-1,0,1)\chi_3 + \pi(N,0,1)\chi_4 + \pi(N-1,0,0)\chi_7$$
$$= \pi(N,0,0)[\frac{\chi_4\chi_1}{\chi_7(\chi_3+\chi_4)}\frac{\chi_7}{\chi_4}\chi_3 + \frac{\chi_7}{\chi_4}\chi_4 + \frac{\chi_4\chi_1}{\chi_7(\chi_3+\chi_4)}\chi_7].$$

8. $[N,0,M]$:

$$\pi(N,0,M) = [\chi_7+\chi_1+\chi_3+\chi_4] = \pi(N-1,0,M+1)\chi_3$$
$$+\pi(N-1,0,M)\chi_7 + \pi(N+1,1,M-1)\chi_2 + \pi(N,0,M+1)\chi_4$$
$$= \pi(N,0,M)[\frac{\chi_4\chi_1}{\chi_7(\chi_3+\chi_4)}\frac{\chi_7}{\chi_4}\chi_3 + \frac{\chi_4\chi_1}{\chi_7(\chi_3+\chi_4)}\chi_7$$
$$+\frac{\chi_7(\chi_3+\chi_4)}{\chi_4\chi_1}\frac{\chi_1}{\chi_2}\frac{\chi_4}{\chi_7}\chi_2 + \frac{\chi_7}{\chi_4}\chi_4].$$

9. $[M,\ell,0]$:

$$\pi(N,\ell,0)[\chi_7+\chi_1+\chi_2]=\pi(N-1,\ell,1)\chi_3+\pi(N,\ell,1)\chi_4$$
$$+\pi(N-1,\ell,0)\chi_7+\pi(N,\ell-1,0)\chi_1$$
$$=\pi(N,\ell,0)[\frac{\chi_4\chi_1}{\chi_7(\chi_3+\chi_4)}\frac{\chi_7}{\chi_4}\chi_3+\frac{\chi_7}{\chi_4}\chi_4+\frac{\chi_4\chi_1}{\chi_7(\chi_3+\chi_4)}\chi_7+\frac{\chi_2}{\chi_1}\chi_1].$$

10. $[M,\ell,N]$:

$$\pi(N,\ell,M)[\chi_7+\chi_1+\chi_2+\chi_3+\chi_4]=\pi(N-1,\ell,M+1)\chi_3+\pi(N+1,\ell+1,N-1)\chi_2$$
$$+\pi(N-1,\ell,M)\chi_7+\pi(N,\ell-1,M)\chi_1+\pi(N,\ell,M+1)\chi_4$$
$$=\pi(N,\ell,M)[\frac{\chi_4\chi_1}{\chi_7(\chi_3+\chi_4)}\frac{\chi_7}{\chi_4}\chi_3+\frac{\chi_7(\chi_3+\chi_4)}{\chi_4\chi_1}\frac{\chi_1}{\chi_2}\frac{\chi_4}{\chi_7}\chi_2+\frac{\chi_1\chi_4}{\chi_7(\chi_3+\chi_4)}\chi_7$$
$$+\frac{\chi_2}{\chi_1}\chi_1+\frac{\chi_7}{\chi_4}\chi_4].$$

11. $[M,K,0]$:

$$\pi(M,K,0)[\chi_7+\chi_2+\chi_1]=\pi(M-1,K,1)\chi_3+\pi(M-1,K,0)\chi_7$$
$$+\pi(M,K,1)\chi_4+\pi(M,K-1,0)\chi_1=$$
$$\pi(M,K,0)[\frac{\chi_1\chi_4}{\chi_7(\chi_3+\chi_4)}\frac{\chi_7}{\chi_4}\chi_3+\frac{\chi_1\chi_4}{\chi_7(\chi_3+\chi_4)}\chi_7+\frac{\chi_7}{\chi_4}\chi_4+\frac{\chi_2}{\chi_1}\chi_1].$$

12. $[M,K,N]$:

$$\pi(M,K,N)[\chi_7+\chi_2+\chi_1+\chi_3+\chi_4]=\pi(M-1,K,N+1)\chi_3+\pi(M-1,K,N)\chi_7$$
$$+\pi(M,K,N+1)\chi_4+\pi(M,K-1,N)\chi_1+\pi(M+1,K,N-1)\chi_1=$$
$$\pi(M,K,N)[\frac{\chi_1\chi_4}{\chi_7(\chi_3+\chi_4)}\frac{\chi_7}{\chi_4}\chi_3+\frac{\chi_1\chi_4}{\chi_7(\chi_3+\chi_4)}\chi_7+\frac{\chi_7}{\chi_4}\chi_4$$
$$+\frac{\chi_2}{\chi_1}\chi_1+\frac{\chi_7(\chi_3+\chi_4)}{\chi_1\chi_4}\frac{\chi_4}{\chi_7}\chi_1].$$

Bibliography

[1] P. V. Afshari, S. C. Bruell, and R. Y. Kain. Modeling a new technique for accessing shared buses. In *Proc. of the Computer Network Performance Symp.*, pages 4–13, New York, NY, USA, 1982. ACM Press.

[2] I. F. Akyildiz. Exact product form solution for queueing networks with blocking. *IEEE Trans. on Computer*, C-36-1:122–125, 1987.

[3] F. Baccelli, W.A. Massey, and D. Towsley. Acyclic fork-join queueing networks. *J. ACM*, 36(3):615–642, 1989.

[4] G. Balbo. *Introduction to Generalized Stochastic Petri Nets in Formal Methods for Performance Evaluation*, chapter 3, pages 33–131. M. Bernardo and J. Hillston (Eds), LNCS, Springer, 2007.

[5] G. Balbo, S. C. Bruell, and S. Ghanta. Combining queueing network and generalized stochastic Petri nets for the solution of complex models of system behavior. *IEEE Trans. on Computers*, 37:1251–1268, 1998.

[6] G. Balbo, S. C. Bruell, and M. Sereno. Product form solution for Generalized Stochastic Petri Nets. *IEEE Trans. on Software Eng.*, 28:915–932, 2002.

[7] G. Balbo, S. C. Bruell, and M. Sereno. On the relations between BCMP Queueing Networks and Product Form Solution Stochastic Petri Nets. *Proc. of 10th Int. Workshop on Petri Nets and Performance Models, 2003.*, pages 103–112, 2003.

[8] S. Balsamo, V. De Nitto Persone', and R. Onvural. *Analysis of Queueing Networks with Blocking.* Kluwer Academic Publishers, 2001.

[9] S. Balsamo, F. de Riz, and A. Marin. Product form queueing networks and product form Stochastic Petri Nets: relations and transformation algorithms. Technical Report CS-2004-12, Dip. Informatica, Università Ca' Foscari Venice.

[10] S. Balsamo and G. Iazeolla. Product-form synthesis of queueing networks. *IEEE Trans. on Software Eng.*, SE-11(2):194–199, 1985.

[11] S. Balsamo and A. Marin. On representing multiclass M/M/k queues by generalized stochastic Petri nets. Technical Report CS-2007-1, Università Ca' Foscari, Venice, Italy, 2007.

[12] S. Balsamo and A. Marin. *Queueing Networks in Formal methods for performance evaluation*, chapter 2, pages 34–82. M. Bernardo and J. Hillston (Eds), LNCS, Springer, 2007.

[13] S. Balsamo and A. Marin. From BCMP Queueing Networks to Generalized Stochastic Petri Nets: An Algorithm and an Equivalence Definition. In *Proc. of ESM*, pages 447–455, Le Havre, FR, 2008.

[14] S. Balsamo and A. Marin. Determining product-form steady state solutions of Generalized Stochastic Petri Nets by the analysis of the reversed process. In *Proc. of ACS/IEEE AICCSA*, Rabat, Marocco, 2009, to appear.

[15] S. Balsamo and A. Marin. On representing multiclass M/M/k queues by generalized stochastic Petri nets. In *Proc. of ECMS/ASMTA-2007 Conf.*, pages 121–128, Prague, Czech Republic, 4-6 June 2007.

[16] S. Balsamo and A. Marin. Representing LCFSPR BCMP service centers with Coxian service time distribution. In *Proc. of Valuetools '07 Conf.*, Nantes, France, October 23-25, 2007.

[17] F. Baskett, K. M. Chandy, R. R. Muntz, and F. G. Palacios. Open, closed, and mixed networks of queues with different classes of customers. *J. ACM*, 22(2):248–260, 1975.

[18] F. Bause. Queueing Petri nets: A formalism for the combined qualitative and quantitative analysis of systems. In *Proc. of 5th Int. Workshop on Petri Nets and Performance Models*, pages 14–23, Toulouse (France), 1993.

[19] F. Bause and P. Buchholz. Product form queueing Petri nets: A combination of product form queueing networks and product form stochastic Petri nets. *Perform. Eval., Elsevier*, 32:265–299, 1998.

[20] M. Bernardo, L. Donatiello, and R. Gorrieri. Modelling and Analyzing Concurrent Systems with MPA. In *Proc. of 2nd Process Algebra and Performance Modelling Workshop*, pages 175–189, 1994.

[21] M. Bernardo, R. Gorrieri, and M. A. Zamboni. A tutorial on EMPA: A theory of concurrent processes with nondeterminism, priorities, probabilities and time. *Theoretical Computer Science*, 202:1–54, 1998.

[22] G. Bolch, S. Greiner, H. de Meer, and K. S. Trivedi. *Queueing networks and Markov chains*. John Wiley, 1998.

[23] P. Bonet, C. Llado, R. Puijaner, and W. Knottenbelt. Pipe v2.5.: a Petri net tool for performance modelling. In *Proc. of 23rd Latin American Conference on Informatics*, San Jose, Costa Rica, October, 2007.

[24] R. Boucherie and N. M. van Dijk. Product-form queueing networks with state dependent multiple job transitions. *Advances in Applied Prob.*, 23:152–187, 1991.

[25] R. J. Boucherie. A characterisation of independence for competing Markov chains with applications to stochastic Petri nets. *IEEE Tran. on Software Eng.*, 20(7):536–544, 1994.

[26] S. C. Bruell and G. Balbo. *Computational Algorithms for Closed Queueing Networks*. The Computer Science Library. Elsevier North Holland, 1980.

[27] S. C. Bruell, G. Balbo, and P. V. Afshari. Mean Value Analysis of mixed, multiple class BCMP networks with load dependent service stations. *Perform. Eval. Elsevier*, 4:241–260, 1984.

[28] P. J. Burke. The output of a queueing system. *Operations Research*, 4(6):699–704, 1956.

[29] J. P. Buzen. Computational algorithms for closed queueing networks with exponential servers. *Commun. ACM*, 16(9):527–531, 1973.

[30] M. Calzarossa and S. Tucci, editors. *Performance Evaluation of Complex Systems: Techniques and Tools, Performance 2002, Tutorial Lectures*, London, UK, 2002. Springer-Verlag.

[31] K. M. Chandy, U. Herzog, and L. Woo. Parametric analysis of queueing networks. *IBM Journal of Res. and Dev.*, 1(1):36–42, 1975.

[32] K. M. Chandy, Jr. J. H. Howard, and D. F. Towsley. Product form and local balance in queueing networks. *J. ACM*, 24(2):250–263, 1977.

[33] K. M. Chandy and A. J. Martin. A characterization of product-form queuing networks. *J. ACM*, 30(2):286–299, 1983.

[34] K. M. Chandy and C. H. Sauer. Computational algorithms for product form queueing networks. *Commun. ACM*, 23(10):573–583, 1980.

[35] G. Chiola, G. Franceschinis, R. Gaeta, and M. Ribaudo. Greatspn 1.7: Graphical editor and analyzer for timed and stochastic petri nets. *Perform. Eval., Elsevier*, 24:47–68, 1995.

[36] G. Chiola, M. A. Marsan, G. Balbo, and G. Conte. Generalized stochastic Petri nets: a definition at the net level and its implications. *IEEE Trans. on Software Eng.*, 19(2):89–107, 1993.

[37] A. Chockalingam and M. Zorzi. Analysis of link-layer backoff algorithms on point-to-point markov fading links: effect of round-trip delays. In *Proc. IEEE ICC'05*, volume 5, pages 3117–3121, Seul, Korea, 30 May - 1 Jun. 2005.

[38] J. W. Cohen. *The single server queue*. Wiley-Interscience, 1969.

[39] J. L. Coleman, W. Henderson, and P. G. Taylor. Product form equilibrium distributions and a convolution algorithm for Stochastic Petri nets. *Perform. Eval., Elsevier*, 26:159–180, 1996.

[40] A. E. Conway, E. de Souza e Silva, and S. S. Lavenberg. Mean Value Analysis by chain of product form queueing networks. *IEEE Trans. Comput.*, 38(3):432–442, 1989.

[41] A. E. Conway and N. D. Georganas. Recal - a new efficient algorithm for the exact analysis of multiple-chain closed queuing networks. *J. ACM*, 33(4):768–791, 1986.

[42] A. E. Conway and N. D. Georganas. *Queueing Networks - Exact Computational Algorithms: A unified Theory Based on Decomposition and Aggregation*. The MIT Press, Cambridge, MA, 1989.

[43] P.J. Courtois. *Decomposability*. Academic Press, New York, 1977.

[44] E. de Souza e Silva and S. S. Lavenberg. Calculating joint queue-length distributions in product-form queuing networks. *J. ACM*, 36(1):194–207, 1989.

[45] E. De Souza e Silva and R. R. Muntz. *Stochastic Analysis of Computer and Communication Systems*, chapter Queueing Networks: Solutions and Applications, pages 319–399. H. Takagi Ed., North Holland, 1990.

[46] J. Fourneau, E. Gelenbe, and R. Suros. G-networks with multiple class negative and positive customers. In *MASCOTS '94: Proc. of the Second Int. Workshop on Modeling, Analysis, and Simulation On Computer and Telecommunication Systems*, pages 30–34, Washington, DC, USA, 1994. IEEE Computer Society.

[47] J. M. Fourneau, B. Plateau, and W. J. Stewart. Product form for stochastic automata networks. In *ValueTools '07: Proc. of the 2nd international conference on Performance evaluation methodologies and tools*, pages 1–10, ICST, Brussels, Belgium, 2007. ICST.

[48] J. M. Fourneau, B. Plateau, and W. J. Stewart. An algebraic condition for product form in stochastic automata networks without synchronizations. *Perform. Eval., Elsevier*, 65:854–868, 2008.

[49] J. M. Fourneau and D. Verchere. G-networks with triggered batch state-dependent movement. In *MASCOTS '95: Proc. of the Third Int. Workshop on Modeling, Analysis, and Simulation On Computer and Telecommunication Systems*, pages 33–37, 1995.

[50] E. Gelenbe. Product form networks with negative and positive customers. *Journal of Applied Prob.*, 28(3):656–663, 1991.

[51] E. Gelenbe. G-networks: a unifying model for neural and queueing networks. *Annals of Operations Research*, 48(5):433–461, October, 1994.

[52] E. Gelenbe and J. M. Fourneau. G-networks with resets. *Perform. Eval., Elsevier*, 49(1-4):179–191, 2002.

[53] E. Gelenbe and I. Mitrani. *Analysis and Synthesis of Computer Systems*. Academic Press, New York, 1980.

[54] W. J. Gordon and G. F. Newell. Cyclic queueing networks with exponential servers. *Operations Research*, 15(2):254–265, 1967.

[55] W. J. Gordon and G. F. Newell. Cyclic queueing networks with restricted length queues. *Operations Research*, 15(2):266–277, 1967.

[56] S. Haddad, P. Moreaux, M. Sereno, and M. Silva. Product-form and stochastic Petri nets: a structural approach. *Perform. Eval., Elsevier*, 59(4):313–336, 2005.

[57] P. Harrison and J. Hillston. Exploiting quasi-reversible structures in Markovian process algebra models. *The Computer Journal*, 38(7):510–520, 1995.

[58] P. G. Harrison. Reversed processes, product forms, non-product forms and a new proof of the BCMP theorem. In *Int. Conf. on the Numerical Solution of Markov Chains (NSMC 2003), Urbana IL, USA, September 2-5 2003*, pages 289–304, September 2003.

[59] P. G. Harrison. Turning back time in Markovian process algebra. *Theoretical Computer Science*, 290(3):1947–1986, January 2003.

[60] P. G. Harrison. Compositional reversed Markov processes, with applications to G-networks. *Perform. Eval., Elsevier*, 57(3):379–408, 2004.

[61] P. G. Harrison. Reversed processes, product forms and a non-product form. *Linear Algebra and Its Applications*, 386:359–381, July 2004.

[62] P. G. Harrison and T. T. Lee. Separable equilibrium state probabilities via time reversal in markovian process algebra. *Theoretical Computer Science*, 346(1):161–182, 2005.

[63] W. Henderson, D. Lucic, and P. G. Taylor. A net level performance analysis of Stochastic Petri Nets. *J. Austral. Math. Soc. Ser. B*, 31:176–187, 1989.

[64] W. Henderson and P. Taylor. Product form in networks of queues with batch arrivals and batch services. *Queueing Systems*, 6:71–88, 1990.

[65] W. Henderson and P. Taylor. Some new results on queueing networks with batch movements. *Journal of Applied Prob.*, 28:409–421, 1990.

[66] H. Hermanns, U. Herzog, and J. P. Katoen. Process algebra for performance evaluation. *Theor. Comput. Sci.*, 274(1-2):43–87, 2002.

[67] H. Hermanns, U. Herzog, and V. Mertsiotakis. Stochastic process algebras: between LOTOS and Markov chains. *Comp. Netw. and ISDN Syst*, 30:901–924, 1998.

[68] J. Hillston. *A Compositional Approach to Performance Modelling*. PhD thesis, Department of Computer Science, University of Edimburgh, 1994.

[69] J. Hillston and N. Thomas. A syntactical analysis of reversible PEPA models. In *Proc. 6th Process Algebra and Performance Modelling Workshop*, Nice, September 11-12 1998. University of Verona, Italy.

[70] J. Hillston and N. Thomas. Product form solution for a class of PEPA models. *Perform. Eval., Elsevier*, 35(3–4):171–192, 1999.

[71] C. Hoare. *Communicating Sequential Processes*. Prentice-Hall, 1985.

[72] K. P. Hoyme, S. C. Bruell, P. V. Afshari, and R. Y. Kain. A tree-structured Mean Value Analysis algorithm. *ACM Trans. Comput. Syst.*, 4(2):178–185, 1986.

[73] J. R. Jackson. Jobshop-like queueing systems. *Management Science*, 10:131–142, 1963.

[74] K. Kant. *Introduction to Computer System Performance Evaluation*. McGraw-Hill, 1992.

[75] K. Kant. *Introduction to Computer System Performance Evaluation*, chapter 4 and 9. McGraw-Hill, 1992.

[76] F. Kelly. *Reversibility and stochastic networks*. Wiley, New York, 1979.

[77] L. Kleinrock. *Queueing Systems*, volume 1 (Theory). John Wiley and Sons, 1975.

[78] S. S. Lam. Queueing networks with capacity constraints. *IBM Journal of Res. and Dev.*, 21(4):370–378, 1977.

[79] S. S. Lam. Dynamic scaling and growth behavior of queuing network normalization constants. *J. ACM*, 29(2):492–513, 1982.

[80] S. S. Lam and Y. L. Lien. A tree-convolution algorithm for the solution of queueing networks. *Commun. ACM*, 26(3):203–215, 1983.

[81] S. S. Lavenberg. *Computer Performance Modeling Handbook*. Academic Press, New York, 1983.

[82] A. M. Law and W. D. Kelton. *Simulation modeling and analysis*. McGraw-Hill, 3rd edition, 2000.

[83] A. A. Lazar and T. G. Robertazzi. Markovian Petri Net Protocols with Product Form Solution. In *IEEE INFOCOM'87, The Conf. on Computer Communications, Proc., Sixth Annual Conference - Global Networks: Concept to Realization*, pages 1054–1062, Washington, DC, 1987. IEEE Computer Society Press.

[84] A. A. Lazar and T. G. Robertazzi. Markovian Petri Net Protocols with Product Form Solution. *Perform. Eval., Elsevier*, 12(1):67–77, Jan 1991.

[85] E. D. Lazowska, J. L. Zahorjan, G. S. Graham, and K. C. Sevcick. *Quantitative system performance: computer system analysis using queueing network models*. Prentice Hall, Englewood Cliffs, NJ, 1984.

[86] J. Y. Le Boudec. A BCMP extension to multiserver stations with concurrent classes of customers. In *SIGMETRICS '86/PERFORMANCE '86: Proc. of the 1986 ACM SIGMETRICS Int. Conf. on Computer performance modelling, measurement and evaluation*, pages 78–91, New York, NY, 1986. ACM Press.

[87] R. Marie. An approximate analytical method for general queueing networks. *IEEE Trans. on Software Eng.*, 5(5):530–538, 1979.

[88] M. A. Marsan, G. Balbo, and G. Conte. *Performance Models of Microprocessor Systems*. MIT Press, 1986.

[89] M. A. Marsan, G. Balbo, G. Conte, S. Donatelli, and G. Franceschinis. *Modelling with generalized stochastic Petri nets*. Wiley, 1995.

[90] B. Melamed. On Poisson traffic processes in discrete-state markovian systems with applications to queueing theory. *Advances in Applied Prob.*, (11):218–239, 1979.

[91] R. Milner. *Communication and Concurrency*. Prentice-Hall, 1989.

[92] M. K. Molloy. Performance analysis using stochastic petri nets. *IEEE Trans. Comput.*, 31(9):913–917, 1982.

[93] R. Muntz and J. Wong. Efficient computational procedures for closed queueing network models. In *Proc. 7th Hawaii Int. Conf. on System Science*, pages 33–36, January 1974.

[94] R. R. Muntz. Poisson departure processes and queueing networks. Technical Report IBM Research Report RC4145, Yorktown Heights, New York, 1972.

[95] T. Murata. Petri nets: Properties, analysis and applications. *Proc. of the IEEE*, 77(4):541–580, 1989.

[96] R. Nelson. The mathematics of product-form queueing networks. *ACM Computing Survey*, 25(3):339–369, 1993.

[97] M. F. Neuts. *Matrix Geometric Solutions in Stochastic Models.* John Hopkins, Baltimore, Md, 1981.

[98] A. S. Noetzel. A generalized queueing discipline for product form network solutions. *J. ACM*, 26(4):779–793, 1979.

[99] D. C. Petriu, J. E. Neilson, C. M. Woodside, and S. Majumdar. Software bottlenecking in client-server systems and rendezvous networks. *IEEE Trans. on Software Eng.*, 21(9):776–782, 1995.

[100] B. Plateau. On the stochastic structure of parallelism and synchronization models for distributed algorithms. *SIGMETRICS Perform. Eval. Rev.*, 13(2):147–154, 1985.

[101] M. Raiser. Mean Value Analysis and Convolution method for queue-dependent servers in closed queueing networks. *Perform. Eval. Elsevier*, 1(1):7–18, 1981.

[102] M. Reiser and C. H. Sauer. Queueing network models: Methods of solution and their program implementation. In (K. M. Chandy and R. T. Yeh), editor, *Current Trends in Programming Methodology.* Prentice-Hall Inc., 1978.

[103] M. Resiser and S. S. Lavenberg. Mean Value Analysis of closed multichain queueing network. *J. ACM*, 27(2):313–320, 1980.

[104] J.A. Rolia and K.C. Sevcick. The methods of layers. *IEEE Trans. on Software Eng.*, 21(8):682–688, 1995.

[105] S. M. Ross. *Stochastic Processes.* John Wiley & Sons, 2nd edition, 1996.

[106] R. Sahner, K. Trivedi, and A. Puliafito. *Performance and Reliability Analysis of Computer Systems - An Example-Based Approach Using the SHARPE Software Package.* Kluwer Academic Publishers, 1996.

[107] C. H. Sauer. Computational algorithms for state-dependent queueing networks. *ACM Trans. Comput. Syst.*, 1(1):67–92, 1983.

[108] C. H. Sauer and K. M. Chandy. *Computer Systems performance modeling.* Prentice-Hall, Englewood Cliffs, 1981.

[109] M. Sereno. On closed support T-invariant and the traffic equations. *J. Appl. Probab.*, 35(2):473–481, 1988.

[110] M. Sereno. Towards a product form solution for stochastic process algebras. *The Computer Journal*, 38(7):622–632, December 1995.

[111] M. Sereno and G. Balbo. Mean Value Analysis of stochastic Petri nets. *Perform. Eval. Elsevier*, 29:35–62, 1997.

[112] R. Shassenberger. The insensitivity of stationary probabilities in networks of queues. *Journal of Applied Prob.*, 10:85–93, 1978.

[113] C. U. Smith. *Performance Engineering of Software Systems*. Addison-Wesley, 1990.

[114] J. Strelen. A generalization of Mean Value Analysis to higher moments: moment analysis. In *SIGMETRICS '86/PERFORMANCE '86: Proc. of 1986 ACM SIGMETRICS Int. Conf. on computer performance modelling, measurement and evaluation*, pages 129–140, New York, NY, 1986. ACM Press.

[115] R. Suri. Robustness of queuing network formulas. *J. ACM*, 30(3):564–594, 1983.

[116] H. M. Taylor and S. Karlin. *An Introduction To Stochastic Modeling*. Academic Press, 3rd edition, 1998.

[117] D. Towsley. Queuing network models with state-dependent routing. *J. ACM*, 27(2):323–337, 1980.

[118] K. S. Trivedi. *Probability and statistics with reliability, queuing and computer science applications*. Wiley-Interscience, second edition, 2002.

[119] S. Tucci and C. Sauer. The tree MVA algorithm. *Perform. Eval. Elsevier*, (5(3)):187–196, August 1985.

[120] N. van Dijk. *Queueing networks and product forms*. John Wiley, 1993.

[121] M. Vernon, J. Zahorjan, and E. D. Lazowska. A comparsion of performance Petri Nets and queueing network models. *Proc. 3rd Int. Workshop on Modelling Techniques and Performance Evaluation*, pages 181–192, 1987.

[122] M. Weber and E. Kindler. *Petri Net Technology for Communication-Based Systems*, chapter The Petri Net Markup Language, pages 124–144. H. Ehrig, W. Reisig, G. Rozenberg, H. Weber, 2003.

[123] P. Whittle. Partial balance and insensitivity. *Journal of Applied Prob.*, 22:168–175, 1985.

[124] A. Zimmermann, J. Freiheit, R. German, and G. Hommel. Petri net modelling and performability evaluation with timenet 3.0. In *TOOLS '00: Proc. of the 11th Int. Conf. on Computer Performance Evaluation: Modelling Techniques and Tools*, pages 188–202, London, UK, 2000. Springer-Verlag.

Index

List of PhD Theses

TD-2004-1 Moreno Marzolla
"Simulation-Based Performance Modeling of UML Software Architectures"

TD-2004-2 Paolo Palmerini
"On performance of data mining: from algorithms to management systems for data exploration"

TD-2005-1 Chiara Braghin
"Static Analysis of Security Properties in Mobile Ambients"

TD-2006-1 Fabrizio Furano
"Large scale data access: architectures and performance"

TD-2006-2 Damiano Macedonio
"Logics for Distributed Resources"

TD-2006-3 Matteo Maffei
"Dynamic Typing for Security Protocols"

TD-2006-4 Claudio Silvestri
"Distributed and Stream Data Mining Algorithms for Frequent Pattern Discovery"

TD-2007-1 Marco Giunti
"Secure Implementations of Typed Channel Abstractions"

TD-2007-2 Francesco Lelli
"Bringing Instruments to a Service-Oriented Interactive Grid"

TD-2007-3 Matteo Mordacchini
"Grid and Peer-to-Peer Resource Discovery Systems"

TD-2008-1 Claudio Lucchese
"High Performance Closed Frequent Itemsets Mining inspired by Emerging Computer Architectures"

TD-2008-2 Giulio Manzonetto
"Models and theories of lambda-calculus"

TD-2009-1 Fernando José Braz
"Warehousing and Mining Aggregate Measures Over Trajectories of Moving Objects"

TD-2009-2 Andrea Marin
"On the relations among product-form stochastic models"

TD-2009-3 Samuel Rota Bulò
"A game-theoretic framework for similarity-based data clustering"