1986

Stochastic High-Level Petri Nets and Applications

Chuang Lin

Dana Cristian Marinescu

Report Number:

86-613

STOCHASTIC HIGH-LEVEL PETRI NETS AND APPLICATIONS

Chuang Lin
Dan Cristian Marinescu

CSD-TR-613
July 1986

Stochastic High-Level Petri Nets and Applications

*Chuang Lin**

Dan Cristian Marinescu

Computer Sciences Department
Purdue University
West Lafayette, IN 47907, USA

ABSTRACT

A new class of stochastic Petri nets is proposed in this paper. The Stochastic High Level Petri nets [SHLPNs] are High Level Petri nets augmented with exponentially distributed firing times. The main advantage of modeling homogeneous systems using SHLPNs is that the resulting models are simpler, more intuitive and have a smaller number of states as our examples show.

1. Introduction

Ramamoorthy [Ram80], Sifakis [Sif77] and others have used Petri nets for the performance evaluation of concurrent systems. The introduction of Stochastic Petri nets [SPNs] proposed by Molloy [Mol82] has triggered a number of interesting developments in this area, for example, the study of Discrete Time Stochastic Petri Nets [Mol85] and of Generalized Stochastic Petri Nets [Mar84 Bal86]. The Generalized Stochastic Petri Nets [GSPNs] have been introduced by Marsan and his coworkers and used for the performance analysis of multiprocessor systems.

Molloy has proved that there is an isomorphism between k-bounded Petri nets with exponentially distributed transition rates and finite Markov processes. Two stochastic systems are isomorphic if:

a- there are one-to-one mappings between the state spaces of the two systems, and between the set of state transitions of the two systems,

b- 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 importance of this result is that the methodology used to find the steady state probabilities of Markov chain can be used for the SPN system in which each marking corresponds to a Markov state.

Though the Stochastic Petri nets do not provide more modeling power than the Markov processes, they can be used as a convenient description of the system being modeled. Since the size of the state space of a Stochastic Petri net is equal to the size of the Markov process space, the complexity of solving an SPN model is the same as in the case of

* On leave from State Planning Committee, Beijing, China

the model based upon the Markov process.

To reduce the complexity of solving an SPN model, Marsan has proposed to define two types of transitions, timed and immediate and to separate the state space of an SPN into two subsets, one containing *vanishing states*, which enable immediate transitions and *tangible states*, which enable timed transitions. The time spent by the system in vanishing states is zero. The existence of vanishing states increases the computational effort by enlarging the size of the transition matrix. The authors claim that they can define a reduced embedded Markov chain over the tangible states only.

This paper takes a different approach, it defines Stochastic High Level Petri nets based upon High Level Petri nets augmented with exponentially distributed transition rates. High Level Petri nets are extensions of regular Petri nets. An example of High Level Petri nets are the Predicate Transition nets [Gen81] in which individual tokens are allowed, and predicates may be associated with some or all transitions. As a general rule, High Level Petri nets lead to simpler models, with a more readable graph than the corresponding regular Petri nets. The size of the state space of Stochastic High Level Petri Nets models can be further simplified due to the introduction of the *compound marking* technique described in the following sections.

In fact, we have concentrated our attention upon *homogeneous systems* which, when modeled using Stochastic High Level Petri nets have subsets of *equivalent* states. Such states can be grouped together in such a way that the Stochastic High Level Petri net model of the system with compound markings contains only one *compound state* for each group of *individual states* in the original Stochastic High Level Petri net model. In this case a *equivalence relation* exists among the Stochastic High Level Petri net model with compound markings and the original Stochastic High Level Petri net model. Both provide the same information about the system being modeled but the SHLPN with the compound marking is a scaled down version of the original SHLPN, it has a lower number of states.

Since the SPN are isomorphic with continuous time Markov chains the Stochastic High Level Petri nets are also isomorphic with continuous time Markov chains. The compound marking concept represents a grouping of markings in the Petri net domain and it corresponds to state grouping in the Markov domain. The necessary and sufficient condition for state grouping is then examined in the context of the Markov domain and its results are then carried back to the Petri net domain in order to prove the correctness of the compound marking concept. In fact we prove that Stochastic High Level Petri nets with compound markings are homomorphic with continuous time Markov chains with a grouping operation.

Three systems are then modeled using the technique described above. We start with an example based upon the classical problem of philosophers who share dinnerware, then we discuss an example related to modeling of communication protocols and conclude with an example, similar to the one treated by Marsan, the analysis of the performance of a multiprocessor system.

The main advantages of introducing SHLPNs with compound markings are: the model of a system has a considerably lower number of states than same model constructed using SPNs or SHLPNs, the graph associated with the model is simpler, easier to read, often it is invariant to system size and the analysis methods for Petri net models can still be applied to SHLPN models. Since the SHLPN model with compound markings is isomorphic with a Markov chain its steady-state probabilities can be determined using techniques well developed by stochastic analysis.

2. Informal Introduction to Stochastic High-Level Petri Nets

In order to introduce Stochastic High Level Petri nets the definition of Stochastic Petri nets will be reviewed. Then Stochastic High Level Petri nets will be introduced informally and an example illustrating the use and the advantages of SHLPNs will be given.

The SPNs are obtained by associating with each transition in a Petri net an average, possibly marking dependent, transition rate for the exponentially distributed firing time. A formal definition of SPN is thus the following:

SPN = (P, T, A, M, λ)

1. P is the set of places
2. T is the set of transitions
3. $P \cap T = \emptyset$, $P \cup T \neq \emptyset$
4. A is the set of input and output arcs; $A \subsetneq (P \times T) \cup (T \times P)$
5. M is the initial marking
6. λ is the set of transition rates

A marking of a Petri net, or of a Stochastic Petri net, is a distribution of tokens on its places. A marking may be viewed as a mapping from the set of places P to the natural numbers N. We can associate with each marking a state of the system and in the followings the terms *state* and *marking* will be used with essentially the same meaning.

The SPNs are isomorphic to continuous time Markov chains due to the memoryless property of the exponential distribution of firing times. The SPN markings correspond to the states of the corresponding Markov chain so that the SPN model allows the calculation of the steady state probabilities of each state.

In SPN analysis as in Markov analysis, ergodic (irreducible) systems are of special interest. For ergodic SPN systems, the steady state probability of the system being in any state always exist and are independent of the initial state. If the firing rates do not depend upon time, a stationary (homogeneous) Markov chain is obtained. In particular, k bounded SPNs are isomorphic to finite Markov chains. In this paper, we consider only ergodic, stationary and k bounded SPNs (or SHLPN) and Markov chains.

As an example consider a system, consisting of five philosophers who alternately think and eat. There are only five forks on a circular table and there is a fork between two philosophers. Each philosopher needs to use the two forks adjacent to him when he eats. Obviously two neighbors cannot eat at the same time. If we suppose that all philosophers have exponentially distributed eating and thinking times with averages $1/\lambda_1$ and $1/\lambda_2$, the philosopher system can be described by the Stochastic Petri net, SPN, shown in Figure 1. The model has fifteen places and ten transitions, all indexed on variable i, $i \in [1,5]$ in the following description:

T_i is the "thinking" place. If T_i holds a token, the i-th philosopher is thinking.

E_i is the "eating" place. If E_i holds a token, the i-th philosopher is eating.

F_i is the "free fork" place. If F_i holds a token, the i-th fork is free.

G_i is the "getting forks" transition. This transition is enabled when the thinking philosopher can get two free forks adjacent to him. The transition firing time is associated with $1/\lambda_1$.

R_i is the "releasing forks" transition. A philosopher releases the forks and return to the thinking stage, after the eating time exponentially distributed with average $1/\lambda_2$.

The Stochastic Petri net model of the philosopher system has a state space size of 11 and its states (markings) are presented in Table 1. The state transition diagram of the corresponding Markov chain is shown in Figure 2. Using the methods mentioned above, the steady state probabilities that the system is in state i, p_i , can be obtained:

$$p_i = \begin{cases} \dfrac{\lambda_2^2}{5\lambda_1(\lambda_1+\lambda_2)+\lambda_2^2} & i = 1 \\ \dfrac{\lambda_1\lambda_2}{5\lambda_1(\lambda_1+\lambda_2)+\lambda_2^2} & i = 2,3,4,5,6 \\ \dfrac{\lambda_1^2}{5\lambda_1(\lambda_1+\lambda_2)+\lambda_2^2} & i = 7,8,9,10,11 \end{cases}$$

Our objective is to model the same system but using a representation which leads to a model with a smaller number of states. High Level Petri nets provide a more compact representation of complex systems hence they represent a natural choice for an alternative representation if the time concept can be embedded into them. Different types of High Level Petri nets [HLPNs] have been proposed, for example Predicate Transition nets[Gen79 Gen81], Coloured Petri nets [Jen81], Relation nets [Rei83], but all of them are conceptually similar. Moreover, the model of a system constructed using one type of HLPN can be informally translated into any other type of HLPN [Jen83 Rei83]. The Predicate Transition nets for example, support the intuition and the modeling elegance since they allow variables which represent arc labels or token attributes to appear in the conditions associated with the firing of a transition. The Stochastic High Level Petri nets are extensions of High Level Petri nets in which each transition has an exponentially distributed firing time associated with it.

The following notation is used throughout this paper: $\oplus$ stands for addition modulo of the sequence number. [x,y] denotes the sequence set of from x to y. The appropriate circle bracket is denoted if the lower or upper bound is excluded from the sequence set. $|\{\ \}|$ denotes the cardinality of a set. The relations between the element and the set, $\in$ and $\notin$ are often used in the predicates.

The SHLPNs will be introduced by means of an example which illustrates the fact that an SHLPN model is a scaled down version of an SPN model, it has a smaller number of places, transitions and states but it still carries the same amount of information about the system being modelled as the original SPN model. Figure 3 presents the SHLPN model of the same philosopher system described in Figure 1 using a SPN. In the SHLPN model each place and each transition stands for a set of places or transitions in the SPN model. The number of places is reduced from 15 to 3, the place T stands for the set { T_i }, E for { E_i } and F for { F_i }, for $i \in [1,5]$. The number of transitions is reduced from ten to two, the transition G stands for the set { G_i } and R for the set { R_i } with $i \in [1,5]$.

The three places contain two types of tokens, as shown in Figure 3. The arcs are labelled by token variables. A token has a number of attributes, the first attribute being its type and the second attribute being its *identity, id.* The tokens residing in the place E have a third attribute and a fourth attribute, the ids of the forks which are currently used by the philosopher. The transition G is associated with the predicate which specifies the correct relation between a philosopher and the two forks used by him. This means that only when the two forks adjacent to him are free, a philosopher can eat.

In a SHLPN model, the transition rate associated with every transition is related to the markings which enable that particular transition. To simplify the design of the model only the transition rate of the individual markings is shown in the graph, instead of the transition rate of the corresponding compound markings. For example in Figure 3, the transition rates are written as: λ_1 for G and λ_2 for transition R.

The problem of determining the compound markings and the transition rates among them is discussed in the followings. The markings (states) of the philosopher system based upon HLPN are given in Table 2. The initial population of different places is: five tokens in T, five tokens of different type in F and no token in E. When one or more philosophers are eating, E contains one or more tokens.

In many systems a number of different processes have an similar structure and behavior. To simplify the system model it is desirable to treat similar processes in a uniform and succinct way. In the HLPN models, a token type may be associated with the process type and the number of tokens with the same type attribute may be associated with the the number of identical processes [Lin86]. A process description, a subnet, can specify the behavior of a type of processes and defines variables unique to each process of that type. Each process is a particular and independent instance of a execution of a process description (subnet).

The tokens present in SHLPNs have several attributes, type, identity, environment, etc. In order to introduce compound markings such attributes are represented by variables with a domain covering the set of values of the attribute.

In the philosopher system we can use a variable i to replace the *identity* attribute of the philosopher and the *environment variable* attribute representing fork tokens to each philosopher process. The domain set of the variable i is [1,5], i.e., the <p,i> represents anyone among <p,1>, <p,2>, <p,3>, <p,4>, <p,5> and the <f,i> represents anyone among <f,1>, <f,2>, <f,3>, <f,4>, <f,5>. The compound marking (state) table of the philosopher system is shown in Table 3. The size of the state space is considerably reduced compared with the previous case. To order to use efficiently this method, we should notice the equivalence of the variable markings. For example, we will recognize the following variable markings : The m_1 and m_2 are equivalent, i.e., they both represent the same set of individual markings. The m_3 is neither equivalent with m_1 nor equivalent with m_2. The equivalent marking concept was also proposed for reachability trees of the HLPN in the reference [Hub85]. Our compound marking concept is convenient for computing the reachability marking set and for understanding the behavior of the system

	T	E	F
m_1	<p,i>, <p,i⊕2>, <p,i⊕3>, <p,i⊕4>	<p,i⊕1,i⊕1,i⊕2>	<f,i>, <f,i⊕3>, <f,i⊕4>
m_2	<p,i⊕1>, <p,i⊕2>, <p,i⊕3>, <p,i⊕4>	<p,i,i,i⊕1>	<f,i⊕2>, <f,i⊕3>, <f,i⊕4>
m_3	<p,i⊕1>, <p,i⊕2>, <p,i⊕3>, <p,i⊕4>	<p,i,i⊕1,i⊕2>	<f,i>, <f,i⊕3>, <f,i⊕4>

modeled.

The markings of Table 3 correspond to the Markov chain states shown in Figure 4 and are obtained by grouping the states from Figure 2. The transition rates between the grouped states (compound markings) can be obtained after determining the number of the possible transition from one individual marking in each compound marking to any individual marking in another compound marking. In our case there is one possible transition from only one individual marking of the compound marking s_1 to each individual marking of the compound marking s_2 with the same rate. So, the transition rate from s_1 to s_2 is $5\lambda_1$. Using a similar argument we can obtain the transition rate from s_2 to s_3 as $2\lambda_1$, from s_3 to s_2 as $2\lambda_2$ and from s_2 to s_1 as λ_2. The steady state probabilities of each compound marking (grouped Markov state) can be obtained as:

$$p_1 = \frac{\lambda_2^2}{5\lambda_1(\lambda_1+\lambda_2)+\lambda_2^2}$$

$$p_2 = \frac{5\lambda_1\lambda_2}{5\lambda_1(\lambda_1+\lambda_2)+\lambda_2^2}$$

$$p_3 = \frac{5\lambda_1^2}{5\lambda_1(\lambda_1+\lambda_2)+\lambda_2^2}$$

The probability of every individual marking of a compound marking is the same and can be easily obtained since the number of individual markings in each compound marking is known.

3. Stochastic High-Level Petri Nets

The previous example has presented the advantage of using High Level Petri nets augmented with exponentially distributed firing times called in this paper Stochastic High Level Petri nets, for modeling of concurrent systems.

This section is organized as follows: first we present a formal definition of Stochastic High Level Petri nets and discuss the isomorphism of SLPNS with continuous time Markov chains. Then the concept of *compound marking* is discussed. The problem of state grouping (lumping) for a continuous time Markov chain is analyzed and it is proved that a SHLPN with compound markings induces a correct state grouping in the Markov domain. As a result there is a Markov chain associated with the compound markings of a SHLPN and the steady state probabilities of each compound marking can be found using the Markov techniques.

3.1. Formal Definition of SHLPNs

A High Level Petri net consists of the following elements:

1. A directed graph (P, T, A) where

   P is the set of places

   T is the set of transitions

   A is the set of arcs; $A \subseteq (P \times T) \cup (T \times P)$

2. A structure set Σ consisting of some types of individual tokens (u_i) together with some operations (op_i) and relations (r_i), i.e., $\Sigma = (u_1,...,u_n ; op_1,...,op_m ; r_1,...,r_k)$

3. A labelling of arcs with a formal sum of n-attributes of token variables (including the zero-attributes indicating a no-argument token)

4. An inscription on some transitions being a logical formula constructed from the operation and relations of the structure Σ and variables occurring at the surrounding arcs.

5. A marking of the places of P with n-attributes of individual tokens.

6. A natural number K which assigns to the places an upper bound for the number of copies of the same token

7. *Firing rule*. Each element of T represents a class of possible changes of markings. Such a change, also called *transition firing*, consists of removing tokens from a subset of places and adding them to other subsets according to the expressions labelling the arcs. A transition is enabled whenever, given an assignment of individual tokens to the variables which satisfies the predicate associated with the transition, all input places carry enough copies of proper tokens, and the capacity K of all output places will not be exceeded by adding the respective copies of tokens. The *state space of the system* consists of the set of all markings connected to the initial marking through such occurrences of firing.

Definition 3.1: A continuous time Stochastic High Level Petri Net, is a HLPN extended with the set of average, markings related, transition rates, $\lambda = \{ \lambda_1 , \lambda_2 , ... \lambda_t \}$.

A one to one correspondence between each marking of a Stochastic High Level Petri net and a state of a Markov chain representing the same system can be established. Following the arguments presented in [Mol82] it can be stated that:

Theorem 3.1: Any finite place, finite transition, Stochastic High Level Petri net is isomorphic to a one-dimensional, continuous time, finite Markov chain.

As in the case of SPNs this isomorphism is based upon the marking sequence and not upon the transition sequence. Any number of transitions between the same two markings are indistinguishable. Clearly, since a marking in a SHLPN depends only upon the current marking and not upon the past history of the system, the SHLPN can be represented by a continuous Markov chain.

3.2. The Compound Marking of a SHLPN

The compound marking concept is based on the fact that a number of entities processed by the system exhibit an identical behavior and they have a single subnet in the SHLPN model. The only distinction between such entities is the *identity* attribute of the token carried by the entity. If, in addition, the system consists of identical processing elements distinguished only by the identity attribute of the corresponding tokens, it is possible to lump together a number of markings in order to obtain a more compact SHLPN model of the system. Clearly, the model can be used to determine the global system performance in case of homogeneous systems when individual elements are indistinguishable.

Definition 3.2: A compound marking of a SHLPN is the result of partitioning an individual SHLPN marking into a number of disjoint sets such that:

- The individual markings in a given compound marking have the same distribution of tokens in places, except for the identity attribute of tokens of the same type,
- All individual markings in the same compound marking have the same transition rates to all other compound markings.

These ideas can be clearly followed in the previous example. If we now consider the example presented in section 5, (Figure 9 shows the SHLPN model of a multiprocessor system) we see that the compound marking indicated as state 2 in Table 6, corresponds to 15 individual markings as shown in Table 8.

Let us now consider a few properties of the compound marking:

-P1. A compound marking enables all transitions enabled by all individual markings lumped into it,

-P2. If the individual reachability set of a SHLPN is finite, its compound reachability set is finite.

-P3. If the initial individual marking is reachable with a nonzero probability from any individual marking in the individual reachability set, the SHLPN initial compound marking is reachable with a nonzero probability from any compound marking in the compound reachability set.

We denote by p_{ij} the probability of a transition from the compound marking i to the compound marking j and by $p_{i_n j_k}$ as the probability of a transition from the individual marking i_n to the individual marking j_k, where $i_n \in i$ and $j_k \in j$. The relation between the transition probability of compound markings and the transition probability of individual markings are:

$$p_{ij} = \sum_k p_{i_n j_k} \qquad \textit{for any } \; i_n \in i \tag{3.1}$$

The relation between the transition rate of compound markings and the transition rate of individual markings are:

$$q_j(t) = \frac{d\left[\sum_i p_{ji}\right]}{dt} = \frac{\sum_i d\left[\sum_k p_{j_n i_k}\right]}{dt} \tag{3.2}$$

$$q_{ij}(t) = \frac{dp_{ij}}{dt} = \frac{\sum_k d(p_{i_n j_k})}{dt} \tag{3.3}$$

If the system can reach a steady-state then the sojourn time in each compound marking is an exponentially distributed random variable with average:

$$\left[\sum_{i \in H} (q_{jk})_i\right]^{-1} \tag{3.4}$$

where H is the set of transitions that are enabled by the compound marking and q_{jk} is the transition rate associated with the transition i firing on the current current compound marking j.

3.3. State Grouping (Lumping) in the Markov Domain

Since there is an isomorphism between Stochastic High Level Petri nets and Markov chains, any compound markings of a SHLPNs corresponds to grouping of states in the Markov domain.

In order to be useful a compound marking must induce a correct grouping in the Markov domain corresponding to the original SHLPN. Otherwise the methodology known from Markov analysis, used to establish whether the system is stable and to determine the steady state probabilities of each compound marking cannot be applied.

We will discuss briefly the state grouping problem in the context of continuous time Markov chains, following the elegant presentation in Iosifescu [Ios80]. Appendix 1 provides a brief review of some concepts used in continuous time Markov chain analysis.

Consider a decomposition of the original state space S into the pairwise disjoint states $S_1, S_2, \ldots S_q$, :

$$S = S_1 \cup S_2 \cup S_3 \cdots \cup S_q \tag{3.5}$$

Given any subset of states, say A, $A \subset S$ and any state $i \in S$ we define the transition probabilities from state i to the subset A as:

$$P(i,A) = \sum_{j \in A} P(i,j) \tag{3.6}$$

and we denote: $\hat{k}$ any state in S_k, and $\hat{l}$ any state in S_l

Theorem 3.2: A necessary and sufficient condition for a continuous time homogeneous

Markov chain to be groupable with respect to the partition (3.5) is that:

$$p(i, S_l) = \hat{p}(\hat{k}, \hat{l}) \tag{3.7}$$

for all states $i \in S_k$ and for all pairs of subsets S_k and S_l, $1 \leq k, l \leq q$.

The transition matrix of the grouped Markov chain is

$$\hat{p} = (\hat{p}(\hat{k}, \hat{l})) \tag{3.8}$$

If the total number of states of the original Markov chain is r, we denote by B, a q by r matrix such that its k-th row is a probability vector whose nonnull components are those corresponding to the states in S_k, $1 \leq k \leq q$. Let C be an r by q matrix such that the nonnull components of its k column are equal to 1 and correspond to the states in S_k.

Theorem 3.3: A necessary and sufficient condition for the grouped (lumped) process to be a Markov chain is:

$$C \times B \times Q \times C = Q \times C \tag{3.9}$$

The transition matrix function of the grouped process is given by

$$\hat{P}(\cdot) = B \times P(\cdot) \times C. \tag{3.10}$$

Let us now examine the definition of compound marking and its properties presented in the previous section as well as the properties of the grouping of Markov states presented above. Clearly, the compound marking of a SHLPN induces a partitioning of the Markov state space which satisfies the conditions for grouping. Hence the compound marking SHLPN is isomorphic with a grouped Markov chain.

Theorem 3.4: A Stochastic High Level Petri Net with a compound marking operation is homomorphic with a continuous time Markov chain with a grouping operation defined by (3.7).

As a conclusion, given the SHLPN model of a system, after constructing a compound marking as defined in the previous section, we can study the behavior of the system applying Markov techniques. In particular we can determine the steady state probabilities for each compound marking (grouped state) if the system is ergodic.

Rather than giving additional rules for constructing a compound marking we describe in detail two examples of modeling with SHLPNs augmented with compound markings.

4. Application of SHLPNs to Communication Protocols Performance Analysis

Stochastic High Level Petri nets are well suited for modeling and performance analysis of communication protocols. To illustrate this type of applications of SHLPNs a transport protocol will be analyzed in this section.

A sliding window flow control mechanism as well as an error control mechanism are embedded into the model shown in Figure 5. On the right hand side of places A and D is the sender; the receiver is on the other side. The processing of received packets is omitted. The sender's window is denoted by W and the receiver's window is one. Acknowledgments and negative acknowledgments are sent in separate control packets. The size a control packet is smaller than that of the data packets.

In the protocol model, the places and attributes of the tokens they contain are:

M is a place representing a buffer which collects the messages to be sent by the transport protocol.

P is a place representing the input buffer receiving the messages from the high layer.

K is a place representing a counter associated with the lower limit of sender's window. Its value is incremented every time an acknowledgment for a previously sent packet is received by the station. The current value is denoted by k.

S is a place representing the transport sequence number of the next packet to be sent. Its value is incremented with every packet sent. The current value is denoted by s.

Q is a place associated with the packet retransmission queue. When sending a packet, a token of type packet is put in this place; when the acknowledgment for the packet is received, the proper token is removed from this place. In case of a negative acknowledgment, a copy of the proper token is retransmitted.

D is a place representing the sending interface with the next lower protocol layer. It may contain tokens of type packet, with two attributes: the token type, p (packet), and the transport sequence number.

R is a place representing the transport sequence number of the next packet to be received, the current value is denoted by r. After receiving an error-free packet, within the receiving window, this counter is incremented.

A is a place representing the receiving interface with the next lower protocol layer. It may contain tokens of the type acknowledgment and/or negative acknowledgment. These tokens have two attributes: the first is the token type, and the other is the sequence number.

In the model represented in Figure 5 the following transitions can be recognized:

T_1 is enabled when the place M contains tokens. The transition rate, λ_1, is dependent upon the marking of place M. This rate is associated with the creation of new messages.

T_2 is enabled when the place P contains tokens and the predicate associated with this transition is satisfied, i.e., the packet to be sent is in the window. The transition rate, λ_2 is related to the marking of place P.

T_3 is enabled when a negative acknowledgment token is received. The transition rate is λ_3.

T_4 When receiving a acknowledgment token with the sequence number i = k, this transition is enabled. In order to maintain the system to be ergodic, a token is put into the place M when it fires. The transition rate is λ_4.

The transitions T_5 and T_6 represent the receiving of packets in error and of error-free packets. The two transitions are in "conflict". Their transition rates are associated with the transmission speed and the packet length. The ratio of their rates is dependent upon the probability of a transmission error.

Let us suppose that the size of the transport sequence number set is 4, and the sender's window size is: $W = 3$. We can select the transport sequence number as the *identity* attribute of the tokens of the type packet. Then $i \in [0,3]$. The values of sequence number counters are expressions in the variable i. In order to simplify the compound marking expression, we use the symbol p_i, a_i and n_i to replace separately <p,i>, <ack,i> and <nak,i> in the state table. Table 4 shows the marking (state) table of the transport protocol model. Using the results presented in the previous section we can obtain the transition rates between the compound states. The state transition table of the transport protocol model is shown in Table 5.

From Table 4 we can observe that there are 16 states corresponding to the marking with no token in place M. The total probability ρ of these states is the probability that the subsystem is busy and cannot accept new messages. In this model messages arrive to the protocol layer according to a Poisson process with mean arrival rate λ_1. This is the transition rate of T_1. When the system is in equilibrium, the actual throughput of the system can be expressed as λ_1 (1 - ρ), with τ the probability of a transmission error.

Let us now consider a system using this protocol which has a 0.05 transmission error probability, 128 bit control packets and 1024 bit data packets. Based upon these values the transition rates of our model are:

$\lambda_2 = 9.0 \quad \lambda_3 = 100.0 \quad \lambda_4 = 100.0$

$\lambda_5 = 5.0 \quad \lambda_6 = 95.0$

The steady state probabilities of the compound markings can be determined for different values of the arrival rate, λ_1. We define the *offered load*, as the total system load including control packets and retransmission packets in addition to the data packets. Figure 6 shows the plot of the throughput versus offered load.

The average packet delay may be obtained using Little's theorem. When the system is busy (there are no tokens in the place M), there are exactly four packets in it and the input rate is the throughput S. Therefore, we can calculate the average delay time T = 4/S. The throughput versus average delay, is plotted in Figure 7.

5. Modeling and Performance Analysis of a Multiprocessor System Using SHLPNs

To assess the modeling power of Stochastic High Level Petri nets we consider now a multiprocessor system as shown in Figure 8a. The performance analysis of a multiprocessor system is becoming an important issue due to the increased availability of such systems. The problem of mapping computations to processors in a multiprocessor environment is nontrivial and involves some apriori knowledge about the expected level of system's performance for a given pattern of behavior of the application. Clearly, the performance of a multiprocessor system depends upon the level of contention for common system resources, the interconnection network and the common memory modules.

There are two basic paradigms for inter-processor communication, determined by the architecture of the system namely message passing and communication through shared memory. The analysis carried out in this section is designed for shared memory communication but it can be extended to accommodate message passing systems. To model the system we assume that each processor executes in a number of *domains and that the execution speed of a given processor is a function of the execution domain.* The model assumes that a random time is needed for the transition from one domain to another.

To simplify the analysis of a multiprocessor configuration used as a case study, we consider that each processor switches between execution in a private domain, when its operands come from the private memory and a common domain when its operands are in one of the common memory modules.

This section is organized as follows: first we describe the basic architecture of a multiprocessor system and the assumptions necessary for system modeling. Then the SHLPN model of the system is presented. We show that though the graph corresponding to the system's model is invariant to the system size (number of processors, busses and common memory modules) the state space grows when the system size increases. Then the methodology to construct a model with a minimal state space is presented, and the equilibrium equations of the system are solved using Markov chain techniques. Based upon the steady state probabilities associated with system's states the performance analysis is carried out.

5.1. System Description and Modeling Assumptions

As shown in Figure 8a, a multiprocessor system consists of a set of n processors, $\mathbf{P} = \{ \mathbf{P}_1, \mathbf{P}_2, \ldots \mathbf{P}_n \}$ interconnected by means of an interconnection network to a set of q common memory modules $\mathbf{M} = \{ \mathbf{M}_1, \mathbf{M}_2, \ldots M_q \}$. The simplest topology of the interconnection network is a set of r busses: $\mathbf{B} = \{ \mathbf{B}_1, \mathbf{B}_2, \ldots \mathbf{B}_r \}$ Each processor is usually connected also to a private memory module through a private bus.

As a general rule, the time to perform a given operation depends whether the operands are in local memory or in the common one. For example in case of a Flex 32 multiprocessor system, assuming that only one processor is active, the time to perform an integer arithmetic operation is 14.66 versus 17.05 μsec. In case of a double precision floating point operation the corresponding times are 25.13 versus 31.5 μsec [Hou86]. When more than one processor is active in common memory the time for a common memory reference will increase due to contention for busses and common memory modules. As a general rule we can assess that:

$$t_{op} (\rho) = t_{op}^{0} + d (\rho) \tag{5.1}$$

with: $t_{op} (\rho)$, the time to perform a given operation when the *load factor* for common resources is ρ, t_{op}^{0} is the corresponding time with no contention for common resources and the last term is determined by the queuing delay due to contention for the system's busses and common memory modules for a given load ρ. A possible definition for this load, adopted in this paper, is the ratio between the time spent in an execution region located in the common domain and the time spent in an execution region located in the private domain.

A common measure of the multiprocessor system performance is the processing power of a system with n identical processors expressed as a fraction of the maximum processing power (n times the processing power of a single processor executing in its private memory). Considering an application which is decomposed into n identical processes, the actual processing power of the system depends upon the ratio between local memory references and common memory ones. For example if for a given application the ratio between local and common memory references is 1 and if a common memory reference is twice as expensive as a local memory one at this load, each processor can work at only 75 percent of its speed on the application discussed in this example.

The purpose of our study is to determine the resource utilization, in particular the processor utilization when the load factor increases and the contention for common resources adds a queuing delay to references for the common memory.

The basic assumptions made for our model are:

a1. All processors exhibit identical behavior for the class of applications considered. It is assumed that the computations performed by all processors are similar and they have the same pattern of memory references. More precisely, it is assumed that each processor spends an exponentially distributed random time with mean $\frac{1}{\lambda_1}$, while executing in its private domain and then an exponentially distributed random time with mean $\frac{1}{\lambda_3}$ while executing in a common domain. To model that common memory references are evenly spread into the set of available common memory modules, it is assumed that after finishing an execution sequence in private memory, each processor draws a random number k, uniformly distributed into the set [1,q], which determines the module where its next common memory reference will be.

a2. The access time to common memory modules have the same distribution for all modules and there is no difference in access time when different busses are used.

a3. When a processor acquires a bus and starts its execution sequence in the common memory, then it releases the bus only after completing its execution sequence in the common domain.

The first assumption is justified since common mapping algorithms tend to decompose a given parallel problem into a number of identical processes one for every processor available in the system. The second and the third assumptions are clearly realistic due to hardware considerations.

5.2. Model Description

Figure 9 presents a Stochastic High Level Petri net model of a multiprocessor system. Though the graph representing the model is invariant to the system size, the state space of the SHLPN clearly depends upon the actual number of: processors, n, common memory modules, q, and busses, r. For our example we assume: n=5, q=3 and r=2, a configuration with five processors, three common memory modules, and two busses, shown in Figure 8b.

The graph consists of five places and three transitions. Each place contains tokens whose type may be be different, as shown in Figure 7. A token has a number of attributes, the first attribute being its type, We recognize three different types: p - processor, m - common memory, b -bus. The second attribute of a token is its *identity, id.* The id attribute is a positive integer with values depending upon the number of objects of a given type. In our example when type=p, id takes values in the set [1,5]. The tokens residing in place Q have a third attribute, the id of the common memory module they are going to refer next.

The general format of a token is: <type, id, i, j,...>. The function *Attr* selects a given attribute of a token. For example the predicate associated with transition G is:

$$Attr\ (p, 3) = Attr\ (m, 2) \tag{5.2}$$

This means that the the third attribute of a p-typed input token should be equal to the second attribute of a m-typed input token, $i = j$ in Figure 9.

The meaning of different places and the tokens they contain are presented in Figure 9. The notation used should be interpreted in the following way: the place P contains the set of tokens of type processor with two attributes, <p, i>, with $i \in [1,5]$. The maximum capacity of place P is equal to the number of processors. The transition E corresponds to an end of execution in the private domain and it occurs with a transition rate exponentially distributed with mean λ_1. As a result of this transition the token moves into place Q where it selects the next common memory reference. A token in place Q has three attributes, <p, i, j> with the first two as before and the third attribute describing the common memory module, $j \in [1,3]$ accessed by processor i. The processor could wait to access the common memory module when either no bus is available or the memory module is busy. Transition G occurs when a processor switches to execution in common domain when the all its input places (P, B, and M) have tokens and when the predicate shown in equation 5.2 is satisfied. The place B contains tokens representing free busses and the place M contains tokens representing free memory modules. The maximum capacities of these places are equal to the number of busses and memory modules. The rate of transition G is λ_2 and it is related to the exponentially distributed communication delay involved in a common memory access. The place A contains tokens representing processes executing in the common domain. The maximum capacity of the places in our graph are:

$$\begin{gathered} Capacity\ (P) = n \\ Capacity\ (Q) = n \\ Capacity\ (M) = q \\ Capacity\ (B) = r \\ Capacity\ (A) = \min\ (n, q, r) \end{gathered} \tag{5.3}$$

The compound markings of the system are presented in Table 6. In order to simplify this table the following convention is used: whenever the attributes of the tokens do not have any effect upon the compound marking, only the number of the tokens present in a given place is shown. When an attribute of a token is present in a predicate only that attribute is shown in the corresponding place if no confusion about the token type is possible.

For example the marking corresponding to state 2 has: 4 tokens in place P (the token type is p according to model description), 2 tokens in place B (type=b), 0 tokens in place A. Only the third attribute, i, of the token present in place Q (the id of the memory module of next reference) is indicated. Also shown are the ids of the tokens present in place M, namely i,j and k.

As a general rule it is necessary to specify in the marking the attributes of the tokens referred to, by any predicate which may be present in the SHLPN. In our case we have to specify the third attribute of tokens in Q and the second attribute of the tokens in M since they appear in the predicate associated with transition G.

Table 7 shows the state transition table of the system. For example state 2 can be reached from the following states: state 1 with the rate $15\times\lambda_1$, state 18 with the rate λ_3 and state 19, with the transition rate equal to λ_3. From state 2, the system goes either to state 3, with the transition rate equal to $8 \times \lambda_1$, to state 4 with rate $4 \times \lambda_1$ or to state 17 with rate λ_2.

State 2 corresponds to the situation when any four processors execute in the private domain and the fifth has selected the memory module of its next common domain reference, to be module i. It should be pointed out that state 2 is a *macro state* obtained due to the use of the compound marking concept and it corresponds to 15 atomic states. These 15 states are distinguished only by the identity attributes of the tokens in two places, P and Q as shown in Table 8. The transition rate from the compound marking denoted as state 1 in Table 6, to the one denoted by state 2 is $15 \times \lambda_1$ since there are 15 individual transitions from one individual marking of state 1 to the 15 individual markings in the compound marking corresponding to state 2.

5.3. Performance Analysis

To determine the average utilization of different system resources it is necessary to solve the equilibrium equations and then the identify the states when each resource is idle and the occupancy of that state, the number of units of that resource which are idle. The following notation is used: $size\,[B\,]_i$ is the occupancy of place B when the system is in state i and p_i the probability of the system being in state i. Then the average utilization of a processor, η_p , a common memory module, η_m, a bus, η_b are defined as:

$$\eta_p = 1 - \sum_{i \in S} \frac{p_i \times size\,[\,Q\,]_i}{n} \tag{5.4}$$

$$\eta_m = 1 - \sum_{i \in S} \frac{p_i \times size\,[\,M\,]_i}{q} \tag{5.5}$$

$$\eta_b = 1 - \sum_{i \in S} \frac{p_i \times size[B]_i}{l} \tag{5.6}$$

We assume a fixed ratio between the transition rate related transition G and that related transition E, $\frac{\lambda_1}{\lambda_2} = 10^{-3}$. This expresses the fact that the communication delay is much smaller than the time spent in executing in the private domain.

The load for common resources is defined as:

$$\rho = \frac{\lambda_1}{\lambda_3} \tag{5.7}$$

We study the resource utilization for $5 \times 10^{-2} \leq \rho \leq 4$.

Figure 10a. shows the processor utilization when the load placed upon shared system resources increases. Even when the load is closed to 0.7 the actual speed up factor of the system is only about 50 percent of its maximum value. Bus utilization increases (Figure 10c) rapidly even for lower values of the load and indicates that the system is communication bounded and no significant performance improvement can be obtained by adding more memory modules. This corroborates with the information provided by the memory utilization curve shown in Figure 10b. which shows a maximum common memory utilization of about 60 percent.

The number of original states is very high, larger than 500, and we have reduced the model to only 51 states. As mentioned earlier the same conceptual model can be used to model a message passing system. In such a case λ_2 will be related to the time necessary to pass a message from one processor to another, including the processor communication overhead at the sender and at the receiving site as well as the transmission time dependent upon the message size and the communication delay. In case of a synchronous message passing system λ_3 will be related to the average blocking time in order to generate a reply.

6. Conclusions

The Stochastic High Level Petri nets introduced in this paper represent a powerful and convenient tool for performance analysis of different communication and computing systems. They are extensions of Stochastic Petri Nets but generally they lead to models with a lower size of the state space. The compound marking concept allows a considerable reduction of the number of states and it induces a correct grouping of states in the Markov domain. Two examples illustrate the advantages of SHLPNs, the first one is related to the performance analysis of a transport protocol. The second example models the impact of contention for system resources in a multiprocessor system.

APPENDIX 1 - Overview of Basic Terms Related to Continuous Time Markov Chains

In the followings whenever we refer to a Markov chain we consider a continuous-time, homogeneous Markov process. It should be noted however that some authors use the term *Markov chain* for a discrete-time process with Markov property and call a continuous-time process with Markov property, *a Markov process*.

A Markov chain with the state space S is characterized by its transition matrix function, $P(\cdot)$. This function associates with any t, a stochastic matrix, $P(t)$ such that for any $t, s \geq 0$ and for any two states $i, j \in S$ the following equation holds:

$$P(s+t) = P(s) \times P(t) \tag{A1.1}$$

An equivalent form of the previous equation is:

$$p(s+t, i, j) = \sum_{k \in S} p(s, i, k) \times p(t, k, j) \tag{A1.1'}$$

The functions $p(t, i, j)$ describe the probability of a transition from state i to state j in an interval of time equal to t and they are uniformly continuous on $[0, \infty)$. The transition matrix function $P(\cdot)$ are differentiable on the same interval. Extremely important for a Markov chain is the *intensity matrix*, $Q(i, j) = (q_{i,j})$ with $i, j \in S$, defined as:

$$Q = P'(0) \tag{A1.2}$$

The elements of the matrix Q are:

$$q(i,i) = \lim_{h \to 0^+} \frac{p(h,i,i)-1}{h} \leq 0 \quad \text{for} \quad j = i \tag{A1.3}$$

The condition:

$$\sum_{j \in S} p(t,i,j) = 1 \qquad i \in S \tag{A1.4}$$

leads to:

$$\sum_{j \in S} q(i,j) = 0, \qquad i \in S \tag{A1.5}$$

or in matrix notation with e the unity matrix:

$$Q \times e = 0 \tag{A1.6}$$

To outline the probabilistic significance of the intensity matrix Q we define the intensity of passage from state i as:

$$q(i) = -q(i,i) \tag{A1.7}$$

Then $q(k,j) \,/\, q(i)$ can be shown to represent the conditional probability of a transition at an arbitrary time from state i to state j given that a transition has in fact taken place. If we define:

$$p^*(i,j) = \begin{cases} (1-\delta(i,j))\dfrac{q(i,j)}{q(i)} & \text{if } q(i) \neq 0 \\ \delta(i,j) & \text{if } q(i) = 0 \end{cases} \tag{A1.8}$$

then we can describe the evolution of the process as follows: if it starts in state i, it remains there a random length of time exponentially distributed with parameters $q(i)$ and then moves to state j with probability $p^*(i,j)$ and so on.

References

[Bal86] G. Balbo, S. C. Bruell and S. Ghanta, *"Combining Queueing Network and Generalized Stochastic Petri Net Models for the Analysis of Some Software Blocking Phenomena,"* IEEE Trans. Software Eng., Vol. **SE-12**, No. 4, PP. 561-576, Apr. 1986

[Gen79] J.H. Genrich and K. Lautenbach, *"The Analysis of Distributed Systems by Means of Predicate/Transition-Nets,"* Lecture Notes on Comp. Sci., Vol. **70**, PP. 123-146, 1979

[Gen81] J.H. Genrich and K. Lautenbach, *"System Modelling with High-Level Petri Nets,"* Theoretical Comp. Sci., Vol. **13**, PP. 109-136, 1981

[Hou86] C. E. Houstis, E. N. Houstis and J. R. Rice, *"Partitioning PDE Computations: Methods and Performance Evaluation,"* 1986

[Hub85] P. Huber, A. M. Jensen, L. O. Jepsen and K. Jensen *"Towards Reachability Trees for High-Level Petri Nets,"* Lecture Nots in Comp. Sci. **188**, PP.215-233, 1985

[Ios80] M. Iosifescu, *"Finite Markov Processes and Their Applications,"* Wiley, 1980

[Jen81] K. Jensen, *"Coloured Petri Nets and the Invariant-Method,"* Theoretical Comp. Sci., Vol. **14**, PP. 317-336, 1981

[Jen83] K. Jensen, *"High-level Petri nets,"* Informatik-Fachberichte, Vol. **66**, PP. 166-180, 1983

[Lin86] Chuang Lin and Dan Cristian Marinescu *"Application of Modified Predicate Transition Nets to Modeling and Simulation of Communication Protocols"* Computer Science Department, Purdue University, Technical Report CSD-TR-599, May 1986

[Mar84] M.A. Marsan, G. Conte and G. Balbo, *"A Class of Generalized Stochastic Petri Nets for the Performance Evaluation of Multiprocessor systems,"* ACM

Trans. Comp. Systems, Vol. **2**, No. 2, PP. 93-122, May 1984

[Mol82] M.K. Molloy, *"Performance Analysis Using Stochastic Petri Nets,"* IEEE Trans. Comp. **C-31**, 9, PP. 913-917, Sep. 1982

[Mol85] M.K. Molloy, *"Discrete Time Stochastic Petri Nets,"* IEEE Trans. Software Eng., Vol. **SE-11**, No. 4, PP. 417-423, 1985

[Ram80] C. V. Ramamoorthy and G. S. Ho, *"Performance Evaluation of Asynchronous Concurrent Systems Using Petri Nets,"* IEEE Trans. Software Eng., Vol. **SE-6**, PP. 440-449, Sept. 1980

[Rei83] W. Reisig, *"Petri Nets with Individual Tokens,"* Informatik-Fachberichte, Vol. **66**, PP. 229-249, 1983

[Sif77] J. Sifakis, *"Petri Nets for Performance Evaluation,"* in Measuring, Modeling, and Evaluating Computer Systems, Proc. **3rd** Int. Symp. IFIP Working Group 7.3, H. Beilner and E. Gelenbe, Eds., PP. 75-93, 1977

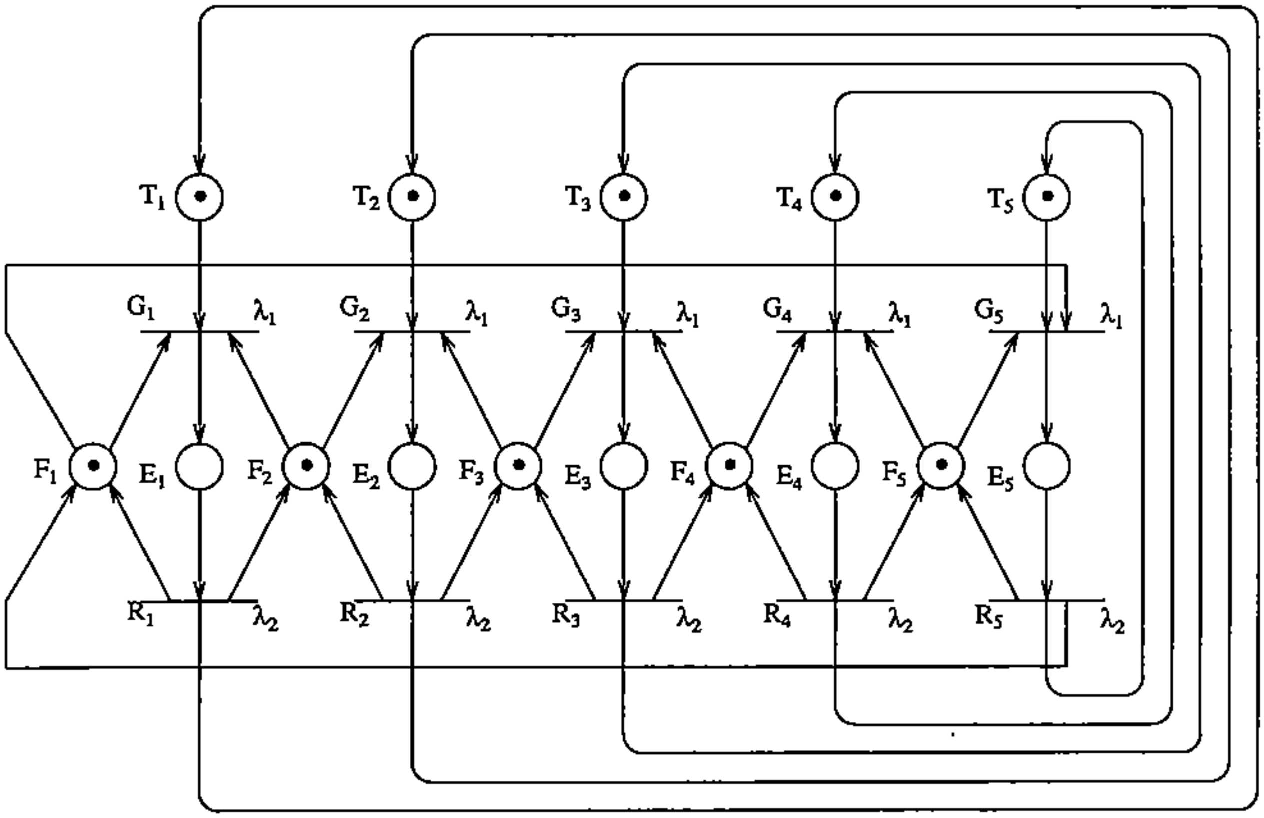
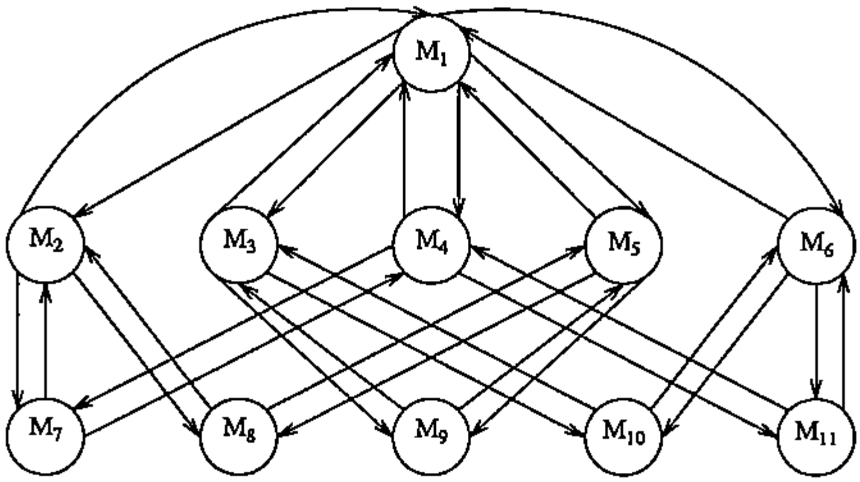


Figure 1. Modeling of the philosopher system with the Stochastic Petri Net

	T_1	T_2	T_3	T_4	T_5	E_1	E_2	E_3	E_4	E_5	F_1	F_2	F_3	F_4	F_5
M_1	1	1	1	1	1	0	0	0	0	0	1	1	1	1	1
M_2	0	1	1	1	1	1	0	0	0	0	0	0	1	1	1
M_3	1	0	1	1	1	0	1	0	0	0	1	0	0	1	1
M_4	1	1	0	1	1	0	0	1	0	0	1	1	0	0	1
M_5	1	1	1	0	1	0	0	0	1	0	1	1	1	0	0
M_6	1	1	1	1	0	0	0	0	0	1	0	1	1	1	0
M_7	0	1	0	1	1	1	0	1	0	0	0	0	0	0	1
M_8	0	1	1	0	1	1	0	0	1	0	0	0	1	0	0
M_9	1	0	1	0	1	0	1	0	1	0	1	0	0	0	0
M_{10}	1	0	1	1	0	0	1	0	0	1	0	0	0	1	0
M_{11}	1	1	0	1	0	0	0	1	0	1	0	1	0	0	0

Table 1. Markings of the philosopher system with the Stochastic Petri Net



The transition rate:
from M_1 to M_i (i=2,3,4,5,6) separately as λ_1, invertly as λ_2
from M_i (i=2,3,4,5,6) to M_j (j=7,8,9,10,11) as λ_1, if any, invertly as λ_2

Figure 2. The Markov state diagram of the philosopher system

State	Place index		
	T	E	F
1	<p,1>, <p,2>, <p,3>, <p,4>, <p,5>	0	<f,1>, <f,2>, <f,3>, <f,4>, <f,5>
2	<p,2>, <p,3>, <p,4>, <p,5>	<p,1,1,2>	<f,3>, <f,4>, <f,5>
3	<p,1>, <p,3>, <p,4>, <p,5>	<p,2,2,3>	<f,1>, <f,4>, <f,5>
4	<p,1>, <p,2>, <p,4>, <p,5>	<p,3,3,4>	<f,1>, <f,2>, <f,5>
5	<p,1>, <p,2>, <p,3>, <p,5>	<p,4,4,5>	<f,1>, <f,2>, <f,3>
6	<p,1>, <p,2>, <p,3>, <p,4>	<p,5,5,1>	<f,2>, <f,3>, <f,4>
7	<p,2>, <p,4>, <p,5>	<p,1,1,2>, <p,3,3,4>	<f,5>
8	<p,2>, <p,3>, <p,5>	<p,1,1,2>, <p,4,4,5>	<f,3>
9	<p,1>, <p,3>, <p,5>	<p,2,2,3>, <p,4,4,5>	<f,1>
10	<p,1>, <p,3>, <p,4>	<p,2,2,3>, <p,5,5,1>	<f,4>
11	<p,1>, <p,2>, <p,4>	<p,3,3,4>, <p,5,5,1>	<f,2>

Table 2. The state table of the philosopher system with individual markings

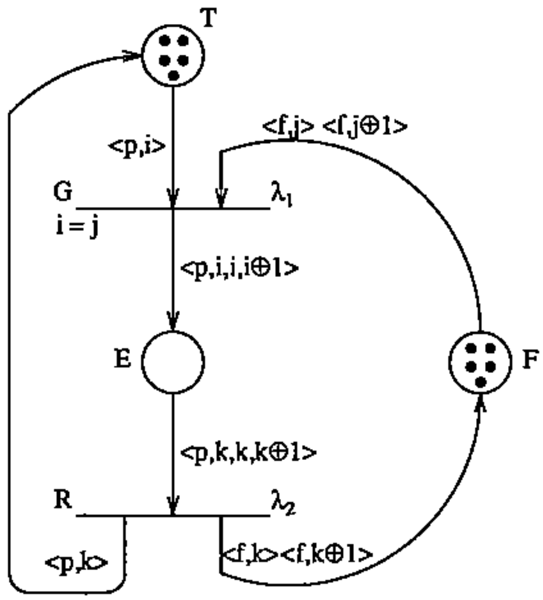


Figure 3. Modeling of the philosopher system with SHLPN

State	Place index		
	T	E	F
1	<p,i>,<p,i⊕1>,<p,i⊕2>,<p,i⊕3>,<p,i⊕4>	0	<f,i>,<f,i⊕1>,<f,i⊕2>,<f,i⊕3>,<f,i⊕4>
2	<p,i⊕1>,<p,i⊕2>,<p,i⊕3>,<p,i⊕4>	<p,i,i,i⊕1>	<f,i⊕2>,<f,i⊕3>,<f,i⊕4>
3	<p,i⊕1>,<p,i⊕3>,<p,i⊕4>	<p,i,i,i⊕1>,<p,i⊕2,i⊕2,i⊕3>	<f,i⊕4>

Table 3. The state table of the philosopher system with compound markings

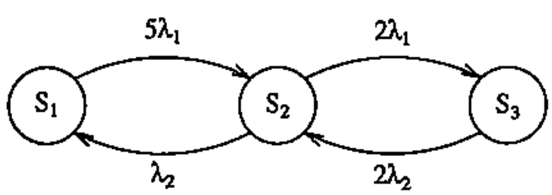


Figure 4. The Markov state diagram of the philosopher system with compound markings

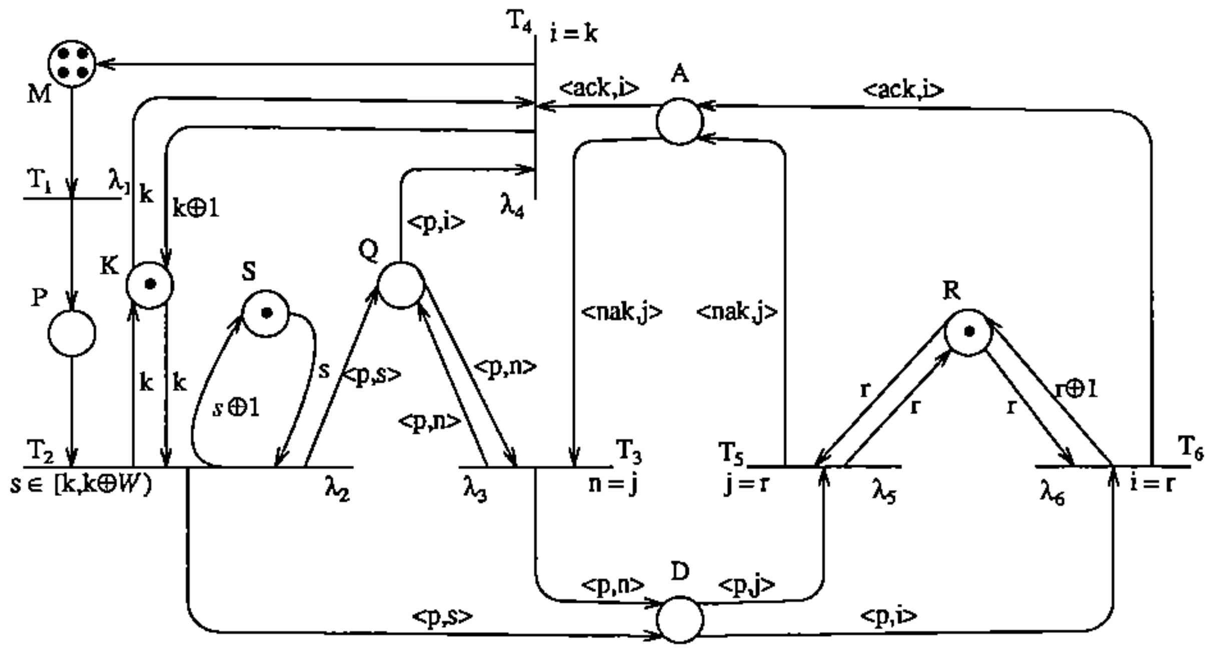


Figure 5. Modeling of the transport protocol with SHLPN

Marking	Place index							
(State)	M	P	K	S	Q	D	R	A
1	4	0	<i>	<i>	0	0	<i>	0
2	3	1	<i>	<i>	0	0	<i>	0
3	2	2	<i>	<i>	0	0	<i>	0
4	1	3	<i>	<i>	0	0	<i>	0
5	0	4	<i>	<i>	0	0	<i>	0
6	0	3	<i>	<i⊕1>	p_i	p_i	<i>	0
7	0	2	<i>	<i⊕2>	$p_i, p_{i\oplus1}$	$p_i, p_{i\oplus1}$	<i>	0
8	0	1	<i>	<i⊕3>	$p_i, p_{i\oplus1}, p_{i\oplus2}$	$p_i, p_{i\oplus1}, p_{i\oplus2}$	<i>	0
9	0	1	<i>	<i⊕3>	$p_i, p_{i\oplus1}, p_{i\oplus2}$	$p_{i\oplus1}, p_{i\oplus2}$	<i⊕1>	a_i
10	0	1	<i>	<i⊕3>	$p_i, p_{i\oplus1}, p_{i\oplus2}$	$p_{i\oplus2}$	<i⊕2>	$a_i, a_{i\oplus1}$
11	0	1	<i>	<i⊕3>	$p_i, p_{i\oplus1}, p_{i\oplus2}$	0	<i⊕3>	$a_i, a_{i\oplus1}, a_{i\oplus2}$
12	1	1	<i⊕1>	<i⊕3>	$p_{i\oplus1}, p_{i\oplus2}$	0	<i⊕3>	$a_{i\oplus1}, a_{i\oplus2}$
13	2	1	<i⊕2>	<i⊕3>	$p_{i\oplus2}$	0	<i⊕3>	$a_{i\oplus2}$
14	1	2	<i⊕2>	<i⊕3>	$p_{i\oplus2}$	0	<i⊕3>	$a_{i\oplus2}$
15	0	3	<i⊕2>	<i⊕3>	$p_{i\oplus2}$	0	<i⊕3>	$a_{i\oplus2}$
16	0	2	<i⊕2>	<i>	$p_{i\oplus2}, p_{i\oplus3}$	$p_{i\oplus3}$	<i⊕3>	$a_{i\oplus2}$
17	1	2	<i⊕3>	<i>	$p_{i\oplus3}$	$p_{i\oplus3}$	<i⊕3>	0
18	1	1	<i⊕3>	<i⊕1>	$p_{i\oplus3}, p_i$	$p_{i\oplus3}, p_i$	<i⊕3>	0
19	1	0	<i⊕3>	<i⊕2>	$p_{i\oplus3}, p_i, p_{i\oplus1}$	$p_{i\oplus3}, p_i, p_{i\oplus1}$	<i⊕3>	0
20	1	0	<i⊕3>	<i⊕2>	$p_{i\oplus3}, p_i, p_{i\oplus1}$	$p_i, p_{i\oplus1}$	<i>	$a_{i\oplus3}$
21	2	0	<i>	<i⊕2>	$p_i, p_{i\oplus1}$	$p_i, p_{i\oplus1}$	<i>	0
22	2	0	<i>	<i⊕2>	$p_i, p_{i\oplus1}$	$p_{i\oplus1}$	<i⊕1>	a_i
23	1	1	<i>	<i⊕2>	$p_i, p_{i\oplus1}$	$p_{i\oplus1}$	<i⊕1>	a_i
24	1	2	<i⊕3>	<i>	$p_{i\oplus3}$	0	<i⊕3>	$n_{i\oplus3}$
25	0	3	<i⊕3>	<i>	$p_{i\oplus3}$	0	<i⊕3>	$n_{i\oplus3}$
26	0	2	<i⊕3>	<i⊕1>	$p_{i\oplus3}, p_i$	p_i	<i⊕3>	$n_{i\oplus3}$
27	0	1	<i⊕3>	<i⊕2>	$p_{i\oplus3}, p_i, p_{i\oplus1}$	$p_i, p_{i\oplus1}$	<i⊕3>	$n_{i\oplus3}$
28	1	1	<i⊕3>	<i⊕1>	$p_{i\oplus3}, p_i$	p_i	<i⊕3>	$n_{i\oplus3}$
29	1	0	<i⊕3>	<i⊕2>	$p_{i\oplus3}, p_i, p_{i\oplus1}$	$p_i, p_{i\oplus1}$	<i⊕3>	$n_{i\oplus3}$
30	0	2	<i⊕2>	<i>	$p_{i\oplus2}, p_{i\oplus3}$	0	<i⊕3>	$a_{i\oplus2}, n_{i\oplus3}$
31	0	1	<i⊕2>	<i⊕1>	$p_{i\oplus2}, p_{i\oplus3}, p_i$	p_i	<i⊕3>	$a_{i\oplus2}, n_{i\oplus3}$
32	0	2	<i⊕1>	<i⊕3>	$p_{i\oplus1}, p_{i\oplus2}$	0	<i⊕3>	$a_{i\oplus1}, a_{i\oplus2}$
33	2	1	<i>	<i⊕1>	p_i	p_i	<i>	0
34	2	1	<i>	<i⊕1>	p_i	0	<i>	n_i
35	2	0	<i>	<i⊕1>	$p_i, p_{i\oplus1}$	$p_{i\oplus1}$	<i>	n_i
36	1	1	<i>	<i⊕2>	$p_i, p_{i\oplus1}$	0	<i⊕1>	a_i
37	1	0	<i>	<i⊕3>	$p_i, p_{i\oplus1}, p_{i\oplus2}$	$p_{i\oplus2}$	<i⊕1>	$a_i, n_{i\oplus1}$
38	3	0	<i>	<i⊕1>	p_i	p_i	<i>	0
39	3	0	<i>	<i⊕1>	p_i	0	<i>	n_i
40	3	0	<i>	<i⊕1>	p_i	0	<i>	a_i
41	2	0	<i>	<i⊕2>	$p_i, p_{i\oplus1}$	0	<i⊕1>	$a_i, n_{i\oplus1}$
42	2	0	<i>	<i⊕2>	$p_i, p_{i\oplus1}$	0	<i⊕2>	$a_i, a_{i\oplus1}$
43	1	0	<i⊕1>	<i>	$p_{i\oplus1}, p_{i\oplus2}, p_{i\oplus3}$	$p_{i\oplus3}$	<i⊕3>	$a_{i\oplus1}, a_{i\oplus2}$
44	1	0	<i⊕1>	<i>	$p_{i\oplus1}, p_{i\oplus2}, p_{i\oplus3}$	0	<i⊕3>	$a_{i\oplus1}, a_{i\oplus2}, n_{i\oplus3}$
45	0	1	<i⊕1>	<i>	$p_{i\oplus1}, p_{i\oplus2}, p_{i\oplus3}$	0	<i⊕3>	$a_{i\oplus1}, a_{i\oplus2}, n_{i\oplus3}$
46	1	0	<i⊕1>	<i>	$p_{i\oplus1}, p_{i\oplus2}, p_{i\oplus3}$	0	<i>	$a_{i\oplus1}, a_{i\oplus2}, n_{i\oplus3}$

Table 4. State table of the transport protocol model

State	Previous state and transition rate	Post state and transition rate
1	$<40,\lambda_4>$	$<2,4\lambda_1>$
2	$<1,4\lambda_1>\ <13,\lambda_4>$	$<3,3\lambda_1>\ <38,\lambda_2>$
3	$<2,3\lambda_1>\ <14,\lambda_4>$	$<4,2\lambda_1>\ <33,2\lambda_2>$
4	$<3,2\lambda_1>\ <15,\lambda_4>$	$<5,\lambda_1>\ <17,3\lambda_2>$
5	$<4,\lambda_1>$	$<6,4\lambda_2>$
6	$<17,\lambda_1>\ <5,4\lambda_2>\ <25,\lambda_3>$	$<7,3\lambda_2>\ <25,\lambda_5>\ <15,\lambda_6>$
7	$<18,\lambda_1>\ <6,3\lambda_2>\ <26,\lambda_3>$	$<8,2\lambda_2>\ <26,\lambda_5>\ <16,\lambda_6>$
8	$<19,\lambda_1>\ <7,2\lambda_2>\ <27,\lambda_3>$	$<27,\lambda_5>\ <9,\lambda_6>$
9	$<20,\lambda_1>\ <16,2\lambda_2>\ <31,\lambda_3>\ <8,\lambda_6>$	$<18,\lambda_4>\ <31,\lambda_5>\ <10,\lambda_6>$
10	$<43,\lambda_1>\ <32,2\lambda_2>\ <45,\lambda_3>\ <9,\lambda_6>$	$<23,\lambda_4>\ <45,\lambda_5>\ <11,\lambda_6>$
11	$<46,\lambda_1>\ <10,\lambda_6>$	$<12,\lambda_4>$
12	$<42,2\lambda_1>\ <11,\lambda_4>\ <23,\lambda_6>$	$<32,\lambda_1>\ <43,\lambda_2>\ <13,\lambda_4>$
13	$<40,3\lambda_1>\ <12,\lambda_4>\ <33,\lambda_6>$	$<14,2\lambda_1>\ <22,\lambda_2>\ <2,\lambda_4>$
14	$<13,2\lambda_1>\ <32,\lambda_4>\ <17,\lambda_6>$	$<15,\lambda_1>\ <23,2\lambda_2>\ <3,\lambda_4>$
15	$<14,\lambda_1>\ <6,\lambda_6>$	$<16,3\lambda_2>\ <4,\lambda_4>$
16	$<23,\lambda_1>\ <15,3\lambda_2>\ <30,\lambda_3>\ <7,\lambda_6>$	$<9,2\lambda_2>\ <17,\lambda_4>\ <30,\lambda_5>\ <32,\lambda_6>$
17	$<33,2\lambda_1>\ <4,3\lambda_2>\ <24,\lambda_3>\ <16,\lambda_4>$	$<6,\lambda_1>\ <18,2\lambda_2>\ <24,\lambda_5>\ <14,\lambda_6>$
18	$<21,2\lambda_1>\ <17,2\lambda_2>\ <28,\lambda_3>\ <9,\lambda_4>$	$<7,\lambda_1>\ <19,\lambda_2>\ <28,\lambda_5>\ <23,\lambda_6>$
19	$<18,\lambda_2>\ <29,\lambda_3>$	$<8,\lambda_1>\ <29,\lambda_5>\ <20,\lambda_6>$
20	$<23,\lambda_2>\ <37,\lambda_3>\ <19,\lambda_6>$	$<9,\lambda_1>\ <21,\lambda_4>\ <37,\lambda_5>\ <43,\lambda_6>$
21	$<33,\lambda_2>\ <35,\lambda_3>\ <20,\lambda_4>$	$<18,2\lambda_1>\ <35,\lambda_5>\ <22,\lambda_6>$
22	$<13,\lambda_2>\ <41,\lambda_3>\ <43,\lambda_4>\ <21,\lambda_6>$	$<23,2\lambda_1>\ <38,\lambda_4>\ <41,\lambda_5>\ <42,\lambda_6>$
23	$<22,2\lambda_1>\ <14,2\lambda_2>\ <36,\lambda_3>\ <10,\lambda_4>\ <18,\lambda_6>$	$<16,\lambda_1>\ <20,\lambda_2>\ <33,\lambda_4>\ <36,\lambda_5>\ <12,\lambda_6>$
24	$<34,2\lambda_1>\ <30,\lambda_4>\ <17,\lambda_5>$	$<25,\lambda_1>\ <28,2\lambda_2>\ <17,\lambda_3>$
25	$<24,\lambda_1>\ <6,\lambda_5>$	$<26,3\lambda_2>\ <6,\lambda_3>$
26	$<28,\lambda_1>\ <25,3\lambda_2>\ <7,\lambda_5>$	$<27,2\lambda_2>\ <7,\lambda_3>$
27	$<29,\lambda_1>\ <26,2\lambda_2>\ <8,\lambda_5>$	$<8,\lambda_3>$
28	$<35,2\lambda_1>\ <24,2\lambda_2>\ <31,\lambda_4>\ <18,\lambda_5>$	$<26,\lambda_1>\ <29,\lambda_2>\ <18,\lambda_3>$
29	$<28,\lambda_2>\ <19,\lambda_5>$	$<27,\lambda_1>\ <19,\lambda_3>$
30	$<36,\lambda_1>\ <16,\lambda_5>$	$<31,2\lambda_2>\ <16,\lambda_3>\ <24,\lambda_4>$
31	$<37,\lambda_1>\ <30,2\lambda_2>\ <9,\lambda_5>$	$<9,\lambda_3>\ <28,\lambda_4>$
32	$<12,\lambda_1>\ <16,\lambda_6>$	$<10,2\lambda_2>\ <14,\lambda_4>$
33	$<38,3\lambda_1>\ <3,2\lambda_2>\ <34,\lambda_3>\ <23,\lambda_4>$	$<17,2\lambda_1>\ <21,\lambda_2>\ <34,\lambda_5>\ <13,\lambda_6>$
34	$<39,3\lambda_1>\ <36,\lambda_4>\ <33,\lambda_5>$	$<17,2\lambda_1>\ <35,\lambda_2>\ <33,\lambda_3>$
35	$<34,\lambda_2>\ <37,\lambda_4>\ <21,\lambda_5>$	$<28,2\lambda_1>\ <21,\lambda_3>$
36	$<41,2\lambda_1>\ <45,\lambda_4>\ <23,\lambda_5>$	$<30,\lambda_1>\ <37,\lambda_2>\ <23,\lambda_3>\ <34,\lambda_4>$
37	$<36,\lambda_2>\ <20,\lambda_5>$	$<31,\lambda_1>\ <20,\lambda_3>\ <35,\lambda_4>$
38	$<2,\lambda_2>\ <39,\lambda_3>\ <22,\lambda_4>$	$<33,3\lambda_1>\ <39,\lambda_5>\ <40,\lambda_6>$
39	$<41,\lambda_4>\ <38,\lambda_5>$	$<34,3\lambda_1>\ <38,\lambda_3>$
40	$<42,\lambda_4>\ <38,\lambda_6>$	$<13,3\lambda_1>\ <1,\lambda_4>$
41	$<44,\lambda_4>\ <22,\lambda_5>$	$<36,2\lambda_1>\ <23,\lambda_3>\ <39,\lambda_4>$
42	$<46,\lambda_4>\ <22,\lambda_6>$	$<12,2\lambda_1>\ <40,\lambda_4>$
43	$<12,\lambda_2>\ <44,\lambda_3>\ <20,\lambda_6>$	$<10,\lambda_1>\ <22,\lambda_4>\ <44,\lambda_5>\ <46,\lambda_6>$
44	$<43,\lambda_5>$	$<45,\lambda_1>\ <43,\lambda_3>\ <41,\lambda_4>$
45	$<44,\lambda_1>\ <10,\lambda_5>$	$<10,\lambda_3>\ <36,\lambda_4>$
46	$<43,\lambda_6>$	$<11,\lambda_1>\ <42,\lambda_4>$

Table 5. State transition table of the transport protocol model

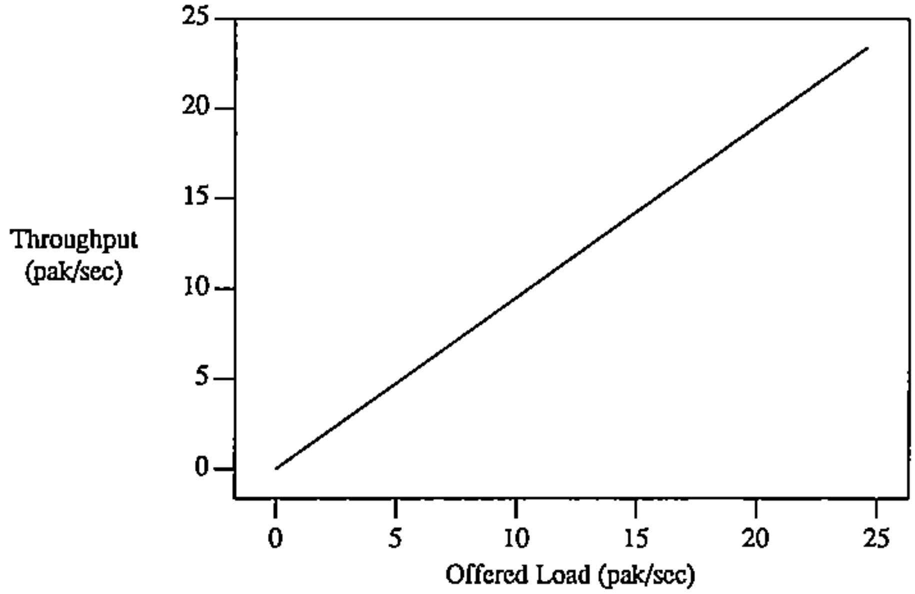


Figure 6. Throughput versus offered load for the transport protocol

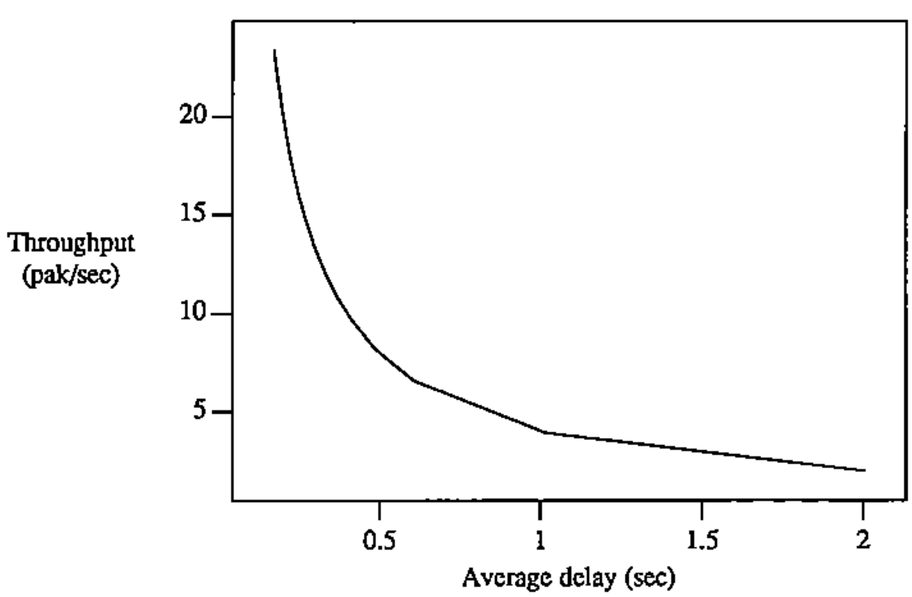


Figure 7. Throughput versus average delay for the transport protocol

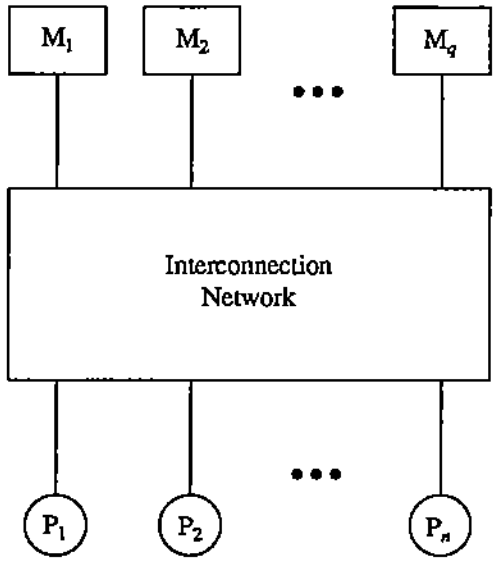


Figure 8a. A shared memory multiprocessor system

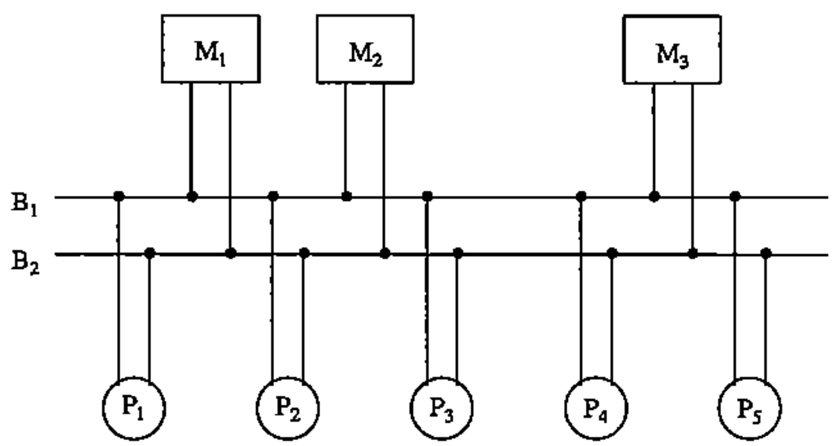
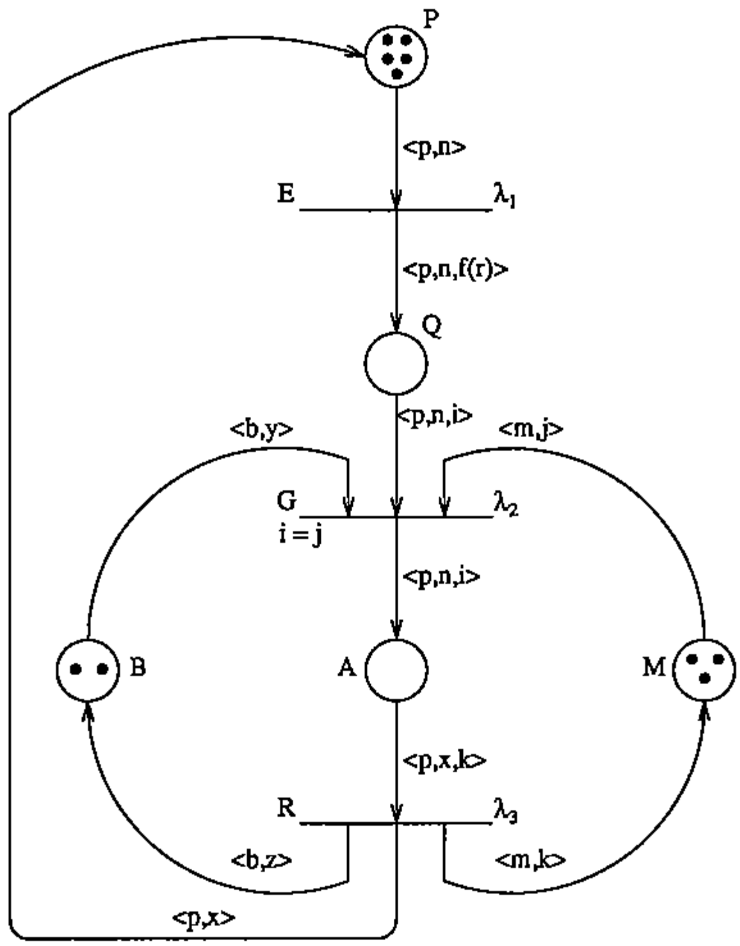


Figure 8b. The configuration of the multiprocessor system used in SHLPN modeling



PLACE AND TRANSITION SIGNIFICANCE

- **P** is the "private memory" place. When the place P holds tokens, the corresponding processors are active in their own memory.
- **Q** is the "queuing" place. When the place Q holds tokens, the corresponding processors are queued for the requiring common memory.
- **A** is the "accessing" place. When the place A holds tokens, the corresponding processors are accessing common memory.
- **M** is the "idle common memory" place. When the place M holds tokens, the corresponding common memories are idle.
- **B** is the "available bus" place. When the place B holds tokens, the corresponding busses are available.
- **E** is the "end of activity" transition. When it fires, a processor ends its activity in its private memory.
- **G** is the "getting common memory" transition. The transition enables when the processor can get the requiring common memory and a bus. The transition firing time is associated with $1/\lambda_2$.
- **R** is the "releasing common memory" transition. After accessing common memory, the processor releases the common memory and bus and returns to the private memory. This activity is represented by the transition R.

Figure 9. Modeling of the multiprocessor system with a SHLPN

Marking	Place index				
(State)	P	Q	M	B	A
1	5	0	i, j, k	2	0
2	4	i	i, j, k	2	0
3	3	i, j	i, j, k	2	0
4	3	i, i	i, j, k	2	0
5	2	i, j, k	i, j, k	2	0
6	2	i, i, j	i, j, k	2	0
7	2	i, i, i	i, j, k	2	0
8	1	i, i, j, k	i, j, k	2	0
9	1	i, i, i, j	i, j, k	2	0
10	1	i, i, j, j	i, j, k	2	0
11	1	i, i, i, i	i, j, k	2	0
12	0	i, i, i, j, k	i, j, k	2	0
13	0	i, i, j, j, k	i, j, k	2	0
14	0	i, i, i, i, j	i, j, k	2	0
15	0	i, i, i, j, j	i, j, k	2	0
16	0	i, i, i, i, i	i, j, k	2	0
17	4	0	j, k	1	i
18	3	j	j, k	1	i
19	3	i	j, k	1	i
20	2	j, k	j, k	1	i
21	2	i, j	j, k	1	i
22	2	i, i	i, k	1	j
23	2	i, i	j, k	1	i
24	1	i, j, k	j, k	1	i
25	1	i, i, j	i, k	1	j
26	1	i, i, j	j, k	1	i
27	1	i, i, i	i, k	1	j
28	1	i, i, i	j, k	1	i
29	0	i, i, j, k	j, k	1	i
30	0	i, i, i, k	i, k	1	j
31	0	i, i, j, j	i, j	1	k
32	0	i, j, j, k	j, k	1	i
33	0	i, i, i, i	i, k	1	j
34	0	i, i, i, j	j, k	1	i
35	0	i, i, j, j	j, k	1	i
36	0	i, i, i, j	i, k	1	j

(continued)

Marking	Place index				
(State)	P	Q	M	B	A
37	0	i, i, i, i	j, k	1	i
38	3	0	k	0	i, j
39	2	k	k	0	i, j
40	2	i	k	0	i, j
41	1	i, i, k	i, k	1	j
42	1	i, k	k	0	i, j
43	1	i, j	k	0	i, j
44	1	i, i	k	0	i, j
45	0	i, i, k	k	0	i, j
46	0	i, i, i	i	0	j, k
47	0	i, j, j	j	0	i, k
48	0	i, j, k	k	0	i, j
49	0	i, i, i	k	0	i, j
50	0	i, i, j	k	0	i, j
51	1	k, k	k	0	i, j

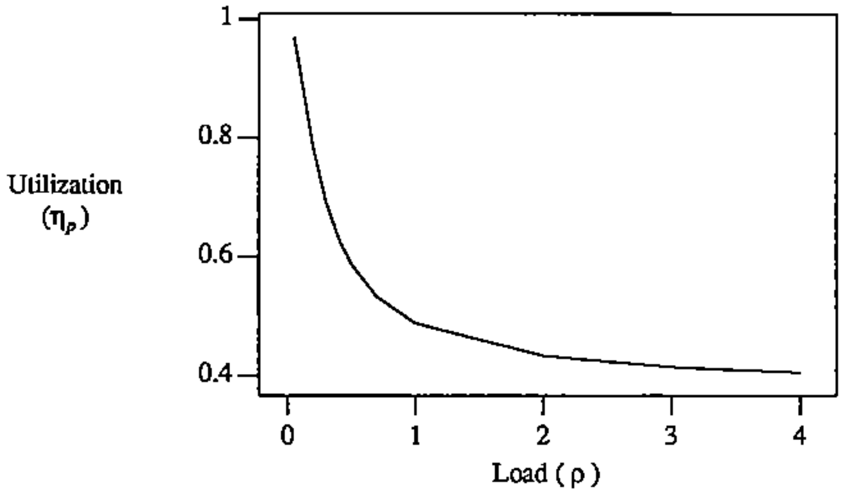
Table 6. State table of the multiprocessor system model

State	Previous state and transition rate	Post state and transition rate
1	$<17,\lambda_3>$	$<2,15\lambda_1>$
2	$<1,15\lambda_1>$ $<18,\lambda_3>$ $<19,\lambda_3>$	$<3,8\lambda_1>$ $<4,4\lambda_1>$ $<17,\lambda_2>$
3	$<2,8\lambda_1>$ $<20,\lambda_3>$ $<21,\lambda_3>$	$<5,3\lambda_1>$ $<6,6\lambda_1>$ $<18,2\lambda_2>$
4	$<2,4\lambda_1>$ $<22,\lambda_3>$ $<23,\lambda_3>$	$<6,6\lambda_1>$ $<7,3\lambda_1>$ $<19,2\lambda_2>$
5	$<3,3\lambda_1>$ $<24,\lambda_3>$	$<8,6\lambda_1>$ $<20,3\lambda_2>$
6	$<3,6\lambda_1>$ $<4,6\lambda_1>$ $<25,\lambda_3>$ $<26,\lambda_3>$ $<41,\lambda_3>$	$<8,2\lambda_1>$ $<9,2\lambda_1>$ $<10,2\lambda_1>$ $<21,2\lambda_2>$ $<22,\lambda_2>$
7	$<4,3\lambda_1>$ $<27,\lambda_3>$ $<28,\lambda_3>$	$<9,4\lambda_1>$ $<11,2\lambda_1>$ $<23,3\lambda_2>$
8	$<5,6\lambda_1>$ $<6,2\lambda_1>$ $<29,\lambda_3>$ $<32,\lambda_3>$	$<12,\lambda_1>$ $<13,2\lambda_1>$ $<24,2\lambda_2>$ $<41,2\lambda_2>$
9	$<6,2\lambda_1>$ $<7,4\lambda_1>$ $<30,\lambda_3>$ $<34,\lambda_3>$ $<36,\lambda_3>$	$<12,\lambda_1>$ $<14,\lambda_1>$ $<15,\lambda_1>$ $<26,3\lambda_2>$ $<27,\lambda_2>$
10	$<6,2\lambda_1>$ $<31,\lambda_3>$ $<35,\lambda_3>$	$<13,\lambda_1>$ $<15,2\lambda_1>$ $<25,4\lambda_2>$
11	$<7,2\lambda_1>$ $<33,\lambda_3>$ $<37,\lambda_3>$	$<14,2\lambda_1>$ $<16,\lambda_1>$ $<28,4\lambda_2>$
12	$<8,\lambda_1>$ $<9,\lambda_1>$	$<29,3\lambda_2>$ $<30,2\lambda_2>$
13	$<8,2\lambda_1>$ $<10,\lambda_1>$	$<31,\lambda_2>$ $<32,4\lambda_2>$
14	$<9,\lambda_1>$ $<11,2\lambda_1>$	$<33,\lambda_2>$ $<34,4\lambda_2>$
15	$<9,\lambda_1>$ $<10,2\lambda_1>$	$<35,3\lambda_2>$ $<36,2\lambda_2>$

(continued)

State	Previous state and transition rate	Post state and transition rate
16	<11,λ_1>	<37,5λ_2>
17	<2,λ_2> <38,2λ_3>	<1,λ_3> <18,8λ_1> <19,4λ_1>
18	<3,2λ_2> <17,8λ_1> <39,2λ_3> <40,λ_3>	<2,λ_3> <20,3λ_1> <21,3λ_1> <22,3λ_1> <38,λ_2>
19	<4,2λ_2> <17,4λ_1> <40,λ_3>	<2,λ_3> <21,6λ_1> <23,3λ_1>
20	<5,3λ_2> <18,3λ_1> <42,λ_3>	<3,λ_3> <24,2λ_1> <39,2λ_2> <41,4λ_1>
21	<6,2λ_2> <18,3λ_1> <19,6λ_1> <42,λ_3> <43,2λ_3>	<3,λ_3> <24,2λ_1> <25,2λ_1> <26,2λ_1> <40,λ_2>
22	<6,λ_2> <18,3λ_1> <44,λ_3> <51,2λ_3>	<4,λ_3> <25,2λ_1> <27,2λ_1> <40,2λ_2> <41,2λ_1>
23	<7,3λ_2> <19,3λ_1> <44,λ_3>	<4,λ_3> <26,4λ_1> <28,2λ_1>
24	<8,2λ_2> <20,2λ_1> <21,2λ_1> <48,2λ_3>	<5,λ_3> <29,λ_1> <32,2λ_1> <42,2λ_2>
25	<10,4λ_2> <21,2λ_1> <22,2λ_1> <47,λ_3> <50,λ_3>	<6,λ_3> <32,λ_1> <35,λ_1> <36,λ_1> <43,2λ_2>
26	<9,3λ_2> <21,2λ_1> <23,4λ_1> <45,λ_3> <50,λ_3>	<6,λ_3> <29,λ_1> <34,λ_1> <35,λ_1> <44,λ_2>
27	<9,λ_2> <22,2λ_1> <46,2λ_3> <49,λ_3>	<7,λ_3> <30,λ_1> <33,λ_1> <36,λ_1> <44,3λ_2>
28	<11,4λ_2> <23,2λ_1> <49,λ_3>	<7,λ_3> <34,2λ_1> <37,λ_1>
29	<12,3λ_2> <24,λ_1> <26,λ_1>	<8,λ_3> <45,2λ_2>
30	<12,2λ_2> <27,λ_1> <41,λ_1>	<9,λ_3> <45,3λ_2> <46,λ_2>
31	<13,λ_2> <41,λ_1>	<10,λ_3> <47,4λ_2>
32	<13,4λ_2> <24,2λ_1> <25,λ_1> <41,λ_1>	<8,λ_3> <47,λ_2> <48,2λ_2>
33	<14,λ_2> <27,λ_1>	<11,λ_3> <49,4λ_2>
34	<14,4λ_2> <26,λ_1> <28,2λ_1>	<9,λ_3> <49,λ_2>
35	<15,3λ_2> <25,λ_1> <26,λ_1>	<10,λ_3> <50,2λ_2>
36	<15,2λ_2> <25,λ_1> <27,λ_1>	<9,λ_3> <50,3λ_2>
37	<16,5λ_2> <28,λ_1>	<11,λ_3>
38	<18,λ_2>	<17,2λ_3> <39,3λ_1> <40,6λ_1>
39	<20,2λ_2> <38,3λ_1>	<18,2λ_3> <42,4λ_1> <51,2λ_1>
40	<21,λ_2> <22,2λ_2> <38,6λ_1>	<18,λ_3> <19,λ_3> <42,2λ_1> <43,2λ_1> <44,2λ_1>
41	<8,2λ_2> <20,4λ_1> <22,2λ_1> <45,λ_3> <47,λ_3>	<6,λ_3> <30,λ_1> <31,λ_1> <32,λ_1> <42,2λ_2> <51,λ_2>
42	<24,2λ_2> <39,4λ_1> <40,2λ_1> <41,2λ_2>	<20,λ_3> <21,λ_3> <45,λ_1> <47,λ_1> <48,λ_1>
43	<25,2λ_2> <40,2λ_1>	<21,2λ_3> <48,λ_1> <50,2λ_1>
44	<26,λ_2> <27,3λ_2> <40,2λ_1>	<22,λ_3> <23,λ_3> <45,λ_1> <49,λ_1> <50,λ_1>
45	<29,2λ_2> <30,3λ_2> <42,λ_1> <44,λ_1>	<26,λ_3> <41,λ_3>
46	<30,λ_2> <51,λ_1>	<27,2λ_3>
47	<31,4λ_2> <32,λ_2> <42,λ_1> <51,2λ_1>	<25,λ_3> <41,λ_3>
48	<32,2λ_2> <42,λ_1> <43,λ_1>	<24,2λ_3>
49	<33,4λ_2> <34,λ_2> <44,λ_1>	<27,λ_3> <28,λ_3>
50	<35,2λ_2> <36,3λ_2> <43,2λ_1> <44,λ_1>	<25,λ_3> <26,λ_3>
51	<39,2λ_1> <41,λ_2>	<22,2λ_3> <46,λ_1> <47,2λ_1>

Table 7. State transition table of the multiprocessor sysytem model



Load(ρ)	Utilization(η_p)
0.050000001	0.970678091
0.100000001	0.911530435
0.200000003	0.785639346
0.300000012	0.693749726
0.400000006	0.631783068
0.500000000	0.588843763
0.699999988	0.534519076
1.000000000	0.490343213
2.000000000	0.435558975

Figure 10a. The utilization of a processor versus load

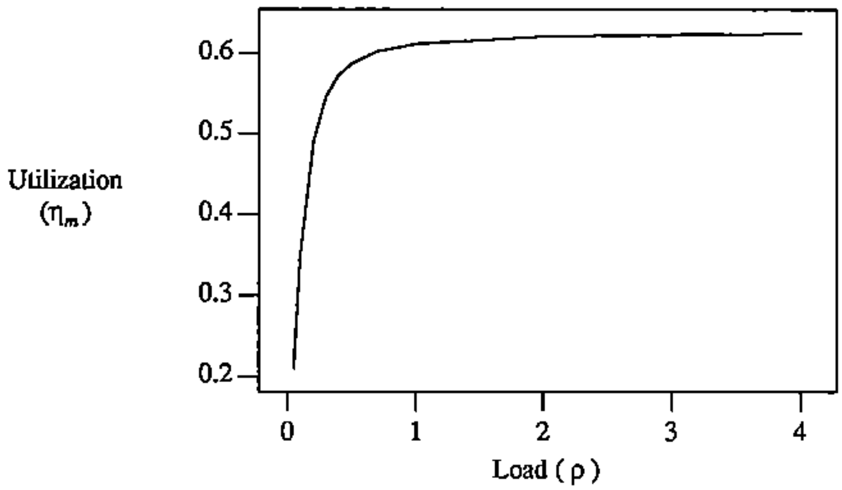


Figure 10b. The utilization of common memory versus load

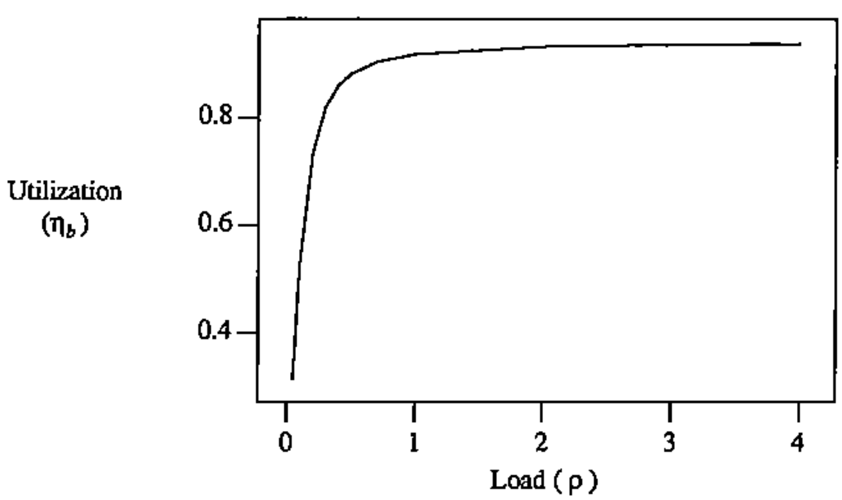


Figure 10c. The utilization of a bus versus load

P				Q
<p,2>	<p,3>	<p,4>	<p,5>	<p,1,1>
				<p,1,2>
				<p,1,3>
<p,1>	<p,3>	<p,4>	<p,5>	<p,2,1>
				<p,2,2>
				<p,2,3>
<p,1>	<p,2>	<p,4>	<p,5>	<p,3,1>
				<p,3,2>
				<p,3,3>
<p,1>	<p,2>	<p,3>	<p,5>	<p,4,1>
				<p,4,2>
				<p,4,3>
<p,1>	<p,2>	<p,3>	<p,4>	<p,5,1>
				<p,5,2>
				<p,5,3>

Table 8. The 15 individual markings (states) for places P and Q, corresponding to the compound marking defined as macro state 2 in Table 6.