6-th Int. Conference on Distributed Computer Systems (DCS'86), Cambridge MA, May 19–23, 1986, pp.452–457.

MODIFIED D-TIMED PETRI NETS, TIMEOUTS, AND MODELLING OF COMMUNICATION PROTOCOLS

W.M. Zuberek

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

Abstract

Modified D-timed Petri nets are Petri nets with "special" arcs to interrupt firing transitions, and with deterministic firing times; these special arcs are called "interrupt" arcs. It is shown that the behaviour of simple modified bounded free-choice D-timed Petri nets can be represented by finite probabilistic state graphs, stationary probabilities of states can thus be obtained by standard techniques used for analysis of Markov chains. An immediate application of such a model is performance analysis of systems of interacting asynchronous processes, and in particular communication protocols. Places of Petri nets model queues of messages, transitions represent events in communication networks, interrupt arcs conveniently model timeouts, and probabilities associated with free-choice classes correspond to relative frequencies of random events. A simple protocol based on unnumbered messages and acknowledgements is used as an illustration of analysis.

1. INTRODUCTION

Petri nets [1,6,16] have been successfully used in modelling [3,8,9,15], validation [4,11] and analysis [7,9] of 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. A basic Petri net, however, is not complete enough for the study of systems performance since no assumption is made on the duration of system events. Timed Petri nets have been introduced by Ramchandani [17] by assigning firing times to the transitions of Petri nets. Sifakis [20] introduced another definition of a timed Petri net by assigning time to places of a net. Garg [10] used the same approach for specification of communication protocols. Merlin and Farber [13] discussed timed Petri nets where a time threshold and maximum delay were assigned to each transition of a net to allow the incorporation of timeouts into protocol models. Berthomieu and Menasche [5,12] used "state classes" to obtain finite representation of behaviour of nets defined by Merlin and Farber; such a description is sufficient for validation and verification studies, but requires further refinement for performance analysis. Razouk [18] and Razouk and Phelps [19] discussed an interesting class of timed Petri nets with enabling as well as firing times, and derived some performance expressions for a simple communication protocol; since the enabling times correspond to the time thresholds of Merlin and Farber, such nets can easily model timeout mechanisms, however, in some cases probabilities of conflicting firings cannot be directly associated with corresponding transitions which can distort the models.

The method described in this paper is a continuation of the approach originated by Ramchandani [17] and subsequently extended by inhibitor arcs and guarded places to model timeouts [21,22]. In extended Petri nets, however, firing transitions cannot be "interrupted:, and timeout signals must be rather "neutralized: than "canceled". In this paper, basic Petri nets are enhanced by "interrupt" (or "cancel") arcs in order to interrupt firing transitions as required in strict modelling of timeouts. Similarly as in [17,21,22], constant (or deterministic) firing times are assigned to transitions of a Petri net, and a state description is introduced which is similar to finite state homogeneous Markov chains. This directly provides many performance measures such as utilization of systems components, average waiting times and turnaround times or average throughput rates, and at the same time preserves the simplicity of model specification and offers automatic generation of the state space.

This paper is organized in 3 main sections. Section 2 contains definitions of basic concepts for modified free-choice bounded singular Petri nets. Modified D-timed Petri nets are introduced in Section 3. Application of modified D-timed Petri nets to modelling and analysis of protocols is discussed in Section 4.

2. MODIFIED PETRI NETS

A modified (basic) Petri net $\mathbf{N}$ is a quadruple $\mathbf{N} = (P, T, A, C)$ where:

P is a finite, nonempty set of places,

T is a finite, nonempty set of transitions,

A is a set of directed arcs which connect places with transitions and transitions with places such that for each transition there is at least one place connected with it

$$\forall t \in T \; \exists p \in P \; : \; (p, t) \in A,$$

C is a set of interrupt arcs which connect places with transitions, $C \subset P \times T$, and A and C are disjoint sets.

A place p is an input (or an output) place of a transition t iff there exists an arc (p, t) (or (t, p), respectively) in the set A. The sets of all input and output places of a transition t are denoted by $Inp(t)$ and $Out(t)$, respectively. Similarly, the sets of input and output transitions of a place p are denoted by $Inp(p)$ and $Out(p)$. Also, a place p is an interrupting place of a transition t iff $(p,t) \in C$. The set of all interrupting places of t is denoted by $Int(t)$, and the set of transitions connected by interrupt arcs with a place p is denoted by $Int(p)$, $Int(p) = \{t | p \in Int(t)\}$.

A modified net $\mathbf{N} = (P, T, A, C)$ is simple iff the interrupt sets of all transitions are disjoint

$$\forall t_i \in T \ \forall t_j \in T - \{t_i\} \ : \ Int(t_i) \cap Int(t_j) = \emptyset$$

where $\emptyset$ denotes the empty set. Only simple modified nets are discussed in this paper. Nonsimple nets require a very similar, but slightly extended description which takes into account conflicts of interrupt sets.

A place pp@ is shared iff it is an input place for more than one transition. A net is conflict-free iff it does not contain shared places. A shared place p is 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 Petri nets are considered in this paper. Some other classes of Petri nets can be described in a very similar way.

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

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

m_0 is an initial marking function which assigns a nonnegative integer number of so called tokens to each place of the net, $m_0 : P \to \{0, 1, ...\}$.

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

A transition t is enabled by a marking m iff every input place of this transition contains at least one token. The set of all transitions enabled by a marking m is denoted by $T(m)$.

Every transition enabled by a marking m can fire. Firing of an enabled transition t can be considered as composed of two steps; in the first step the firing is initiated by removing a single token from each of t's input places, while the second step terminates the firing by adding a token to each of t's output places. Firing of a transition determines a new marking in a net, a new set of enabled transitions, and so on. In modified Petri nets, an initiated firing of transition t is interrupted and canceled if the interrupt set $Int(t)$ is nonempty and all interrupting places of t contain at least one token. Cancellation of an initiated firing of t removes a single token from all interrupt places of t, but no tokens are added to t's output places.

A marking m_j is directly reachable from a marking m_i in a net $\mathbf{N}$ iff there exists a transition t enabled by the marking m_i, $t \in T(m_i)$, such that

$$\forall p \in P \ : \ m_j(p) = \begin{cases} m_{ij}(p) + 1, & \text{if } p \in Out(t) \wedge \\ & \quad min_{p_k \in Int(t)}(m_i(p_k)) = 0, \\ m_{ij}(p), & \text{otherwise,} \end{cases}$$

where

$$\forall p \in P \ : \ m_{ij}(p) = \begin{cases} m_i(p) - 1, & \text{if } p \in Inp(t)) \vee \\ & \quad (p \in Int(t) \wedge \text{mint}(t, m_i) > 0, \\ m_i(p), & \text{otherwise.} \end{cases}$$

where $\text{mint}((t, m) = \min_{p_k \in Int(t)}(m(p_k))$.

A marking m_j is reachable from a marking m_i in a net $\mathbf{N}$ if there exists a sequence of markings $(m_{i_0} m_{i_1} m_{i_2} ... m_{i_n})$ such that $m_{i_0} = m_i$, $m_{i_n} = m_j$, and each marking m_{i_k} is directly reachable from the marking $m_{i_{k-1}}$ for $k = 1, ..., n$.

A set $M(\mathbf{M})$ of reachable markings of a marked Petri net $\mathbf{M} = (\mathbf{N}, m_0)$ is the set of all markings which are reachable from the initial marking m_0 (including m_0).

A marked net $\mathbf{M}$ is bounded if there exists a positive integer k such that each marking in the set $M(\mathbf{M})$ assigns at most k tokens to each place of the net

$$\exists k > 0 \ \forall m \in M(\mathbf{M}) \ \forall p \in P \ : \ m(p) < k.$$

If a marked net $\mathbf{M}$ is bounded, its set of reachable markings $M(\mathbf{M})$ is finite. Only bounded nets are considered in this paper.

An enable function e of a marking m in a net $\mathbf{N}$ and its interrupting function d_e are any functions $e : T \to \{0, 1, ...\}$ and $d_e : T \to \{0, 1, ...\}$ such that

(1) there exists a sequence of transitions $u = (t_{i_1}, t_{i_2}, ..., t_{i_k})$, such that for $j = 1, ..., k$, $t_{i_j} \in T(m_{i_{j-1}})$ where $m_{i_0} = m$ and

$$\forall p \in P \ : \ m_{i_j}(p) = \begin{cases} m_{i_{j-1}}(p) - 1, & \text{if } p \in Inp(t_{i_j}) \\ & \quad \vee \ p \in Int(t_{i_j}) \wedge \\ & \quad \min_{p_k \in Int(t)}(m(p_k)) > 0, \\ m_{i_{j-1}}(p), & \text{otherwise,} \end{cases}$$

(2) the set of transitions enabled by m_{i_k}, $T(m_{i_k})$, is empty,

(3) for each $t \in T$, $e(t)$ is equal to the number of occurrences of t in the sequence u, while $d_e(t)$ is equal to the number of interrupted and canceled (or simply interrupted) firings of t in the sequence u;

i.e., an enable function e is any function which indicates (by nonzero values) all those transitions which can simultaneously initiate their firings (and some transitions may initiate their firings "several times"), while its interrupt function d_e indicates the numbers of interrupted firings. The set of all enable functions of a marking m is denoted by $E(m)$.

A marked net $\mathbf{M}$ is singular iff all enable functions of all reachable markings indicate at most a single firing of a transition

$$\forall m \in M(\mathbf{M}) \ \forall e \in E(m)) \ \forall t \in T \ : \ e(t) \leq 1.$$

To simplify the description of timed Petri nets, only singular free-choice bounded nets are considered in this paper. Nonsingular nets can be described by a rather straightforward extension of the formalism which follows.

3. MODIFIED D-TIMED PETRI NETS

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 [17,21,22] they are deterministic (or constant), i.e., there is a nonnegative (rational) number assigned to each transition of a net. In M-timed

Petri nets (or stochastic Petri nets [2,14]), the firing times are exponentially distributed random variables, and the corresponding firing rates are assigned to transitions of a net.

A modified D-timed free-choice Petri net $\mathbf{T}$ is a triple $\mathbf{T} = (\mathbf{M}, c, f)$ where:

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

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 place p of $\mathbf{N}$:

$$\textstyle\sum_{t \in Out(p)} c(t) = 1,$$

and for all remaining transitions $c(t) = 1$,

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

The behaviour of a modified D-timed Petri net can be represented by a sequence of states where each state describes the current marking as well as the firing transitions of a net. Each termination of a firing changes the current state of a net.

A state s of a modified D-timed Petri net $\mathbf{T}$ is a triple $s = (m, n, r)$ where:

m is a marking function, $m : P \to \{0, 1, ...\}$,

n is a firing-rank function which indicates the number of active firings (i.e., the number of firings which have been initiated but are not yet terminated) for each transition of the net, $n : T \to \{0, 1, ...\}$,

r is a remaining-firing-time function which assigns the remaining firing time to each independent firing (if any) of a transition, i.e., if the firing rank of a transition t is equal to k, $n(t) = k$, the remaining-firing-time function $r(t)$ is a vector of k nonnegative nondecreasing real numbers denoted by $r(t)[1]$, $r(t)[2]$, ..., $r(t)[k]$; r is a partial function and it is undefined for all those transitions t for which $n(t) = 0$.

An initial state s_i of a net $\mathbf{T}$ is a triple $s_i = (m_i, n_i, r_i)$ where n_i is determined as a difference between an enable function e from the set $E(m_0)$ and its interrupt function d_e

$$\forall tinT \; : \; n_i(t) = e(t) - d_e(t), e \in E(m_0),$$

the remaining-firing-time function is equal to the firing times $f(t)$ for all those transitions t for which $n_i(t) > 0$

$$\forall tinT \; : \; r_i(t) = \begin{cases} f(t), & \text{if } n_i(t) > 0, \\ \text{undefined}, & \text{otherwise}; \end{cases}$$

and the marking m_i is defined as

$$\forall p \in P \; : \; m_i(p) = m_0(p) - \sum_{tinOut(p)} e(t).$$

A modified free-choice D-timed net $\mathbf{T}$ may have several different initial states.

A state $s_j = (m_j, n_j, r_j)$ is directly e_k-reachable from the state $s_i = (m_i, n_i, r_i)$, iff:

(1) $e_k \in E(m_{ji})$,

(2) $\forall p \in P \; : \; m_{ji}(p) = m_i(p) - \sum_{t \in Int(p)} a_j(t)$,

(3) $\forall t \in T \; : \; a_j(t) = \min(n_i(t) - a_i(t), g_{ij}(t))$,

(4) $\forall t \in T \; : \; g_{ij}(t) = \min_{p \in Int(t)}(m_{ij}(p))$,

(5) $\forall p \in P \; : \; m_{ij}(p) = m_i(p) + \sum_{t \in Inp(p)} a_i(t)$,

(6) $\forall t \in T \; : \; a_i(t) = \begin{cases} 1, & \text{if } n_i(t) > 0 \wedge r_i(t)[1] = h_i, \\ 0, & \text{otherwise}; \end{cases}$

(7) $h_i = min_{t \in T}(r_i(t)[1])$,

(8) $\forall t \in T \; : \; a_{ij}(t) = a_i(t) + a_j(t)$,

(9) $\forall p \in P \; : \; m_j(p) = m_{ji}(p) - \sum_{t \in Out(p)} e_k(t)$,

(10) $\forall t \in T \; : \; n_j(t) = n_i(t) - a_{ij}(t) + e_k(t) - d_{e_k}(t)$,

(11) $\forall t \in T \; : \; r_j(t)[\ell] = \begin{cases} r_i(t)[\ell + a_{ij}(t)] - h_i, \\ \quad \text{if } 1 \le \ell \le n_i(t) - a_{ij}(t), \\ f(t), \quad \text{if } e_k(t) > d_{e_k}(t) \wedge \\ \quad \ell = n_j(t). \end{cases}$

The state s_j which is directly e_k-reachable from the state s_i is thus obtained by the termination of the "next" firings (i.e., those firings for which the remaining firing time is the smallest one; this time is denoted by $h(s_i)$ (6,7), updating the marking of a net (5), checking if updated interrupting sets discontinue any active firing and performing required modifications (2,3), and then initiating new firings (if any) which correspond to the enable function e_k from the set $E(m_{ji})$ (1, 9, 10 and 11).

Similarly as for marked nets, a state s_j is reachable from a state s_i if there is a sequence of directly reachable states from the state s_i to the state s_j. Also, a set $S(\mathbf{T})$ of reachable states is defined as the set of all states of a net $\mathbf{T}$ which are reachable from the initial states of the net $\mathbf{T}$ (including the initial states). For bounded nets the sets of reachable states are finite.

The state graph $\mathbf{G}$ of a D-timed Petri net $\mathbf{T}$ is a labeled directed graph $\mathbf{G}(\mathbf{T}) = (V, D, b)$ where:

V is a set of vertices which is equal to the set of reachable states of the net $\mathbf{T}$, $V = S(\mathbf{T})$,

D is a set of directed arcs, $D \subset V \times V$, such that (s_i, s_j) is in D iff s_j is directly reachable from s_i,

b is a function which assigns the probability of transition from s_i to s_j to each arc (s_i, s_j) in the set D, $b : D \to [0, 1]$, in such a way that if s_j is directly e_k-reachable from s_i, then

$$b(s_i, s_j) = \prod_{t \in T} c(t)^{e_k(t)}.$$

Example. For a simple Petri net shown in Fig.1 (as usual, places are represented by circles, transitions by squares, interrupt arcs by arcs with black dots instead of arrowheads, the initial marking by dots inside places, and the firing time function as well as the choice function are given as additional descriptions of transitions), the state graph is shown in Fig.2, and the derivation of the set $S(\mathbf{T}_1)$ of reachable states is given in Tab.1.

	m_i	n_i		m_{ij}	e_k		
s_i	1 2 3 4 5	1 2 3 4 5	h_i	1 2 3 4 5	1 2 3 4 5	s_j	$b(s_i, s_j)$
1	0 0 0 0 0	1 0 0 0 0	1.0	0 1 0 1 0	0 1 0 1 0	2	1.00
2	0 0 0 0 0	0 1 0 1 0	10.0	0 0 1 0 0	0 0 0 0 1	3	0.90
					0 0 1 0 0	4	0.10
3	0 0 0 0 0	0 0 0 1 1	5.0	1 0 0 0 1	1 0 0 0 0	1	1.00
4	0 0 0 0 0	0 0 1 1 0	0.0	0 0 0 0 0	0 0 0 0 0	5	1.00
5	0 0 0 0 0	0 0 0 1 0	10.0	1 0 0 0 0	1 0 0 0 0	1	1.00

Tab.1. The set of reachable states for $f(t_4) = 20$ and $m_0 = [1, 0, 0, 0, 0]$.

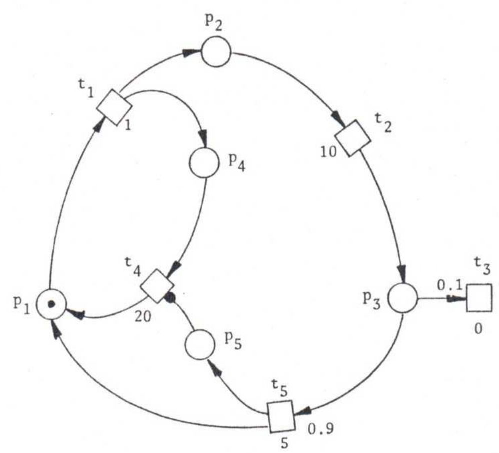

Fig.1. Modified free-choice D-timed Petri net $\mathbf{T}_1$.

It can be observed that the time spent in the state s_5, h_5, is equal to $f(t_4) - f(t_2)$. If the timeout $f(t_4)$ changes, the state graph has the same structure, and the only differences are in the remaining-firing-time components in some of the Petri net states. □

4. ANALYSIS OF PROTOCOLS

The Petri net shown in Fig.1 is a model of a very simple protocol in which messages are exchanged between a sender (place p_1) and a receiver (place p_3), and each received message is confirmed by an acknowledgement sent back to the sender (in the loop $p_1, t_1, p_2, t_2, p_3, t_5, p_1$). There is a nonzero probability that the system can lose (or distort) a message or an acknowledgement; the place p_3 is a free-choice place, and the transition t_3 models a message/acknowledgement "sink"; the probability associated with t_3, $c(t_3)$, represents thus the probability of losing a message or an acknowledgement (or shortly a "token") in the system. A "timeout" is used to recover from lost "tokens". It works in the following way. An event of "sending a message" is modelled by transition t_1. When it fires, single tokens are deposited in p_2 (a "message") and in p_4 (a "timeout"). A token in p_4 immediately starts a firing of the "timeoutU transition t_4. The firing time associated with t_4 is large enough to allow the transfer of a message and an acknowledgement. If there is no loss of tokens, i.e., if t_5 is selected for firing (according to its probability), the transition t_5 will finish its firing before t_4, and then a token in the place p_5 interrupts and cancels the timeout (i.e., the firing of t_4). If, however, a message or acknowledgement has been lost (i.e., if t_3 has been selected for firing instead of t_5), the timeout t_4 ends its firing without interruption, and regenerates the "lost" message in p_1.

The state graph representing the behaviour of this model contains 5 states, as shown in Fig.2 (and Tab.1). It contains one "decision" node (the state s_2) with two outgoing branches and branch probabilities q and $(1-q)$, corresponding to a lost and regenerated token, and a correct transfer, respectively (in the example $q = c(t_3) = 0.1$). It can be observed that the state graph can be reduced (by simply "aggregating" the paths connecting the "decision" node) to a single-node graph shown in Fig.3 in which one branch represents "a loss and recovery of a token" (and replaces the states s_1, s_2, s_4 and s_5), and the second branch represents correct transfers (i.e., it replaces the states s_1, s_2 and s_3). The times associated with the branches (i.e., the total times of "aggregated" paths) correspond to the "transition times", and are equal to 21 and 16, respectively.

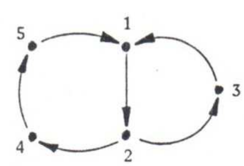

Fig.2. State graph for $\mathbf{T}_1$.

The (reduced) transition graph from Fig.3 can be transformed to an equivalent state graph [22] shown in Fig.4, with the holding times (i.e., times spent in the corresponding states) equal to $h_1 = 21$ and $h_2 = 16$ and transition probabilities q and $1 - q$.

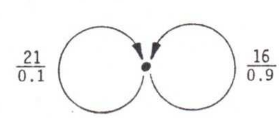

Fig.3. Reduced transition graph for $\mathbf{T}_1$.

The stationary probabilities x_1 and x_2 of the reduced states s_1 and s_2 (Fig.4) can be obtained as the solution of two simultaneous linear equations

$$x_1 = q * x_1 + q * x_2,$$
$$x_1 + x_2 = 1,$$

from which $x_1 = q$ and $x_2 = 1 - q$. In general case, the stationary probabilities $x(s)$ of the reduced states $s \in$

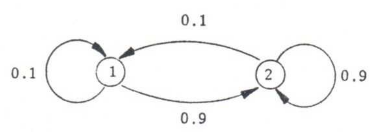

Fig.4. Reduced state graph for $\mathbf{T}_1$.

$S_r(\mathbf{T})$ are obtained by solving a system of simultaneous linear equations

$$\begin{cases} \sum_{(s_j,s_i)\in D_r} b(s_j,s_i) * x(s_j) = x(s_i); \quad i = 1,...,k-1 \\ \sum_{1\le i\le k} x(s_i) = 1 \end{cases}$$

where k is the number of states in the set $S_r(\mathbf{T})$.

Since the stationary probability of correct transfers is equal to $1 - q = 0.9$, and one correct transfer requires $f(t_1) + ft_2) + f(t_5) = 16$ time units, the throughput is equal to 0.9/16=0.0563 messages per time unit.

Fig.5 shows slightly modified net from Fig.1 where additional place p_6 is used for sequencing messages through the channel (represented by t_2). For initial marking m_0 representing 2 messages in the system ($m_0(p_1)+m_0(p_2) = 2$), the derivation of the states is shown in Tab.2, the original state graph in Fig.6, and the reduced state graph in Fig.7. The holding times for reduced states are equal $h_1 = h_2 = 11$, and the transition probabilities q and $(1-q)$, are as before. It can be observed that the reduced state graph is isomorphic to the reduced state graph from Fig.4. Consequently, the stationary probabilities are the same, i.e., $x_1 = q$ and $x_2 = 1 - q$.

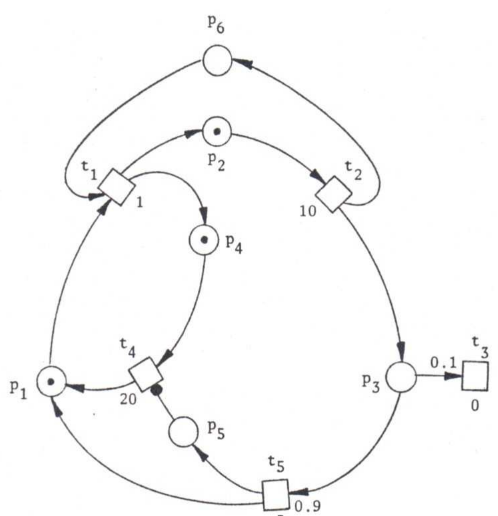

Fig.5. Modified free-choice D-timed Petri net $\mathbf{T}_2$.

Since a correct transfer of a single message requires $\max(f(t_1) + f(t_2), f(t_5)) = 11$ time units (the messages are sent sequentially through the channel, but their acknowledgements may be sent concurrently with subsequent messages), the throughput is equal to 0.9/11=0.0818 messages per time unit in this case. It can be observed, that the throughput cannot be further increased by increasing the number of messages in the system because the effective transfer time is equal to the sequential part of the transfer (which becomes the bottleneck in the system) and the messages will simply wait in the sender's queue (p_1). Further improvement can be obtained, however, by changing the length of messages [21].

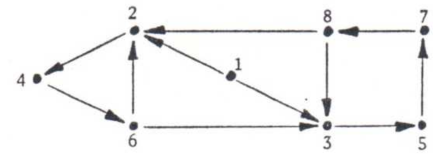

Fig.6. State graph for $\mathbf{T}_2$.

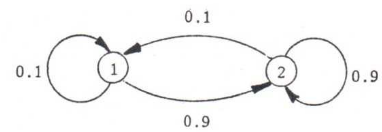

Fig.7. Reduced state graph for $\mathbf{T}_2$.

Many other results can be derived in a very similar way.

5. CONCLUDING REMARKS

It has been shown that the behaviour of a class of D-timed Petri nets with interrupt arcs can be represented by finite-state probabilistic graphs which are homogeneous finite-state discrete-time Markov chains. Escape arcs provide a mechanism for interrupting firing transitions, which can conveniently model timeouts of communication protocols. Protocol models are usually quite simple (in fact, they are significantly simpler than in previous models [21,22]), and their parameters correspond in a very natural way to components or activities of modelled systems (e.g., the numbers of messages, timeout signals, etc.). The results can be obtained in a rather general form, and repeated analyses with different sets of parameters can indicate the effects of performance changes corresponding to different queueing disciplines, or different priorities of messages, etc. The state space can easily be generated from model specifications, and since the states of the modelling net directly correspond to the "states" of the modelled system, a verification step is provided which is not available in simulation or analytical modelling.

The stationary probabilities of systems states are obtained by solving a system of linear equations. Therefore it may seem that the Petri net approach is feasible only for rather small systems. It should be observed, however, that the state graphs generated by timed Petri nets can be substantially reduced by relatively simple graph transformations [22], and the transformations can easily be integrated with state generation algorithms. Moreover, additional "structural" regularities can be used for further simplifications; for example, quite often [22] the (reduced) state graphs contain several identical subgraphs in which case complete subgraphs can be substituted by single vertices to obtain "preliminary" solutions, subsequently refined, as in hierarchical decomposition methods.

The class of timed Petri nets discussed in the paper is restricted in several ways (singular, free-choice, bounded nets), some of the restrictions, however, can be removed by rather straightforward extensions of the formalism.

s_i	m_i 1 2 3 4 5 6	n_i 1 2 3 4 5	h_i	m_{ij} 1 2 3 4 5 6	e_k 1 2 3 4 5	s_j	$b(s_i, s_j)$
1	1 0 0 0 0 0	0 1 0 1 0	10.0	1 0 1 0 0 1	1 0 0 0 1	2	0.90
					1 0 1 0 0	3	0.10
2	0 0 0 0 0 0	1 0 0 1 1	1.0	0 1 0 1 0 0	0 1 0 1 0	4	1.00
3	0 0 0 0 0 0	1 0 1 1 0	0.0	0 0 0 0 0 0	0 0 0 0 0	5	1.00
4	0 0 0 0 0 0	0 1 0 2 1	4.0	1 0 0 0 1 0	0 0 0 0 0	6	1.00
5	0 0 0 0 0 0	1 0 0 1 0	1.0	0 1 0 1 0 0	0 1 0 1 0	7	1.00
6	1 0 0 0 0 0	0 1 0 1 0	6.0	1 0 1 0 0 1	1 0 0 0 1	2	0.90
					1 0 1 0 0	3	0.10
7	0 0 0 0 0 0	0 1 0 2 0	9.0	1 0 0 0 0 0	0 0 0 0 0	8	1.00
8	1 0 0 0 0 0	0 1 0 1 0	1.0	1 0 1 0 0 1	1 0 0 0 1	2	0.90

Tab.2. The set of reachable states for $f(t_4) = 20$ and $m_0 = [1, 1, 0, 1, 0, 0]$.

REFERENCES

1. T. Agerwala, Putting Petri nets to work; IEEE Computer Magazine, vol.12, no.12, pp.85-94, 1979.
2. 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.
3. J-L. Baer, G. Gardarin, C. Girault, G. Roucairol, The two-step commitment protocol - modeling, specification and proof methodology; Proc. 5-th Int. Conf. on Software Engineering, San Diego CA, pp.363-373, 1981.
4. G. Berthelot, R. Terrat, Petri net theory for the correctness of protocols; IEEE Trans. on Communications, vol.30, no.12, pp.2497-2505, 1982.
5. B. Berthomieu, M. Menasche, An enumerative approach for analyzing time Petri nets; Information Processing 83, R.E.A. Mason (ed.), pp.41-45, IFIP 1983.
6. W. Brauer (ed.), **Net theory and applications**; Proc. of the Advanced Course on General Net Theory of Processes and Systems, Hamburg 1979; Lecture Notes in Computer Science 84, Springer Verlag 1980.
7. J.P. Courtiat, J.M. Ayache, B. Algayres, Petri nets are good for protocols; Computer Communication Review, vol.14, no.2, pp.66-74, 1984.
8. A. Danthine, Modeling and verification of end-to-end protocols; in: **New Advances in Distributed Computer Systems**, pp.125-158, Reidel Publ. Co. 1982.
9. M. Diaz, Modeling and analysis of communication and cooperation protocols using Petri net based models; Computer Networks, vol.6, no.6, pp.419-441, 1982.
10. K. Garg, An approach to performance specification of communication protocols using timed Petri nets; Proc. 4-th International Conf. on Distributed Computer Systems, San Francisco CA, 1984.
11. W. Juergensen, S.T. Vuong, Formal specification and validation of ISO transport protocol component using Petri nets; Computer Communication Review, vol.14, no.2, pp.75-82, 1984.
12. M. Menasche, B. Berthomieu, Time Petri nets for analyzing and verifying time dependent communication protocols; in: **Protocol Specification, Testing and Verification, III**, pp.161-172, North-Holland 1983.
13. P.M. Merlin, D.J. Farber, Recoverability of communication protocols - implications of a theoretical study; IEEE Trans. on Communications, vol.24, no.9, pp.1036-1049, 1976.
14. M.K. Molloy, Performance analysis using stochastic Petri nets; IEEE Trans. on Computers, vol.31, no.9, pp.913-917, 1982.
15. A. Pattavina, S. Trigila, Combined use of finite-state machines and Petri nets for modelling communicating processes; Electronic Letters, vol.20, no.22, pp.915-916, 1984.
16. J.L. Peterson, **Petri net theory and the modeling of systems**, Prentice-Hall 1981.
17. C. Ramchandani, Analysis of asynchronous concurrent systems by timed Petri netsU; Project MAC Technical Report MAC-TR-120, Massachusetts Institute of Technology, Cambridge MA, 1974.
18. R.R. Razouk, The derivation of performance expressions for communication protocols from timed Petri nets; Computer Communication Review, vol.14, no.2, pp.210-217, 1984.
19. R.R. Razouk, C.V. Phelps, Performance analysis using timed Petri nets; Proc. Int. Conf. on Parallel Processing, Columbus OH, pp.126-128, 1984.
20. J. Sifakis, Use of Petri nets for performance evaluation; in: **Measuring, modelling and evaluating computer systems**, pp.75-93, North-Holland 1977.
21. W.M. Zuberek, Extended D-timed Petri nets, timeouts, and analysis of communication protocols, Proc. ACM Annual Conference, Denver CO, 1985.
22. W.M. Zuberek, Inhibitor D-timed Petri nets and performance analysis of communication protocols; INFOR - Special Issue on Operation Research and Data Communications (to appear), 1986.