36th Midwest Symposium on Circuits and Systems; Detroit MI, 16-18 August 1993, pp.930-934.

THROUGHPUT ANALYSIS OF SIMPLE CLOSED TIMED PETRI NET MODELS

W.M. Zuberek

Department of Computer Science
Memorial University of Newfoundland
St. John's, Canada A1C–5S7

A b s t r a c t

It is shown that for some classes of closed timed Petri net models the steady–state behavior can be determined on the basis of structural properties only (i.e., without reachability analysis), using the concept of throughput and simple rules of operational analysis. Throughput analysis is based on the average values of firing times rather than firing time distribution functions, so the same approach can be used to a variety of net models. Bounded as well as unbounded (but ergodic) net models can be analyzed by the proposed approach.

1. INTRODUCTION

Petri nets [Mu89,Re85] have been proposed to model systems of events in which it is possible for some events to occur concurrently, but there are constraints on the occurrence, precedence, or frequency of these occurrences. Examples of such systems include multiprocessor and distributed systems, communication networks and data flow architectures. In order to study performance aspects of Petri net models, the duration of activities must also be taken into account and included into model specifications. Timed Petri nets have been introduced by assigning firing times to the transitions of Petri nets (t–timed nets) [Ram74,Zu86,Ho86,Zu88] or by assigning time to places of a net (p–timed nets) [Sif77]. It should be noted that a fundamental difference between these two approaches is in the mechanism of transition firings. In t–timed nets, transition firings are not instantaneous events; a firing occurs in 'real-time', i.e., the tokens are removed from transition's input places at the beginning of the firing period, and they are deposited to the output places at the end of this period (sometimes this is called a 'three–phase' firing mechanism). In p–timed nets, the 'firing time' determines the period of time that tokens must 'wait' (in places) before firing can occur, and the firing is is an instantaneous event, as in ordinary nets (so the 'firing times' are – in fact – 'enabling times'). A simple consequence of this difference is that the 'firing' process in a p–timed net can easily be discontinued (using a simple configuration of conflicting transitions), while in t–timed nets there in no 'access' to tokens once the firing started, so a special type of inhibitor arcs has been proposed to provide this capability [Zu88]. On the other hand, in p–timed nets, the conflict resolution policies cannot be defined independently of timing specifications [C3S89], while in t–timed nets conflict resolution and timing specification are quite independent aspects. Stochastic Petri nets [Na80] and generalized stochastic nets [ACB84] are also p–timed nets. And there is a variety of generalized, augmented, extended and enhanced net models [CMT89,Du85,Ho86] which belong to one of these basic categories of nets 'with time'.

This paper discusses free–choice timed Petri nets in which (deterministic or stochastic) firing times are associated with transitions of a Petri net, as in [Ram74,Zu86,Ho86,Zu88]. For such models, the 'state space' is a discrete–space, discrete–time or continuous–time (depending upon the character of firing times of transitions) homogeneous semi–Markov process [Zu86,Zu88]. If this process is ergodic, the stationary probabilities of the states can be determined [Ki90], and many performance measures, such as utilization of systems components, average waiting times and turnaround times or average throughput rates, characterizing the steady–state behavio of the model, can be derived from stationary probabilities of states.

Analysis of net models based on the derivation of the state space is known as the reachability analysis. Although reachability analysis is quite general (e.g., it can easily handle state–dependent routing probabilities as well as state–dependent timing properties), it becomes inefficient for large state spaces (for some models, the number of states grows exponentially with model parameters, which is known as the 'state explosion problem'). Also, reachability analysis is usually restricted to bounded nets. Therefore, approaches that avoid the generation of the state space are recently gaining popularity. Such approaches are based on structural properties of models. Although structural approaches do not provide as much information as the reachability approach does, quite often, however, all the detailed results of reachability analysis are not really needed, and more synthetic performance measures, that can also be provided by structural approaches, are quite satisfactory. Structural approaches can be used to obtain exact or approximate performance measures, e.g., lower and upper performance bounds [BG85,C3S89].

In performance analysis, one of basic performance measures is throughput of a model or a model's component. Many other performance characteristics can be obtained from throughputs, for example, the throughput and the maximum service rate of a component determine its utilization factor, which – in turn – is an indicator of systems saturation. The throughput is one of the elements used in the Little's formula, etc. Throughput analysis of net models is similar to operational methods developed for analysis of queueing systems

[Bu76,DB78]. It is based on the first moment (the average value) of random variables, and general relationships which do not depend upon probability distribution functions.

The paper is organized in four main sections. Section 2 recalls basic concepts for free–choice timed Petri nets. Section 3 briefly describes fundamental concepts of operational analysis, while Section 4 introduces the concept of throughputs in nets and derives throughput values for several classes of closed net models. Section 5 illustrates the proposed approach using a simple model of a multiprogrammed computer system.

2. TIMED PETRI NETS

This section recalls and formalizes concepts used in subsequent parts of this paper. It is rather concise since more detailed discussion is provided elsewhere [Zu86,Zu88].

A Petri net (or a Petri net structure) N is a triple N=(P, T, A) where:

P is a finite, nonempty set of places,

T is a finite, nonempty set of transitions,

A is a set of directed arcs, $A \subseteq P \times T \cup T \times P$ such that for each transition there is at least one place connected with it.

For each place p and each transition t, the input and output sets are defined as follows:

$$Inp(p) = \{t \in T | (t,p) \in A\}, \quad Out(p) = \{t \in T | (p,t) \in A\},$$
$$Inp(t) = \{p \in T | (p,t) \in A\}, \quad Out(t) = \{p \in P | (t,p) \in A\}.$$

and this notation is extended on sets of places and transitions.

A marked Petri net $\mathcal{M}$ is a pair $\mathcal{M} = (\mathcal{N}, m_0)$ where:

$\mathcal{N}$ is a Petri net, $\mathcal{N} = (P, T, A)$,

m_0 is an initial marking function, $m_0 : P \to \{0, 1, ...\}$.

Let any function $m : P \to \{0, 1, ...\}$ be called a marking in a net $\mathcal{N} = (P, T, A)$.

A place p is shared iff it is an input place for more than one transition. A shared place p is free–choice (or extended free–choice) iff the input sets of all transitions sharing p are identical. A net is free–choice iff all its shared places are free–choice. Only free–choice nets are considered in this paper since in most cases free–choice nets are sufficient for modeling random events, e.g., random faults in communication networks or any random events described by discrete distributions.

Since the relation of sharing a free–choice place is an equivalence relation in T, it determines a partition of T into a set of free–choice equivalence classes denoted by $Free(T) = \{T_1, T_2, ..., T_k\}$.

A transition t is enabled by a marking m iff every input place of this transition contains at least one token. Every transition enabled by a marking m can fire. When a transition fires, a token is removed from each of its input places (but not inhibitor places) and a token is added to each of its output places. This determines a new marking in a net, a new set of enabled transitions, and so on.

In timed Petri nets each transition takes a 'real time' to fire, i.e., there is a 'firing time' associated with each transition of a net. The firing times can be defined in several ways. In D–timed Petri nets [Zu88] they are deterministic (or constant), i.e., there is a nonnegative number assigned to each transition of a net which determines the duration of transition's firings. In M–timed Petri nets [Zu86] (or stochastic Petri nets [Na80,ACB84]), the firing times are exponentially distributed random variables, and the corresponding firing rates are assigned to transitions of a net. In this paper, the firing times associated with transitions of the net are the average values of firing times.

A free–choice timed Petri net $\mathcal{T}$ is a triple $\mathcal{T} = (\mathcal{M}, c, f)$ where:

$\mathcal{M}$ is a free–choice marked Petri net, $\mathcal{M} = (\mathcal{N}, m_0)$, $\mathcal{N} = (P, T, A)$,

c is a choice function which assigns a 'free–choice' probability to each transition t of the net in such a way that for each free–choice equivalence class $T_i \in Free(T)$ the sum of these probabilities is equal to 1,

f is a firing time function which assigns the nonnegative (average) firing time $f(t)$ to each transition t of the net, $f : T \to \mathbf{R}^{\oplus}$, and $\mathbf{R}^{\oplus}$ denotes the set of nonnegative real numbers.

The behavior of a timed Petri net can be represented by a sequence of 'states' where each 'state' describes the distribution of tokens in places and firing transitions of the net; detailed definitions of states and state transitions for D–timed and M–timed nets are given in [Zu88] and [Zu86], respectively. The states and state transitions can be combined into a graph of reachable states; this graph is a semi–Markov process defined by the timed net $\mathcal{T}$.

A timed net is ergodic iff the semi–Markov process defined by it is ergodic. Only ergodic timed nets are considered in this paper.

3. ELEMENTS OF OPERATIONAL ANALYSIS

Two principal reasons for the introduction of operational analysis are [FSZ83]:

- the need for providing performance analysis with mathematical relationships involving experimentally measurable quantities and characterizing a system's performance during a given time interval,
- the possibility of deriving performance results by using only measurable quantities under assumptions that can be experimentally verified.

The time period in which a system is observed is called the 'observation interval'. The variables measured (or observed) over an observation interval are called 'operational variables'. Operational analysis tries to provide a framework in which system performance can be studied using only operational variables and mathematical relationships among them (called 'operational laws').

There are two main assumptions on which operational analysis is based. First, the system, during observation interval, is in operational equilibrium. i.e., flow balanced; the average number of processing requests that the system or one of its components receives must be equal to the the average number of requests processed by the system or its components. Second, the arrival rate of processing requests is independent of the length of queues at other resources (homogeneous arrivals); similarly, the mean time between completions is independent of the length of queues at system resources (homogeneous services). Essentially, the homogeneity assumptions state that there must not be interactions between the behavior of a component and that of the rest of the system except for those occurring through the local queue of this component. These assumptions are approximately satisfied by real systems if they are observed by sufficiently long periods [FSZ83].

Operational variables are either basic quantities, which are directly measured during the observation interval, or derived quantities, which are computed form the basic quantities. For a typical single–server queueing system the basic operational variables are:

- T – the observation interval,
- $A(T)$ – the number of arrivals occurring during T,
- $C(T)$ – the number of completions occurring during T,
- $B(T)$ – the total amount of time during which the system is busy during T ($B(T) \leq T$),
- $Q(T)$ – the total time spent in the system by all requests.

Typical derived quantities include:

- $\lambda = A(T)/T$ – the arrival rate (i.e., the average number of arrivals per time unit),
- $\theta = C(T)/T$ – the output rate or the throughput (i.e., the average number of completions per time unit),
- $u = B(T)/T$ – the utilization factor (i.e., the fraction of time when the system is busy),
- $s = B(T)/C(T)$ – the mean service time,
- $\mu = 1/s$ – the service rate,
- $R_t = Q(T)/A(T)$ – the mean response time (i.e., the average time spent in the system),
- $N = Q(T)/T$ – the mean number of requests in the system.

Operational laws or operational identities are relations between operational variables which must hold for every observation interval:

- $N = \lambda R_t$ – Little's law,
- $u = s\theta$ – utilization law,
- $\theta_i = \theta_1 A_i(T)/A_1(T)$ – forced flow law (for systems composed of a number of components indicated by subscripts; subscript '1' indicates the 'reference' component).

4. THROUGHPUTS IN NETS

Intuitively, throughput of a place p in a timed net $\mathcal{T}$, $\theta_{\mathcal{T}}(p)$, is equal to the average number of tokens entering p in a unit time, or leaving p (or t) in a unit time; in the steady–state of the net, the average numbers of tokens entering and leaving p must be equal since no 'accumulation' of tokens can occur. Similarly, throughput of a transition t in a net $\mathcal{T}$, $\theta_{\mathcal{T}}(p)$, is equal to the average number of new (or completed) transition's firings in a unit time. It should be noted that the throughput of a transition does not depend upon the number of incoming or outgoing arcs.

More formally, the throughput of a timed net $\mathcal{T}$ is defined as a function $\theta : P \cup T \to \mathbf{R}^{\oplus}$ which assigns a nonegative number to each place and each transition of the net in such a way that:

$$\forall (x \in P \cup T)\ \theta(x) = \lim_{n\to\infty} \frac{n}{\tau_n(x)}$$

where $\tau_n(x)$ denotes the time instant at which the n-th consecutive token enters (or leaves) the place x or at which the transition x initiates (or terminates) its n-th firing.

It follows immediately from the definition of throughput that [Zu92]:

- the throughput of a place p is equal to the sum of throughputs of its input transitions as well as the sum of throughputs of its output transitions:
  $\forall (p \in P)\ \theta(p) = \sum_{t_i \in Inp(p)} \theta(t_i) = \sum_{t_j \in Out(p)} \theta(t_j)$,
- for each non–shared place p, the throughput of p's output transition is equal to the throughput of p:
  $\forall (p \in P)\ Out(p) = \{t\} \Rightarrow \theta(t) = \theta(p)$,
- for each free–choice place p, throughputs of p's output transitions are determined by the choice function c:
  $\forall (T_i \in Free(T))\ \forall (p \in Inp(T_i))\ \forall (t \in T_i)\ \theta(t) = c(t)\theta(p)$.

An elementary net is a connected net in which there is exactly one input place and exactly one output place for each transition of the net, and one input transition and one output transition for each place of the net. In other words, the (directed) graph of an elementary net is a (simple) cycle. It follows from property 1 that in elementary nets the throughputs of all transitions and all places are the same. To determine the value of these throughputs, the Little's law can be applied to an elementary net considered as an open system (by braking one of the arcs) in which the mean response time is equal to the sum of (the average) firing times of all transitions.

For a timed elementary net $\mathcal{T}$:

$$\forall(x \in P \cup T)\ \theta(x) = \frac{\sum_{p\in P} m_0(p)}{\sum_{t\in T} f(t)}$$

A state graph (or a FSM net) is a connected net in which there is exactly one input and one output place for each transition of the net (so only the places can have several incoming and/or outgoing arcs associated with them). State graphs are obviously free–choice, so the selections of conflicting firings are random variables described by the choice function c, from which the relative frequencies of visiting (relative with respect to one of elements) called the 'visiting ratios', $v(x)$, $x \in P \cup T$, can be determined by solving a system of simultaneous equations:

$$\begin{cases} \forall(p \in P)\ v(p) = \sum_{t\in Inp(p)} v(t) \\ \forall(t \in T)\ v(t) = c(t)v(Inp(t)) \\ v(x_0) = 1 \text{ where } x_0 \text{ is the 'reference element'} \end{cases}$$

For a timed state graph $\mathcal{T}$:

$$\forall(x \in P \cup T)\ \theta(x) = v(x)\frac{\sum_{p\in P} m_0(p)}{\sum_{t\in T} v(t)f(t)}$$

Analysis of nets in which transitions can have more than one input arc must take into account 'synchronization delays' which do not exist in state graphs (and elementary nets). Since firing of a transition removes (single) tokens from all input places simultaneously, some tokens may wait (in places) for the enabling of a transition. Such waiting times will affect the response time of timed models. Moreover, since response times usually depend upon the throughput of a server, the behavior of such models must be characterized by nonlinear dependencies with respect to throughputs.

More specifically, in a very simple model of a single server queueing system shown in Fig.1 place p_1 represents the queue of waiting requests, the transition t_1 is the server with the average service time $f(t_1)$, place p_2 indicates (by a single token) that the server is idle. Whenever there is a (waiting) request in p_1 and the server is idle, t_1 is enabled and starts its firing by removing (single) tokens from p_1 and p_2. It can be observed that multiple (initial) tokens in p_2) would represent multiserver systems.

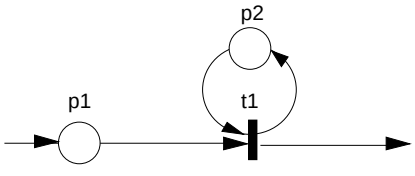

Fig.1. Petri net model of a single server system.

For a given service rate, the response time of such a system depends upon the arrival rate (and throughput which, in the steady–state is equal to the arrival rate). For example, in the case of $M/M/1$ system, i.e., a system with exponentially distributed interarrival and service times, the response time is [Ki90]:

$$R_t = \frac{s}{1-\theta s}$$

where s is the average service time and θ the throughput. Consequently, the server represented by the transition t_1 and place p_2 in Fig.1 can be modeled by a single–input single–output transition with the average firing time equal to the response time of the original system. Such delay–equivalent transformations can be very useful in analysis of net models.

5. EXAMPLE

The M–timed net shown in Fig.2 is a model of an interactive multiprogramming system. p_1 and t_1 model the users with $f(t_1)$ representing the average 'terminal time' or 'thinking time'; the initial marking of p_1 represents the number of (active) users. p_6 and t_2 represent the central server with its waiting queue p_2, while p_5 and t_5 model a disk server with its queue p_4; the number of (identical) processors in the central server is determined by the initial marking of the place p_6, and the number of (identical) disk drives by the initial marking of the place p_5. The place p_3 is a free–choice place with two choices, termination of job execution (transition t_3 with probability q) or continuation of execution (transition t_4 with probability $1-q$).

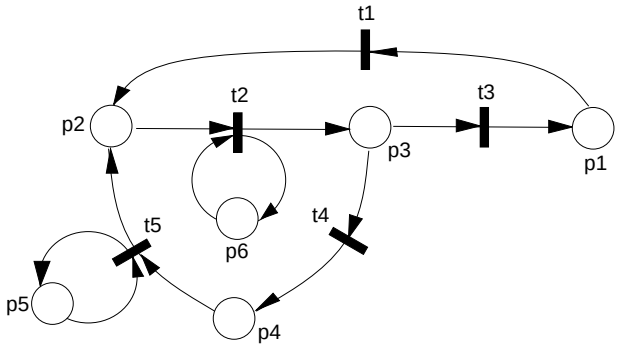

Fig.2. Petri net model of a multiprogramming system.

Let $m(p_6) = m(p_5) = 1$. Then the delay–equivalent replacement of t_2 and p_6 is a simple transition t_2' with throughput–dependent average firing time $f(t_2') = f(t_2)/(1-\theta(t_2)f(t_2))$. Similarly, t_5 and p_5 can be replaced by t_5' with $f(t_5') = f(t_5)/(1-\theta(t_5)f(t_5))$. Moreover, the visit ratios for t_2, t_4 and t_5 with respect to t_1 are $v(t_5) = v(t_4) = (1-q)/q$ and $v(t_2) = 1/q$, so the total average processing time of each job can be assigned to a single transition, for example t_3', replacing t_2', p_3, t_4, p_4 and t_5', as shown in Fig.3:

$$f(t_3') = v(t_2)f(t_2') + v(t_4)f(t_4) + v(t_5)f(t_5') + f(t_3)$$

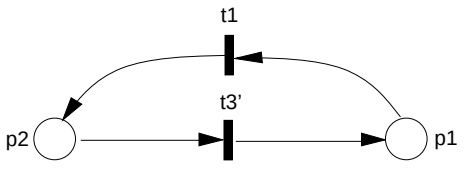

Fig.3. Simplified Petri net model of a multiprogramming system.

The throughput for the elementary net from Fig. 3 is simply $\theta = m_0(p_1)/(f(t_1)+f(t_3'))$. Taking into account the forced flow law, $\theta(t_2) = v(t_2)\theta(t_1)$ and $\theta(t_5) = v(t_5)\theta(t_1)$, so:

$$\theta = \theta(t_1) = m_0(p_1)/[f(t_1) + v(t_2)f(t_2)/(1 - \theta v(t_2)f(t_2))+ \\ v(t_4)f(t_4) + v(t_5)f(t_5)/(1 - \theta v(t_5)f(t_5)) + f(t_3)]$$

which defines the value of θ in terms of model parameters, i.e., the firing time function f, the initial marking function m_0 and the choice function c.

6. CONCLUDING REMARKS

It has been shown that, for a class of closed timed Petri net models, steady–state throughputs in the model can be determined on the basis of structural properties of the net (and – of course – model parameters defined by the m_0, c and f functions). The approach avoids the complexity (the "state explosion" problem) of the reachability analysis and it can be used to nets with different distribution functions associated with different transitions of the same nets; for example, some transitions may have deterministic firing times while other transitions may use stochastic firing times. For stochastic firing times, the actual distribution functions may be needed for delay–equivalent simplifications of net models.

Although the discussion in this paper was restricted to nets with simple arcs, it is believed that similar results can be derived for generalized nets, i.e., nets with multiple arcs (or arc weights).

The proposed approach can be used for analysis of unbounded nets, but the ergodicity condition is strictly required. As shown in [Zu92], the ergodicity of the solution can be 'verified' by checking additional requirements (e.g., utilization factors or performance bounds).

An attractive aspect of throughput analysis is the possibility of obtaining the solution in a symbolic form rather than as a numerical value. Different combinations of parameter values, sensitivity of the results with respect to different parameters or functional dependencies can be investigated very conveniently when symbolic solutions are available.

Acknowledgement

The Natural Sciences and Engineering Research Council of Canada partially supported this research through Operating Grant A8222.

References

[ACB84] M. Ajmone Marsan, G. Conte, G. Balbo, "A class of generalized stochastic Petri nets for the performance evaluation of multiprocessor systems"; ACM Trans. on Computer Systems, vol.2, no.2, pp.93–122, 1984.

[BG85] S.C. Bruell, S. Ghanta, "Throughput bounds for generalized stochastic Petri net models"; Proc. Int. Workshop on Timed Petri Nets, Torino, Italy, pp.250–261, 1985.

[Bu76] J.P. Buzen, "Fundamental operational laws of computer system performance"; Acta Informatica, vol.7, no.2, pp.167–182, 1976.

[C3S89] J. Campos, G. Chiola, J.M. Colom, M. Silva, "Tight polynomial bounds for steady–state performance of marked graphs"; Proc. Int. Workshop on Petri Nets and Performance Models, Kyoto, Japan, pp.200–209, 1989.

[CMT89] G. Ciardo, J. Muppala, K. Trivedi, "SPNP - stochastic Petri net package"; Proc. Int. Workshop on Petri Nets and Performance Models, Kyoto, Japan, pp.142–151, 1989;

[DB78] P.J. Denning, J.P. Buzen, "The operational analysis of queueing network models"; ACM Computing Surveys, vol.10, no.3, pp.225-261, 1978.

[Du85] J.B. Dugan, A. Bobbio, G. Ciardo, K. Trivedi, "The design of a unified package for the solution of stochastic Petri net models"; Proc. Int. Workshop on Timed Petri Nets, Torino, Italy, pp.6–13, 1985.

[FSZ83] D. Ferrari, G. Serazzi, A. Zeigner, "Measurement and tuning of computer systems"; Prentice–Hall 1983.

[Ho86] M.A. Holliday, "Deterministic time and analytical models of parallel architectures"; Ph.D. Thesis, Computer Science Department, University of Wisconsin - Madison, Technical Report #652, 1986.

[Ki90] P.J.B. King, "Computer and communication systems performance modelling"; Prentice–Hall 1990.

[Mu89] T. Murata, "Petri nets: properties, analysis and applications"; Proceedings of IEEE, vol.77, no.4, pp.541–580, 1989.

[Na80] S. Natkin, "Les réseaux de Petri stochastique"; Thèse de Docteur Ingenieur, CNAM, Paris, France, 1980.

[Ram74] C. Ramchandani, "Analysis of asynchronous concurrent systems by timed Petri nets"; Project MAC Technical Report MAC–TR–120, Massachusetts Institute of Technology, Cambridge MA, 1974.

[Re85] W. Reisig, "Petri nets – an introduction"; Springer Verlag 1985.

[Si77] J. Sifakis, "Use of Petri nets for performance evaluation"; in: "Measuring, modelling and evaluating computer systems", pp.75–93, North–Holland 1977.

[Zu86] W.M. Zuberek, "M–timed Petri nets, priorities, preemptions, and performance evaluation of systems"; in: "Advances in Petri Nets 1985" (Lecture Notes in Computer Science 222), G. Rozenberg (ed.), pp.478–498, Springer Verlag 1986.

[Zu88] W.M. Zuberek, "D–timed Petri nets and modelling of timeouts and protocols"; Transactions of the Society for Computer Simulation, vol.4, no.4, pp.331–357, 1988.

[Zu92] W.M. Zuberek, "Throughput analysis in timed Petri nets"; Proc. Midwest Symp. on Circuits and Systems, Washington DC, pp.1576–1580, 1992.