FROM BCMP QUEUEING NETWORKS TO GENERALIZED STOCHASTIC PETRI NETS: AN ALGORITHM AND AN EQUIVALENCE DEFINITION

Simonetta Balsamo, Andrea Marin
Dipartimento di Informatica
Universita' Ca' Foscari di Venezia
via Torino, 155 Venice, Italy
{balsamo,marin}@dsi.unive.it

KEYWORDS
GSPN, BCMP theorem, Coxian random variable, multiclass system

ABSTRACT

In this paper we define an algorithm that converts a BCMP queueing network (QN) with multiple classes of customers into a generalized stochastic Petri net (GSPN). Product-form property of BCMP networks is preserved by the GSPN model. The algorithm can be embedded in a hybrid formalisms modelling tool. In fact a product-form model can be partially expressed in terms of queueing network and partially in terms of GSPN. Then the algorithm determines an equivalent GSPN representation of the whole system, which can be eventually analyzed using exact techniques (taking advantages of the product-form property) or by GSPN simulators. It is worthwhile pointing out that multiple classes QNs are considered; hence their GSPN representation is not trivial, as queueing disciplines must be represented.

INTRODUCTION

In this paper we consider different formalisms to model systems for performance evaluation purposes. We consider the integration of different types of models into a unique framework, in order to take advantage of the relative merit of each of the modelling formalism involved. To this aim we consider the relation between the class of queueing network models and the generalized stochastic Petri nets. Specifically, we propose an algorithm to transform a class of queueing networks into the corresponding stochastic Petri net model.

Queueing networks (QNs) are useful stochastic models for the performance evaluation of systems that consist of a set of customers which compete for a set of resources. The exact analysis of such kind of models, if possible, is usually computationally unfeasible due to the high cardinality of the set of all possible system states. Some classes of QNs have been introduced which allow a closed expression for the stationary probabilities, i.e., it can be expressed as product of functions that depend on the state and on the type of each node. The main result is known as BCMP theorem (Baskett et al. 1975) and it considers open, closed and mixed QNs that consist of multiple class of customers, probabilistic routing, Poisson arrivals and four types of service centers.

Generalized Stochastic Petri Nets (GSPNs) are stochastic models that represent systems with concurrency and synchronization and they are usually defined as models at a lower level of abstraction with respect to QNs. In fact the GSPN semantic is strictly defined in terms of transitions, places and rules of firing, that are the GSPN components. We formally introduce the GSPN definition in the next section. In the general case, GSPNs are hard to study because of the generation of their reachability sets which is an NP-hard problem. Product-form GSPNs are studied in (Coleman et al. 1996, Balbo et al. 2002) and they define the stationary state probability as the product of functions depending on the marking of each place. However, these GSPNs still do not overcome the problem of deciding whether a marking is reachable from the model initial state or not. As a consequence this limits the applicability of the algorithms defined for product-form GSPNs (Sereno and Balbo 1997, Coleman et al. 1996). Hence the solution of such models can be derived by simulation in many practical complex cases.

In this paper we present an algorithm that transforms a multiple class BCMP queueing network into a GSPN model. The task is not obvious for at least two reasons:
1- Multiple class QNs cannot be simply associated to state machines as for single class QNs. In fact theoretical results (Baskett et al. 1975, Chandy et al. 1977) have shown that the queueing discipline influences the performance measures in multiple class QNs. Hence the GSPN that models a specific multiple class queueing station depends on the correspondent queueing discipline, and the equivalence between the QN and GSPN models must be proved.
2- BCMP QNs allow some classes of stations to serve the customers with Coxian distributed service time. Moreover the service time distribution can be class dependent. Also in this case state machines cannot be used, and a different GSPN model has to be defined.

The proposed algorithm is based on some equivalence results presented in previous papers (Balbo et al. 2003, Balsamo and Marin 4-6 June 2007; October 23-25, 2007)

that we briefly recall in the section on related works. The proposed technique has the following properties:
- the modelling approach is modular and hierarchical. Indeed we define some GSPN blocks corresponding to queueing stations that can be combined into more complex systems preserving the product-form property.
- the GSPN models correspondent to the BCMP service stations can be combined with other no-BCMP product-form models. We have proved in (Balsamo and Marin 4-6 June 2007; October 23-25, 2007) that any system that holds $M \Rightarrow M$ Muntz (1972) property can be combined with our BCMP-equivalent models preserving the product-form solution.
The algorithm is based on the following idea:
- For each BCMP queueing center type it defines a correspondent equivalent GSPN model.
- The GSPN models obtained from the queueing center translations are connected by arcs and immediate transitions according to the QN routing probability matrix.
Theoretical results proved in (Balbo et al. 2003, Balsamo and Marin 4-6 June 2007; October 23-25, 2007) define the equivalence property between the GSPN and QN models, in terms of stationary state probability and average performance indices. Even if translating a QN into a GSPN increases the model complexity and reduces its readability, there are also some relevant advantages. First, the class of GSPN models is very expressive and its semantic if formally defined at a very low level. This means that given a model definition, the state representation can be obtained automatically. In general this does not happen when studying QNs where the system state has to be derived from a high level description of the queueing discipline. Therefore the class of GSPN models represents a suitable candidate for being the base model for a hybrid modelling tool. A second reason to use GSPN is related to the former one. We observe that product-form BCMP QNs exact analyzers or simulators usually do not allow the modeler to define new queuing centers with specific discipline. For example, to the best of our knowledge, modelling tools based on QNs do not allow the modeler to introduce in the network an MSCCC service center type defined in (Le Boudec 1986) that extends the BCMP theorem. Using GSPNs, one can define a model representing MSCCC discipline and then can embed it into the net. Therefore the performance indices can be derived, possibly by the product-form analysis.

MODELS IN PRODUCT FORM

In this section we introduce the formalisms that we use in the following and we briefly review the main definitions of the QNs and the GSPNs. We will limit our description to BCMP queueing networks.

BCMP Queueing networks

A queueing network consists of a set $\mathcal{C} = \{c_1, \ldots, c_N\}$ of N service centers or stations. Each service center has a scheduling discipline. BCMP QNs allow the following service disciplines: First Come First Served (FCFS), Last Come First Served with preemptive resume (LCFSPR), Processor Sharing (PS) and Infinite Servers (IS). In a QN, the customer enters a service center, waits in the queue for the service, gets the service, and finally it either exits the network or enters another service center. Customers moves among the service centers according to routing probabilities. At a given time, every customer belongs to a class, but there can be class switchings, i.e., a customer can change its class after being served at a station. The class of the customer influences the routing probabilities and the service time at the stations. We denote by R the number of classes. The classes are labeled by $1, \ldots, r, \ldots R$ and can be partitioned into chains. A chain permanently characterizes a customer. In order to simplify the notation we consider BCMP QNs with multiple chains but only one class for chain. Hence in this context the terms class and chain becomes synonymous. In the section on supported extension of the algorithm, we show how it is possible to model class switching. We use the following notation for QNs:
- $p_{ij}^{(c)}$ with $1 \leq i, j \leq N$ and $1 \leq c \leq R$ is the probability that a chain c customer goes to station j after being served by station i
- $p_{i0}^{(c)}$ with $1 \leq i \leq N$ and $1 \leq c \leq R$ is the probability that a chain c customer exits the network after being served by station i. Then the normalizing condition holds, i.e., $\sum_{j=0}^{N} p_{ij}^{(c)} = 1$, for $1 \leq i \leq N$ and $1 \leq c \leq R$
- $\mu_i^{(c)}$ with $1 \leq i \leq N$ and $1 \leq c \leq R$ is the mean service rate for a chain c customer at station i. If the service time is Coxian distributed and $L_i^{(c)}$ is the number of stages for chain c customers at station i, then $\mu_{\ell i}^{(c)}$, $1 \leq \ell \leq L_i^{(c)}$ denotes the mean service rate for a chain c customer at stage of service ℓ of station i.
- If chain c is open, i.e., external arrivals and departures from the system are allowed, then $\lambda^{(c)} > 0$ denotes the external arrival rate for class c customers. The arrival probability at node i and chain c is denoted by $p_{0i}^{(c)}$. It is defined such that $\sum_{i=1}^{N} p_{0i}^{(c)} = 1$, for $1 \leq c \leq R$. Then $\lambda^{(c)} p_{0i}^{(c)}$ is the external arrival rate of chain c customers to station i. If chain c is closed then $p_{0i}^{(c)} = 0$.

BCMP theorem considers four types of scheduling disciplines with some constraints. FCFS stations must have exponentially distributed service time. The service time must be chain-independent, i.e., $\mu = \mu^{(c)}$ for $1 \leq c \leq R$. LCFSPR, PS and IS station types have less restrictive conditions. The service time can be Coxian distributed and the mean service rate can depend on the customer being served. Let $\mathbf{n} = (\mathbf{n_1}, \ldots, \mathbf{n_N})$ be a vector whose components are R-dimension vector of vectors and where component $n_i^{(r)}$ represents the number of class r customers at station i. Then BCMP theorem (Baskett et al. 1975) states that, under stability conditions, the stationary probability distribution is given by:

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

where $g_i(\mathbf{n_i})$ is a function defined according to station i type, $d(\mathbf{n})$ is a function defined for state dependent arrival rates and G is a normalizing constant. From the stationary state distribution one can derive several average performance indices.
BCMP theorem holds if the service rates depend on the state of the network. Several load-dependent service time functions have been defined in (Baskett et al. 1975). However, for the sake of clarity, we first study stations with a finite number of identical servers and scheduling discipline IS, PS, LCFSPR, FCFS.

Generalized Stochastic Petri Nets
In this section we briefly recall the Generalized Stochastic Petri Nets (GSPN). We consider the notation for GSPN introduced in (Marsan et al. 1995). In order to allow marking dependent probabilities for solving conflicts among immediate transitions we use the techniques discussed in (Chiola et al. 1993). Let us define a marked Stochastic Petri Net which consists of a 8-tuple as follows:

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

where:
- $\mathcal{P} = \{P_1, \dots, P_M\}$ is the set of M places,
- $\mathcal{T} = \{t_1, \dots, t_N\}$ 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, $1 \le i \le N$, $1 \le j \le M$,
- $O(t_i, P_j) : \mathcal{T} \times \mathcal{P} \to \mathbb{N}$ is the output function, $1 \le i \le N$, $1 \le j \le M$,
- $H(t_i, P_j) : \mathcal{T} \times \mathcal{P} \to \mathbb{N}$ is the inhibition function, $1 \le i \le N$, $1 \le j \le M$,
- $\Pi(t_i) : \mathcal{T} \to \mathbb{N}$ is a function that specifies the priority of transition t_i, $1 \le i \le N$,
- $\mathbf{m} \in \mathbb{N}^M$ denotes a marking or state of the net, where m_i represents the number of tokens in place P_i, $1 \le i \le N$,
- $w(t_i, \mathbf{m}) : \mathcal{T} \times \mathbb{N}^M \to \mathbb{R}$ 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,
- $\mathbf{m_0} \in \mathbb{N}^M$ represents the initial state of the GSPN, i.e., the number of tokens in each place at the initial state.

We consider ordinary nets, i.e., functions I, O and H take values in $\{0, 1\}$. 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: $\mathbf{I}(t_i) = (i_1, \dots, i_M)$ where $i_j = I(t_i, P_j)$, $\mathbf{O}(t_i) = (o_1, \dots, o_M)$ where $o_j = O(t_i, P_j)$ and $\mathbf{H}(t_i) = (h_1, \dots, h_M)$ 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 $m_i \ge I(t_a, P_i)$ and $m_i < H(t_a, P_i)$ for $i = 1, \dots, M$ and no other transition of higher priority is enabled. We consider just two priority levels, 0 and 1. 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 $\mathbf{m_0}$. 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 it is defined as: $p(t_i, \mathbf{m}) = w(t_i, \mathbf{m}) / \sum_{t_j \in \mathcal{T}_\alpha} w(t_j, \mathbf{m})$.
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 $\mathbf{O}_j(t_i)$, the output function by $O_j(t_i, P_j)$ and its firing probability by $d(t_i, j)$ where $\sum_j d(t_i, j) = 1$. Note that this is not a real extension to the model definition.
A GSPN is represented by a graph with the following conventions: timed transitions are white filled boxes, immediate transitions are black filled boxes, places are circles, if $I(t_i, P_j) > 0$ we draw an arrow from P_j to t_i labeled with $I(t_i, P_j)$, if $O(t_i, P_j) > 0$ we draw an arrow from t_i to P_j labeled 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 labeled 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 $j = 1, \dots, M$.For ordinary nets we do not use labels for the arrows.
GSPN analysis consists in finding the steady state probability for each tangible marking of the reachability set, from which one can derive other average performance indices. Some analysis techniques are presented in (Marsan et al. 1995). GSPN in product-form are studied in (Balbo et al. 2002) and they are defined as GSPNs reducible to SPNs in Coleman, Henderson et al. product-form (Coleman et al. 1996). GSPN product-form can be also identified as a special case of Boucherie product-form definition (Boucherie 1994). The product-from GSPN models that we introduce with the proposed algorithm do not belong to any of the previous product-form classes.

RELATED WORKS
In this section we review some previous results on relations between BCMP queueing network and GSPNs. Hybrid modelling and combining different classes of

stochastic models has been studied in literature by considering various types of models. For example in (Balbo et al. 1998) the authors illustrate a hybrid GSPN/QN modelling technique although product-form is not deeply explored. In (Bause 1993) the author studies an hybrid formalism SPN/QN. He considers a SPN with product-form (Coleman et al. 1996) and then introduces a new place type which exhibits a queueing station behavior. Then he shows that the whole system maintains the product-form property. Our approach differs from the previous ones for several reasons:
- it considers multiple class QNs in product-form.
- it combines GSPNs and QNs so it allows the modeler to use immediate transitions.
- an hybrid model is translated into a standard GSPN model. Hence existing GSPN analyzers can be used in order to simulate the net or to obtain exact results.

The definition of a GSPN model that is equivalent to a multiple-class queueing station is presented in (Balsamo and Marin 4-6 June 2007; October 23-25, 2007) and we shall now informally recall the main idea. Let us consider a multiple class FCFS station. In order to define the GSPN model, when we consider the system state, as mentioned above, we cannot just count the number of customers in the stations for every class, as this technique would ignore the queueing discipline, leading to incorrect results as discussed in (Balbo et al. 2003). For each customer class we use a place for representing the customers in queue, and a place for the customer in service. Another place stores as many tokens as the free servers are. If there is a free server and a set of waiting customers an immediate transition puts a customer in service. The problem is how to choose which customer has to get the service. In (Balsamo and Marin 4-6 June 2007) we showed that we can choose the customer to put in service probabilistically, according to the uniform distribution. This is obtained by an appropriate definition of the immediate transition weights. Service time is simply modelled by a timed transition. Figure 1 illustrates the GSPN model equivalent to a two classes FCFS station with 3 identical servers.

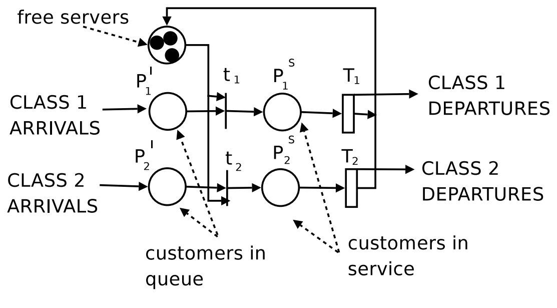

Figure 1: FCFS equivalent representation by GSPN

Representing an LCFSPR queueing station is more complex than FCFS, because it allows Coxian distributed service times, so we have to represent every stage of service. In the GSPN model, as soon as a token representing a new customer arrives, two things can happen: 1) a customer is preempted. This happens if there are not free servers. The customer to preempt is chosen probabilistically with uniform distribution. This is modelled by immediate transitions. 2) The customer enters in service immediately. This happens if there is at least one free server. This is modelled by immediate transitions. When a token representing a customer leaves the net for a service completion, a server becomes free if there are not preempted customers, or a preempted customers is resumed otherwise. The customer to be resumed is chosen probabilistically with uniform distribution. When a customer is preempted, the correspondent token has to be stored in an appropriate place that will identify the customer class and the stage of serviced reached.

The equivalence between the models and the QN stations is defined and proved in (Balsamo and Marin 4-6 June 2007; October 23-25, 2007) and is based on $M \Rightarrow M$ property. (Muntz 1972). GSPN models can be combined with any other models holding $M \Rightarrow M$ property obtaining a product-form solution. In order to decide whether a system holds $M \Rightarrow M$ property the space state has to be built, so it cannot be considered a structural property. Deciding how this property can be translated into general structural GSPN conditions is still an open problem.

ALGORITHM DEFINITION

We shall now define the algorithm that converts a BCMP QN with multiple classes of customers into a product-form GSPN. 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 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.
- $\Omega = \{c_1, \dots c_N\}$ 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 for a customer 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.
- $\lambda = (\lambda_1, \dots, \lambda_R)$ 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 $\mathbf{K} = (K_1, \dots, K_R)$ 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 correspondent service center number. For example $P^S_{r,\ell,i}$ is a place defined in the i-th service center translation, correspondent 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 correspondent place names. For example m^S_i is the number of initial tokens in place P^S_i.
- The arcs are specified in terms on input, output, 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 $\leftarrow$ 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 and 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 2). 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 us consider station i and suppose that $p^{(r)}_{ij} > 0$ and $p^{(r)}_{ik} > 0$. 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 3. The main structure of the algorithm is simple and is shown by Algorithm 1. The main cycle of

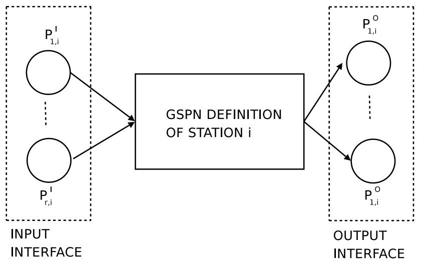

Figure 2: Modularity of station equivalent GSPN blocks

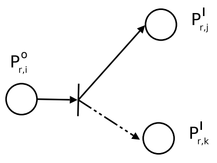

Figure 3: Modelling the QN probabilistic routing. In this example the output vector of transition $t^Z_{r,i}$ is determined probabilistically and $O_0(t^Z_{r,i}, P^I_{r,j}) = 1$, $O_0(t^Z_{r,i}, P^I_{r,k}) = 1$ and $d(t^Z_{r,i}, 0) = p^{(r)}_{ij}$, $d(t^Z_{r,i}, 1) = p^{(r)}_{ik}$.

the algorithm considers each service center of the QN and executes 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.
FCFSBlock is defined by Algorithm 2. 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 3. 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 resume of a customer. 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 4. *PSBlock* is similar to *ISBlock* so the same Algorithm 4

```
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 1: Main program

```
/* 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 2: FCFSBlock

```
/* Add a place for the free servers                 */
$\mathcal{P} := \mathcal{P} \cup \{P_i^S\}$;
$\mathcal{P} := \mathcal{P} \cup \{P_i^T\}$;
foreach $i \in c_i.\mathcal{R}$ do
    /* Set up the arrival places and
       transitions                                  */
    $\mathcal{P} := \mathcal{P} \cup \{P_{r,i}^I, P_{r,i}^T\}$;
    $\mathcal{T} := \mathcal{T} \cup \{t_{r,i}^I\}$;
    $I(t_{r,i}^I, P_{r,i}^I) := 1$;
    $O(t_{r,i}^I, P_{r,i}^T) := 1$;
    $O(t_{r,i}^I, P_i^T) := 1$;
    $w(t_{r,i}^I) := 1$; $\Pi(t_{r,i}^I) := 1$;
    /* Set up the output places                     */
    $\mathcal{P} := \mathcal{P} \cup \{P_{r,i}^O\}$;
    /* Add the other needed places and
       transitions for each class                   */
    for $\ell := 1$ to $c_i.L_r$ do
        $\mathcal{P} := \mathcal{P} \cup \{P_{r,\ell,i}^Q, P_{r,\ell,i}^S\}$;
        /* Add transitions which model the
           service time                             */
        $\mathcal{T} := \mathcal{T} \cup \{T_{r,\ell,i}\}$;
        $w(T_{r,\ell,i}) \leftarrow c_i.\mu_\ell^{(r)} * m_{r,\ell,i}^S$;
        $\Pi(T_{r,\ell,i}) := 0$;
        $I(T_{r,\ell,i}, P_{r,\ell,i}^S) := 1$;
        if $\ell \neq c_i.L_r$ then
            $O_0(T_{r,\ell,i}, P_{r,\ell+1,i}^S) := 1$;
            $d(T_{r,\ell,i}, 0) := c_i.a_\ell^{(r)}$;
        end
        $O_1(T_{r,\ell,i}, P_i^S) := 1$;
        $O_1(T_{r,\ell,i}, P_i^O) := 1$;
        $d(T_{r,\ell,i}, 1) := c_i.b_\ell^{(r)}$;
        /* Add transitions modelling the
           preemption (label $P$)                   */
        $\mathcal{T} := \mathcal{T} \cup \{t_{r,\ell,i}^P\}$; $I(t_{r,\ell,i}^P, P_{r,\ell,i}^S) := 1$;
        $I(t_{r,\ell,i}^P, P_i^T) := 1$; $H(t_{r,\ell,i}^P, P_i^S) := 1$;
        $O(t_{r,\ell,i}^P, P_{r,\ell,i}^Q) := 1$; $w(t_{r,\ell,i}^P) \leftarrow m_{r,\ell,i}^S$;
        $\Pi(t_{r,\ell,i}^P) := 1$;
        /* Add transitions modelling the resume
           (label $R$)                              */
        $\mathcal{T} := \mathcal{T} \cup \{t_{r,\ell,i}^R\}$; $I(t_{r,\ell,i}^R, P_{r,\ell,i}^Q) := 1$;
        $I(t_{r,\ell,i}^R, P_i^S) := 1$; $H(t_{r,\ell,i}^R, P_i^T) := 1$;
        $O(t_{r,\ell,i}^R, P_{r,\ell,i}^S) := 1$; $w(t_{r,\ell,i}^R) \leftarrow m_{r,\ell,i}^Q$;
        $\Pi(t_{r,\ell,i}^R) := 1$;
    end
    /* Add transitions modelling the customers
       entering in stage of service 1               */
    $\mathcal{T} = \mathcal{T} \cup \{t_{r,0,i}^R\}$; $I(t_{r,0,i}^R, P_i^S) := 1$;
    $I(t_{r,0,i}^R, P_i^T) := 1$; $I(t_{r,0,i}^R, P_{r,i}^T) := 1$;
    $O(t_{r,0,i}^R, P_{r,1,i}^S) := 1$; $w(t_{r,0,i}^R) := 1$;
    $\Pi(t_{r,0,i}^R) := 1$;
    $M_i^S := 0$;
end
```

Algorithm 3: LCFSPRBlock

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 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.

```
/* 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 $\ell$ 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 4: ISBlock

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 6.

SUPPORTED EXTENSIONS

The proposed algorithm that transforms BCMP QNs into GSPNs can support some 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

```
/* model the QN routing by GSPN                        */
foreach $c_i \in \Omega$ do
    foreach $r \in c_i.\mathcal{R}$ do
        /* Model internal routing                      */
        $\mathcal{T} := \mathcal{T} \cup \{t^Z_{r,i}\}$;
        $I(t^Z_{r,i}, P^O_{r,i}) := 1$;
        $w(t^Z_{r,i}) := 1; \Pi(t^Z_{r,i}) := 1$;
        $f := 0$;
        foreach $c_j \in \Omega$ 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 5: ROUTINGBlock

```
for $r := 1$ to $R$ do
    if $\lambda_r > 0$ then
        /* Open chain                                  */
        $\mathcal{T} := \mathcal{T} \cup \{T_{r,0}\}$;
        $w(T_{r,0}) := \lambda_r$;
        $f := 0$;
        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 6: CHAINSBlock

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 of 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.

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 4 (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 Le Boudec (1986) and it is proved and it holds $M \Rightarrow M$ 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, denoted 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 4 (b). The parameters of the GSPN are completely defined by the various steps of the algorithm.

As the three blocks hold $M \Rightarrow M$ 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.

CONCLUSIONS

In this paper we have defined an algorithm that given a BCMP QN returns an equivalent GSPN. The algorithm computational complexity is linear with the number of stations of the QN and with number of no-zero elements of its routing probability matrix (in the worst case, without class switching it is of $O(N^2R)$ operations, and with class switching it is $O((NR)^2)$ operations, where N is the number of the QN stations and R the number of classes). The proposed algorithm is defined in terms of a mathematical definition of the models, it can be easily rewritten in order to deal with well-defined languages for representing PNs and their extensions such as PNML Weber and Kindler (2003). Further research can have two directions. From the theoretical viewpoint an open problem is the definition of general structural sufficient conditions on GSPN models that ensure that a model holds the $M \Rightarrow M$ property. This could allow an automatic verification of the conditions to combine a GSPN block with others ones holding $M \Rightarrow M$ property to obtain a product-form model. From the practical viewpoint, open research concerns possible language extensions to represent GSPNs in order to be able to represent the following features:

- 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 holds $M \Rightarrow M$ 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 holding 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. When the product-form expression is not known, the model can be still studied by simulation, and the theoretical results guarantee that the performance indices are the same of the original hybrid model.

REFERENCES

Balbo G.; Bruell S.C.; and Ghanta S., 1998. *Combining queueing network and generalized stochastic Petri nets for the solution of complex models of system behavior. IEEE Trans on Computers*, 37, 1251–1268.

Balbo G.; Bruell S.C.; and Sereno M., 2002. *Product Form Solution for Generalized Stochastic Petri Nets. IEEE Trans on Software Eng*, 28, 915–932.

Balbo G.; Bruell S.C.; and Sereno M., 2003. *On the rela-*

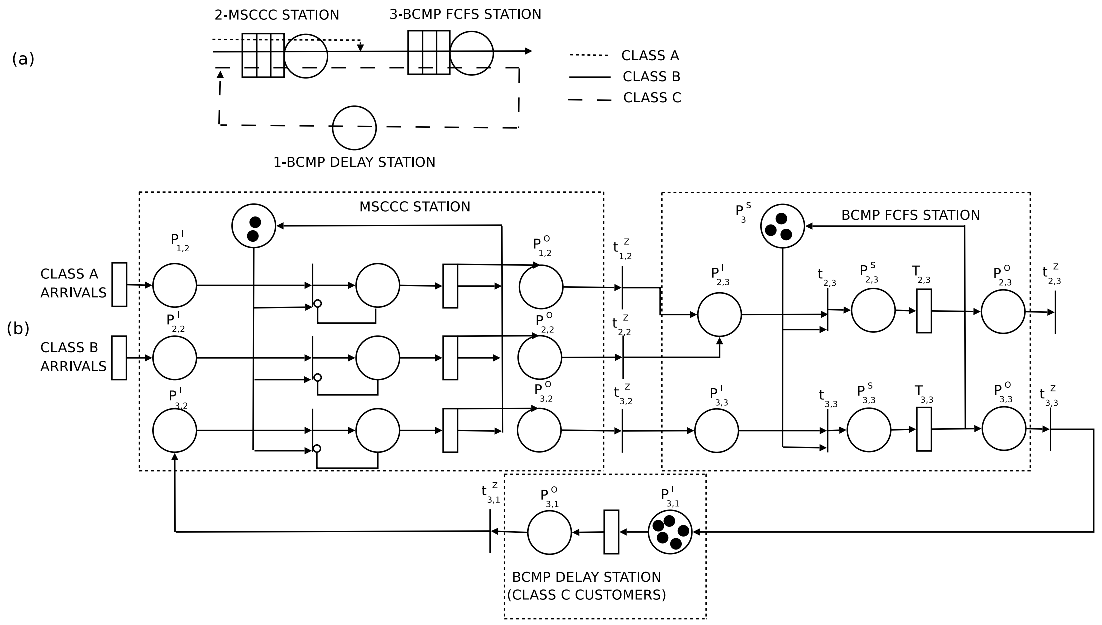

Figure 4: (a) System modelled by a no-BCMP queueing network. (b) System modelled by a product-form GSPN.

tions between BCMP Queueing Networks and Product Form Solution Stochastic Petri Nets. Proc of 10th International Workshop on Petri Nets and Performance Models, 2003, 103–112.

Balsamo S. and Marin A., 4-6 June 2007. *On representing multiclass M/M/k queues by generalized stochastic Petri nets.* In *Proc. of ECMS/ASMTA-2007 Conference.* Prague, Czech Republic, 121–128.

Balsamo S. and Marin A., October 23-25, 2007. *Representing LCFSPR BCMP service centers with Coxian service time distribution.* In *Proc. of Valuetools '07 conference.* Nantes, France.

Baskett F.; Chandy K.M.; Muntz R.R.; and Palacios F.G., 1975. *Open, Closed, and Mixed Networks of Queues with Different Classes of Customers. J ACM*, 22, no. 2, 248–260.

Bause F., 1993. *Queueing Petri Nets: A Formalism for the Combined Qualitative and Quantitative Analysis of Systems.* In *5th International Workshop on Petri Nets and Performance Models.* Toulouse (France), 14–23.

Boucherie R.J., 1994. *A Characterisation of independence for competing Markov chains with applications to stochastic Petri nets. IEEE Tran on Software Eng*, 20, no. 7, 536–544.

Chandy K.M.; John H. Howard J.; and Towsley D.F., 1977. *Product Form and Local Balance in Queueing Networks. J ACM*, 24, no. 2, 250–263.

Chiola G.; Marsan M.A.; Balbo G.; and Conte G., 1993. *Generalized stochastic Petri nets: a definition at the net level and its implications. IEEE Trans on Software Eng*, 19, no. 2, 89–107.

Coleman J.L.; Henderson W.; and Taylor P.G., 1996. *Product form equilibrium distributions and a convolution algorithm for Stochastic Petri nets. Performance Evaluation*, 26, 159–180.

Le Boudec J.Y., 1986. *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.* ACM Press, New York, NY, 78–91.

Marsan M.A.; Balbo G.; Conte G.; Donatelli S.; and Franceschinis G., 1995. *Modelling with generalized stochastic Petri nets.* Wiley.

Muntz R.R., 1972. *Poisson Departure Processes and Queueing Networks.* Tech. Rep. IBM Research Report RC4145, Yorktown Heights, New York.

Sereno M. and Balbo G., 1997. *Mean Value Analysis of stochastic Petri nets. Performance Evaluation*, 29, 35–62.

Weber M. and Kindler E., 2003. *Petri Net Technology for Communication-Based Systems*, H. Ehrig, W. Reisig, G. Rozenberg, H. Weber, chap. The Petri Net Markup Language. 124–144.