ROMAI J., **2**, 1(2006),
181–188

USING STOCHASTIC PETRI NETS IN PERFORMANCE ANALYZE OF DISTRIBUTED SYSTEMS

Norocel Petrache

Abstract This paper shows how Stochastic Petri Nets (SPNs) (Petri Nets with timed transitions and random, negative exponentially distributed firing delays) can be used to investigate performances of distributed systems. Because SPN systems are isomorphic to continuous time Markov chains (CTMCs), their analysis can be performed by construction of the state transition rate matrix. A model of a simple shared memory system and the software tool GreatSPN are described.

1. PETRI NETS AND TEMPORAL CONCEPTS

The concept of time was intentionally avoided in the original work by C. A. Petri, because of the effect that timing may have on the behaviour of PNs: the association of timing constraints with the activities represented in PN models or systems may prevent certain transitions from firing, thus invalidating the important assumption that all possible behaviours of a real system are represented by the structure of the PN.

Temporal concepts were introduced in Petri nets models in the mid 1970s. This initiated a hot debate about the suitability of such an extension, and about the most appropriate ways to make it. Both Petri nets with timed places and timed transitions were proposed; in the latter case there was either a time delay (before the atomic occurrence of a transition) or a time interval (between the consumption of input tokens and the generation of output tokens). Whether deterministic or stochastic timing should be used was also an issue of the discussion.

The firing of a transition in a PN model corresponds to the event that changes the state of the real system. This change of state can be due to the completion of some activity. Transitions can be used to model activities, so that transition enabling periods correspond to activity executions and transition firings correspond to activity completions. Hence, time can be naturally associated with transitions. Transitions with associated temporal specifications are called *timed transitions*; they are graphically represented by boxes or thick bars.

A timed transition T can be associated with a local clock or timer. When T is enabled the associated timer is set to an initial value. The timer is then

decremented at constant speed and the transition fires when the timer reaches the value zero. The timer can thus be used to model the duration of an activity whose completion induces the state change that is represented by the change of marking produced by the firing of T.

It is important to note that the activity is assumed to be in progress while the transition is enabled. This means that in the evolution of complex nets, an interruption of the activity may take place if the transition loses its enabling condition before it can actually fire. The activity may be resumed later on, during the evolution of the system in the case of a new enabling of the associated transition. This may happen several times until the timer goes down to zero and the transition finally fires.

When introducing time into PN models and systems, it would be extremely useful not to modify the basic behaviour of the underlying untimed model. By so doing, it is possible to study the timed PNs exploiting the properties of the basic model as well as the available theoretical results. The addition of temporal specifications therefore must not modify the unique and original way of expressing synchronization and parallelism that is peculiar to PNs. In untimed PN systems, the choice of which transition to fire in a free-choice conflict is completely nondeterministic. In the case of timed PN systems, the conflict resolution depends on the delays associated with transitions and is obtained through the so-called *race policy*: when several timed transitions are enabled in a given marking M, the transition with the shortest associated delay fires first.

An important issue that arises at every transition firing when timed transitions are used in a model is how to manage the timers of all the transitions that do not fire. From the modelling point of view, the different policies that can be adopted link the past history of the systems to its future evolution considering various ways of retaining memory of the time already spent on activities. The question concerns the memory policy of transitions, and defines how to set the transition timers when a state change occurs, possibly modifying the enabling of transitions. Two basic mechanisms can be considered for a timed transition at each state change:
- continue: the timer associated with the transition holds the present value and will continue later on the count-down;
- restart: the timer associated with the transition is restarted, i.e., its present value is discarded and a new value will be generated when needed.

The main reason for the introduction of temporal specifications into PN models is the interest for the computation of *performance indexes*. The introduction of temporal specifications in PN models must not reduce the modelling capabilities with respect to the untimed case (it must be done so as not to modify the basic behaviour and specifically the non-determinism of the choices occurring in the net execution). If the temporal specification is given in a de-

terministic way, the behaviour of the model is deterministically specified, and the choices due to conflicting transitions may be always solved in the same way; this would make many of the execution sequences in the untimed PN system impossible. Instead, if the temporal specifications are defined in a stochastic manner, by associating independent continuous random variables with transitions to specify their firing delays, the non-determinism is preserved (in the sense that all the execution sequences possible in the untimed PN system are also possible in the timed version of the PN system).

2. STOCHASTIC PETRI NETS

A simple sufficient condition to guarantee that the qualitative behaviour of PN models with timed transitions is identical to the qualitative behaviour of the underlying untimed PN model (that is, to guarantee that the possible transition sequences are the same in the two models) is that the delays associated with transitions be random variables whose distributions have an infinite support. If this is the case, a probabilistic metrics transforms the *nondeterminism* of the untimed PN model into the *probabilism* of the timed PN model, so that the theory of discrete-state stochastic processes in continuous time can be applied for the evaluation of the performance of the real system described with the timed PN model.

Discrete-state stochastic processes in continuous time may be very difficult to characterize from a probabilistic viewpoint and virtually impossible to analyse. Their characterization and analysis become reasonably simple in some special cases that are often used just because of their mathematical tractability. The simplicity in the characterization and analysis of discrete-state stochastic processes in continuous time can be obtained by eliminating the amount of *memory* in the dynamics of the process. This is the reason for the adoption of the *negative exponential probability density function* (pdf) for the specification of random delays. The negative exponential is the only continuous pdf that enjoys the *memoryless property*, i.e., the only continuous pdf for which the residual delay after an arbitrary interval has the same pdf as the original delay.

Stochastic Petri Nets (SPNs) were proposed, simultaneously, by S. Natkin [Nat80] and M. K. Molloy [Mol82]. The new net class paved the way to the utilization of Petri nets for performance evaluation. In a SPN, a firing delay is associated with each transition. Firing delays are instances of random variables that have a negative exponential probability distribution. The rates are sufficient to characterize the pdf of the transition delays (the only parameter of the negative exponential pdf is its *rate*, obtained as the inverse of the *mean*). The selection of the transition instance to fire among the set of enabled ones follows a race policy (the transition that has drawn the least delay is the one

that fires). Each timed transition can be used to model the execution of an activity in a distributed environment; all activities execute in parallel (unless otherwise specified by the PN structure) until they complete. At completion, activities induce a local change of the system state that is specified with the interconnection of the transition to input and output places.

No special mechanism is necessary for the resolution of timed conflicts because the probability that two timers expire at the same instant is zero. The negative exponential pdf is a continuous function defined in the interval [0,1), that integrates to one. The lack of discontinuities in the function makes the probability of any specific value x being sampled equal to zero (obviously, the probability that a value is sampled between two distinct values $x_1 \geq 0$ and $x_2 > x_1$ is positive). Thus, the probability of two timers expiring at the same time is null (given the value sampled by the first timer, the probability that the second one samples the same value is zero). The memoryless property of the negative exponential pdf, at any time instant, makes the residual time until a timer expires statistically equivalent to the originally sampled timer reading. Thus, whether a new timer value is set or not at every change of marking makes no difference from the point of view of the probabilistic metrics of the SPN. A formal definition of SPN is the following.

Definition 2.1 *SPN = (P, T, A, M, λ), where P is the set of places, T is the set of transitions, $P \bigcap T = \theta$, $P \bigcup T \neq \theta$, A is the set of input and output arcs, $A \subseteq (P \times T) \bigcup (T \times P)$, M is the initial marking and λ is the set of transition rates.*

A marking of a stochastic Petri net is a distribution of tokens in its places; a marking may be viewed as a mapping from the set of places to the natural numbers. A stochastic Petri net is said to be *k-bounded* if there exists a finite nonnegative integer k such that for every marking M in the reachability set, and for every place P_i $M(P_i) < k$. With each marking we can associate a state of the system and in the following the terms *state* and *marking* are used with essentially the same meaning.

Molloy [Mol81] has proved that there is an *isomorphism* between k-bounded SPNs and finite Markov processes. Two stochastic systems are *isomorphic* if
- there are one-to-one mappings between the state space of the two systems, and between the set of state transitions of the two systems,
- the probability of a transition from one state to another in one system equals the probability of a transition between the corresponding states of the other system.

The SPNs are isomorphic to continuous time Markov chains due to the memoryless property of the negative exponential distribution of firing times. The SPN markings correspond to states of the corresponding Markov chain. Since the size of the state space of a SPN is equal to the size of Markov process space, the complexity of solving an SPN model is the same as in the case of

the model based upon the Markov process; the methodology used to find the steady-state solution for a Markov chain can be used for an SPN system in which each marking corresponds to a Markov state. Although the SPNs do not provide more modelling power than Markov processes, they can be used as a convenient description of the system being modelled.

3. THE STOCHASTIC PROCESS ASSOCIATED WITH A SPN

SPN systems are isomorphic to continuous time Markov chains (CTMCs); k-bounded SPN systems are isomorphic to finite CTMCs. The CTMC associated with a given SPN system is obtained by applying the following rules:

1. the CTMC state space $S = \{s_i\}$ corresponds to the reachability set $RS(M_0)$ of the PN associated with the SPN ($s_i \leftrightarrow M_i$)

2. the transition rate from state s_i (corresponding to marking M_i) to state s_j (M_j) is obtained as the sum of the firing rates of the transitions that are enabled in M_i and whose firings generate marking M_j.

It is possible to develop algorithms for the automatic construction of the infinitesimal generator (also called *the state transition rate matrix*) of the isomorphic CTMC, starting with the SPN description. Denoting this matrix by $\mathbf{Q}$, with w_k the firing rate of T_k, and with $E_j(M_i) = \{T_h \in E(M_i) : M_i[T_h > M_j\}$ the set of transitions whose firings bring the net from marking M_i to marking M_j, the components of the infinitesimal generator are

$$q_{ij} = \begin{cases} -\sum\limits_{k:T_k \in E(M_i)} w_k, & i = j \\ \\ \sum\limits_{k:T_k \in E_j(M_i)} w_k, & i \neq j \end{cases}$$

In SPN analysis, as in Markov analysis, *ergodic* (irreductible) systems are of special interest. A k-bounded SPN system is said to be ergodic if it generates an ergodic CTMC; in [FN85] is showed that a SPN system is ergodic if M_0, the initial marking, is a *home state* (a marking $M \in RS(M_0)$ is a home state if and only if $\forall N \in RS(M_0), M \in RS(N)$). For ergodic SPN systems, the steady-state probability of the system being in any state always exists and is independent of the initial state.

Let the row vector η represent the steady-state probability distribution on markings of the SPN; to compute η we solve the linear system expressed in matrix form as

$$\begin{cases} \eta Q &= 0, \\ \\ \eta 1^T &= 1, \end{cases}$$

where **0** is row vector with all its components equal to zero and 1^T is a column vector with all its components equal to one, used to enforce the normalization condition. We will use η_i instead of $\eta(M_i)$ to denote the steady state probability of marking M_i. The *sojourn time* is the time spent by the system in a given marking. Because a CTMC can be associated with the SPN system, the sojourn time in marking M_i is exponentially distributed with rate q_i. The pdf of the sojourn time in a marking corresponds to the pdf of the minimum among the firing times of the transitions enabled in the same marking. The probability that a given transition $T_k \in E(M_i)$ fires first in marking M_i has the expression: $P\{T_k/M_i\} = \frac{w_k}{q_i}$ and the average sojourn time in marking M_i is given by $E[SJ_i] = \frac{1}{q_i}$.

To allow the efficiency of the SPN system to be evaluated, *performance indices* can be computed by using *reward functions* with different interpretations: the probability of a particular condition, the mean number of firings per unit of time of a given transition, the expected value of the number of tokens in a given place etc. If r(M) is a reward function, the average reward can be computed with the following weighted sum

$$R = \sum_{i:M_i \in RS(M_0)} r(M_i)\eta_i.$$

In order to illustrate this analysis step, the following example [MBCDF94] is presented.

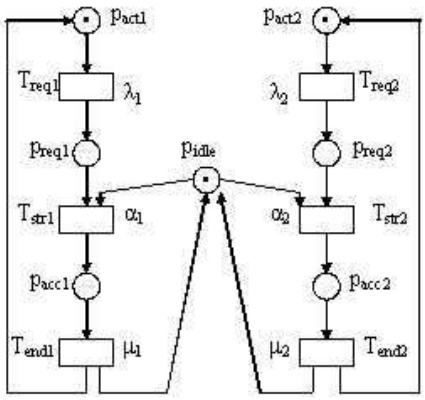

Fig. 1. A shared memory system

In fig. 1, two processors try to access a common shared memory; in the initial marking, the two processors are both in a locally active state ($p_{act1} + p_{act2} + p_{idle}$). Processor 1 works locally for an exponentially distributed random amount of time with average $\frac{1}{\lambda_1}$, and then requests an access to the common memory (T_{req1} fires). If common memory is available, (place *pidle* is marked),

the acquisition of the memory starts and takes an average of $\frac{1}{\alpha_1}$ units of time to complete. After transition T_{str1} fires, processor 1 uses the common memory (which is not available for processor 2) for $\frac{1}{\mu_1}$ units of time (on average) and when transition T_{end1} fires, the net is returning to its initial state. A similar processing cycle is possible for processor 2.

The evolution of the net describes the interleavings between the activities of the processors. A conflict exists when transitions T_{str1} and T_{str2} are both enabled (both processors want to simultaneously access the common memory). Transition T_{str1} fires with probability $P\{T_{str1}\} = \frac{\alpha_1}{\alpha_1+\alpha_2}$, whereas transition T_{str2} fires with probability $P\{T_{str2}\} = \frac{\alpha_2}{\alpha_1+\alpha_2}$. The conflict is resolved when the first transition fires and the speed at which the PN model exits from this marking is the sum of the individual speeds of the two transitions. The reachability set is $\{M_0 = p_{act1} + p_{idle} + p_{act2},\ M_1 = p_{req1} + p_{idle} + p_{act2},\ M_2 = p_{acc1} + P_{act2},\ M_3 = p_{acc1} + p_{req2},\ M_4 = p_{req1} + p_{idle} + p_{req2},\ M_5 = p_{act1}+p_{idle}+p_{req2},\ M_6 = p_{act1}+p_{acc2},\ M_7 = p_{req1}+p_{acc2}\}$ and the infinitesimal generator of the Markov chain is Q=

$$\begin{bmatrix} -\lambda_1-\lambda_2 & \lambda_1 & 0 & 0 & 0 & \lambda_2 & 0 & 0 \\ 0 & -\alpha_1-\lambda_2 & \alpha_1 & 0 & \lambda_2 & 0 & 0 & 0 \\ \mu_1 & 0 & -\lambda_2-\mu_1 & \lambda_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\mu_1 & 0 & \mu_1 & 0 & 0 \\ 0 & 0 & 0 & \alpha_1 & -\alpha_1-\alpha_2 & 0 & 0 & \alpha_2 \\ 0 & 0 & 0 & 0 & \lambda_1 & -\alpha_2-\lambda_1 & \alpha_2 & 0 \\ \mu_2 & 0 & 0 & 0 & 0 & 0 & -\lambda_1-\mu_2 & \lambda_1 \\ 0 & \mu_2 & 0 & 0 & 0 & 0 & 0 & -\mu_2 \end{bmatrix}$$

The system of linear equations whose solution yields the steady-state distribution over the CTMC states is

$$\begin{cases} (\eta_0, \eta_1, \eta_2, \eta_3, \eta_4, \eta_5, \eta_6, \eta_7)Q = 0, \\ \sum_{i=0}^{7} \eta_i = 1. \end{cases}$$

Assuming $\lambda_1 = 1, \lambda_2 = 2, \alpha_1 = \alpha_2 = 100, \mu_1 = 10$ and $\mu_2 = 5$, we obtain $\eta_0 = 0.61471, \eta_1 = 0.00842, \eta_2 = 07014, \eta_3 = 0.01556, \eta_4 = 0.00015, \eta_5 = 0.01371, \eta_6 = 0.22854, \eta_7 = 0.04876$.

Since one token is present in place p_{idle} in markings M_0, M_1, M_4 and M_5, the average number of tokens in p_{idle} is $E[M(p_{idle})] = \eta_0 + \eta_1 + \eta_4 + \eta_5 = 0.637$ and the utilization of the shared memory is $U[shared_memory] = 1.0 - E[M(p_{idle})] = 0.363$.

Since T_{req1} is enabled only in M_0, M_5 and M_6, the rate of access to the shared memory from the first processor (the throughout of T_{req1}) is $f_{req1} = (\eta_0 + \eta_5 + \eta_6)\lambda_1 = 0.8570$.

These results show that Stochastic Petri Nets can be used as a tool for computing performance indices that allow the efficiency of modelled systems to be evaluated.

Developed at Universita di Torino, GRaphical Editor and Analyzer for Timed and Stochastic Petri Nets (GreatSPN) [Chi91] is a software package for the modeling, validation, and performance evaluation of distributed systems using Stochastic Petri Nets. It runs on Linux (gcc, Motif 1.2 and X11R6 are required) and it offers graphical model editing, definition of timing and stochastic specifications, parameters, and performance measures, Markovian solvers for steady-state, graphical representation of performance results and graphical interactive simulation of stochastic models. The package is available for free for universities and non-profit organizations (http://www.di.unito.it/ greatspn/index.html).

References

[Chi91] G. Chiola , *GreatSPN 1.5 software architecture*, in Proc. 5th Int. Conf. Modeling Techniques and Tools for Computer Performance Evaluation, Torino, Italy, 1991.

[FN85] G. Florin, S. Natkin, *Les reseaux de Petri stochastiques*, Technique et Science Informatiques **4**, 1, 143-160.

[MBBCCC85] M. A. Marsan, G. Balbo, A. Bobbio, G. Chiola, G. Conte, A.C. Cumani, *On Petri nets with stochastic timing*, in Proc. Int. Workshop Timed Petri Nets, Torino, Italy, July 1985, 80-87.

[MBCDF94] M. A. Marsan, G. Balbo, G. Conte, S. Donatelli, G. Franceschinis, *Modelling with generalised stochastic Petri nets*, Università degli studi di Torino, Dipartimento di Informatica, Torino, Italy, 1994.

[Mol81] M.K. Molloy, *On the integration of delay and throughout measures in distributed processing models*, Ph. D. Thesis, Univ. of California, Los Angeles, Sept. 1981.

[Mol82] M. K. Molloy, *Performance analysis using stochastic Petri nets*, IEEE Trans. Comput., vol. C-31, Sept. 1982, 913-917, .

[Nat80] S. Natkin, *Le reseaux de Petri stochastiques et leur application a l'evaluation des systems informatiques*, Thèse de Docteur Ingenieur, CNAM, Paris, 1980.