Proxel-Based Simulation of Stochastic Petri Nets

Sanja Lazarova-Molnar, Graham Horton[1]
Otto-von-Guericke-Universität Magdeburg
[sanja | graham]@isg.cs.uni-magdeburg.de

Abstract

This paper discusses the analysis of stochastic Petri nets using the proxel-based simulation method. The paradigm of the proxel ("probability element") was recently introduced in order to provide a new algorithmic approach to analysing discrete-state stochastic models such as are represented by stochastic Petri nets (SPNs) or queueing systems. Proxel-based simulation is not related to either of the standard simulation approaches: it is in no way analogous to discrete-event simulation, and, although it is based on the model's underlying stochastic process and makes use of supplementary variables, it does not require the use of differential equations. Instead, the proxels trace the movement of probability from one state of the model to another using discretized time steps. Since stochastic Petri nets are a powerful and widespread tool for modelling stochastic processes, we are interested in finding out how the proxel-based method applies to them. The formal foundations of the analysis of SPNs with the use of the proxel-based method are the subject of this paper.

1 Goals of the Paper

The goal of this paper is to present a new way of analysing stochastic Petri nets using the recently introduced proxel-based method. The formal connections between the proxel-based method and stochastic Petri nets will be established, so that the analysis of the Petri nets can based upon them. Because of the novelty of the proxel-based method, an extensive description of the method will be presented, as well as a short description of the formalism of stochastic Petri nets. We will show that the proxel-based simulation can be interpreted as a discrete-time Markov chain (DTMC). Petri Nets were interesting for us because of their popularity and power for describing stochastic processes. Their broad definition and this being the first attempt for their analysis using the proxel-based method are the reasons that we will focus on a restricted class of Petri nets. Results from experiments with an implementation of the method will be presented.

[1] Fakultät für Informatik, Institut für Simulation und Graphik, D-39016 Magdeburg, Germany

2 Introduction to SPNs and Proxels

2.1 Modelling with Petri Nets

Stochastic Petri nets are popular and powerful tool for modelling and analysing complex stochastic systems. A stochastic system is usually viewed as a function of time, which means that we are interested in computing its dynamic behaviour. In many cases it is a discrete-event system, meaning that the changes in the system occur at certain points in time as consequences of the completion of certain activities in the system. Activities in SPNs are associated with transitions, which can either fire instantly, as soon as certain conditions become enabled (immediate transitions), or can be associated with probability distribution functions which determine the firing delays (timed transitions). Besides the transitions, an SPN is defined by a finite number of places, a finite number of arcs and an initial state (the initial marking of the Petri net). Note that our definition of SPNs is not restricted to exponential distributions. Formally, the class of SPNs that will be treated in this paper can be described in the following way:

$$SPN = (P, T, A, G, m_0)$$

- $P = \{P_1, P_2, \dots, P_n\}$, the set of places, drawn as circles
- $T = \{ T_1, T_2, \dots, T_m \}$, the set of timed transitions along with their distribution functions, drawn as bars
- $A = A^I \cup A^O \cup A^H$, the set of arcs, where A^O is the set of output arcs, A^I is the set of input arcs and A^H is the set of inhibitor arcs and each of the arcs has a multiplicity assigned to it,
- $G = \{g_1, g_2, \dots, g_r\}$, the set of guard functions which are associated with different transitions,
- m_0 – the initial marking of the Petri net.

Each transition is represented as $T_i = (F, mp)$, where $mp \in \{enabling, age\}$ is the memory policy of the transition, and F is a cumulative distribution function. The sets of arcs are defined such that

$$A^O = \{a^o{}_1, a^o{}_2, \dots, a^o{}_k\},\ A^I = \{a^i{}_1, a^i{}_2, \dots, a^i{}_j\}, \text{ and } A^H = \{a^h{}_1, a^h{}_2, \dots, a^h{}_l\}, \text{ where}$$
$$A^H, A^O \subseteq P \times T, A^I \subseteq T \times P.$$

We denote by $M = \{m_0, m_1, m_2, \dots \}$ the set of all reachable markings of the Petri net. Each marking is a vector made up of the number of tokens in each place in the Petri net, $m_i = (\#P_1, \#P_2, \dots, \#P_n)$. The set of all reachable markings is the discrete state space of the Petri net. The changes from one marking to another are consequences of the firing of enabled transitions which move (destroy and create) tokens, creating the dynamics in the Petri net. This makes the firing of a transition analogous to an event in a discrete-event system. The markings of a Petri net, viewed as nodes, and the possibilities of movement from one to another, viewed as arcs, form the reachability graph of the Petri net.

In this paper, for reasons of simplicity, we describe the case for SPNs without immediate transitions. Proxel-based simulation of SPNs with immediate transitions is possible, and has been described in [Mol03].

Another issue that requires attention are the memory policies of the transitions. Every transition has a memory policy assigned to it, which determines the behaviour of the transition's enabling time when it becomes disabled. It can either be age or enabling memory policy, which specifies whether the time one transition was enabled until another fired, will be remembered (age memory policy), or reset (enabling memory policy).

2.2 The Proxel-based Approach to Simulation

Proxels were recently introduced [Hor02] as a new technique for analysing discrete-state stochastic models such as queueing systems or stochastic Petri nets. For this class of models, the analysis is usually carried out using Monte Carlo simulation. By contrast, proxel-based simulation is deterministic, and works with the state-space of the model, which is extended to include supplementary variables that represent transition age intensities. Normally, this approach entails constructing and solving partial differential equations [Ger00], but proxels yield a purely algorithmic approach to the simulation, in which differential equations are avoided completely. The goal of the new approach is to develop an easily-understood, deterministic algorithm, which, for certain classes of models at least, may prove to be competitive with discrete-event simulation.

A proxel is a basic computational unit for the simulation algorithm, which represents the probability that at a given time, the simulation model has reached a specific state via a specific path. We use the term "Proxel" as an abbreviation of "probability element" by analogy to the well-known "pixel" (picture element) in Computer Graphics.

The idea behind the simulation algorithm is to discretise the continuous stochastic process of the SPN using a discrete time step *dt*. This yields a computational model consisting of a set of discrete states at each time point, each of which has a certain probability of occurring. The proxel simulation algorithm creates the discrete states on the fly and tracks probability as it is redistributed between these states as time progresses.

We illustrate the idea intuitively using the simple SPN and its reachability graph shown in Figure 1. This Petri net has three markings, m_0, m_1, and m_2. The delays for each transition are described by probability distributions and the initial marking is m_0.

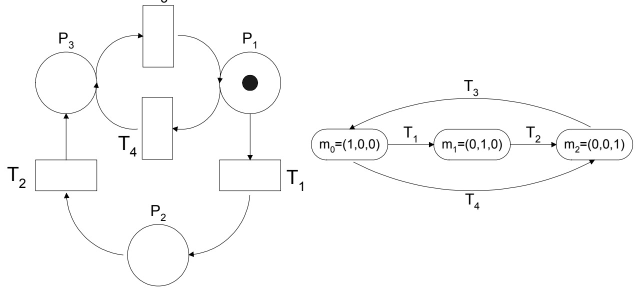


Figure 1: Example model

We now choose a discrete time step *dt*. This user-defined parameter must be chosen carefully, since it affects the accuracy of the simulation; specifically, it must be chosen so that the probability of the model making two or more state changes during any time interval (*t*, *t*+*dt*) is significantly smaller than the probability of one state change only. The proxel-based method makes the approximating assumption that at most one state change occurs during a time interval of length *dt*.

Figure 2 shows the states reached by the model at times 0, *dt*, and 2**dt*. The state is composed of two parts - the marking of the SPN and the length of time that a transition has been enabled without firing. The latter is known as the age intensity; it is needed to compute the probabilities for each state change, as will be explained in the next section. For this simple model, only one age intensity variable is needed. For more general models, several age intensity variables will be necessary.

At time $t = 0$, the model is in state (m_0, 0), where the second component is the time that the enabled transitions have been enabled. At time $t = dt$, the model can have done any of three things – transition to marking m_1 or m_2, or remain in marking m_0. The probability for each of these occurrences can be computed from the distribution functions describing the firing delays of the transitions. States at time $t = 2*dt$ and later are generated correspondingly. Proxel-based simulation is the repeated computation of the new set of states and their probabilities as simulation time progresses.

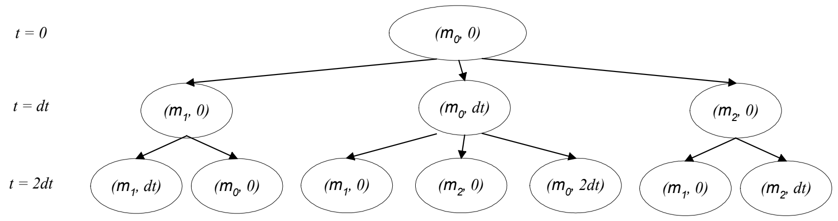


Figure 2: The first few states reached by the model

The set of reachable markings of the Petri net corresponds to the set of discrete states of the system. The state changes are associated with transitions. The state space is created based on the reachability graph. The analysis begins with the probability equal to 1 for the Petri net being in the initial marking at time zero. Based on the transitions enabled in the initial marking and their distribution functions, this probability of one is distributed among the successor states at subsequent time steps, as will be explained in detail in the next section.

3 Proxel-based Simulation of SPNs

3.1 Definitions

Proxels are elements that completely define the state of the system at discrete points in time. We assume that the system is modelled as a stochastic Petri net, having the restrictions that we specified in the second section. Let *PN* be a stochastic Petri net, defined as follows:

$$PN = (P, T, A, m_0), \text{ with}$$

$$P = \{P_1, P_2, \ldots, P_n\},\ T = \{T_1, T_2, \ldots, T_m\}.$$

A proxel *Px* is defined as the following:

$$Px = (S, t, R, Pr).$$

$S = (m, \vec{\tau})$ is the state of the system, where *m* is the marking and $\vec{\tau}$ is the *age intensity vector,* which contains the elapsed enabling time of a set of transitions. *t* is the global simulation time and *Pr* denotes the probability that the model is in state *S* at time *t*, given that it has been reached through the sequence of states $R = (S_1, S_{2, \ldots,} S_s)$. The null sequence is denoted by $\varnothing = ()$. The age intensity vector $\vec{\tau}$ is needed for a complete definition of the state of the Petri net, because the firing rate of each transition is dependent on how long it has been enabled. The *instantaneous rate function* IRF, denoted by $\mu(\tau)$, is used for this purpose. It takes as a parameter τ, the enabling time of the transition that caused the change from the previous to the current marking, and is computed from the distribution function that is associated with the transition. Henceforth, the term "state" will refer to the vector $S = (m, \vec{\tau})$.

For each transition $t_i = (F_i, mp_i)$ with distribution function F_i and memory policy mp_i, the IRF $\mu_i(\tau)$ is calculated according to:

$$\mu_i(\tau) = \frac{F_i'(\tau)}{1 - F_i(\tau)}$$

The instantaneous rate function is used for calculating the probability that the transition will fire within the time interval $(\tau, \tau{+}dt)$, if it has been active for time τ. We approximate this probability by $\mu_i(\tau){*}dt$. This is the fact that makes it possible to compute the probabilities for each state as time progresses.

The proxel-based method distributes the initial probability of 1 for being in the initial state among all of the subsequent states of the model. The proxels are the computational units that store and keep track of the flow of probability from one state to another. This means that the proxel-based simulation repeatedly generates from each proxel the set of successive proxels, until the end of the simulation time has been reached. In this manner, a tree structure of proxels is created, which will be referred to as the "proxel tree". The proxel tree is actually the state space of the model in terms of proxels.

During the process of generating the proxel tree, at any discrete time step, the same state may be generated many times, each time via a different sequence of predecessors, i.e. a different route R_i. In order to obtain the total probability for that state and to optimise the storage of the proxels, the probabilities of all of that state's instances are summed up and a new proxel that represents the state is generated. This proxel is stored as a representative of the corresponding state and its route parameter is set to the union of all the routes that lead to that state at the current discrete time step. This can be formally represented in the following way:

$$Px = (S, t, \bigcup_{i:R_i \xrightarrow{t/dt} S} R_i \text{ , P(model in } S \text{ at time } t))$$

$$\text{P(model in state } S \text{ at time } t) = \sum_{i:R_i \xrightarrow{t/dt} S} \text{P(model in } S \text{ at time } t \mid S \text{ reached via } R_i)$$

$$\xrightarrow{n} = \text{“leads to in } n \text{ steps”}$$

Then, at each discrete time step, this procedure repeats. In many cases, after a certain number of discrete time steps, a steady state is reached, which means that the sums of the probabilities of the proxels that represent the same discrete state of the system converge to a stationary value. The formal representation of the probabilities of the different discrete states at different time steps as well as the calculation of the proxels' probabilities in the next discrete time step based on the previous one is the following:

$$\text{P(Model in marking } m \text{ at time } t) = \sum_{\tau:\ S_k = (m,\tau)} \text{P(Model in state } S_k \text{ at time } t)$$

$$\text{P(Model in state } S_k \text{ at time } t+dt) = \sum_{i,j:R_j \xrightarrow{t/dt} S_i} \text{P(Model in state } S_i \text{ at time } t)*\mu(\tau_{ik})*dt$$

where τ_{ik} is the age intensity of the transition that caused the state change from S_i *to* S_k. $\mu(\tau_{ik})$ is 0 if the state change from S_i to S_k is not possible.

One important aspect when constructing the proxel tree are the memory policies of the transitions in the Petri net. Age policy transitions "remember" their activation times when they become disabled owing to another transition firing. When re-enabled, the clock that measures the time until the transition fires resumes from where it left off. For this reason, every age policy transition may require an additional component in the age intensity vector, which adds to the complexity of the simulation.

3.2 Example

In this section an example will be presented of how the proxel-based method functions when the Petri net contains both types of transition memory policies. For this purpose, the Petri net model shown in Figure 3 will be used. This SPN represents a round-robin approach to processing three queues of jobs, under the assumption that there is always at least one job in each queue. This example was chosen because it contains three age-policy transitions, which lead to a large age intensity vector.

The initial marking m_0=(1, 0, 0) is used. There are six transitions, defined as follows:

- $T_i = (F_i, \textit{enabling})$, $i = 1, 2, 3$
- $T_i = (F_i, \textit{age})$, $i = 4, 5, 6$

Three of the transitions are of age policy, and at each point in time there is one enabled transition of enabling policy, which means that the dimension of the age intensity vector will be four. The reachability graph of the Petri net is shown in Figure 4.

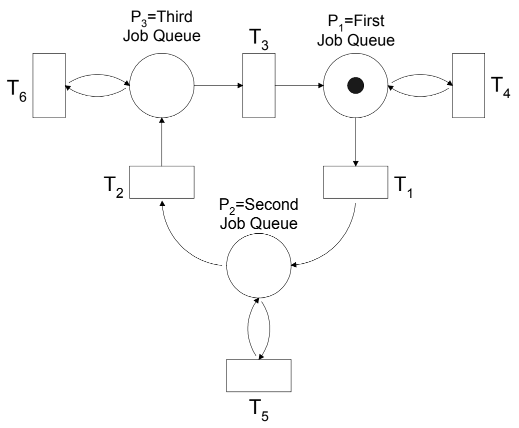


Figure 3: SPN model of round-robin processing of three queues

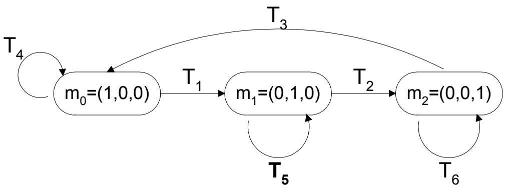
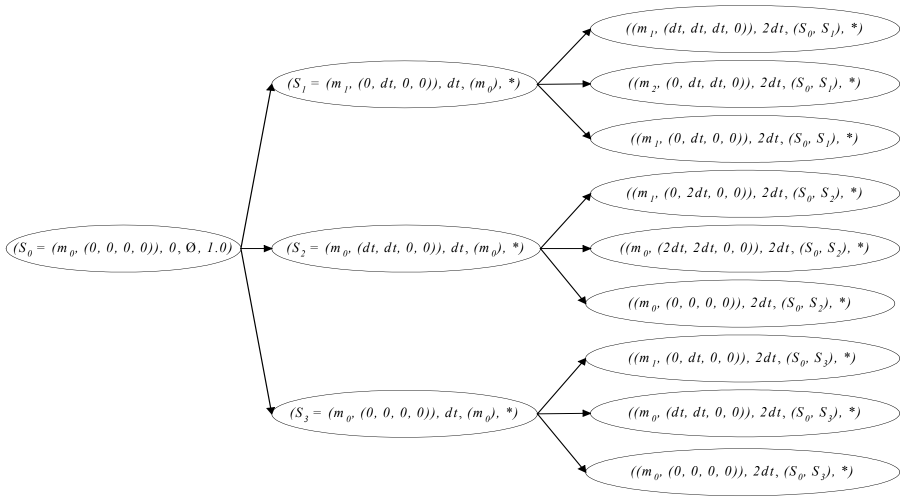


Figure 4: Reachability graph of the Petri net in Figure 3

The age intensity vector is created with the following mapping ($T_1/T_2/T_3$, T_4, T_5, T_6), i.e. the first three transitions which have enabling memory policy are all mapped to the first component of the vector, while each of the three age policy transitions is mapped to a different component.

The beginning part of the proxel-tree for this model is presented in Figure 5. The values of the probabilities are not stated explicitly in the figure, but are replaced by asterisks. In Figure 6, a result from a proxel-based simulation of this Petri net is presented. The value that was used for *dt* is 0.05, and the simulation was run up to time t = 30, i.e. there were 600 discrete time steps. The computation took 59 seconds on a 1.2 GHz Pentium III notebook. The distribution functions associated with the transitions that were used in the simulation are the following:

- F_1 ~ *Uniform*(0.2, 0.3), F_2 ~ *Deterministic*(0.7), F_3 ~ *Uniform*(0.15, 0.35),
- F_4~*Deterministic*(2.5), F_5~*Uniform*(3.5, 4.5), F_6~*Uniform*(2.0, 3.0).



$(S_0 = (m_0, (0, 0, 0, 0)), 0, \emptyset, 1.0)$
$(S_1 = (m_1, (0, dt, 0, 0)), dt, (m_0), *)$
$(S_2 = (m_0, (dt, dt, 0, 0)), dt, (m_0), *)$
$(S_3 = (m_0, (0, 0, 0, 0)), dt, (m_0), *)$
$((m_1, (dt, dt, dt, 0)), 2dt, (S_0, S_1), *)$
$((m_2, (0, dt, dt, 0)), 2dt, (S_0, S_1), *)$
$((m_1, (0, dt, 0, 0)), 2dt, (S_0, S_1), *)$
$((m_1, (0, 2dt, 0, 0)), 2dt, (S_0, S_2), *)$
$((m_0, (2dt, 2dt, 0, 0)), 2dt, (S_0, S_2), *)$
$((m_0, (0, 0, 0, 0)), 2dt, (S_0, S_2), *)$
$((m_1, (0, dt, 0, 0)), 2dt, (S_0, S_3), *)$
$((m_0, (dt, dt, 0, 0)), 2dt, (S_0, S_3), *)$
$((m_0, (0, 0, 0, 0)), 2dt, (S_0, S_3), *)$

Figure 5: The first three levels of the proxel tree based on the SPN in Figure 3

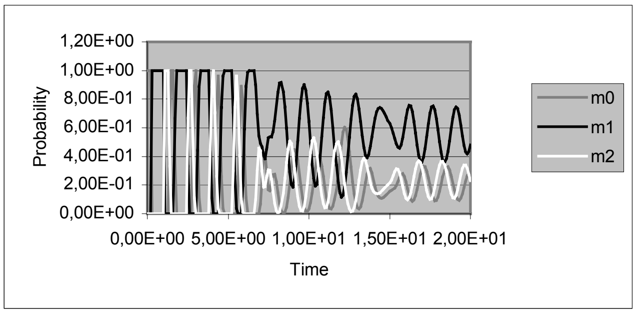


Figure 6: Solution of the example SPN obtained by proxel-based simulation

4 Markov Chain Interpretation of the Proxel-Based Method

Markov chains are one of the most popular tools for modelling and analysing stochastic systems. They possess both simplicity and the ability to model various classes of stochastic systems. It can be shown using a supplementary variable approach that the proxel-based simulation method can be interpreted as a discrete-time Markov chain

$$\pi_k = \pi_{k-1} P$$

where P is the transition probability matrix for one step and π_k is the state occupancy probability row-vector at the k-th time step.

We begin by once again considering a discrete time interval *dt* of the stochastic process of the SPN. If *dt* is sufficiently small, then the probability that more than one state change occurs during this interval is negligible. Recall that the state $S = (m, \vec{\tau})$ of the SPN is defined by a marking m and a vector of age intensities $\vec{\tau}$. For two states S_i, S_j, where S_i is defined at discrete time step k and S_j is defined at discrete time step k+1, we may compute the probability p_{ij} for the change of state from S_i to S_j as follows:

$$p_{ij} = \mu_{ij}(\tau)\,dt \tag{1}$$

where $\mu_{ij}(\tau)$ denotes the instantaneous rate function of the transition leading from marking m_i to marking m_j and τ is the age intensity of that transition. The probability that the SPN remains in marking m_i is then one minus the sum of all such probabilities.

For a given SPN we can thus define a DTMC in which the unknowns correspond to the discretised states of the SPN and the coefficients of the matrix P are given by (1). For simplicity, we will describe the case where the dimension of the age intensity vector is one; this is not a limitation of the proxel-based method.

We consider as an example a simple system which consists of a cashier who can be either free or busy. The Petri net representation together with its reachability graph is shown in Figure 7.

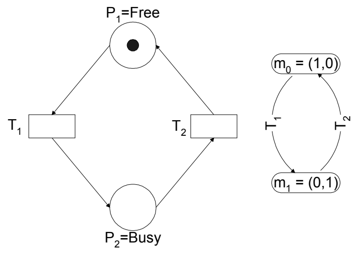


Figure 7: SPN model of the cashier with its reachability graph

We assume for simplicity that transition T_1 and T_2 have exponential delays with rates λ and μ, respectively. This means that the instantaneous rate functions for the both transitions have constant values λ and μ. We also assume, for brevity, that the maximum length of time that each transition can be enabled is $\tau_{max} = 3*dt$. The transition probability matrix for this example is as follows:

$$P = \begin{bmatrix} 0 & 1-\lambda*dt & 0 & 0 & \lambda*dt & 0 & 0 & 0 \\ 0 & 0 & 1-\lambda*dt & 0 & \lambda*dt & 0 & 0 & 0 \\ 0 & 0 & 0 & 1-\lambda*dt & \lambda*dt & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ \mu*dt & 0 & 0 & 0 & 0 & 1-\mu*dt & 0 & 0 \\ \mu*dt & 0 & 0 & 0 & 0 & 0 & 1-\mu*dt & 0 \\ \mu*dt & 0 & 0 & 0 & 0 & 0 & 0 & 1-\mu*dt \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

We will denote by $\pi_{i,j}$ the variable of the DTMC which corresponds to the Petri net being in marking m_i and the age intensity of the transition being $j*dt$. Then, π_k is given by

$$\pi_k = \begin{bmatrix} \pi_{00} & \pi_{01} & \pi_{02} & \pi_{10} & \pi_{11} & \pi_{12} \end{bmatrix}$$

When the DTMC is explicitly created, then a decision on the maximum age intensity τ_{max} must be made. It is obvious that when this point is reached, the transition is forced to fire, which is unrealistic. The proxel-based method avoids this difficulty by adapting the length of the solution vector as needed; when probabilities for states at a new discrete time step would be generated whose value falls below a certain threshold, then these are simply ignored. On the other hand, the proxel simulator will continue to prolong the enabling times of transitions which are still producing non-negligible probabilities. The DTMC for more complex SPNs can be generated analogously. The number of states of the chain can grow exponentially in the number of concurrently enabled transitions – a well-known drawback of the supplementary variable approach.

5 Discussion and Experimental Results

5.1 Discussion of the Algorithm

At first glance, it appears that the method has exponential complexity in time which would prohibit its use in practice. This, however, is not the case. The reasons are as follows:

- the proxel-tree initially grows by a constant number of proxels at each time step; after a certain amount of simulation time has elapsed, it stops to grow altogether, and
- at each time step, several proxels can which occur represent the same state. In the implemented program, these are mapped to one proxel (and their probabilities are summed).

In its basic form, the method has an exponential complexity in the dimension of the age intensity vector. In certain cases, where the instantaneous rate functions associated with the transitions have finite support such as uniform or deterministic distributions, then the method takes advantage of this property and performs quite efficiently, since the value of 0 for the IRF means that the proxel tree does not increase in size. This is one of the advantages of the proxel-based method – that it still performs economically for certain classes of problems. In order to demonstrate this, we present experimental results with respect to time and memory complexity for SPNs containing distributions with and without infinite support. In the latter case, uniform and deterministic distribution functions will be used instead.

5.2 Experimental Results

A number of experiments were made based on the example in section 3.2 in order to show some of the features of the method in terms of computation time, accuracy and memory complexity, so that a better idea of the method can be obtained. The experiments were made on a workstation with a 2.26 GHz Pentium IV processor with 1 GB RAM. The results that are presented in Figure 8 show the probabilities for the model being in the three different markings when using time steps of different sizes. Based on this experiment, it can be concluded that the method is first order accurate with respect to *dt*. This is a valuable feature of the method, because it creates the possibility of calculating an accurate solution value by just linearly extrapolating two solution values obtained with greater values of *dt* to a solution value at *dt*=0. In this manner, accurate solutions can be obtained at significantly reduced cost. It is worth noting that Monte Carlo simulations do not permit an analogous approach.

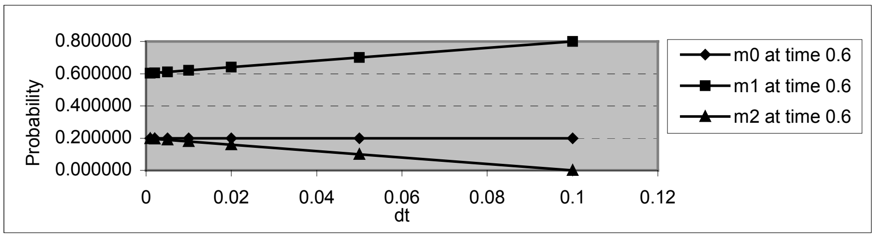


Figure 8: Accuracy of the proxel-based method

Figure 9 shows the times that were needed to run the simulation up to $t = 5$ using different sizes of time step *dt* for the SPNs with transitions with IRFs with infinite and finite support. In the first case (case A), one of the transitions, namely T_4, has a Weibull distribution function assigned to it, and in the second case (case B) it has been replaced with a uniform distribution function whose mean value is approximately the same.

Size of *dt*	Computation Time in Case A	Number of Proxels in Case A	Computation Time in Case B	Number of Proxels in Case B
0.01	486.52	40778212	1.33	483881
0.02	15.33	2547138	0.093	45639
0.03	1.56	411357	0.015	10368
0.04	0.31	114991	< 0.01	3764
0.05	0.16	56606	< 0.01	2494

Figure 9: Computation times and numbers of proxels needed in the two cases

Figure 9 shows the beneficial effect of distributions with finite support: the total number of proxels generated by the algorithm is considerably smaller than for the computation that included the Weibull distribution. The number of proxels grows superlinearly with the number of time steps. This is due to the increase in the number of different values of the age intensities that are active concurrently. The table also shows that a highly accurate result could be obtained at small cost by extrapolating the solutions obtained, for example, with $dt = 0.03$ and $dt = 0.05$.

6 Summary and Outlook

The proxel based method is an alternative method for analysing the behaviour of discrete stochastic models such as stochastic Petri nets. It follows the behaviour of the model considering every possible development and calculating the probabilities of the state changes based on the distribution functions that describe them. The method is applied under the approximating assumption that just one state change occurs during any interval of length *dt*.

In this paper, the analysis of stochastic Petri nets using the proxel-based method was described. There were formalities and relations that were necessary to be established in order to carry out this kind of analysis. The method was interpreted as a DTMC whose dimension is dynamically adapted. Experiments were carried out in order to test the method's properties. The results obtained showed that the proxel-based method is of first-order accuracy and exploits density functions with finite support, in which case it performs very efficiently. The method also performs well for models with rare events [Hor02][Laz03], for which Monte Carlo simulations can need a very large number of replications.

The SPN specification presented in this paper excludes the use of immediate transitions. We believe that these can also be treated using proxels. The probabilities are computed directly from the transition probabilities and the simulation time is not advanced. This would, however, prohibit the use of vanishing loops in the SPN, which is, however, commonplace and not a serious limitation.

The method, however, is still in its early stage, which leaves a large space for improvements, which we believe will extend the class of problems for which it performs efficiently. Many details with respect to its memory and computational complexity optimisation are subjects of future research. One of them is using an adaptable size of the time step [Hor03], which we believe can increase the method's performance.

We believe that the proxel-based method is useful when fast analysis and rough approximations of the solutions of stochastic systems are required (which can be controlled with the size of *dt*), as well as when a highly accurate solutions are needed and the time is not an issue. The extrapolation of the solutions obtained by using different values for *dt* is also an option that can lead to fast and efficient solutions. This shows that the proxel-based method is very flexible.

References

[Hor02] G. Horton: *A new paradigm for the numerical simulation of stochastic Petri nets with general firing times.* Proceedings of the European Simulation Symposium 2002, Dresden. Society for Computer Simulation, 2002.

[Ger00] R. German: *Performance Analysis of Communication Systems. Modeling with Non-Markovian Stochastic Petri Nets*. John Wiley & Sons, Ltd, 2000.

[Laz03] S. Lazarova-Molnar, Graham Horton: *An Experimental Study of the Behaviour of the Proxel-Based Simulation Algorithm.* Simulation und Visualisierung 2003. SCS Verlag 2003.

[Hor03] G. Horton, S. Lazarova-Molnar: *A Reduced-Stiffness Method for the Transient Solution of Discrete-Time Markov Chains.* Submitted to Numerical Solution of Markov Chains, University of Illinois at Urbana-Champaign, 2003.

[Mol03] S. Lazarova-Molnar, G. Horton: *Proxel-Based Simulation of Stochastic Petri Nets containing Immediate Transitions.* On-Site Proceedings of the Satellite Workshop of ICALP 2003 in Eindhoven, Netherlands. Forschungsbericht Universität Dortmund. Dortmund 2003.