Journal of the Korean
OR/MS Society
Vol. 16, No.2, Dec., 1991

Dynamic Incidence Matrix Representation of Timed Petri Nets and Its Applications for Performance Analysis

J. G. Shon*, C. S. Hwang**, and D.-K. Baik**

Abstract

We propose a dynamic incidence matrix(DIM) for reflecting states and time conditions of a timed Petri net(TPN) explicitly.

Since a DIM consists of a conventional incidence matrix, two time-related vectors and two state-related vectors, we can get the advantages inherent in the conventional incidence matrix of describing a static structure of a system as well as another advantage of expressing time-dependent state transitions.

We introduce an algorithm providing the DIM with a state transition mechanism. Because the algorithm is, in fact, an algorithmic model for discrete event simulation of TPN models, we provide a theoretical basis of model transformation of a TPN model into a DEVS(Discrete EVent system Specification) model.

By executing the algorithm we can carry out perfomance analysis of computer communication protocols which are represented TPN models.

1. Introduction

The Petri net was developed as a tool for system modeling in 1962 by C.A. Petri[1]. Since then, many modifications and extensions have been studied in order to increase the modeling and analyzing power of Petri nets. Among these efforts, a timed Petri net(TPN) is useful to describe time conditions, such as time delays, which are necessary for performance evaluation and scheduling problems of dynamic systems[5].

There are two major approaches used for analyzing systems modeled by Petri nets to prove

* *Dept. of Computer Science, Korea Air & Corr. Univ.*

** *Dept. of Computer Science, Korea Univ.*

some properties including liveness, safeness, and boundedness: one is based on the reachability tree, and the other on matrix representation[2].

In the matrix-based approach, a Petri net is defined by the incidence matrix which is a composite matrix of a forward incidence matrix and a backward incidence matrix representing the input and output functions, respectively. Although this matrix representation promises to facilitate analysis of Petri net models, it has some serious problems[2], [4]. One of them is a limitation in that it can express only static structure of Petri nets and not state or time, which are essential parts of a time-dependent system modeling tool, such as a TPN.

In this paper, we propose the dynamic incidence matrix(DIM) for TPNs to incorporate the state and time component into the ordinary incidence matrix. By this DIM, a TPN can be allowed to describe states and time conditions explicitly.

In order to provide the DIM with state transition mechanism, we present an algorithm based on the DIM for finding the next state together with time-dependent information about the state transition. Since this algorithm is also used for the discrete event simulation of TPN models, we provide a theoretical basis by which a TPN model can be transformed into a DEVS model. And we show the dynamic expressibility of a DIM representation by applying the DIM scheme to performance analysis of two well-known communication protocols.

2. Classification and Limitations

Many various definitions of Petri nets have been developed by different researchers in different ways. Among those definitions the following definition is often used to explain the structure of Petri nets[6].

Definition 1

A *Petri net* is a quadruple $C=<P, T, F, B>$

where

P is a set of places, $P \neq \phi$

T is a set of transitions, $T \neq \phi$, $P \cap T = \phi$

$F : P \times T \rightarrow N$ is the forward incidence function

(N is the set of nonnegative integers.)

$B : P \times T \rightarrow N$ is the backward incidence function

Representation : A Petri net is described as a digraph, in which the nodes are the places and transitions represented by circles and bars, respectively. There is a directed arc from place p_i to transition t_j iff $F(p_i, t_j) = d_{ij} \neq 0$, where d_{ij} is called the weight of the arc. There is also a

directed arc from transition t_r to place p_k iff $B(p_k, t_r) = d_{kr} \neq 0$, where d_{kr} is called the weight of the arc.

Marking : A marking M of a Petri net is a function $M : P \rightarrow N$ such that $M(p_i)$ is the number of tokens in $p_i \in P$. A token is represented by a black dot within a circle. A Petri net with an initial marking M_0 is called an marked Petri net and denoted as $C = <P, T, F, B, M_0>$.

Input place set : $I(t_i) = \{p_i \in P \mid F(p_i, t_i) \neq 0\}$ for each $t_i \in T$.

Output place set: $O(t_i) = \{p_k \in P \mid B(p_k, t_i) \neq 0\}$ for each $t_i \in T$.

Enabled transition: $t_i \in T$ is enabled under a given marking M iff $M(p_i) \geq F(p_i, t_i)$ for each $p_i \in P$.

Next marking M' after firing of t_j : When M is a current marking and t_i is enabled under M, the next marking M' can be found by $M'(p_i) = M(p_i) - F(p_i, t_i) + B(p_i, t_i)$ for each $p_i \in P$. In this case, M' is said to be reachable from M by t_i.

2.1 Classification of Timed Petri Nets

Although a Petri net itself allows the modeling of concurrency, nondeterminism, and communication, it has been used to represent only the logic behavior of systems, with no timing or performance considerations. To make it easier to analyze the performance characteristics of such systems, several modifications of Petri nets have been needed so that time could be represented in Petri net models. Those various efforts have been reported in many literatures, and we intend to classify them here. We will use the term 'timed Petri net(TPN)' with the generic meaning of 'Petri net with some deterministic/stochastic timing.'

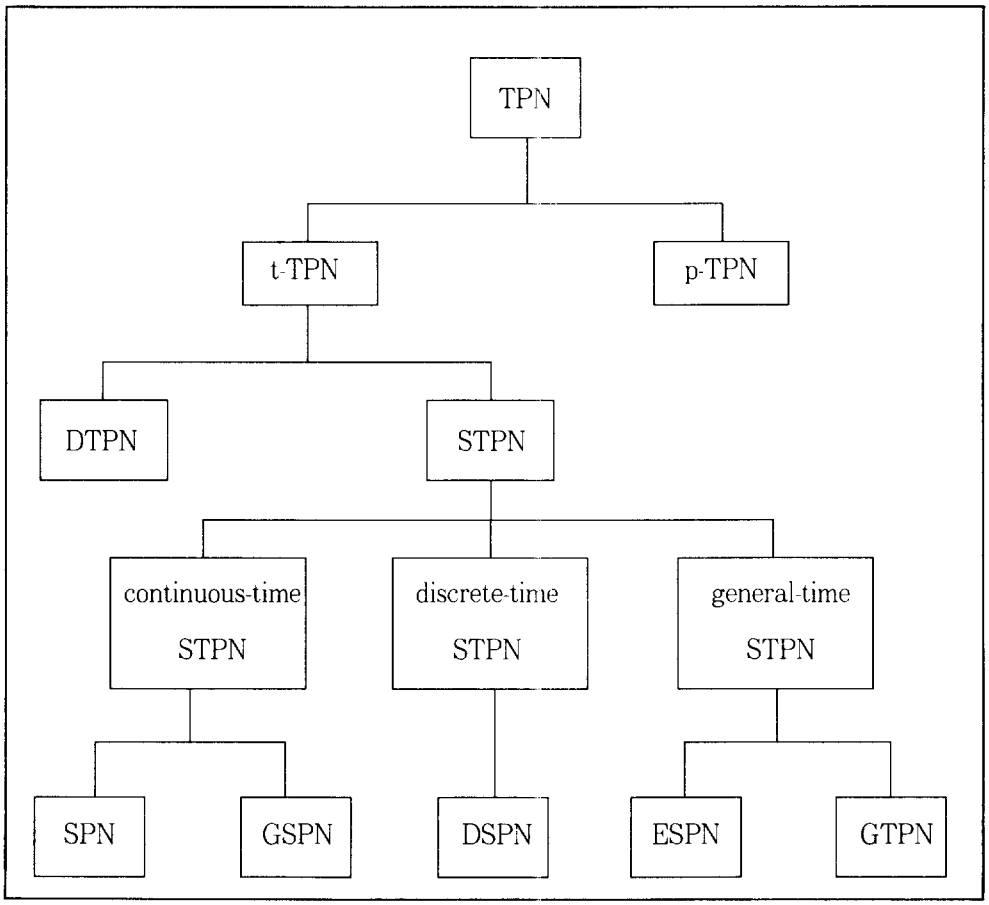
TPNs can be separated into t-timed Petri nets(t-TPNs)[7], [8], [9] and p-timed Petri nets(p-TPN)[10] depending on whether time is assigned to each transition or each place. t-TPNs in turn can be divided into deterministic TPNs(DTPNs; this category is called simply TPNs in many literatures[7], [11]) and stochastic TPNs(STPNs) according as time assigned is a fixed value(or interval) or a random variable, respectively.

Again we can classify STPNs into continuous-time, discrete-time, and general-time STPNs according to the kind of distribution of the random variable: exponential, geometric, and general, respectively. While the SPN(stochastic Peri net)[12] and the GSPN(generalized SPNs)[13] are included in continuous-time STPNs, the DSPN(discrete-time SPN)[14] in discrete-time STPNs. Finally, the ESPN(extended SPN)[15] and the GTPN(generalized TPN)[16] are in the general-time STPN category. We summarize this in Fig.1.

In this paper, we define a TPN in somewhat generic sense as follows; that is, the waiting time may be assumed to be (enable time)+(firing time).

Definition 2

A *timed Petri net* is a six-tuple $C=<P, T, F, B, M_0, \tau>$, where $\tau : T \rightarrow R_0$ is a function from T into the nonnegative reals R_0 such that $\tau(t_i)$ is the waiting time of a transition $t_i \in T$.



<Fig.1> Classification of Timed Petri Nets

2.2 Limitations of The Incidence Matrix

As an alternative definition of a Petri net, $C= <P, T, D^-, D^+>$ may be used, where $D^- = [d^-_{ij}]$ is called the forward incidence matrix with $d^-_{ij}=F(p_i, t_j)$ and $D^+=[d^+_{ij}]$ is called the backward incidence matrix with $d^+_{ij}=B(p_i, t_j)$. Because of some reasons described in [2] and the need for more memory space for matrices D^- and D^+, the definition $C=<P, T, D>$ may be preferrable to the definition $C=<P, T, D^-, D^+>$.

In definition $C=<P, T, D>$, matrix $D=[d_{ij}]$ is called the incidence matrix such that $D = D^+ - D^-$ and $d_{ij}= d^+_{ij} - d^-_{ij} = B(p_i, t_j) - F(p_i, t_j)$.

Although the incidence matrix is a well-arranged representation of a Petri net for analyzing behavioral and/or structural properties[3], it has certain limitations:

(1) It does not reflect self-loops[2], [4].

(2) It does not provide information about the order of transition firings[2].

(3) It is not sufficient for the reachability problem[2].

(4) It does not have complete representiveness[4].

(5) It can reflect neither the time element nor the marking element of the timed Petri net.

To eliminate limitation (5), we present another matrix representation for reflecting both time and marking in the following chapter.

3. Dynamic Incidence Matrix of A Timed Petri Net

A model is said to have *dynamic expressibility* when it can express a system by the mechanisms of

(1) state transitions which occur by change over time, and

(2) snapshot representation of a state at any time.

That is, if a model has dynamic expressibility, then the model can show when the current state is transferred, what the next state is, how long a system has been in the current state, and how long a system must remain in the current state. According to this view, a TPN model does not have dynamic expressibility because it cannot satisfy either of the two conditions.

In order to endow TPN models with dynamic expressibility, we will introduce the extended state transition function and the total state set of a TPN model after defining some terms for TPN $C=<P, T, F, B, M_0, \tau>$ as follows[18]:

reachability set $R(M_0)$: The set of all markings reachable from M_0 is called the reachability set of C and is denoted by $R(M_0)$.

enable set $E(M)$ *under* M : For each marking $M \in R(M_0)$, the set of all enabled transitions under M is called the enable set under M and is denoted by E(M).

extended time fuction τ_s : The extended time function τ_s of a TPN is defined as $\tau_s : R(M_0) \rightarrow R_0$ such that, for each $M \in R(M_0)$, $\tau_s(M) = \min\{\tau(t_i) \mid t_i \in E(M)\}$, where R_0 is the set of nonnegative reals.

3. 1 Dynamic Expressibility in a Global View

Since the state of a TPN model is represented by a marking, a state M can be transferred to the next state M' after firing of an enabled transition, say t_i, of which the waiting time value $\tau(t_i)$ is minimum among the t_i's in E(M). That is, there exists a state transition function $\delta : R(M_0) \rightarrow R(M_0)$ such that, for each $M \in R(M_0)$, $\delta(M)$ is the next state under M after a transition

t_i with $\tau(t_i)=\tau_s(M)$ fires. This state transition function δ satisfies conditon(1) of dynamic expressibility.

The other condition will be satisfied by considering the total state set of a TPN model.

Definiton 3

Given a TPN $C=<P, T, F, B M_0, \tau>$, the *total state set* of C is the set Q defined as follows:

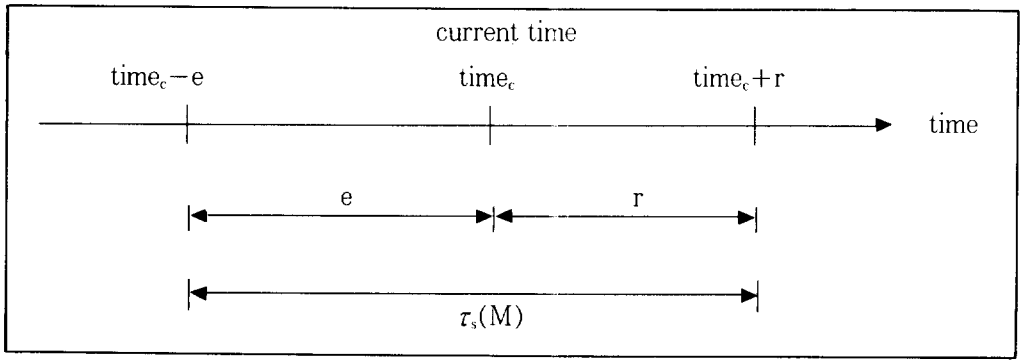
$Q=\{(M, e) \mid M\in R(M_0), e\in R_0, 0\le e\le\tau_s(M)\}$.

The pair (M, e) represents a sequential state M together with the elapse time e during which the system has been in that state M. As an alternative to Q for the total state set, we present Q' as follows:

$Q'=\{(M, r) \mid M\in R(M_0), r\in R_0, 0\le r\le\tau_s(M)\}$,

where $r=\tau_s(M)-$(current time). The pair (M, r) represents a sequential state M along with the remain time r during which the system will remain in that state M. So the pair(M, e) or (M, r) is called a *snapshot state.*

If $time_c$ is the current model time, then the system comes into the current state M at $time_c-e = time_c+r-\tau_s(M)$, and state M will be tarnsferred at $time_c+\tau_s(M)-e = time_c+r$ (Fig.2).



<Fig. 2> Elapse time e and remaining time r under the current state M

The pair (M, e) (or (M, r)) is considered a snapshot representation of a system at the current time. Hence the total state set Q (or Q') includes all possible snapshots so as to satisfy the condition(2) of dynamic expressibility.

From definition 3, our state trasition function δ can be extended to be defined on the total state set Q.

Definition 4

Given a TPN $C=<P, T, F, B, M_0, \tau>$ with the total state set Q, the *state transition function* δ of C is defined as

$\delta : Q\rightarrow Q$ such that $\delta(M, e)=(M', 0)$ when $e=\tau_s(M)$.

For Q', δ is similarly defined as

$\delta : Q'\rightarrow Q'$ such that $\delta(M, r)=(M', \tau_s(M'))$ when $r=0$.

Consequently, a TPN model C has dynamic expressibility if C adopts the state transition function δ (in Definition 4) and the total state set Q(in Definition 3).

3.2 Dynamic Expressibility in An Individual View

In a similar manner as described above, we can define elapse time function ε and remaining time function σ for each transition $t_i \in T$, individually.

Definition 5

Given a TPN $C=<P, T, F, B, M_0, \tau>$, *elapse time function* ε of C is defined as $\varepsilon : T \rightarrow R_0$ such that, for each $t_i \in T$, $\varepsilon(t_i)$ is the elapse time from the time at which t_i has begun to be enabled to the current time.

And the *remaining time function* σ of C is defined as $\sigma : T \rightarrow R_0$ such that, for each $t_i \in T$, $\sigma(t_i)$ is the remaining time from the current time to the time at which t_i will fire.

From this definition, each transition can be expressed individually by elapse time function ε or remaining time function σ. Therefore each transition may have dynamic expressibility in individual sense. If the number of set T, denoted by o(T), is n, then elapse time function ε may be represented by a n-vector $E=(\varepsilon(t_1), \varepsilon(t_2), \cdots, \varepsilon(t_n))$ called the elapse time vector, and remaining time function σ by a n-vector $F=(\sigma(t_1), \sigma(t_2), \cdots, \sigma(t_n))$ called the remaining time vector.

Given a TPN model $C=<P, T, F, B, M_0, \tau>$ with the total state set Q and the current state $M \in R(M_0)$, the following properties are satisfied:

(1) If the current snapshot state $(M, e) \in Q$,
then $e = \varepsilon(t_i)$ for all $t_i \in E(M)$, and

(2) If the current snapshot state $(M, r) \in Q'$,
then $r = \min\{\sigma(t_i) \mid t_i \in E(M)\}$.

These properties mean that a snapshot state of a system can be represented partly by function ε or σ; that is, dynamic expressibility in global sense can be achieved by dynamic expressibility in an individual sense if a state transition function is assumed.

However, we need a new representation of a TPN model which can reflect dynamic expressibility individually (and then globally), because no conventional representations of a TPN model have dynamic expressibility. A new representation ought to reflect the following:

(1) a state transition mechanism as described in Definition 4,

(2) a snapshot state by a certain mechanism, say function ε or σ in Definition 5, as well as

(3) static structure including P, T, F, O, M_0, and τ.

3.3 Definition of The Dynamic Incidence Matrix

We define the dynamic incidence matrix(DIM) of a TPN model so as to represent and analyze TPN models dynamically; the DIM shows us static structure as well as a snapshot state of a TPN model because the matrix includes function σ described in Definition 5. In addition to the DIM, we present Find_Next_State algorithm in section 4.3 as the state transition mechanism described in Definition 4.

A timed Petri net $C=<P, T, F, B, M_0, \tau>$ with a conventional incidence matrix $D=[d_{ij}]$ is denoted by $C=<P, T, D, M_0, \tau>$.

Definition 6

Let $C=<P, T, D, M_0, \tau>$ be a timed Petri net with initial marking M_0. When $o(P)=m-2$ and $o(T)=n-2$, the *dynamic incidence matrix* $A=[a_{ij}]$ of C is defined as an $m\times n$ matrix whose typical element

$$a_{ij}=\begin{cases} d_{ij} & (1\le i\le m-2,\ 1\le j\le n-2)\\ \tau(t_j) & (i=m-1,\ 1\le j\le n-2)\\ \sigma(t_j) & (i=m,\ 1\le j\le n-2)\\ M_0(p_i) & (1\le i\le m-2,\ j=n-1)\\ M(p_i) & (1\le i\le m-2,\ j=n)\\ \text{not defined} & (i=m-1, m, \text{ and } j=n-1, n). \end{cases}$$

We name four vectors and one matrix related in Definition 6 as follows:

waiting time vector W : the row vector $W=(a_{m-1,1}, a_{m-1,2}, \cdots, a_{m-1,n-2})=(\tau(t_1), \tau(t_2), \cdots, \tau(t_{n-2}))$ is called the waiting time vector.

remaining time vector F : the row vector $F=(a_{m,1}, a_{m,2}, \cdots, a_{m,n-2})=(\sigma(t_1), \sigma(t_2), \cdots, \sigma(t_{n-2}))$ is called the remaining time vector.

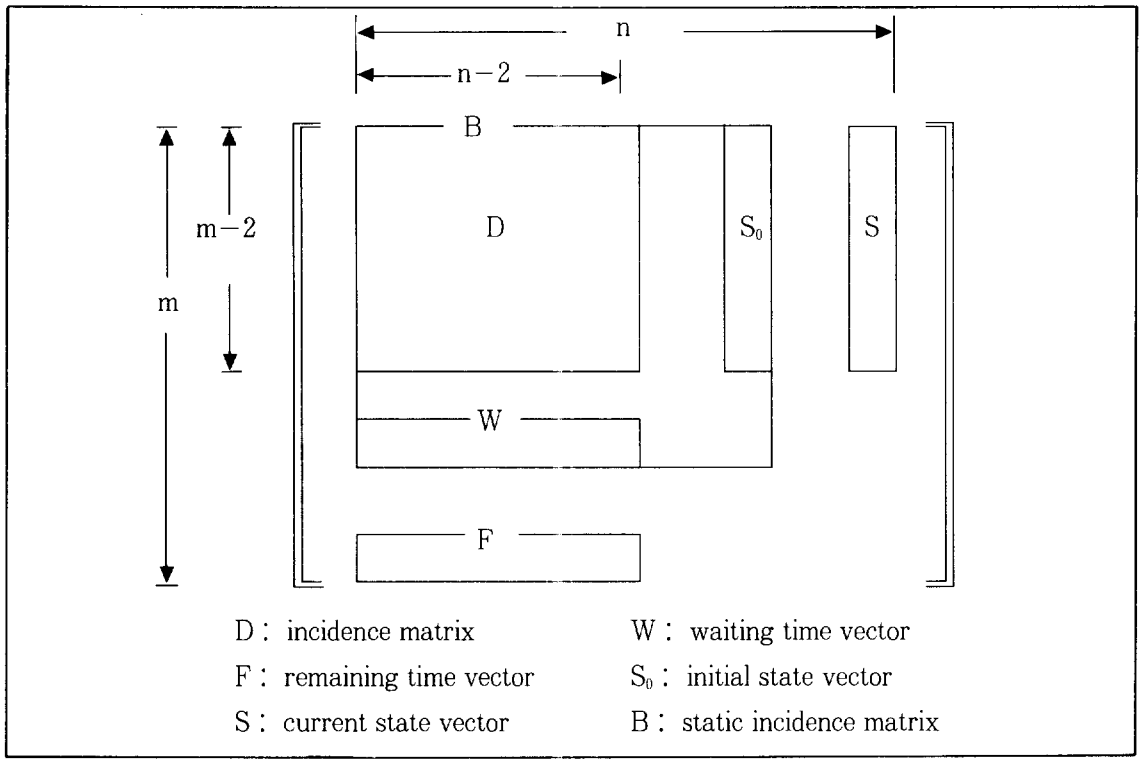
initial state vector S_0 : the column vecor $S_0=(a_{1,n-1}, a_{2,n-1}, \cdots, a_{m-2,n-1})=(M_0(p_1), M_0(p_2), \cdots, M_0(p_{m-2}))$ is called the inital state vector.

current state vector S : the column vector $S=(a_{1,n}, a_{2,n}, \cdots, a_{m-2,n})=(M(p_1), M(p_2), \cdots, M(p_{m-2}))$ is called the current state vector.

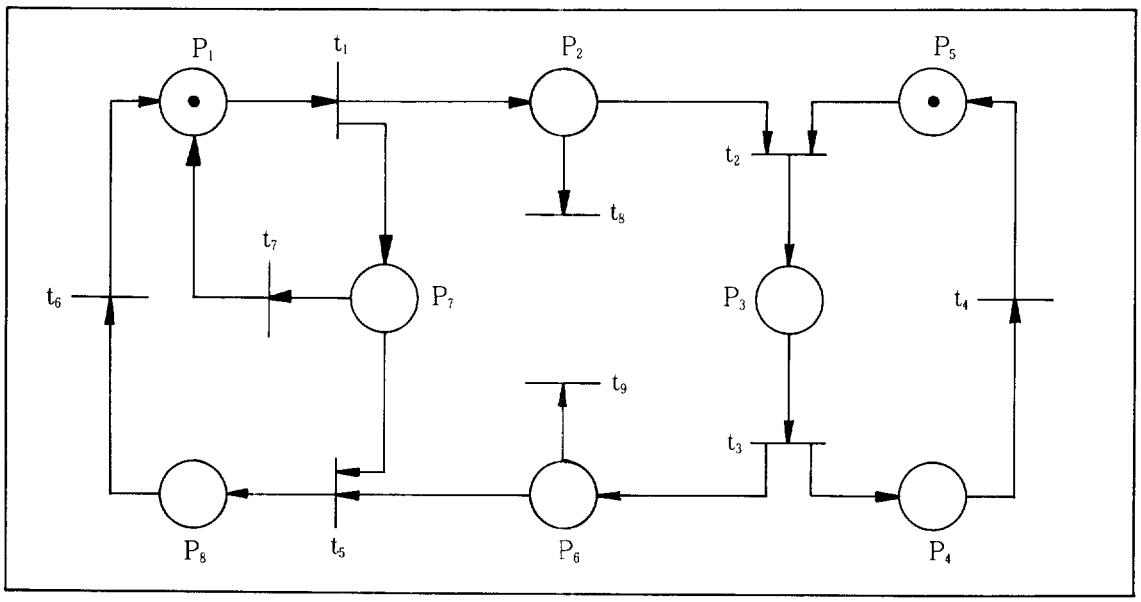
static incidence matrix(SIM) B : the $(m-1)\times(n-1)$ submatrix $B=[b_{ij}]$ of the DIM A, whose typical element $B_{ij}=a_{ij}(i\neq m,\ j\neq n)$, is called the static incidence matrix of TPN model C.

A DIM representation of a TPN model consists of the incidence matrix and the above vectors, and an outline of a DIM is depicted in Fig.3.

A simple protocol is modeled by the TPN model in Fig.4 where time conditions of each transitions are assumed as follows: $\tau(t_1)=20$, $\tau(t_2)=100$, $\tau(t_3)=15$, $\tau(t_4)=30$, $\tau(t_5)=100$, $\tau(t_6)=35$, $\tau(t_7)=200$, $\tau(t_8)=100$, and $\tau(t_8)=100$. The following Fig.5 illustrates the DIM of the TPN model in Fig.4 when the model is at initial state $M=M_0$.



<Fig.3> Outline of a dynamic incidence matrix(DIM)



<Fig.4> A TPN model of a simple protocol

	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	$M_0(p_i)$	$M(p_i)$
p_1	−1	0	0	0	0	1	1	0	0	1	1
p_2	1	−1	0	0	0	0	0	−1	0	0	0
p_3	0	1	−1	0	0	0	0	0	0	0	0
p_4	0	0	1	−1	0	0	0	0	0	0	0
p_5	0	−1	0	1	0	0	0	0	0	1	1
p_6	0	0	1	0	−1	0	0	0	−1	0	0
p_7	1	0	0	0	−1	0	−1	0	0	0	0
p_8	0	0	0	0	1	−1	0	0	0	0	0
p_9	0	0	0	0	1	−1	0	0	0	0	0
$\tau(t_i)$	20	100	15	30	100	35	200	100	100	−	−
$\sigma(t_i)$	20	100	15	30	100	35	200	100	100	−	−

<Fig.5> The DIM of the TPN model at initial state $M=M_0$ in Fig.4

4. Simulation of TPN Models with The DIM

The DEVS(Discrete EVent system Specification) is a formalism introduced by Zeigler[17] in order to provide a formal basis for specifying the models within discrete event simulation languages such as SIMSCRIPT, SIMULA, and GPSS.

A DEVS is a 6-tuple structure

$M=<X, Y, S, \delta, \lambda, t_a>$

where

X is a set of external input events,

Y is a set of output events,

S is a set of sequential states,

δ is a function, called the state transition function,

λ is a function, called the output function, and

t_a is a function, called the time advance function

with the following constraints:

(1) t_a is a mapping from S into R_0, the non-negative reals with infinity:

$t_a : S \rightarrow R_0$

(2) The total state set of the system specified by M is

$Q=\{(s, e) \mid s\in S, 0 \leq e \leq t_a(s)\}$

(3) The transition function δ consists of two parts:

a) The internal transition function $\delta_{int} : Q \rightarrow Q$

b) The external transition function $\delta_{ext} : Q \times X \rightarrow Q$

4.1 Model Transformation

In order to transform a marked TPN model into a DEVS simulation oriented model, we need to have some preparation: extended time function τ_s (in section 3.1) and definitions of a closed TPN and an open TPN.

We define a closed TPN as a TPN without external input/output places, and an open TPN as a TPN with external input/output places.

Theorem A marked TPN model can be represented by a DEVS model[18].

Proof: The basis of the proof is to show that a marked TPN model, say C, is defined with the DEVS structure, $<X, Y, S, \delta, \lambda, T_a>$.

Since a state in C is represented by a marking, the reachability set $R(M_0)$ plays the role of a sequential state set S under DEVS formalism. While a state transition can occur when an event arrives in a DEVS model, a state(marking) can be transferred to the next state(marking) when the enable transition with a minimum value of τ_s fires in a TPN model.

Hence we consider a set $G=\{(M, e) \mid M \in R(M_0), 0 \leq e \leq \tau_s(M)\}$ as the total state set Q in DEVS formalism. In a TPN model, a state $M \in R(M_0)$ is transferred to the next state M' after firing of the transition with a minimum τ_s. Formalizing this, we get the following: there exists a state transition function δ_{int} such that $\delta_{int} : G \rightarrow G$ by $\delta_{int}(M, e)=(M', 0)$ when $e= \tau_s(M)$. And this δ_{int} plays the part of the internal state transition function δ_{int} in the DEVS framework.

Now we divide this proof into two cases according to whether a TPN model is of closed form or open form.

1) in case of a closed TPN model C

Since C does not communicate with the outside world, there is no necessity to include X, Y, and λ as components of the model structure, hence $X = Y = \phi$ and $\lambda = \phi$ (meaning that λ need not be defined). Because $X=\phi$, the external transition function δ_{ext} is no longer needed, so that $\delta_{ext}=\phi$. Therefore a closed TPN C is represented by the DEVS $M_c = < \phi, \phi, R(M_0), \delta_{int}, \phi, \tau_s>$.

2) in case of an open TPN model C

Since C does communicate with the outside world, X, Y, and λ should be defined. X is the set of those elements which represent the deposit of one or more tokens in the input place(s) of C. Y is the set of those elements which represent a property expressed by tokens in the output place(s) of C. λ is a function of $R(M_0)$ into Y. Those three components of the DEVS model

structure may not be specifically defined, but in a specific application they would be specified by a modeler. δ_{ext} is a function of $Q \times X$ into Q defined as $\delta_{ext}(M, e, x)=(M', 0)$ in which an external event x, meaning that a token is to be deposited in an input place of C, causes M to be the next state M' when $e < \tau_s(M)$. Hence an open TPN C is represented by the DEVS $M_c = < X, Y, R(M_0), \delta, \lambda, \tau_s >$. Q.E.D.

4.2 An Algorithm for Finding The Next State

In this section we present an algorithm model(Fig.6) for finding the next state of a TPN model $C=<P, T, F, B, M_0, \tau>$ under current state M. This algorithm is based on DIM A of TPN model C.

For a DEVS based simulation, we include in the algorithm a step 3.1 which can be extended as follows:

3.1.1 Set $time_l := time_c$

3.1.2 Set $time_c := time_c + a_{mk}$

where, $time_l$ and $time_c$ stand for the time of the last firing and the current time, respectively.

After the resulting state vector S' is updated, we can find the next state iteratively. By this algorithm, we can tell what the next state is as well as when it will be transferred. As seen, the DIM representation of a TPN model is simple and easy to understand, especially for modelers who are familiar with the incidence matrix of a Petri net[18].

Algorithm Find_Next_State
Input: DIM A of TPN model C with $m-2$ places and $n-2$ transitions
Output: changed state vector S' of A
1. Find the enable set E under the current state S
   1.1 Find a set $K=\{i \mid a_{in}>0, 1\le i\le m-2\}$
   1.2 For each $i \in K$, find a set
   $L_i = \{j \mid a_{ij}<0, a_{in} \ge -a_{ij}, 1\le j\le n-2\}$
   1.3 For each $j\in L_i$, find a set
   $L_{ij} = \{k \mid a_{kj} < 0, k \ne i, 1\le k\le m-2\}$
   1.4 Decide whether t_i is enabled or not:
   if $L_{ij}=\phi$
   then put t_i in E
   else
   for each $k \in L_{ij}$
   if $a_{kn}\ge -a_{kj}$
   then put t_i in E
   endif
   endfor
   endif
2. Find the immediate transition t_k out of E
   2.1 Find a set $I=\{k \mid a_{mk} = \min\{a_{mj} \mid t_i \in E\}\}$
   2.2 If the set I has only one element
   then t_k is the immediate transition
   else
   call a tie-breaking procedure
   endif
3. Find the next state vecotr S'
   3.1 Set current time advanced as a_{mk} unit
   3.2 For each $i(1\le i\le m-2)$
   $a_{in} := a_{in} + a_{ik}$
4. End Find_Next_State

<Fig.6> Algorithmic Model

5. Applications : Performance Analysis

5.1 Performance Analysis of A Simple Protocol

The simple protocol in Fig.4 is sutdied to analyze some performance measures by executing its TPN model. The TPN model reflects (1) message loss, (2) retransmission, (3) timeout. The transition t_8 expresses message loss and t_9 acknowledgement loss. t_7 plays the role of retransmission when timeout occurs.

We collect the simulated data during the execution of the TPN model represented by the DIM. When the channel has failure rate x % of the message loss, we evaluate performance measures including mean successful throughput and mean response time.

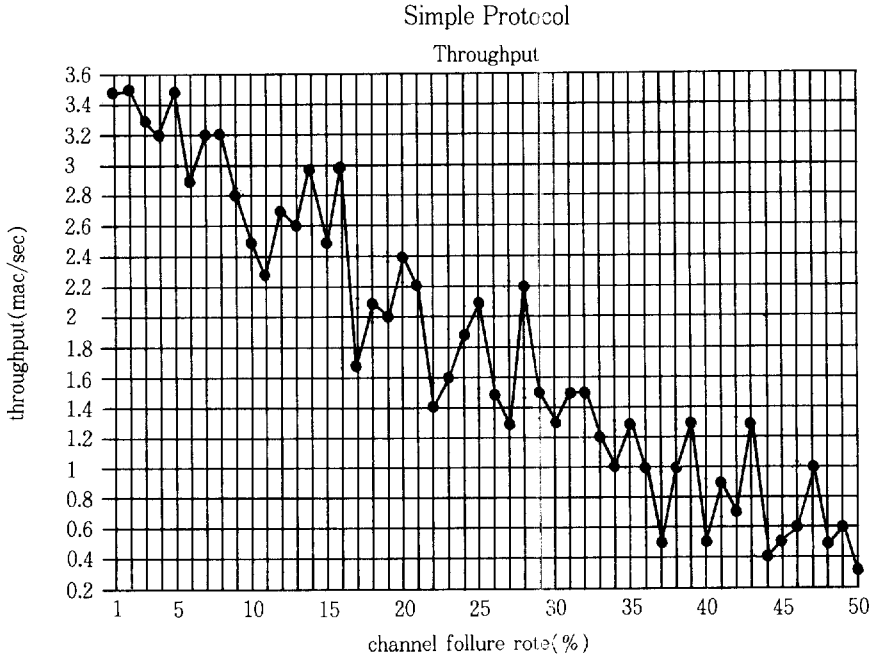
For the sake of simplicity in performance analysis, we assume that all processed (transmitted, lost, retransmitted, or created) data have the same length, say one message, or one packet. When the simulation time is t_{sim}, we specify performance measures as follows:

(1) mean throughput(T_m):

$$T_m = W/t_{sim} = N_m t_{sim} / t_{sim} = N_m (\text{message/sec})$$

where W is the total amount of successful data during t_{sim}, and N_m is the mean number of successful transmissions.

About the TPN moldel of the simple protocol, we can calculate T_m as the total firing number of t_6(creating a new message after receiving the ACK message) during t_{sim} divided by t_{sim}. The relation between the mean throughput and channel failure rate is depicted in Fig.7



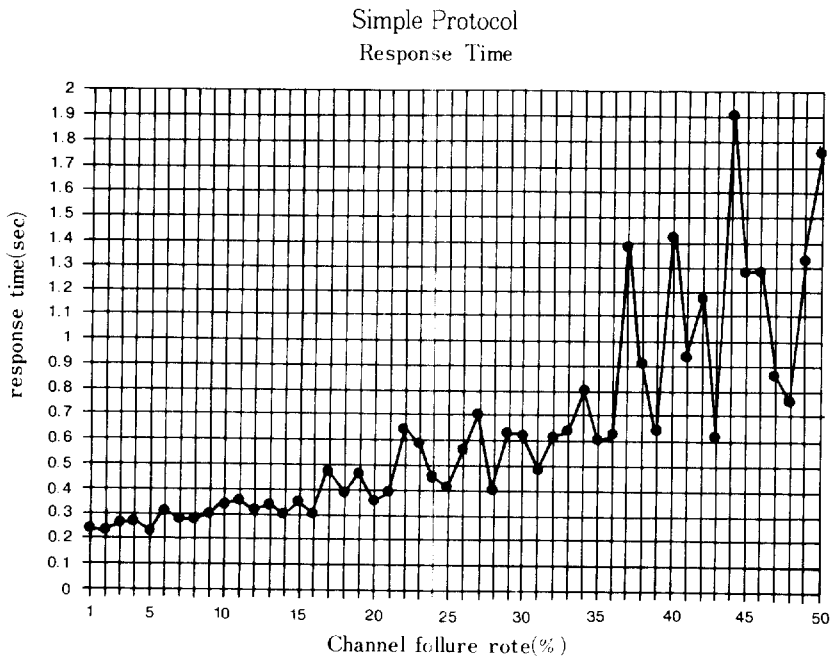
<Fig.7> Mean throughput of the simple protocol in Fig.4

(2) mean response time(R_m):

R_m is defined as the mean duration time between a sending state (M=(1,5)) and a receiving state (M=(5,8)). We can evaluate R_m from the following formula:

$$R_m = \left(\sum_{i=1}^{n} r_i \right)/n$$

where each r_i is the i-th actual response time, and n is the number of responses during t_{sim}. Fig.8 illustrates the relation between the mean response time and the channel failure rate.



<Fig.8> Mean response time of the simple protocol in Fig.4

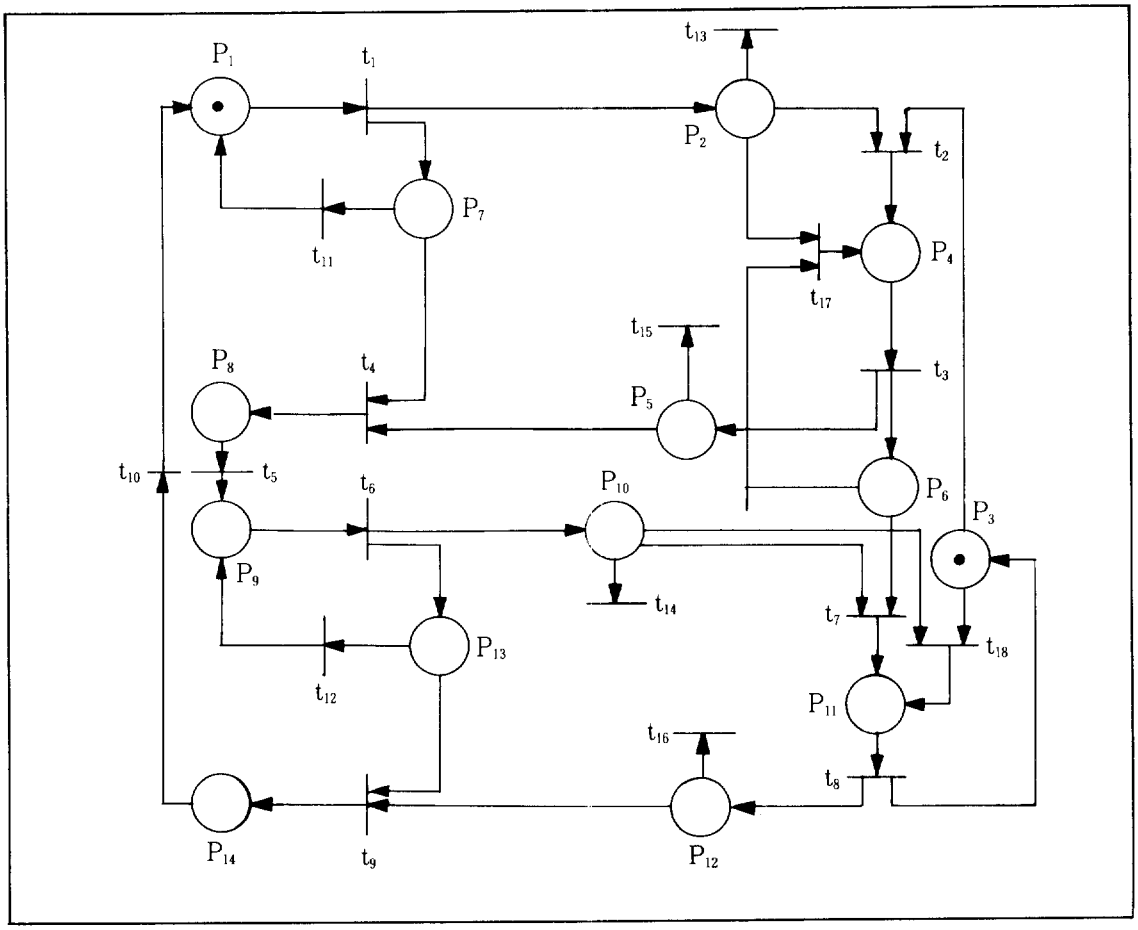
5.2 Performance Analysis of The Alternating Bit Protocol

The timed Petri net Fig.9 represents the alternating bit protocol. This protocol involves two parties, a sender and a receriver, connected through a link. The sender sends messages to the receiver, and the receiver responses with acknowledgements. Each message carries a control bit (0 or 1) whose value alternates for consecutive messages, and each acknowledgement carries a bit equal to the one carried by the message it acknowledges. This TPN is similar to the two instances of the TPN shown in Fig.4. The differences are t_{17} and t_{18} representing the event of receiving messages out of order.

About this alternating protocol, we can calculate T_m similarly as in the pervious section. T_m is described as the total firing number of t_5 and t_{11}(creating a new message whatever the value of its control bit) during t_{sim} divided by t_{sim}. The relation between the mean throughput and the channel failure rate is depicted in Fig.11.

We can also define mean response time R_m as the mean of two duration times: one is between

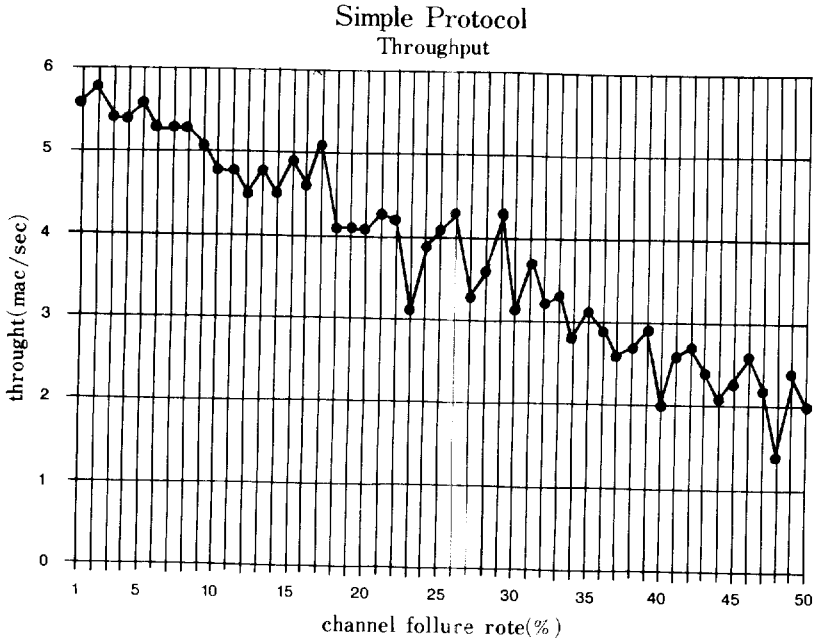
a sending state M=(1,3) and a receiving state M=(6,8) and the other is between a sending state M=(6,9) and a receiving state M=(3,14). Fig.12 illustrates the relation between the mean response time and the channel failure rate.



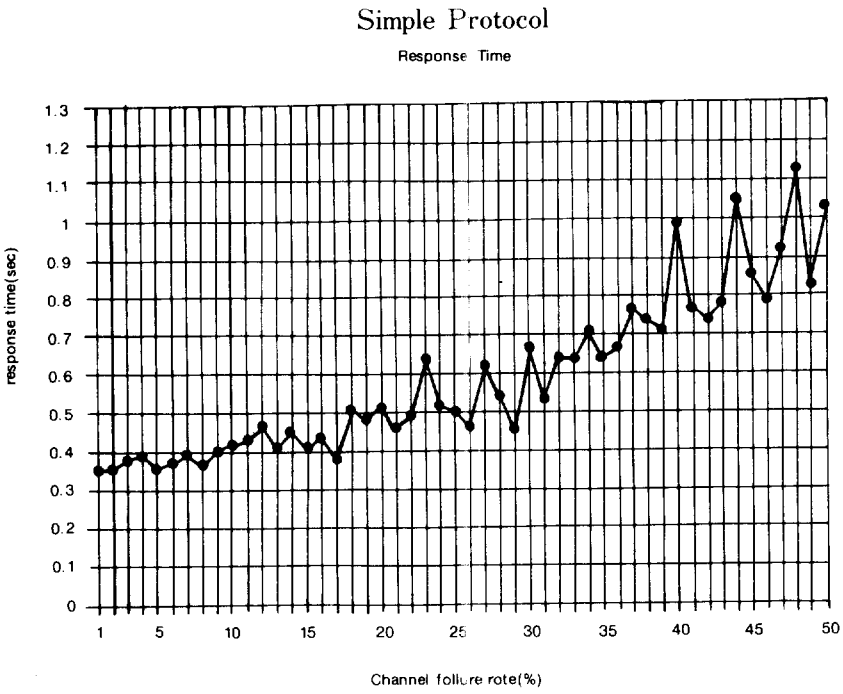
<Fig.9> A TPN model of the alternating bit protocol

	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}	t_{11}	t_{12}	t_{13}	t_{14}	t_{15}	t_{16}	t_{17}	t_{18}	$M_0(p_i)$	$M(p_i)$
p_1	-1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0
p_2	1	-1	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	-1	0	0	0
p_3	0	-1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	-1	1	0
p_4	0	1	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
p_5	0	0	1	-1	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0
p_6	0	0	1	0	0	0	-1	0	0	0	0	0	0	0	0	0	-1	0	0	0
p_7	1	0	0	-1	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0
p_8	0	0	0	1	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
p_9	0	0	0	0	1	-1	0	0	0	0	0	1	0	0	0	0	0	0	0	0
p_{10}	0	0	0	0	0	1	-1	0	0	0	0	0	0	-1	0	0	0	-1	0	0
p_{11}	0	0	0	0	0	0	1	-1	0	0	0	0	0	0	0	0	0	1	0	1
p_{12}	0	0	0	0	0	0	0	1	-1	0	0	0	0	0	0	-1	0	0	0	0
p_{13}	0	0	0	0	0	1	0	0	-1	0	0	-1	0	0	0	0	0	0	0	1
p_{14}	0	0	0	0	0	0	0	0	1	-1	0	0	0	0	0	0	0	0	0	0
$\tau(t_i)$	20	50	15	50	35	20	50	15	50	10	100	50	50	0	50	50	50	50	-	-
$\sigma(t_i)$	20	50	15	50	35	20	50	15	50	10	100	50	50	0	50	50	50	50	-	-

<Fig.10> The DIM of the TPN model in Fig.8
when the model is at certain intermediate state M



<Fig.11> Mean throughput of the alternating bit protocol



<Fig.12> Mean response time of the alternating bit protocol

6. Conclusions

In this paper, we propose the dynamic incidence matrix(DIM) for TPN models to describe explicitly states and time conditions which are not expressed by the conventional incidence matrix. The DIM has dynamic expressibility since this matrix representation has remaining time function σ and Find_Next_State algorithm as mechanisms for snapshot states and state transitions, respectively.

While the SIM(static incidence matrix) part of a DIM can express the static properties of a TPN, the other part, which consists of the current state vector S and the remaining time vector F, can express the dynamic properties of a TPN model. Thus it is possible to express the snapshot state of a system at any time, and describe not only what the next state is but also when the next state is transferred.

According to the theoretical basis of model transformation of a TPN model into a DEVS moldel, Find_Next_State algorithm is also used for discrete event simulation of TPN models. By simulating TPN models, we carry out performance analysis of a simple protocol and the alternating bit protocol especially about mean throughput and mean response time.

REFERENCES

[1] C. Petri, "Kommunikation mit automatem," *Ph.D. dissertation*, University of Bonn, Bonn, West Germany, 1962.

[2] J. L. Peterson, *Petri Net Theory and Modeling of Systems*, Englewood Cliffs, NJ: Prentice-Hall, 1981.

[3] T. Murata, "Petri Nets:Properties, Analysis and Applications," *Proc. of the IEEE*, Vol. 77, No. 4, pp. 541-580, Apr. 1989.

[4] S. K. Das *et al.*, "Reflexive Incidence Matrix(RIM) Representation of Petri Nets," *IEEE Trans. Soft. Eng.*, Vol. SE-13, No. 6, pp.643-653, Jun. 1987.

[5] J. Carlier *et al.*, "Modelling scheduling Problems with timed Petri nets," in *Lecture Notes in Computer Science 188(ed. G. Rozenberg)*, pp.62-82, Springer-Berlag, 1985.

[6] M. Hack, "Decidability Questions for Petri Nets," *Ph.D. dissertation*, MIT, Cambridge, Massachusetts, Dec. 1975.

[7] C. Ramchandani, "Analysis of asynchronous concurrent system by timed Petri nets," MIT, Cambridge, *Tech. Rep.* 120, Feb. 1974.

[8] P. M. Merlin and D. J. Faber, "Recoverability of communication protocols—Implications of a theoretical study," *IEEE Trans. Commun.* pp.1036-1043, Sept. 1976.

[9] R. R. Razouk and C. V. Phelps, "Perfomance analysis using timed Petri nets," in *Proc. 1984 Int. Conf. Parallel Processing*, pp.126-129, Aug. 1984.

[10] J. Sifakis, "Use of Petri nets for performance evaluation," *Acta Cybernet.*, Vol. 4, No. 2, pp.185-202, 1978.

[11] C. V. Ramamoorthy and G. S. Ho, "Performance evaluation of asynchronous concurrent processes using Petri nets," *IEEE Trans. Soft. Eng.*, Vol. SE-6, No. 5, pp.440-449, Sept. 1980.

[12] M. K. Molloy, "Perfomance Analysis using stochastic Petri nets," *IEEE Trans. Comput.*, Vol. C-31, pp.913-917, Sept. 1982.

[13] M. A. Marsan, G. Balbo, and G. Conte, "A class of generalized stochastic Petri nets," *ACM Trans. Comput. Syst.*, Vol. 2, pp.93-122, May 1984.

[14] M. K. Molloy, "Discrete time stochastic Petri nets," *IEEE Trans. Soft. Eng.*, Vol. SE-11, pp.417-423, Apr. 1985.

[15] J. B. Dugan, K. S. Trivedi, R. M. Geist, and V. F. Nicola, "Extended stochastic Petri nets: Applications and analysis," in *Performance 84*, Paris, France, pp.507-519, Dec. 1984.

[16] M. A. Holiday and M. K. Vernon, "A Generalized Timed Petri Net Model for Performance Analysis," *IEEE Trans. Soft. Eng.*, Vol. SE-13, No. 12, pp. 1297-1310, Dec. 1987.

[17] B. P. Zeigler, *Theory of Modelling and Simulation*, John Wiley & Sons, Inc., 1976.

[18] J. G. Shon, D.-K. Baik, and C. S. Hwang, "An Efficient Simulation of Communication Protocols Modelled by Timed Petri Nets," *Proc. of 5th Int'l Jonit Workshop on Computer Communication*, pp.31-40, Kyongju, Korea, Jul. 1990.