# Analysis and Evaluation of non-Markovian Stochastic Petri Nets

András Horváth[1], Antonio Puliafito[2], Marco Scarpa[2], and Miklós Telek[1]

[1] Department of Telecommunications
Technical University of Budapest, 1521 Budapest, Hungary
[2] Istituto di Informatica e Telecomunicazioni
Università di Catania, 95125 Catania, Italy

**Abstract.** In order to extend their applicability to more complex situations, in this paper we present a new approach for the analysis of non-Markovian Stochastic Petri Net (*NMSPN*) models, which is based on a discrete time approximation of the stochastic behavior of the marking process. The proposed approach, which resulted in a new modeling tool for the analysis of *NMSPN*s called WebSPN, allows to analyze a wider class of *PN* models with *prd, prs* and *pri* concurrently enabled generally distributed transitions. This implies the possibility of dealing with very complex systems with arbitrarily distributed events with very complex interrelations among each other. The adopted technique is described, an application example is solved and the results are carefully analyzed in order to demonstrate the validity of the proposed approach.

**Key Words:** *Non-Markovian Stochastic Petri Nets, Discrete time Markov chain (DTMC), expansion techniques, performance and dependability analysis*

## 1 Introduction

Petri nets are commonly viewed as a valid tool for the qualitative and quantitative study of computer systems [2]. Over the years, many stochastic extensions to the basic Petri net model have been proposed. Dealing with non-exponentially distributed events is an extension that widened the field of applicability of this modeling approach to real situations. There are a great number of real circumstances in which deterministic or generally distributed event times occur. Events such as timeouts in a protocol, service times in a manufacturing system and memory access or instruction execution in a low-level hardware or software have durations which are constant or have a very low coefficient of variation. Choi et al. have shown that the marking process underlying a Stochastic Petri Net (*SPN*), where at most one generally distributed transition is enabled in each marking, belongs to the class of Markov Regenerative Processes (*MRGP*s) [7]. Following the line opened in [7], different approaches have been proposed to deal with non-Markovian systems [8, 14, 16].

All the above literature on Markov Regenerative *SPN*s (*MRSPN*s) implicitly assumed an *enabling* memory policy (as it is defined in [1]). The transient analysis of a class of *NMSPN*s with age memory policy ([1]) was provided in [5], and a preemption mechanism, different than the ones considered in [1], was introduced and analyzed in [4]. Following the common terminology used when dealing with queuing systems,

the stochastic behavior of the transitions of a *SPN* model has been classified as *preemptive repeat different (prd)*, *preemptive resume (prs)* and *preemptive repeat identical (pri)*, respectively. These stochastic extensions have increased the descriptive power of *SPN*s, as well as the computational effort required for their solution.

Many *SPN* modeling tools have recently been proposed or developed (e.g. ESP [11], GreatSPN [6], SPNP [9], DSPNExpress [16], TimeNet [13], UltraSAN [10]). Some of the above tools have also implemented the possibility of including some non-Markovian features thus extending the range of applicability of PNs. Their main limitations regard the kind and number of generally distributed firing time (GEN) transitions and their associated preemption policy. A very limited number of simultaneously enabled GEN transitions is allowed. And usually it reduces to only one. Further, the *preemptive repeat different* (*prd*) policy is the only adopted. The *preemptive resume* (*prs*) and the recently proposed *preemptive repeat identical* (*pri*) policies [4], although very powerful, are basically not yet implemented. The first restriction can be relaxed by the analytical results available for the analysis of *PN* with non-overlapping *prs* general transitions [5], and there is an active research to find the proper way to analyze *PN* with concurrently active general transitions [16, 17].

A possible approach for the analysis of *SPN* models, with concurrently active *prs* and *prd* general transitions, is through the continuous time Phase type (*CPH*) approximation of generally distributed firing times [11]. With this technique, the marking process of the *NMSPN* is approximated by a continuous time Markov chain with an expanded state space [11].

In this paper, we discuss a modeling technique for the analysis of *NMSPN*s that relaxes some of the restrictions present in currently available *SPN* analysis packages. This approach is based on a discrete time approximation of the stochastic behavior of the marking process, hence it can be considered as a discrete time version of the phase type expansion technique. A similar approach can be found in [9], where Discrete Deterministic and Stochastic PNs (DDSPNs) are presented and race policies equivalent to our *prd* and *prs* policies are considered. The main differences with our approach consist in the intrinsic assumption of a discrete time model, the lack of the *pri* policy and the absence of a full implementation of the proposed algorithm. A new modeling tool for the analysis of non-Markovian stochastic Petri nets, called *WebSPN* [15] has been successfully implemented. The approach we propose offers some new features which result in the possibility to analyze a wider class of *NMSPN* models. The main advantages of this method consist of the possibility to evaluate *SPN*s with transitions of *pri* type with finite firing time, besides the more traditional *prd* and *prs*, and to analyze models with concurrently enabled generally distributed transitions of any kind.

## 2 Introducing Petri Nets and Preemption Policies

A timed Petri net is a tuple PN=$(\mathcal{P}, \mathcal{T}, \mathcal{G}, \mathcal{A}, \mathcal{I}, \mathcal{O}, \mathcal{H}, M_0)$ where: $\mathcal{P}$ is the set of places; $\mathcal{T}$ is the set of transitions; $\mathcal{G}$ is the set of random variables $\gamma_g$ associated to transitions; $\mathcal{A}$ is the set of age variables $a_g$ associate to transitions; $\mathcal{I}, \mathcal{O}$ and $\mathcal{H}$ are

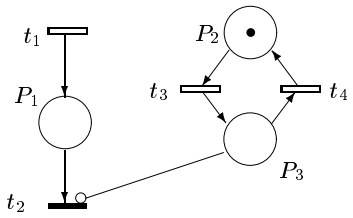


**Fig. 1.** Petri net model of one server.

respectively the set of input, output and inhibitor functions ($\mathcal{I} \subset \mathcal{P} \times \mathcal{T}$, $\mathcal{O} \subset \mathcal{T} \times \mathcal{P}$, $\mathcal{H} \subset \mathcal{P} \times \mathcal{T}$), providing their multiplicity; $M_0$ is the initial marking[1].

The firing of an enabled transition $t_k$, in a given marking $M_i$, generates another marking $M_j$. $M_j$ is said directly reachable from $M_i$ ($M_i \rightarrow^{t_k} M_j$). Starting from the initial marking $M_0$, the transitive closure of $\rightarrow$ generates the reachability graph $RG(M_0)$ (the set of all reachable markings from $M_0$).

A consistent way to introduce memory into a *SPN* is provided in [1] and extended in [5]. Each timed transition $t_g$ is assigned a general random firing time $\gamma_g$ with a cumulative distribution function $G_g(t)$. A clock, associated to each transition, counts the time in which the transition has been enabled. An *age variable* $a_g$ associated to the timed transition $t_g$ keeps track of the clock count. A timed transition fires as soon as the memory variable $a_g$ reaches the value of the firing time $\gamma_g$.

A timed transition has to be characterized both in terms of the distribution function of the random firing time and also of its behavior when a preemption occurs. Thus, a preemption policy is required to fully describe the behavior of a timed transition. In this paper we prefer an informal approach to the definition of preemption policies through the example of Figure 1.

The Petri net on Figure 1 models a server with exponential arrivals (transition $t_1$) and general service time (transition $t_2$). Waiting customers are represented by the tokens in place $P_1$. The server is randomly preempted by higher priority jobs (transition $t_3$) for an exponentially distributed amount of time (transition $t_4$), as shown by the inhibitor arc from place $P_3$ to transition $t_2$.

When a customer arrives to a server, a specific service requirement $\gamma_g$ has to be completed. The amount of computation required is sampled from the distribution function $F_g(t)$ of the service time. The optimal case is when the server is able to complete the job before an interruption occurs. However, the server may be interrupted after processing only a portion of the submitted job. In this case the whole behavior is strongly affected by the preemption policy and the whole performances will depend on the strategy adopted to deal with the preempted job, as described in the following:

- The server drops the customer it was dealing with before the interruption.
- The server goes back to the preempted customer who still maintains the original work requirement $\gamma_g$.

[1] A marking $M_i$ is a tuple, whose cardinality is $||\mathcal{P}||$, recording the number of tokens in each place.

– The server also returns to the same customer who still has the same work requirement $\gamma_g$.

According to [5] and [4], the previous policies are referred to as *preemptive repeat different (prd)*, *preemptive resume (prs)* and *preemptive repeat identical (pri)*, respectively. Note that in [1] the authors indicated the *prd* and *prs* type policies as enabling and age type. The *pri* policy was introduced for the first time in [4]. The *prd* policy is the only considered in the available tools modeling non-Markovian *SPN* [16, 13, 10]. The ESP tool [11] allows to deal with *prs* policy through a continuous time *PH* approximation. Recently German developed a tool with Mathematica package where the *prs* policy is also implemented adopting the method of supplementary variable [12].

From the previous discussion it is clear that the main difficulty in analyzing stochastic Petri nets with general transitions is related to the fact that the underlying discrete state marking process is no longer a CTMC, as its future evolution depends on the past history. Below, we call general (GEN) transitions both the transitions with generally distributed firing time (including the deterministic ones) and the exponentially distributed firing time transitions of *pri* type. For a transition with exponentially distributed firing time the *prd* and the *prs* policies have the same effect, due to the memoryless property. We denote these transitions as EXP transitions. For a transition with deterministic firing time, the *prd* and the *pri* policies have the same effect, since a resampling of the firing time results in the same firing time sample each time.

According to this memory concept, at any time the marking and the individual memory associated with the GEN transitions of a *NMSPN* only determine the future stochastic behavior of the *NMSPN*. This means that the marking process together with the memory process of the GEN transitions is a Markov process.

Below we make a distinction between enabled and active transitions. In Fact, a GEN transition may be active (the age variable $a_g$ is between 0 for a *prs* transition or the threshold value $\gamma_g$ is already set up for a *pri* transition) but not enabled.

The main idea behind our proposed discrete time approach is to discretize the continuous memory process and the time to obtain a Discrete time Markov chain (DTMC) that approximates the stochastic behavior of the compound Markov process. The time access is divided into equal intervals of size $\delta$, while we use the concept of discrete phase type distributions (*DPH*) to discretize the memory process when it is possible.

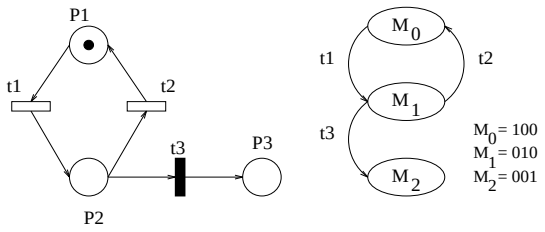


**Fig. 2.** *SPN* with one GEN transition

## 3 Discrete-time approach

In order to provide a better explanation on how to approximate the stochastic behavior of continuous time *SPN*s, the *SPN* shown in Figure 2 is considered as an example. This *SPN* models a system that alternates between two conditions: a fully operative state (token in place $P_2$), where useful work can be produced, and a failure state (token in place $P_1$), where the system does not perform any work. The EXP transitions $t_1$ and $t_2$ describe the changes in the system state from operational to failed, and vice versa. Transition $t_3$ models the duration of the work to be performed, and it is assumed to be non-exponentially distributed. In this example the *DPH* [3] distribution, depicted in Figure 3a, with generator $P = \{P_{ij}\}$ and initial probability $\alpha = \{1, 0, 0, \ldots\}$ is used to approximate the firing time of $t_3$. According to this *DPH* structure, the firing of transition $t_3$ can happen when the *DPH* is either in phase 2 or in phase 4, because in those phases there are arcs towards the absorbing phase 5 of *DPH*. We want to stress that the *DPH* of Figure 3 is used only as an example to show how our approach works, but, in general, there are no restriction to the usable *DPH*s to approximate the firing time of a GEN transition. Similarly, figure 3b depicts the *DPH* we have adopted to approximate the firing of an exponentially distributed transition, where $\lambda$ is the firing rate and $\delta$ is the approximation step.

In this paper we assume that the *DPH* distribution starts from the first phase with probability 1. This assumption is not restrictive since any acyclic *DPH* distribution can be represented with an acyclic *DPH* distribution of the same order starting from the first phase with probability 1 [3], and a general *DPH* distribution of order $n$ with generator $P$ and initial probability vector $\alpha$ can be represented as $P'$ and $\alpha'$ of order $n + 1$, where

$$P' = \begin{array}{|c|c|} \hline 0 & \alpha P \\ \hline 0 & P \\ \hline \end{array} \qquad \text{and} \qquad \alpha' = \{1, 0, \ldots, 0\}$$

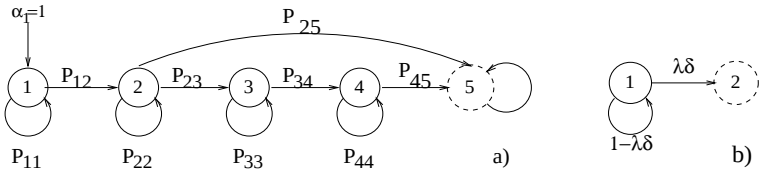


**Fig. 3.** The *DPH* approximation of the firing time of $t_3$

### 3.1 *SPN* with one generally distributed *prd* transition

Let us suppose that the GEN transition $t_3$ is associated with a *prd* memory policy. Using *DPH* distributions, the state of the expanded DTMC is defined as a pair of non negative integers $(i, u)$, where $i$ is the index of a marking ($M_i \in RG(M_0)$), and $u$ is a phase of the *DPH* associated with the GEN transition. Thus, $u$ is used to capture

the "memory" that is necessary to model the GEN transitions. $u = \diamond$ denotes that the process is in a state where the general transition is not influential (i.e. it has no memory). If $1 \leq u \leq \nu$, the GEN transition is enabled. The pair $(i, u)$ will be called *descriptor* and identifies the state of the expanded DTMC.

Figure 4 gives the DTMC constructed to approximate the stochastic behavior of the Petri net depicted in Figure 2. The chain is derived from the reachability graph. All the states in the reachability graph are examined, and the DTMC is generated depending on the transitions enabled in each of them. Each marking in the original continuous process produces a set of states of the expanded DTMC characterized by the same index $i$ in the descriptor $(i, u)$. All the states with the same marking index in its descriptor constitute a *macrostate*. Of course, the expanded process has as many macrostates as the number of markings of the continuous process. In Figure 4 the three macrostates are outlined by ellipses with the name of the marking depicted nearby.

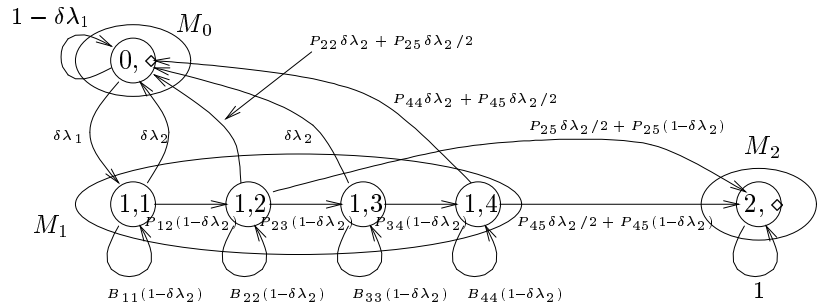


**Fig. 4.** DTMC approximation of the *SPN* on Figure 2 with *prd* transition

In marking $M_0$ only one EXP transition is enabled, so that memory does not need to be maintained, and the macrostate has only one state, the state $(0, \diamond)$. From this macrostate only two arcs can exit, relating to the firing or not of the EXP transition. Similar considerations can be done with reference to the marking $M_2$. In this marking, no transitions are enabled, thus no memory is needed.

Conversely, in marking $M_1$ an EXP and a GEN transition are enabled; then the marking is expanded to describe the evolution of the GEN transition using the *DPH*. The macrostate corresponding to this marking has the same number of states as the number of phases of the *DPH*: the states with descriptor $(1, u)$, with $1 \leq u \leq 4$. The absorbing state of *DPH* is not used to expand the marking into the macrostate, because it represents the firing of the GEN transition (therefore a change into a new marking). The states in the macrostate describe the evolution of the transition $t_3$.

State $(0, \diamond)$ corresponds to the initial marking $M_0$ shown in Figure 2. Because of the presence of transition $t_1$, in marking $M_0$ only two events can occur in a time slot: $t_1$ does not fire, $t_1$ fires. Thus the state $(0, \diamond)$ has two outgoing arcs: one of them enters the same state to model the event related to the fact that $t_1$ does not fire in $\delta$, the other one produces a state change since it is related to the firing of $t_1$. The firing of $t_1$ produces a change in marking $M_1$, where the GEN transition $t_3$ becomes enabled.

Since $t_3$ is a *prd* transition, when it becomes enabled its age memory starts from zero. This means that the DTMC enters the first phase of the *DPH*, in the example the state with descriptor $(1,1)$. Since a step of the DTMC corresponds to a time slot of length $\delta$, the one step probabilities of the two outgoing arcs model the firing or not of the enabled EXP transition in an interval of length $\delta$. Using the first order approximation for the exponential function it is easy to realize that: $p_{(0,\diamond)\to(1,1)} = P\{t_1 \text{ fires } |(0,\diamond))\} = \delta\,\lambda_1$ and $p_{(0,\diamond)\to(0,\diamond)} = P\{t_1 \text{ does not fire } |(0,\diamond))\} = 1 - \delta\,\lambda_1$, where $\lambda_1$ is the rate of the EXP transition $t_1$.

As we have already said, the macrostate with states $(1,u)$, with $1 \leq u \leq 4$, corresponds to the marking $M_1$. The marking process remains in such marking till one of the two transitions $t_2$ or $t_3$ fires. If both of them do not fire, the marking does not change. This means that the DTMC stays into the macrostate, and only passages between two phases of the *DPH* are possible. The one step probability must take into account that the EXP transition does not fire, thus the probability between a state with descriptor $(1,u)$ to one with descriptor $(1,v)$ is: $p_{(1,u)\to(1,v)} = P_{uv}(1 - \delta\lambda_2)$

The outgoing arcs from the macrostate are due to some firing: when $t_2$ fires, the DTMC goes to a state with descriptor $(0,\diamond)$, because this firing causes the marking process to go to the marking $M_0$; whereas the firing of $t_3$ is described from the arcs towards the absorbing phase of the *DPH*, so that two arcs from states with descriptors $(1,2)$ and $(1,4)$ towards the state with descriptor $(2,\diamond)$ are used. The one step probability is easily computed with reference to the firing events and to the *DPH* structure. The only thing to note is that in a time slot $\delta$ both transitions $t_2$ and $t_3$ can fire, then the simultaneous firing event has to be considered. In the case when both transitions fire in the same time slot, we uniformly distribute the probability of firing between the two possible destination states with descriptor $(1,\diamond)$ and $(3,\diamond)$. This is where the factors $P_{25}\,\delta\,\frac{\lambda_2}{2}$ and $P_{45}\,\delta\,\frac{\lambda_2}{2}$ come from. This problem will be extensively discussed in Section 4.

### 3.2 *SPN* with one generally distributed *prs* transition

If the GEN transition is associated with a *prs* policy, the DTMC structure has to be organized in order to keep track of the amount of time the *prs* transition spent in an enabled condition before being preempted. This is because the transition has to restart with the same age memory value once it becomes enabled again. For this purpose, a different expanded DTMC is needed. Figure 5 shows the DTMC that approximates the stochastic behavior of the *SPN* depicted in Figure 2 when $t_3$ has a *prs* memory policy.

The only difference with regard to the *prd* case is the macrostate related to the marking $M_0$. With a *prs* policy, four states with descriptors $(0,u)$, with $1 \leq u \leq 4$, are added to the macrostate. The purpose of these descriptors is remembering the value of the age memory of transition $t_3$ when it is disabled by the firing of the EXP transition $t_2$.

Thus, from each state with descriptor $(1,u)$, with $1 \leq u \leq 4$, the DTMC can transit either to the state with descriptor $(0,u)$ (with one step probability $p_{(1,u)\to(0,u)} = P_{uu}\delta\lambda_2$) or to the state with descriptor $(0,u+1)$, where $1 \leq u \leq 3$ (with probability $p_{(1,u)\to(0,u+1)} = P_{u(u+1)}\delta\lambda_2$).

Of course, transition $t_3$ cannot fire from any of the states corresponding to marking $M_0$, as it is not enabled in such marking. ¿From each of these states it is possible

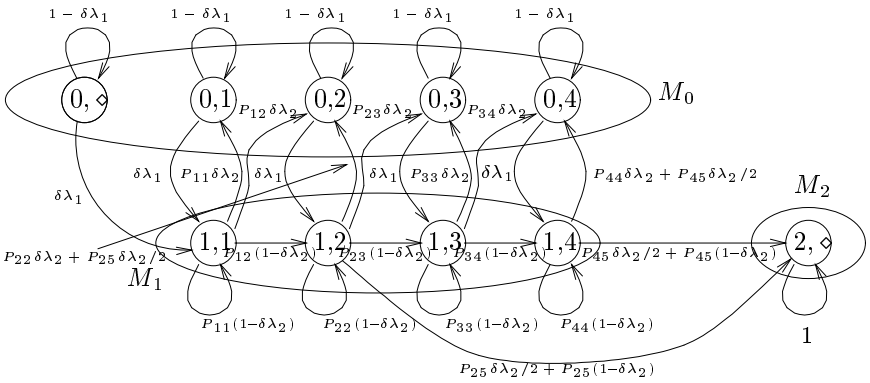


**Fig. 5.** *DTMC* approximation of the *SPN* with *prs* transition

either to exit with probability $p_{(0,u)\to(1,u)} = \delta\lambda_1$, with $u = 1, 2, 3, 4$, when transition $t_2$ fires, or to remain in the same state with probability $p_{(0,u)\to(0,u)} = 1 - \delta\lambda_1$, with $u = 1, 2, 3, 4$, when $t_2$ does not fire in a time slot $\delta$.

The same considerations made with regard to the firing of $t_3$ with a *prd* policy are valid in this case.

### 3.3 *SPN* with one generally distributed *pri* transition

If a *pri* policy is assumed for the GEN transition $t_3$, an interrupted job must be repeated with an identical work requirement. To capture the stochastic behavior of this case, a different expanded DTMC is constructed.

To model a transition $t_k$ with an associated *pri* preemption policy, the following quantities are computed $Q_i^k = F^k(i\delta) - F^k((i-1)\delta)$. $Q_i^k$ derives from the firing time distribution $F^k(t)$ of transition $t_k$, and approximates the firing probability of transition $t_k$ in the $i$-th $\delta$ interval. For making the model solvable in practice, the firing time distribution of a *pri* transition is supposed to have finite support, in order to avoid the computation of an infinite number of nonzero $Q_i^k$ values, and to construct an approximate discrete process with infinite state space. Let us denote the number of nonzero $Q_i^k$ quantities with $q^k$; it depends on $\delta$, and its value is $q^k = \frac{d^k}{\delta}$, where $d^k$ is the length of the support of the firing time probability distribution function $F^k(t)$. In case of infinite support, a truncation of $F^k(t)$ may be used.

The stochastic behavior of an enabled *pri* type transition is described by two continuous variables: the actual sample of the firing time and the remaining firing time, or alternatively, the actual sample of the firing time and the amount of time during which the transition has been enabled. In the proposed expansion method, the descriptor $(i, u, w)$ with $u \leq w$ is used in order to describe the state of the process, where $i$ indicates the marking, $u$ indicates the duration of time while the transition is enabled (measured in integer numbers of time slots $\delta$), and $w$ is the sampled value (measured in integer numbers of time slots $\delta$). The descriptor $(i, 0, w)$ indicates that the *pri* transition is disabled but it has not fired, so that the sampled firing time $w$ is maintained; after becoming enabled again, the process enters state $(i, 1, w)$. The

descriptor $(i, \diamond, \diamond)$ is used for states where the process has no memory. In other words, the marking itself completely determines the state of the process.

The evolution of the GEN transition $t_k$ with *pri* memory policy in isolation can be described by $q^k$ columns. The $w$-th column consists of $w$ states with descriptors $(i, u, w)$, where $1 \leq u \leq w$. Recalling that $w$ is the sampled firing time, when the discrete process enters a state with descriptor $(1, 1, w)$, $w$ slots of time have to pass before the firing. This is exactly the time spent to transit among the states of the column.

Figure 6 shows the DTMC that approximates the behavior of the *SPN* shown in Figure 2. In this case, the macrostate corresponding to the marking $M_1$ consists of the states approximating the GEN transition $t_3$, as described before. From the state with descriptor $(0, \diamond, \diamond)$, the DTMC enters the macrostate corresponding to marking $M_1$, and specifically the column selected according to the probability $Q_w$. Since this happens if the EXP transition $t_1$ fires in a time slot, the one step probability is: $p_{(0,\diamond,\diamond)\to(1,1,w)} = Q_w \delta \lambda_1$.

The macrostate referred to the marking $M_0$ has $q$ states with descriptor $(0, 0, w)$ reached by the DTMC when the GEN transition is disabled by the firing of the conflicting transition $t_2$. These states are used to remember the correct sampled firing value, so when the GEN transition is enabled again the correct column is reached. The one step probabilities between two states in this macrostate are computed according to the firing events related to the EXP transition $t_2$, also enabled in marking $M_1$, as in the other cases.

The GEN transition $t_3$ fires when $u = w$ in the descriptor. When this happens, the DTMC transits in the state with descriptor $(2, \diamond, \diamond)$.

## 4 General solution

In the last three subsections we have described a method to build a DTMC to approximate the stochastic behavior of Petri Nets containing only one GEN transition. Using a similar approach, in this section we show how to derive the underlying DTMC for *SPN*s with more than one GEN transitions simultaneously enabled. A similar idea can be followed to deal with the case of more EXP transitions simultaneously enabled in the same time slot $\delta$. The following notation has to be introduced:

- $N^D$, $N^S$ and $N^I$ is the number of *prd*, *prs* and *pri* transitions in the *SPN*, respectively;
- $\mathcal{A}^D(i)$, $\mathcal{A}^S(i)$ and $\mathcal{A}^I(i)$ are the set of enabled *prd*, *prs*, *pri* GEN transitions in marking $M_i$, respectively;
- $P^k_{i,j}$ is the probability of moving from phase $i$ to phase $j$ in the *DPH* structure of the transition $t_k$; it describes how a *prd* or *prs* GEN transition changes its phase;
- $Q^k_i$ is the approximated probability that the *pri* GEN transition $t_k$ fires in the $i$-th $\delta$ interval;
- $L_k$ is the number of phases in the *DPH* structure of the *prd* or *prs* transition $t_k$.

As already discussed, we need one variable to handle transitions with *prd* and *prs* policy (to store the current phase of the expanded DTMC), and two variables to handle transitions with *pri* policy (one to store the age of the transition, and the other to store the sampled value of the firing time).

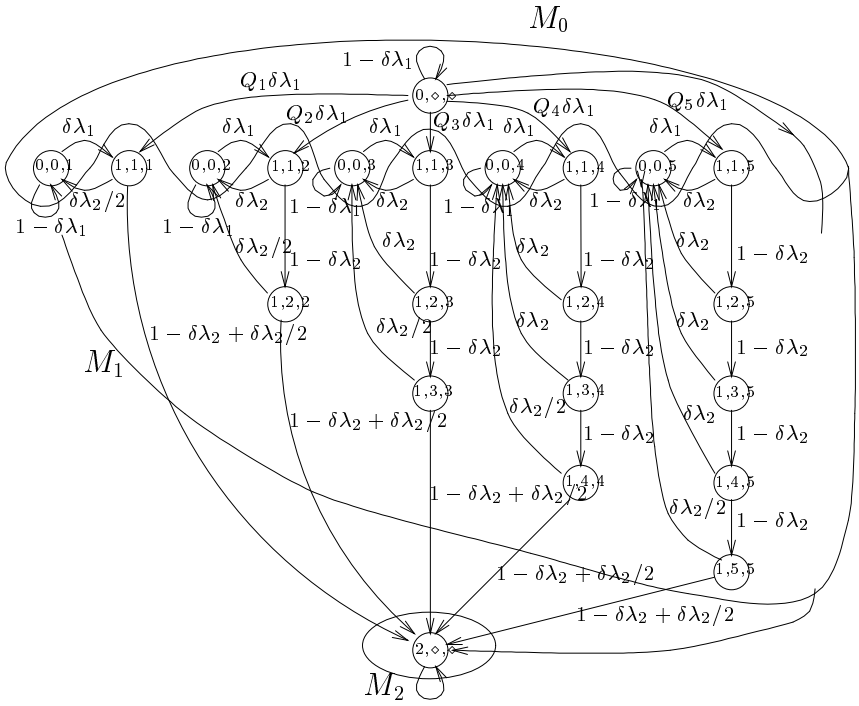


**Fig. 6.** DTMC approximation of the *SPN* with *pri* transition

When a *pri* transition gets enabled, the associated random variable is sampled and the age variable is set to 1 [2]. If the *pri* transition gets preempted in the next state, the age variable is reset to 0 and the associated sampled value remains the same.

Thus a generic state of the DTMC will be $Z^r = (j, D^r, S^r, I^r, X^r)$, where

- $j$ is the index of marking $M_j$ of the *SPN*;
- $\boldsymbol{D}^r$ is a vector of length $||\mathcal{T}||$, the number of the transitions in the *SPN*, storing the phases in which a *prd* transition is allowed to be; in particular, its $k$-th element ($D_k^r$) is the phase of transition $t_k$ when the DTMC is in the state $Z^r$; the sign $\diamond$ in the $k$-th position indicates that the *prd* GEN transition $t_k$ has no memory (it is not enabled).
- $\boldsymbol{S}^r$ is the same as $D^r$ but for *prs* GEN transitions; $S_k^r = \diamond$ means that the *prs* transition $t_k$ is not active, thus it has no memory;
- $\boldsymbol{I}^r$ is a vector of length $||\mathcal{T}||$. The $k$-th element of $\boldsymbol{I}^r$ ($I_k^r$) is the age of the *pri* GEN transition $t_k$ when the DTMC is in the state $Z^r$; similarly to the case of *prs* transitions, $I_k^r = \diamond$ indicates that transition $t_k$ is not active;
- $\boldsymbol{X}^r$ is a vector whose $k$-th element ($X_k^r$) is the sampled value of the *pri* GEN transition $t_k$ when the DTMC is in the state $Z^r$.

[2] Note that as time increases by $\delta$, at step $i$ the total elapsed time is $i * \delta$. This explains why only the index indicating the time interval has to be recorded.

Given a state $Z^r = (i, \boldsymbol{D}^r, \boldsymbol{S}^r, \boldsymbol{I}^r, \boldsymbol{X}^r)$ of the DTMC, we consider first the case when none of the enabled transitions fires in a time slot $\delta$, and then we show the more complex case when some firings occur.

### 4.1 Initial states and probability vector

When the algorithm starts to generate the approximated discrete process, a set of initial states are created together with an initial probability vector. The number of initial states depends on the transitions enabled in the initial marking $M_0$. The *prd* and *prs* transitions are considered without memory when the process starts, thus, using the assumption that the *DPH* distribution starts from the first phase, the memory variables associated to these transitions are described by the following equations:

$$D_k^0 = \begin{cases} 1 & t_k \in \mathcal{A}^D(0) \\ \diamond & \text{otherwise} \end{cases}, \qquad S_k^0 = \begin{cases} 1 & t_k \in \mathcal{A}^D(0) \\ \diamond & \text{otherwise} \end{cases} \tag{1}$$

Instead, if *pri* transitions are enabled in the initial marking $M_0$, a set of states has to be created to remember the different levels of sampled values $Q_i^k$, with $t_k \in \mathcal{A}^I(0)$. To explain how to build the initial states, a new notation has to be introduced. Let $q^k$ be the maximum value of $X_k^r$; $q^k$ is the number of different possible values, $Q_i^k$, for the firing probability of $t_k$ (if the *pri* transition $t_k$ is not enabled in the marking $M_0$, then $q^k = 0$). The number of initially built states is $s = \prod_{t_k \in \mathcal{A}^I(0)} q^k$, and each of them corresponds to a different combination of the possible values assumed by $X_k$.

To formally construct the descriptor, we define a function that associates each possible state with an index starting from the values assumed by the components of $\boldsymbol{X}$. Let $k_l$ be the index of the $l$-th *pri* transition enabled in $M_0$ ($t_{k_l} \in \mathcal{A}^I(0)$). With this formalism the index is $r = \sum_{l=0}^{||\mathcal{A}^I(0)||-1} (X_{k_l} - 1)\, l$

Vice versa given a value of index $r$, the combination that generated it can be found. We denote this function $v(r, l)$.

With these definitions, it is possible to describe all the components of the state descriptors generated at the beginning. The different components of states $Z^r = (i, \boldsymbol{D}^r, \boldsymbol{S}^r, \boldsymbol{I}^r, \boldsymbol{X}^r)$, $\forall r = 0, \cdots, s-1$ are the vectors $\boldsymbol{D}^r = \boldsymbol{D}^0$, and $\boldsymbol{S}^r = \boldsymbol{S}^0$, whereas $\boldsymbol{I}^r$ and $\boldsymbol{X}^r$ assume the following value:

$$I_k^r = \begin{cases} 1 & t_k \in \mathcal{A}^I(0) \\ \diamond & \text{otherwise} \end{cases}, \qquad X_k^r = \begin{cases} v(r, l(k)) & t_k \in \mathcal{A}^I(0) \\ \diamond & \text{otherwise} \end{cases} \tag{2}$$

where $l(k)$ is the position of transition $t_k$ among the enabled *pri* transitions in $M_0$.

The generic element of the initial probability vector is:

$$\Pi_r(0) = \prod_{t_k \in \mathcal{A}^I(0)} Q_{v(r,l(k))}^k, \quad \forall r = 0, \cdots, s \tag{3}$$

### 4.2 No firing

In this section we describe how to generate a new state of the expanded discrete process starting from a given state of the expanded process itself, in the occurrence of no firing of the enabled transitions.

Let $Z^r = (i, \boldsymbol{D}^r, \boldsymbol{S}^r, \boldsymbol{I}^r, \boldsymbol{X}^r)$ be a descriptor of state of the discrete process. Under the assumption that no transition fires in a time slot $\delta$ in $M_i$, the marking of the *PN* remains the same, and the *DPH* of the enabled transitions change their phase accordingly to their description. This implies that the *prd* and *prs* enabled transitions are not allowed to enter their absorbing state (phase change to the absorbing phase means that the transition fires), and all the enabled *pri* transitions have an age value less than the firing value originally sampled ($I_t^r < X_t^r$). The phase of the disabled *prd* transitions is indicated as $\diamond$ (they have no memory), while a phase indicator exists for the *prs* and *pri* transitions.

Let $Z^r = (i, \boldsymbol{D}^r, \boldsymbol{S}^r, \boldsymbol{I}^r, \boldsymbol{X}^r)$ be the descriptor of the actual considered state, and $Z^{r'} = (i, \boldsymbol{D}^{r'}, \boldsymbol{S}^{r'}, \boldsymbol{I}^{r'}, \boldsymbol{X}^{r'})$ the descriptor of the state we want to generate. Note that the first component of the descriptor $Z^{r'}$ is the same of $Z^r$ because no firing is supposed and the marking does not change. The different components of the descriptor $Z^{r'} = (i, \boldsymbol{D}^{r'}, \boldsymbol{S}^{r'}, \boldsymbol{I}^{r'}, \boldsymbol{X}^{r'})$ are built as follows:

$$D_k^{r'} = \begin{cases} a,\ 1 \le a \le L_k & t_k \in \mathcal{A}^D(i) \\ \diamond & t_k \notin \mathcal{A}^D(i) \end{cases} \tag{4}$$

where $a = \texttt{next}(t_k, D^r)$, being $\texttt{next}(t, p)$ a function that computes the index of a phase of the *DPH* associated to transition $t$ reachable from the phase with index $p$.

Equation (4) means that a new phase ($1 \le a \le L_k$) of the *prd* transitions enabled in marking $M_i$ ($t_k \in \mathcal{A}^D(i)$) is considered in the new state $Z^{r'}$. Otherwise ($t_k \notin \mathcal{A}^D(i)$) the memory is reset ($D_k^{r'} = \diamond$).

$$S_k^{r'} = \begin{cases} b,\ 1 \le b \le L_k & t_k \in \mathcal{A}^S(i) \\ S_k^r & t_k \notin \mathcal{A}^S(i) \end{cases} \tag{5}$$

Equation (5) is very similar to the (4), but if in marking $M_i$ a *prs* transition is disabled ($t_k \notin \mathcal{A}^S(i)$), the same value of the memory is maintained ($S_k^{r'} = S_k^r$).

The vectors for *pri* transitions are computed as follows:

$$I_k^{r'} = \begin{cases} I_k^r + 1 & t_k \in \mathcal{A}^I(i) \\ I_k^r & t_k \notin \mathcal{A}^I(i) \end{cases}, \qquad X_k^{r'} = X_k^r. \tag{6}$$

These last two equations describe how to manage the two variables for the *pri* transitions: the first one updates the age memory either incrementing it, if the transition is enabled in marking $M_i$, or maintaining the same value if the transition is not enabled (note that $I_k^r = \diamond$ if transition has not memory, and $I_k^r = 0$ if it has memory but it is not enabled); the second equation says that the residual time for the transition firing is the same with respect to that in state $Z^r$, since it is not enabled.

The state transition probability from $Z^r$ to $Z^{r'}$ in a time slot is computed as the product of the probability that none of the EXP transitions will fire times the probabilities that the *prd* and *prs* GEN transitions change their phase. This transition probability can be expressed as follows:

$$P_{Z^r \to Z^{r'}} = \underbrace{\prod_{k \in \mathcal{A}^D(i)} B^k_{D_k^r, D_k^{r'}}}_{\text{en. prd tr.-s}} \underbrace{\prod_{l \in \mathcal{A}^S(i)} B^l_{S_l^r, S_l^{r'}}}_{\text{en. prs tr.-s}}, \tag{7}$$

Note that EXP transitions are represented with the *DPH* depicted Figure 3b, and they are considered by the same standard as GEN transitions. Their presence is taken into account in the equation (7) by the first term ( $\prod_{k \in \mathcal{A}^D(i)} B^k_{D^r_k, D^{r'}_k}$ ) of that equation. Below we will not make special considerations on the EXP transitions and we will manage them by using the associated *DPH*.

### 4.3 Firing of one or more transitions

In this section we deal with the problem of one or more transitions firing in a time slot $\delta$. In order to address this goal, we decide first to identify the main steps and then to proceed with their formal treatment. Assuming that the expanded DTMC is in state $Z^r$, we want to identify all the reachable states $Z^{r'}$ and all the transitions probabilities associated to the connecting arcs.

Given a state $Z^r = (i, \boldsymbol{D}^r, \boldsymbol{S}^r, \boldsymbol{I}^r, \boldsymbol{X}^r)$, only a subset of the enabled transitions is allowed to fire in a time slot $\delta$. Transition $t_k \in \mathcal{A}^D(i)$ (an enabled *prd* transition) can fire if $P^k_{D^r_k, L_k} > 0$, i.e. if the probability of immediately reaching the absorbing state from phase $D^r_k$ is positive; similarly, a transition $t_k \in \mathcal{A}^S(i)$ can fire if $P^k_{S^r_k, L_k} > 0$. A transition $t_k \in \mathcal{A}^I(i)$ is allowed to fire if $I^r_k = X^r_k$, i.e. if its age is equal to the sampled firing time. We use these conditions to define the following set:

$$\begin{aligned} \mathcal{F}^r = \{ t_k | & (t_k \in \mathcal{A}^D(i) \wedge P_{D^r_k, L_k} > 0) \vee \\ & (t_k \in \mathcal{A}^S(i) \wedge P_{S^r_k, L_k} > 0) \vee \\ & (t_k \in \mathcal{A}^I(i) \wedge I^r_k = X^r_k) \} \end{aligned} \tag{8}$$

$\mathcal{F}^r$ is the set of all the transitions that are allowed to fire when the process is in state $Z^r$. The elements of this set, whose cardinality is $||\mathcal{F}^r||$, can be grouped into $2^{||\mathcal{F}^r||} - 1$ different subsets, corresponding to all the possible combinations of the transitions allowed to fire in marking $M_i$. A generic subset of the $\mathcal{F}^r$ will be indicated as $\overline{\mathcal{F}}^r_p$, with $p = 1, \cdots, 2^{||\mathcal{F}^r||} - 1$.

Considering the generic state $Z^{r'} = (j, \boldsymbol{D}^{r'}, \boldsymbol{S}^{r'}, \boldsymbol{I}^{r'}, \boldsymbol{X}^{r'})$ reachable from $Z^r$ when the transitions belonging to $\overline{\mathcal{F}}^r_p$ fire, the values of the components of its descriptor are computed as follows:

$$D^{r'}_k = \begin{cases} 1 & t_k \in (\mathcal{A}^D(j) \cap \overline{\mathcal{F}}^r_p) \cup (\mathcal{A}^D(j) \backslash \mathcal{A}^D(i)) \\ a, 1 \leq a < L_k & t_k \in (\mathcal{A}^D(i) \backslash \overline{\mathcal{F}}^r_p) \\ \diamond & \text{otherwise} \end{cases} \tag{9}$$

where, as in the equation (4), $a = \texttt{next}(t_k, D^{r'}_k)$. The first term in the equation (9) sets the phase of transition $t_k$ to 1 if $t_k$ fires in marking $M_i$ and is re-enabled in marking $M_j$ (i.e. $t_k \in \mathcal{A}^D(j) \cap \overline{\mathcal{F}}^r_p$), or it is not enabled in marking $M_i$ and becomes enabled in marking $M_j$ ($t_k \in \mathcal{A}^D(j) \backslash \mathcal{A}^D(i)$); the second term updates the new phase of the transition $t_k$ when it is not fired in marking $M_i$ ($t_k \in \mathcal{A}^D(i) \backslash \overline{\mathcal{F}}^r_p$);

$$S^{r'}_k = \begin{cases} \diamond & t_k \in (\overline{\mathcal{F}}^r_p \backslash \mathcal{A}^S(j)) \\ 1 & t_k \in \mathcal{A}^S(j) \cap \overline{\mathcal{F}}^r_p \\ b, 1 \leq b < L_k & t_k \in (\mathcal{A}^S(i) \cap \mathcal{A}^S(j)) \backslash \overline{\mathcal{F}}^r_p \\ S^r_k & \text{otherwise} \end{cases} \tag{10}$$

where $b = \texttt{next}(t_k, S_k^r)$; in equation (10) the set $\overline{\mathcal{F}}_p^r \backslash \mathcal{A}^S(j)$ identifies the transitions fired in marking $M_i$ and not enabled again in marking $M_j$, so they do not need to maintain their memory ($S_k^{r'} = \diamond$); when transition $t_k$ becomes enabled ($t_k \in \mathcal{A}^S(j) \cap \overline{\mathcal{F}}_p^r$), its phase is set to 1; when the transition is enabled in both the markings $M_i$ and $M_j$ without firing ($t_k \in \mathcal{A}^S(i) \cap \mathcal{A}^S(j)) \backslash \overline{\mathcal{F}}_p^r$) its memory is updated by changing the phase of the associated *DPH*. In all the other cases, the same phase is maintained ($S_k^{r'} = S_k^r$) because the transition is not enabled, but it is still active;

$$I_k^{r'} = \begin{cases} \diamond & t_k \in (\overline{\mathcal{F}}_p^r \backslash \mathcal{A}^I(j)) \\ 1 & t_k \in \mathcal{A}^I(j) \cap \overline{\mathcal{F}}_p^r \\ I_k^r + 1 & t_k \in (\mathcal{A}^I(i) \cap \mathcal{A}^I(j)) \backslash \overline{\mathcal{F}}_p^r \\ I_k^r & \text{otherwise} \end{cases} \tag{11}$$

Equation (11) describes the updating of the memory associated with a *pri* transition $t_k$. We recall that the vector $\boldsymbol{I}^{r'}$ stores the amount of time since when the transitions are enabled (measured in time slots $\delta$). Thus, the $k$-th so one component is increased by a unit ($I_k^{r'} = I_k^r + 1$) when the corresponding transition $t_k$ remains enabled ($\mathcal{A}^I(i) \cap \mathcal{A}^I(j)) \backslash \overline{\mathcal{F}}_p^r$). The other terms of equation (11) have the same meaning of those of the equation (10);

$$X_k^{r'} = \begin{cases} \diamond & t_k \in (W \backslash \mathcal{A}^I(j)) \\ x & t_k \in (\mathcal{A}^I(j) \cap W) \vee (t_k \in \mathcal{A}^I(j) \wedge X_k^r = \diamond) \\ X_k^r & \text{otherwise} \end{cases} \tag{12}$$

The equation (12) is introduced for considering the sampled firing values of the *pri* transitions (the $q^k$ values introduced in section 4.1); also in this case the sampled firing time is measured in time slots. The second term is referred to the enabling of the *pri* transition $t_k$ either when it fires in marking $M_i$ and it becomes enabled again ($t_k \in \mathcal{A}^I(j) \cap \overline{\mathcal{F}}_p^r$), or it becomes enabled in marking $M_j$ when it has no memory ($(t_k \in \mathcal{A}^I(j)) \wedge (X_k^r = \diamond)$); when these conditions are true, the new descriptor must store one of the $q^k$ possible sampled values from the cdf $F^k(\cdot)$. The other two terms are more trivial: the first term of the equation 12 is used when the process does not need to maintain any memory on the evolution of the transition $t_k$ because it has already fired and is not enabled again ($t_k \in \overline{\mathcal{F}}_p^r \backslash \mathcal{A}^I(j)$); the third one is used to store the memory of the process when $t_k$ is not enabled, but it is still active.

Due to the time discretization approach we have adopted, the cdf associated to each timed transition will have a time discontinuity at the end of each time slot $\delta$. Thus, if in marking $M_i$ several transitions are enabled, there is a non null probability that they simultaneously fire. The probability that the transitions belonging to $\overline{\mathcal{F}}_p^r$ simultaneously fire can be expressed as follows:

$$f_p^r = \text{P}\{\text{all transitions in} \overline{\mathcal{F}}_p^r \text{ fire} \mid Z^r \} = \prod_{k \in (\overline{\mathcal{F}}_p^r \cap \mathcal{A}^D(i))} P_{D_k^r, L_k}^k \prod_{l \in (\overline{\mathcal{F}}_p^r \cap \mathcal{A}^S(i))} P_{S_k^r, L_k}^l \tag{13}$$

Equation (13) does not include any reference to *pri* transitions Because, if *pri* fire, their contribution to $f_p^r$ is equal to 1. This probability will cause the switching from the generic marking $M_i$ (where the process is now) to a marking $M_j$. Since the transitions in $\overline{\mathcal{F}}_p^r$ may be in conflict, the marking $M_j$ is reached with a probability $W_{ij}$

(the method to compute the reached marking $M_j$ and the probability $W_{ij}$ is deeply analyzed in [18]).

The probability associated with the arc from $Z^r$ to $Z^{r'}$ is evaluated as:

$$P_{Z^r \to Z^{r'}} = p_{ij} \prod_{h \in (\mathcal{A}^D(i) \setminus \overline{\mathcal{F}}_p^r)} P^h_{D_h^r, D_h^{r'}} \prod_{k \in (\mathcal{A}^S(i) \cap \mathcal{A}^S(j)) \setminus \overline{\mathcal{F}}_p^r} P^t_{S_k^r, S_k^{r'}} \prod_{t \in \mathcal{A}^I(j) \wedge X_k^r = \diamond} Q^l_{X_l^{r'}} \tag{14}$$

where each term has the following meaning:

- $p_{ij} = f_p^r \cdot W_{ij}$ is the probability that the process arrives in marking $M_j$, starting from marking $M_i$, due to the firing of the transitions belonging to $\overline{\mathcal{F}}_p^r$ in a time slot of size $\delta$;
- the second (third) term takes into account the changing of phase of the enabled *prd* (*prs*) transitions;
- the last term is used for considering the case that some *pri* transitions become enabled, thus a new value has to be sampled according to the associated distribution function.

## 5 The algorithm

The algorithm is based on a discretization of the continuous random variables for approximating the continuous process. The phase type distributions, used in case of *prd* and *prs* GEN transitions, are given by the users, whereas the probabilities $Q_i^k$ are directly computed from the cdf associated with the *pri* transition $t_k$.

The main steps of the implemented solution method are the following:

1. generation of the reachability graph (with tangible and vanishing states) and reduction of the reachability graph to tangible states only;
2. generation and analysis of the expanded DTMC;
3. evaluation of the final measures at the net level, based on the solution of the expanded DTMC.

According to the results shown in the previous sections, given the reachability graph and the discrete phase type distributions associated to the GEN transitions, the elementary step 2 of the approximation method is as follows:

- **Initialization Step**
  Initialization consists of creating the set of states originated in the initial marking $M_0$. Equations (1), and (2) are used to compute these states; equation (3) is used to compute the initial state probability vector on the generated states. Note that if no *pri* transition is enabled in $M_0$, only one state is built in this step of the algorithm. The created states are put in a list of states to expand (`list_expand`).
- **Iteration Step**
  1. a state $Z^r$ to be expanded is extracted from the `list_expand` list;
  2. new expanded states $Z^{r'}$ are computed in case of no firing events using the equations (4), (5), and (6).
  3. using equation (7) the transition state probabilities from $Z^r$ to $Z^{r'}$ are computed and stored;

4. all the states $Z^{r'}$, not previously created, are stored in `list_expand`;
5. sets $\overline{\mathcal{F}}_p^r$, with $p = 1, \cdots, 2^{||\mathcal{F}^r||}-1$, are computed; according to these sets, other reachable markings $M_j$ are computed, and expanded states $Z^{r'}$ are built using equations (9), (10), (11), and (12) (these states are the states associated to the firing of some transitions);
6. using equation (14) the transition probabilities from $Z^r$ to $Z^{r'}$ are computed and stored;
7. all states $Z^{r'}$, not previously created, are stored in `list_expand`; the state $Z^r$ is stored in another list named `expanded`;
8. if the list `list_expand` is not empty, the algorithm proceeds with step 1, otherwise it terminates.

Similarly to [9], the system behavior is approximated by a Discrete Time Markov Chain (DTMC) over an expanded state space determined by the cross product of the system states (the markings of the Petri net) and the discretized values of the associated age variables. This approach is also closely related with the *DPH* expansion method proposed by Cumani in [11]. The main difference is that, in this case, the system behavior is approximated by an expanded DTMC while in the *PH* approximation case an expanded CTMC is obtained. The present approach inherits some similarities also from the supplementary variable approach [14], since the supplementary (age) variables are constrained to assume values in a discretized set.

## 6 Numerical results

For testing the described method, two kinds of experiments were done:

1. in the first experiment a preemptive M/G/1/2/2 queue model, whose customers belong to different user classes, was solved. This Petri net model belongs to the *MRSPN* class and is analytically solvable; the results (transient analysis of state probabilities) were compared with the solution obtained by solving the same example using the Laplace transform method [5, 4];
2. the second experiment involves the preemptive M/G/1/2/2 queue model again; this time all the transitions in the model have a non exponentially distributed firing time, but one, and the model cannot be solved with any of the available analytical techniques. Thus the obtained results were validated by simulation.

The purpose of these experiments is to show that the results obtained by applying the method described in the previous section can be compared with those produced by other analytical solution methods, when available. Moreover, more general classes of models, not analytically solvable by others techniques, can be studied. The tool WebSPN[3] [15] was used to solve the models under exam.

### 6.1 Experiment 1 - Preemptive M/G/1/2/2 queue with different customers

The *SPN* of Figure 7a models an M/G/1/2/2 queue in which the jobs submitted by customer 2 have higher priority and preempt the jobs submitted by customer 1. The

[3] The tool WebSPN is accessible through Internet at the address http://sun195.iit.unict.it/websp

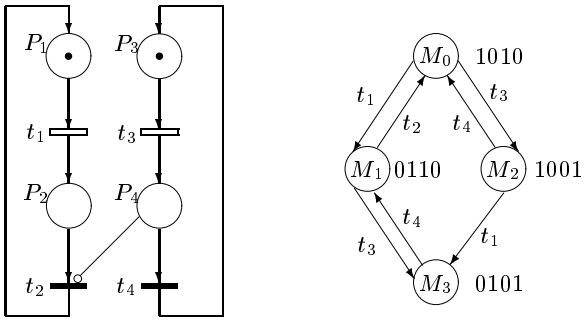


**Fig. 7.** Preemptive M/G/1/2/2 queue with two classes of customers

associated reachability graph is shown in Figure 7b. Place $P_1$ ($P_3$) represents customer 1 (2) thinking, while place $P_2$ ($P_4$) indicates job 1 (2) under service. Transitions $t_1$ and $t_3$ are EXP and model the submission of jobs of type 1 or 2, respectively. $t_2$ is a GEN transition and represents the completion of service of the lower priority job. Transition $t_4$ models the service time of a higher priority job. Its firing time is exponentially distributed. The inhibitor arc from $P_4$ to $t_2$ models the described preemption mechanism: as soon as a type 2 job joins the queue, the type 1 job under service (if any) is interrupted.

The server can adopt all kinds of preemption policies. Assuming a *prd* and *prs* memory policy, the model was solved with the following numerical values:

- firing rate of the EXP transitions $t_1$ and $t_3$: $\lambda_1 = \lambda_3 = 0.5$;
- the service times of both the lower and higher priority jobs (represented by $t_2$ and $t_4$) are deterministically distributed with firing time 1.0;
- time slot: $\delta = 0.05$.

The results obtained solving the model either with inverse Laplace transform and discrete expansion technique are depicted in Figure 8 a) and b) respectively. The symbols $\times$, $*$, $\diamond$, and $\square$ are used to plot the results obtained with Laplace transform method, whereas the continuous lines refer to the results obtained with the discrete expansion approach.

From these graphs it is evident that the method works well and the results are almost coincident with those computed with the inverse Laplace method, that is extensively discussed in literature.

The model of Figure 7 was solved also assuming a *pri* policy associated to the transition $t_2$. The deterministic cdf was not used with *pri* policy because the behavior of the preempted transition would be the same as in the *prd* case, since every time the transition is preempted it loses its memory and remembers the sampled firing time. But the possible firing time sampled by a deterministic cdf is always the same, thus remembering the firing time has no effect. In this case, the firing time was uniformly distributed between 0.5 and 1.0. The obtained results are depicted in Figure 8 c). As in the previous cases (the *prd* and the *prs* cases) also this experiment produced

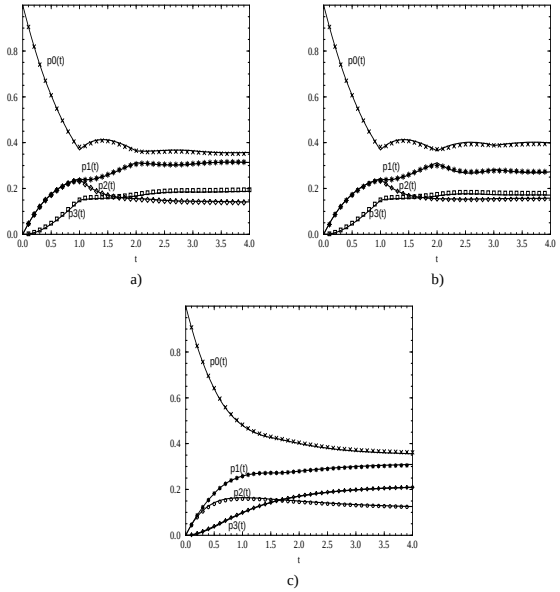


**Fig. 8.** State probabilities of the Petri net of Figure 7 when $t_2$ has a *prd* (a), *prs* (b) and *pri* (c) memory policy

the same results of those obtained with the inverse Laplace method. The comparison with the expansion method using *CPH* is not possible due to the fact that the *pri* preemption policy cannot be modeled using the *CPH* approximation.

### 6.2 Experiment 2 - Preemptive M/G/1/2/2 queue with different customers

The second experiment was done by assigning deterministically distributed firing time to transition $t_3$. In this case, in the state $M_1$ there are two transitions ($t_2$ and $t_3$) with generally distributed firing time, and this model cannot be solved using neither the Markov regenerative theory nor the supplementary variable method. The results are thus compared with those obtained from a simulator.

The following numerical value are used:

- firing rate of EXP transition $t_1$: $\lambda_1 = 1.0$;
- firing time of deterministic transition $t_3$: $\tau = 0.5$;
- service time of lower (transition $t_2$) and higher (transition $t_4$) priority job: uniformly distributed between 0.5 and 1.0.

Figure 9 shows the results obtained. The results of the simulation are depicted as two dashed lines, identifying the interval of confidence (95%) of the computed measure (the probability that the process is in state $M_0$). The continuous lines are the results obtained with the discrete expansion approach. Also in this case the three kind of policy are adopted for transition $t_2$, and a $\delta = 0.05$ was used to discretize the

model. As it can be noted, the results of discretization are always inside the interval of confidence computed by simulation, showing that the discrete expansion produces a correct result.

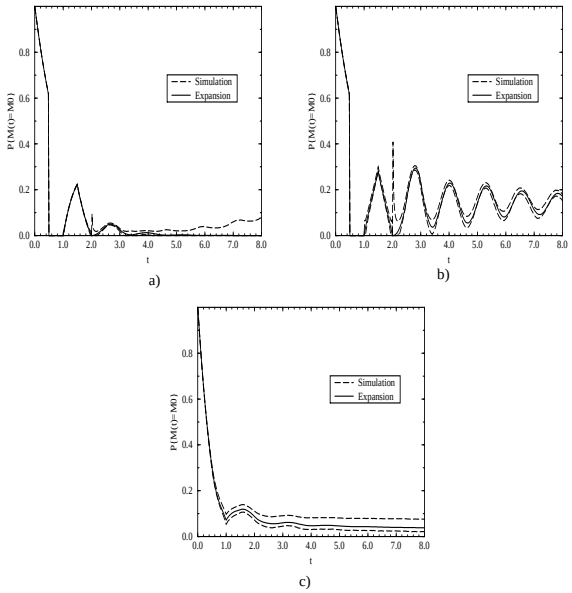


**Fig. 9.** State probabilities of the Petri net of Figure 7 when $t_2$ has a *prd* (a), *prs* (b) and *pri* (c) memory policy.

## 7 Conclusions

A numerical approach for the solution of *NMSPN*s has been proposed. It is based on a discrete time approximation of the stochastic behaviour of the marking process, which results in the possibility of analyzing a wider class of *SPN* models with *prd, prs* and *pri* concurrently enabled generally distributed transitions. In case of *prd, prs* policies distributions with infinite support are considered, for *pri* policy the firing distribution is limited to finite support distributions. We obtained that a *pri* type transition, which can be described, in transform domain, by the inclusion of a single transform variable [4], requires the inclusion of 2 memory variables in time domain. This explains why the representation of *pri* transitions is quite expensive.

We discussed the way the time-discretization algorithm works both in the case of only one general transition in the model and also when an arbitrary number of GEN transitions are simultaneously active.

The described algorithm has been implemented and embedded in the WebSPN tool, for specification and automatic solution of non-Markovian *SPN*. Due to the use of the Java programming language, WebSPN ([15]) is easily accessible from any node connected with the Internet as long as it possesses a Java-enabled Web browser.

## References


1. M. Ajmone Marsan, G. Balbo, A. Bobbio, G. Chiola, G. Conte, and A. Cumani. The effect of execution policies on the semantics and analysis of stochastic Petri nets. *IEEE Transactions on Software Engineering*, SE-15:832–846, 1989.
2. M. Ajmone Marsan, G. Balbo, and G. Conte. A class of generalized stochastic Petri nets for the performance evaluation of multiprocessor systems. *ACM Transactions on Computer Systems*, 2:93–122, 1984.
3. A. Bobbio, A. Horváth, M. Scarpa, and M. Telek. Acyclic discrete phase type distributions. Part 1: Properties and canonical forms. Technical report, Technical University of Budapest, 1999.
4. A. Bobbio, V.G. Kulkarni, A. Puliafito, M. Telek, and K. Trivedi. Preemptive repeat identical transitions in Markov Regenerative Stochastic Petri Nets. In *6-th International Conference on Petri Nets and Performance Models - PNPM95*, pages 113–122. IEEE Computer Society, 1995.
5. A. Bobbio and M. Telek. Markov regenerative SPN with non-overlapping activity cycles. In *International Computer Performance and Dependability Symposium - IPDS95*, pages 124–133. IEEE CS Press, 1995.
6. G. Chiola. *GreatSPN* 1.5 Software architecture. In G. Balbo and G. Serazzi, editors, *Computer Performance Evaluation*, pages 121–136. Elsevier Science Publishers, 1992.
7. Hoon Choi, V.G. Kulkarni, and K. Trivedi. Markov regenerative stochastic Petri nets. *Performance Evaluation*, 20:337–357, 1994.
8. G. Ciardo, R. German, and C. Lindemann. A characterization of the stochastic process underlying a stochastic Petri net. *IEEE Transactions on Software Engineering*, 20:506–515, 1994.
9. G. Ciardo and R. Zijal. Discrete deterministic and stochastic petri nets. Technical report, NASA, 1996.
10. J.A. Couvillon, R. Freire, R. Johnson, W.D. Obal, M.A. Qureshi, M. Rai, W. Sanders, and J.E. Tvedt. Performability modeling with UltrasSAN. *IEEE Software*, 8:69–80, September 1991.
11. A. Cumani. Esp - A package for the evaluation of stochastic Petri nets with phase-type distributed transition times. In *Proceedings International Workshop Timed Petri Nets*, pages 144–151, Torino (Italy), 1985. IEEE Computer Society Press no. 674.
12. R. German. Markov Regenerative Stochastic Petri Nets with general execution policies: supplementary variable analysis and a prototype tool. In *10th International Conference on Modelling Techniques and Tools for Computer Performance Evaluation (TOOLS'98)*. Palma de Mallorca, Spain, September 14-18, 1998.
13. R. German, C. Kelling, A. Zimmermann, and G. Hommel. *TimeNET - A toolkit for evaluating non-markovian stochastic Petri nets*. Report No. 19 - Technische Universität Berlin, 1994.
14. R. German and C. Lindemann. Analysis of stochastic Petri nets by the method of supplementary variables. *Performance Evaluation*, 20:317–335, 1994.
15. A. Horváth, A. Puliafito, M. Scarpa, M. Telek, O. Tomarchio. Design and evaluation of a web-based non-markovian stochastic Petri net Tool. In *The Thirteen International Symposium on Computer and Information Science (ISCIS'98)*, Belek (Antalia, Turkey), October 1998.
16. C. Lindemann. DSPNexpress: a software package for the efficient solution of deterministic and stochastic Petri nets. *Performance Evaluation*, 22:3–21, 1995.
17. A. Puliafito, M. Scarpa, and K.S. Trivedi. Petri nets with k simultaneously enabled generally distributed timed transitions. *Performance Evaluation*, 32 n.1, February 1998.
18. M. Scarpa. Non Markovian Stochastic Petri Nets with Concurrent Generally Distributed Transitions. *PhD Thesis*, Universitá di Torino, Torino, Italy, 1999.