THE UNIVERSITY OF MICHIGAN

COMPUTING RESEARCH LABORATORY

A SEMI-MARKOV MODEL FOR MEMORY INTERFERENCE IN MULTIPROCESSORS

T.N. Mudge, H.B. Al-Sadoun and B.A. Makrucki

CRL-TR-44-84

1079 East Engineering Bldg.
Ann Arbor, MI 48109

A SEMI-MARKOV MODEL FOR MEMORY INTERFERENCE IN MULTIPROCESSORS

by

T. N. Mudge, Senior Member, IEEE[†],
Humoud B. Al-Sadoun, Student Member, IEEE[†],

and

B. A. Makrucki, Member, IEEE[‡]

Abstract

This paper presents a discrete time model of memory interference in multiprocessors. The model, termed the semi-Markov memory interference model, explicitly describes each processing element's behavior by means of a semi-Markov process. The model requires as input the number of processing elements and the number of memory modules in the multiprocessor, the mean think time of the processing elements, the first and second moments of the connection time between processing elements and memories, and the probability mass function characterizing the destination of processing element's requests for memories. The model produces as output the memory bandwidth, processing element utilization, memory module utilization, average queue length at a memory and average waiting time experienced by a processing element while waiting to access a memory. Thus, it is possible to analyze the interaction of variable connection time, think time and the distribution of the destination of the memory requests on the system performance. This modeling capability is attained without having to employ a complex Markov chain. Indeed, the number of states in the semi-Markov process describing a processing element is dependent only on the probability mass function describing the destination of the memory requests. For instance, in the simplest and most common case when requests are directed to each memory module equiprobably, a four state semi-Markov process is sufficient regardless of the think and connection time distributions. The accuracy and capability of the model is illustrated with three examples.

Index terms--Memory interference, multiprocessors, memory bandwidth, performance evaluation, semi-Markov processes, Markov chains.

This work was supported in part by National Science Foundation under Grant MCS-8009315.

[†] Computing Research Lab, EE&CS Department, The University of Michigan, Ann Arbor, MI 48109.

[‡] AT&T Bell Laboratories, Holmdel, NJ 07733.

1. Introduction

In a multiprocessor system main memory may consist of a number of memory modules connected with processing elements through an interconnection network. Processing elements may access any memory module. Two types of memory conflict, or memory interference, can occur [1]. Type one conflicts arise when several processing elements attempt to access an idle memory module simultaneously. In this situation one of the processing elements is selected, according to a predefined selection strategy, to access the memory module while the other processing elements wait until the selected processing element is done. Type two conflicts arise when one or more processing elements attempt to access a busy memory module. In this situation the processing elements wait until the memory module becomes idle before they attempt again an access. Both types of conflicts have a negative effect on the overall performance of the multiprocessor system by, among other things, reducing the memory bandwidth and increasing the average queueing time for memory.

This paper introduces a discrete time model based on a widely accepted set of assumptions that characterize multiprocessor behavior as a stochastic process; see [1]-[12]. The model describes the behavior of each processing element as a semi-Markov process. The model, termed the semi-Markov memory interference (SMI) model, can be used to determine memory interference effects on the multiprocessor system's performance. The system of interest, depicted in Figure 1, is a multiprocessor system which has N processing elements (PE's) and M memory modules (MM's). The processing elements communicate with the memory modules through what can be regarded as an $N \times M$ crossbar interconnection network. The whole system is synchronized with a system clock whose period is referred to as the "system cycle," or, where context allows, simply the "cycle."

[1] In this paper we will not consider conflicts that may arise in the interconnection network due to connectivity limitations, i.e., a virtual crossbar interconnection is assumed.

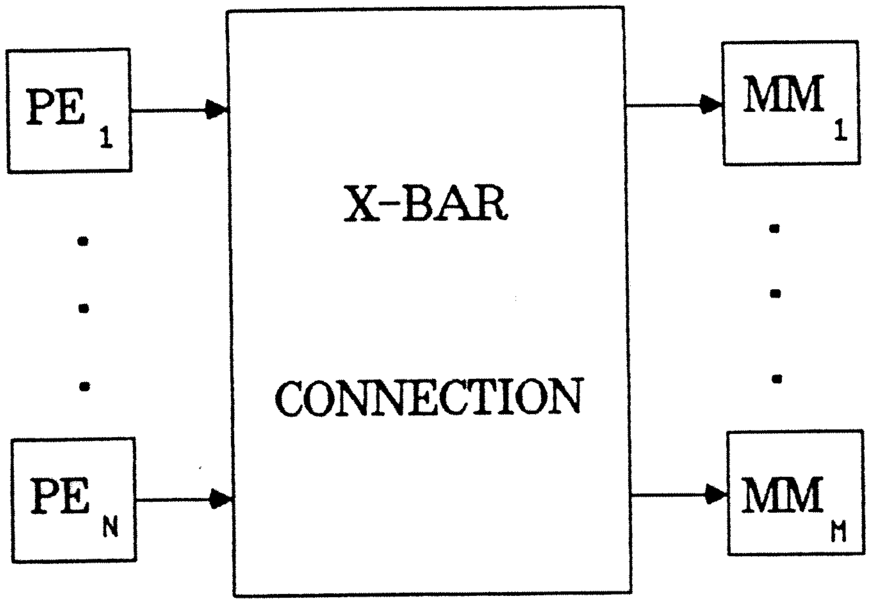


Figure 1. Block diagram of a multiprocessor system.

The literature contains a number of discrete time memory interference models for such multiprocessor systems. In most of these discrete time models, system operation is characterized as follows. At the beginning of the system cycle a processing element, which has no pending request, makes a request to access a memory module with probability r. The memory module is chosen at random from the set of memory modules with probability $1/M$. If a type one conflict occurs at the memory module, that memory selects, with equal probability, one of the conflicting processing elements to access it. The processing elements that are not selected attempt again to access the same memory in the next system cycle. This retry will generally occur in the presence of new requests for access. The connection between the processing element and the memory module lasts for one system cycle; at the end of this cycle the processing element releases the memory module. Only type one conflicts take place under these assumptions. A processing element will have at most one pending request waiting for an access at any time. The behavior of the processing elements are considered to be independent and statistically identical.

The behavior of the multiprocessor system, described above, can be viewed as a stochastic process. Markov chains have been used to describe this process; see [1] and [4]. The drawback of this approach is the unmanageably large state space of the resulting Markov chains: even for moderately sized systems the state space size is enormous. This obstacle led to the development of models in which further simplifying assumptions were adopted; see [2]-[9], [11]. One of the main themes of these approximate models is the reduction of the state space while maintaining reasonable agreement with simulation results. In [9] a classification was proposed in which the approximate models were categorized according to the approach used in their formulations. These classes are: probabilistic models, typified by the models in [2], [3], [7], [8], [11]; rate-adjusted models, typified by the model in [5]; queueing system models, typified by the models in [4], [6]; and finally, the steady state flow models, typified by the model in [9].

Some of the approximate models have been extended to cover the case in which memory requests are not sent equiprobably to every memory module; see [5], [8], [9], [11]. In these extensions the destination of the request originating from a processing element is chosen according to a probability mass function. Another extension is reported in [4] and [5]. In these models the description of processing element behavior includes *a think time,* i.e., the time elapsed between releasing a connection with a memory module and requesting the next connection with any memory module. Think times are specified with a probability mass function. Think times need not be geometrically distributed as is implicitly assumed in the original system operation assumptions outlined above. The study in [4] concluded that the memory bandwidth of the multiprocessor system will be affected primarily by the first moment of the think time but will be relatively insensitive to higher moments of the distribution. Such conclusions were deduced after examining the results of trace driven simulations in which six different distribution functions were used to characterize the think time. A third extension of system operation is introduced in [10] and [12] where the connection time between a processing element and a memory module can last for more than the single system cycle. In [10] it is assumed that the connection will be held for a fixed number of system cycles. This extension was motivated by cache memory models in which the cache line is a fixed size block. The study in [12] allowed variable connection times between processing elements and memory modules and showed the effect of variance in the connection times, an important consideration if cache coherency checks are to be modeled. In [12] a Markov chain model was used to describe the behavior of the processing element.

The SMI model consolidates the modeling capabilities of all the above models into one model. It further extends these capabilities by providing expressions for the average length of memory request queues and the average waiting time experienced by a processing element attempting to access a memory. Thus, for the first time it is possible to analyze the interaction of variable connection time, arbitrary think time distribution and the distribution of the destination of the memory requests on the system performance. This additional modeling capability is attained without having to employ a complex Markov chain as in [12]. Indeed, the number of states in the semi-Markov process describing a processing element is dependent only on the probability mass function describing the destination of the memory requests. For instance, in the simplest and most common case when requests are directed to each memory module equiprobably, a four state semi-Markov process is sufficient regardless of the think and connection time distributions. Furthermore, the SMI model explicitly describes each processing element's interaction with the memory modules and can therefore provide direct information about processing element behavior.

In addition to the above discrete time models, continuous time models have been proposed. These are typified by the models reported in [13] and [14]. In these models connection times were approximated as exponentially distributed random variables. Processing element's think times were also described as exponentially distributed random variables. The multiprocessor system considered in these models has a different structure than the one adopted by the discrete time models. The continuous time models considered the case of a multiple bus interconnection network rather than a crossbar. Such structures have been studied in the discrete time environment in [15] using simulation results. Recently [16]-[18] proposed discrete time models to study such structures. Continuous time models are usually less accurate than discrete time models when discrete time events are being modeled; see [19] and [20] for discussions. Furthermore, the approximation of the connection and think times as exponentially distributed random variables limits application to those systems where the exponential approximation holds.

The paper is organized as follows: Section 2 describes the assumptions that characterize the operation of the multiprocessor system; Section 3 defines the performance measures that can be obtained from the model; Section 4 develops the SMI model under the assumptions of Section 3 and illustrates it with two examples; Section 5 generalizes the assumptions, extends the SMI model accordingly and illustrates it with one example; Section 6 concludes the paper.

2. The System Operation Assumptions

A processing element in the multiprocessor system, shown in Figure 1, may be in any of three states: *thinking*, when it is working on an internal task with no memory request outstanding; *accessing*, when it is connected to a memory module; and *waiting*, when it is waiting in the queue of a memory module for that memory to become available. The memory module can be in any of two states: *busy*, when a processing element is connected to it; and *idle*, when there is no processing element connected to it. The following notation will be used throughout this paper: the i^{th} processing element will be denoted by PE_i; the j^{th} memory module will be denoted by MM_j; a discrete random variable will be denoted by its name with a ˜ above it, e.g., the discrete random variable Z will be donated by $\tilde{Z}$; the cumulative distribution function (CDF) of $\tilde{Z}$ will be denoted by $Z(x)$, i.e., $Z(x) = Pr[\tilde{Z} \leq x]$; the probability mass function (pmf) of $\tilde{Z}$, will be denoted by $z(x)$, i.e., $z(x) = Pr[\tilde{Z} = x]$; the mean value of $\tilde{Z}$ will be denoted by $\bar{Z}$; and the n^{th} moment of $\tilde{Z}$ will be denoted as $\overline{Z^n}$.

System operation will be characterized by the following assumptions:

Ia. The behavior of the PE's can be modeled as identical stochastic processes.

IIa. The PE's think for an integer number of system cycles. The thinking period of any PE is characterized by a discrete independent random variable, $\tilde{T}$.

IIIa. Each PE will submit a memory request after its thinking period; requests originating from the same processing element are independent of each other. The destination memory module of the request originated from any PE will be determined by a discrete independent random variable, $\tilde{D}$, which is uniformly distributed between 1 and M.

IVa. When the first type of memory conflict occurs, the memory module selects, equiprobably, one of the conflicting processing elements to gain access. The blocked processing element(s) wait until the connection is completed and then they resubmit their requests to the same memory module.

Va. When the second type of memory conflict occurs, the blocked processing element(s) wait until the connection is completed and then they resubmit their requests to the same memory module.

VIa. The connection time between any processing element, *PE*, and any memory module, *MM*, is characterized by a discrete independent random variable, $\tilde{C}$.

Empirical evidence reported in [3]-[5], supports the assumptions in the case where $\tilde{C}$ is a deterministic random variable with a value of one. Further work reported in [21] supports the assumptions in the more general case where $\tilde{C}$ is a discrete random variable with arbitrary distribution. The assumptions Ia through VIa will be referred to as the uniform case assumptions because of the uniform distribution of $\tilde{D}$. The model and examples in Section 4 will consider this case. In Section 5 these assumptions will be relaxed and a more general case studied. In particular, each *PE* will have its own distribution for the random variables $\tilde{T}$ and $\tilde{D}$, and each *PE* and *MM* pair will have its own distribution for the random variable $\tilde{C}$.

In order to obtain numerical information from the SMI model developed later, the values of M, N, $\overline{T}$, $\overline{C}$ and $\overline{C^2}$, must be obtained through measurements or by hypothesis. These quantities can be regarded as input parameters of the SMI model; knowledge of the full distributions of $\tilde{T}$ and $\tilde{C}$ is not necessary for solving the SMI model.

3. Performance Measures

A number of performance measures can be derived from the SMI model. These measures are: the memory bandwidth, BW; the i^{th} processing element utilization, PU_i; the j^{th} memory module utilization, MU_j; the average queue length of the j^{th} memory module queue, L_j; and the average waiting time experienced by the i^{th} processing element in the j^{th} memory module queue, W_{ij}.

- The memory bandwidth, BW, is defined as the average number of busy memory modules when the multiprocessor system reaches steady state. This is the same as the average number of accessing processing elements when the multiprocessor system reaches steady state. Hence, BW can be expressed as follows:

$$BW = \lim_{t\to\infty} \left[\sum_{i=1}^{N} Pr[PE_i \textit{ is accessing at time } t] \right]$$

- The i^{th} processing element utilization, PU_i, is the probability that the i^{th} processing element is thinking or accessing a memory module when the multiprocessor system reaches steady state. Hence, PU_i can be expressed as follows:

$$\begin{aligned} PU_i &= \lim_{t\to\infty} Pr[PE_i \textit{ is thinking or accessing at time } t] \\ &= 1 - \lim_{t\to\infty} Pr[PE_i \textit{ is waiting at time } t] \end{aligned}$$

  Some of the memory interference models, [10] and [22], which were motivated by cache studies of multiprocessor systems, define PU_i as the probability that the i^{th} processing element is thinking when the multiprocessor system reaches steady state. In this study, we consider that memory accessing contributes to the progress of the computation and is therefore counted as useful work. The alternative quantity can be readily derived from the SMI model if it is required, see Example 1.

- The j^{th} memory module utilization, MU_j, is the probability that the j^{th} memory module is busy when the multiprocessor system reaches steady state. This is the same as the probability that any processing element is accessing the j^{th} memory module when the multiprocessor system reaches steady state. Hence, MU_j can be expressed as follows:

$$MU_j = \lim_{t\to\infty} \sum_{i=1}^{N} Pr[PE_i \textit{ is accessing } MM_j \textit{ at time } t]$$

The memory bandwidth, BW, can be expressed in terms of the memory utilizations as follows:

$$BW = \sum_{j=1}^{M} MU_j$$

- The average queue length at the j^{th} memory module, L_j, can be defined as the expected number of processing elements waiting to access the j^{th} memory module. Hence, L_j can be expressed as follows:

$$L_j = \sum_{i=1}^{N} \lim_{t\to\infty} Pr[PE_i \textit{ is waiting to access } MM_j]$$

- The average waiting time experienced by the i^{th} processing element while waiting to access the j^{th} memory module, is denoted by W_{ij}.

As will be seen the last two measures fall out naturally from the SMI model.

4. The Semi-Markov Memory Interference (SMI) Model

A Markov chain which models a multiprocessor system according to the assumptions outlined in Section 2 has an unmanageably large state space, see [1] and [4]. To simplify this we first adopt a technique presented in [12]. In that work separate identical Markov chains are used to describe the behavior of each *PE*, and the coupling between the N chains appears in the transition probabilities between the states in each chain. Solving the model requires only one of the chains to be considered which dramatically reduces the solution complexity. Moreover, because the chains are coupled, independence of *PE*'s does not have to be assumed, resulting in a more realistic model (assumption Ia does not imply independence). The number of states in the model of [12] can still grow large, in some cases, because it depends on the number of discrete values $\tilde{T}$ and $\tilde{C}$ can take on. This can be avoided, resulting in a further simplification, by replacing the Markov chains by semi-Markov processes. These only have four states for the uniform case regardless of the distributions for $\tilde{T}$ and $\tilde{C}$. In addition, the semi-Markov processes permit computation of the average queue length at each *MM* and the average waiting time experienced by a *PE*.

A detailed discussion of semi-Markov processes can be found in [23]-[25]. Briefly, a semi-Markov process (SMP) is a stochastic process which can be in any one of K states $1, 2, \ldots, K$. Each time it enters state i it remains there for a random amount of time (the sojourn time) having mean η_i and then makes a transition into state j with probability p_{ij}. As a special case, a discrete time Markov chain is an SMP with a deterministic sojourn time of value one. If the SMP has an irreducible embedded Markov chain that consists of ergodic states, then the limiting probability of being in state i, denoted by P_i, can be expressed as follows:

$$P_i = \frac{\pi_i \ \eta_i}{\sum_{j=1}^{K} \pi_j \ \eta_j} \tag{1}$$

where π_i is the limiting probability of state i in the embedded Markov chain. All the SMP's that appear in this paper have irreducible embedded Markov chains with ergodic states, therefore, equation (1) will always be applicable. The rate of leaving state i, λ_i, is defined as the reciprocal of the average time elapsed between two consecutive departures from state i. The rate can be obtained using the following equation:

$$\lambda_i = \frac{P_i}{\eta_i} = \frac{\pi_i}{\sum_{j=1}^{K} \pi_j \ \eta_j} \tag{2}$$

Since the average sojourn time in any one of the states of the SMP's that appear in this paper is at least one system cycle, then λ_i falls in the range [0,1] and it is possible to view λ_i as the probability of leaving state i at the beginning of a system cycle.

As mentioned above the SMI model uses an SMP to approximate the behavior of a *PE* which functions according to the system operation assumptions outlined in Section 2, i.e., the uniform case assumptions. Therefore, *N* SMP's will approximate the behavior of the multiprocessor system. The states of the SMP denote the different states of any *PE*, and they can be partitioned to four disjoint subsets [2]. The first subset is the ***thinking subset***, $S^{th} = \{0\}$. The process enters state 0 and remains there for a duration of time with mean value η_0, equivalent to the thinking time of the *PE* (see Figure 2). A memory request is modeled by the SMP leaving state 0. The destination state depends on the state of the requested *MM*. The second subset is the ***accessing subset***, $S^{ac} = \{1\}$. The process enters state 1 and remains there for a duration of time with mean value η_1, equivalent to the connection time between the *PE* and any *MM*. From state 1 the process returns to state 0, i.e., the *PE* resumes thinking after it has completed its memory access. The third subset is the ***full waiting subset***, $S^{fw} = \{2\}$. The process enters state

[2] For the moment these subsets are singletons. Later generalizations increase their cardinality.

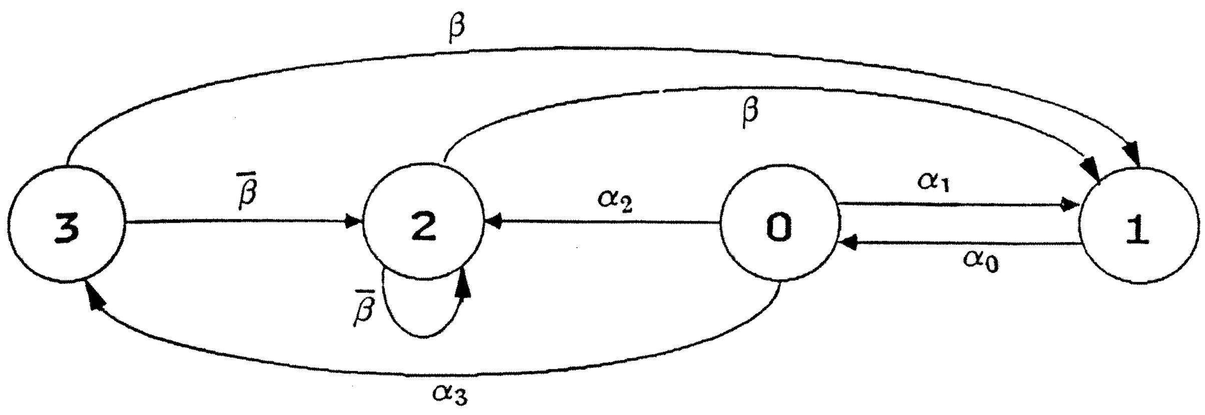


Figure 2. SMP that describes PE behavior with the uniform case assumptions.

2 when the *PE* requests an idle *MM* simultaneously with at least one other request, and the *PE* fails to be selected by the module, i.e., a type one conflict occurs and another *PE* is selected to have access to the *MM*. In this case the *PE* has to wait for the full duration of the connection time between the *MM* and the selected *PE*; this duration has a mean value of η_2. The original *PE* will retry to access the same *MM* when the selected *PE* releases the module. If it succeeds, the process enters state 1, otherwise the process reenters state 2. The fourth subset is the *residual waiting subset*, $S^{rw} = \{3\}$. The process enters state 3 when the *PE* requests a busy *MM*, i.e., when a type two conflict occurs. The *PE* has to wait for the remaining (residual) connection time before retrying to access the *MM*; the mean value for the residual time is η_3. The process then enter state 1 if the *PE* succeeds in accessing the *MM*, or it enters state 2 if it fails to obtain a connection. Clearly, the SMP description does not include which module the *PE* is accessing or which module the *PE* is waiting to access. This does not represent an approximation of the *PE*'s behavior because of the symmetry in the uniform case. In non-symmetric cases, as will be shown in the next section, the SMP has to represent this

information. The underlying approximation of the SMI model is in describing any *PE* behavior independently from the other *PE*'s while compensating for the coupling between the *PE*'s behaviors in the transition probabilities between the states of the SMP (the coupling results from the *PE*'s sharing the *MM*'s).

In order to derive numerical information from the SMI model, the values of N, M, the first moment of $\tilde{T}$, and the first two moments of $\tilde{C}$ must be obtained through measurement or, if it is considered satisfactory, by hypothesis. These quantities can be regarded as the input parameters to the model. These parameters are defined as follows:

$$\begin{aligned} N &\triangleq \textit{the number of PE's} \\ M &\triangleq \textit{the number of MM's} \\ \overline{T} &\triangleq \textit{the first moment of } \tilde{T} \\ \overline{C} &\triangleq \textit{the first moment of } \tilde{C} \\ \overline{C^2} &\triangleq \textit{the second moment of } \tilde{C} \end{aligned}$$

The average sojourn times, of the different states of the SMP, can be obtained from the parameters of the model as follows:

$$\eta_j = \begin{cases} \overline{T} & j = 0 \\ \overline{C} & j = 1 \\ \overline{C} & j = 2 \\ \dfrac{\overline{C^2} - \overline{C}}{2\,(\overline{C} - 1)} & j = 3 \end{cases} \tag{3a}$$

The average sojourn times in the states 0, 1 and 2 arises directly from the definition of these states. The average sojourn time in state 3 is obtained by using the following theorem.

Theorem 1:

The average residual accessing time of a busy memory seen by a requesting *PE* at the beginning of a system cycle, η_3, can be expressed as $\dfrac{\overline{C^2} - \overline{C}}{2(\overline{C} - 1)}$, where $\overline{C}$ and $\overline{C^2}$ are the first and the second moments of the connection time, $\tilde{C}$, respectively.

Proof:

The definition of probability mass function of $\tilde{C}$, given in Section 2, states that:

$$c(i) = Pr[\tilde{C} = i]$$

Two events, A_i and B, will be used in this proof. They are defined as follows: A_i is the event that a requesting *PE* will see a residual accessing time of i cycles; B is the event that a requesting *PE* will find a busy *MM*. Therefore, the average residual accessing time of a busy memory seen by a requesting *PE*, η_3, can be expressed as follows:

$$\eta_3 = \sum_{i=1}^{S-1} i \, Pr[A_i \,|B]$$

Where S is the maximum connection time between a *PE* and any *MM*. Since A_i and B are dependent events, Bayes' rule can be used to obtain the following equation:

$$Pr[A_i \,|B] = \frac{Pr[A_i \bigcap B]}{Pr[B]}$$

The term $Pr[A_i \bigcap B]$ is determined as follows:

$$Pr[A_i \bigcap B]$$

$$= \sum_{j=1}^{S-i} Pr[\textit{the accessing PE submitted, } j \textit{ cycles ago, an accessing request of } i{+}j \textit{ cycles}]$$

$$= \sum_{j=1}^{S-i} c(i+j) = \sum_{j=i+1}^{S} c(j)$$

While the term $Pr[B]$ is determined as follows:

$$Pr[B]$$
$$= \sum_{j=1}^{S-1} Pr[\textit{the accessing PE obtained the connection, } j \textit{ cycles ago, } \text{for } \textit{at least } j+1 \textit{ cycles}]$$
$$= \sum_{j=1}^{S-1} \sum_{k=j+1}^{S} c(k) = \sum_{j=1}^{S} (j-1)\, c(j)$$

Therefore, η_3 can be expressed as follows:

$$\eta_3 = \sum_{i=1}^{S-1} i \frac{\sum_{j=i+1}^{S} c(j)}{\sum_{j=1}^{S} (j-1)\, c(j)} = \frac{\sum_{i=1}^{S-1} i \sum_{j=i+1}^{S} c(j)}{\sum_{j=1}^{S} (j-1)\, c(j)} = \frac{\sum_{j=1}^{S} \frac{j\,(j-1)}{2}\, c(j)}{\sum_{j=1}^{S} (j-1)\, c(j)}$$

$$= \frac{\overline{C^2} - \overline{C}}{2\,(\overline{C} - 1)} \qquad \textit{Q.E.D.} \ \square$$

It is convenient to introduce three terms that will be used in formulating the model. These terms are: R, *WIN* and *BUSY*. The term R is defined as the probability that a *PE* makes a request to access a particular *MM* at the beginning of a system cycle. Hence, it is the probability that one of two events occurs at the beginning of a system cycle. The first event is that the *PE* will direct a new request to that *MM*, i.e., the processor leaves state 0. And the second event is that the *PE* will resubmit a previously blocked request to access that particular *MM*, i.e., it leaves state 2 or 3. Therefore, R is expressed as follows:

$$R = \frac{1}{M} \left(\lambda_0 + \lambda_2 + \lambda_3 \right)$$

The term *WIN* is the probability that an idle *MM* selects the *PE*'s request over other requests (if there are any) at the beginning of a system cycle. The term *WIN* is derived by the following argument: the probability that a *PE* will not request a particular *MM* is $1 - R$; the probability that none of the N *PE*'s request that *MM* is $(1 - R)^N$; therefore the probability that a particular *MM* is requested by at least one of the *PE*'s is $[1 - (1 - R)^N]$; the expected number of *PE*'s which requested that *MM* at the beginning of a system cycle is $N\,R$. Therefore, *WIN* can be expressed as follows:

$$WIN = \frac{1}{NR}\left[1 - (1-R)^N\right] \tag{4a}$$

Finally, the term *BUSY* is defined as the probability that a *PE* finds a particular *MM* busy at the beginning of a system cycle. Therefore, one of the other (N-1) PE's is accessing that *MM* and is not on the point of releasing it. In other words, the requesting *PE* experienced a memory conflict of type one. Hence, *BUSY* is the probability that one of (N-1) PE's is accessing a particular *MM* and the accessing *PE* is not on the point of releasing the *MM*. Thus, *BUSY* can be expressed as follows:

$$BUSY = \frac{N-1}{M}\,(P_1 - \lambda_1) = \frac{N-1}{M}\,(\bar{C}-1)\,\lambda_1 \tag{5a}$$

The last step follows from equation (2).

The transition probabilities between the states of the SMP can be derived as the following functions of *BUSY* and *WIN*:

$$\alpha_j = \begin{cases} 1 & j=0 \\ (1-BUSY)\,WIN & j=1 \\ (1-BUSY)\,(1-WIN) & j=2 \\ BUSY & j=3 \end{cases} \tag{6a}$$

$$\beta = (1-BUSY)\,WIN$$
$$\bar{\beta} = 1-\beta$$

The derivation proceeds as follows. When the process, shown in Figure 2, leaves the thinking state it enters any one of the other states: it enters the accessing state with probability α_1 if the *MM* is idle and the *PE*'s request is selected; or it enters the full waiting state with probability α_2 if the *MM* is idle and the *PE*'s request fails to be selected; or it enters the residual waiting state with probability α_3 if the *MM* is busy. The process always enters the thinking state after it leaves the accessing state ($\alpha_0 = 1$). The process leaves the residual waiting state or the full waiting state to enter the accessing state with probability β if the requested MM is idle and the PE's request is selected; otherwise it will enter the full waiting state with probability $\bar{\beta}$. (Although α_1 and β are equal we distinguish them in preparation for the general case discussed in the next section.)

The embedded Markov chain can be solved and the π's can be represented as functions of the transition probabilities, i.e., of *BUSY* and *WIN*. Then, from equation (2) λ_1 may be defined as a function of R and *WIN*. Therefore, using equation (5a) the term *BUSY* can be expressed as follows:

$$BUSY = \frac{(N-1)(\bar{C}-1)\, WIN\, R}{1 + (N-1)(\bar{C}-1)\, WIN\, R} \tag{7a}$$

The SMP limiting probabilities can be derived by substituting the limiting probabilities of the embedded Markov chain (π's) into equation (1). Therefore, the SMP limiting probabilities can be expressed as functions of R and the transition probabilities as shown below:

$$P_j = \begin{cases} \eta_0\, M\, \beta\, R & j = 0 \\ \eta_1\, M\, \beta\, R & j = 1 \\ \left[\alpha_2 + \dfrac{(\bar{\beta})^2}{\beta}\right] \eta_2\, M\, \beta\, R & j = 2 \\ \alpha_3\, M\, \eta_3\, \beta\, R & j = 3 \end{cases} \tag{8a}$$

It can be seen from the above equations that we have a set of non-linear equations to be solved. The non-linearity is introduced because the transition probabilities are defined as functions of the SMP's limiting probabilities; meanwhile, the SMP's limiting probabilities are defined as functions of the transition probabilities. An iterative algorithm can be used to solve these equations. The algorithm will iterate on the value of R and then the performance measures of the system can be derived. The algorithm breaks down as follows:

1. Calculate the average sojourn times of the states using equation (3a).

2. Choose an initial value for R in the range $0 < R < 1$ (in our experiment $R = 0.5$ was used).

3. Calculate the terms *WIN* and *BUSY* using equations (4a) and (7a) respectively.

4. Calculate the transition probabilities using equation (6a).

5. Calculate new value for R by summing the four equations of equation (8a) to one. Then R can be expressed as follows:

$$R = \frac{1}{\left[\eta_0 + \left[1 + \alpha_2 + \frac{(\bar{\beta})^2}{\beta} \right] \eta_1 + \alpha_3 \, \eta_3 \right] \; M \, \beta}$$

6. Repeat steps 3 through 5 until R has the desired accuracy[3].

The solution for R may be used to calculate the limiting probabilities of the states using equation (8a). These can in turn be used to calculate the performance measures of Section 3, as follows:

$$\begin{aligned} BW &= N \, P_1 \\ PU_i &= P_0 + P_1 \\ MU_j &= \frac{N}{M} \; P_1 \\ L_j &= \frac{N}{M} \; \left(P_2 + P_3 \right) \\ W_{ij} &= \eta_2 \left[\frac{1}{\beta} - (1 + BUSY) \right] + \eta_3 \, BUSY \end{aligned}$$

where $1 \leq i \leq N$ and $1 \leq j \leq M$. The last equation is the only one that does not follow directly from the definition of the states of Figure 2. It can be derived by calculating the expected value of $\tilde{W}_{ij}$ in the usual way from the pmf of $\tilde{W}_{ij}$. The pmf of $\tilde{W}_{ij}$ can be expressed as follows:

[3] This is a simple fixed-point iteration scheme. Higher order iterations scheme could be used but were found unnecessary in the experiments discussed in this paper. Four iterations were usually sufficient.

$$\begin{aligned}
Pr[\tilde{W}_{ij} = 0] &= \alpha_1 & \\
Pr[\tilde{W}_{ij} = k\eta_2] &= \alpha_2 (\bar{\beta})^{k-1} \beta & k \geq 1 \\
Pr[\tilde{W}_{ij} = (\eta_3 + k\eta_2)] &= \alpha_3 (\bar{\beta})^{k} \beta & k \geq 0
\end{aligned}$$

The derivation of the above equations proceeds as follows. The probability that the process moves from state 0 to state 1 without waiting is α_1. The probability that the process moves from state 0 to state 1 and makes k visits to state 2 is $\alpha_2 (\bar{\beta})^{k-1} \beta$, where $k \geq 1$. The probability the process moves from state 0 to state 1 and makes one visit to state 3 and k visits to state 2 is $\alpha_3 (\bar{\beta})^{k} \beta$, where $k \geq 0$. These three cases exhaust all the possible values for $\tilde{W}_{ij}$.

As a final note, it can be shown that, when $\bar{C} = \overline{C^2} = 1.0$, the SMI model reduces to the model of [5]. In other words, the SMI model can be considered to be a generalization of the rate adjusted models.

Example 1:

In this example, we will use the SMI model to study a multiprocessor system with private cache memories. The multiprocessor system is shown in Figure 3. Each *PE* in the system consists of a processor and a private cache memory. The *MM*'s form the shared memory. The system operation is characterized as follows. At the end of a system cycle a processor causes a cache fault in its private cache with probability m. The *PE* which has a cache fault chooses, with equal probability, one of the *MM*'s with which to transfer a line. When the connection is established between the *PE* and the *MM* it lasts for a variable number of cycles given by a discrete random variable, $\tilde{C}$. The variability arises because cache coherency requires reads to be performed in some transfers and writes before reads in others. In the case of a memory conflict, of either type, the rejected *PE*'s will resubmit their requests to the same *MM* when that *MM* becomes idle. This retry will generally occur in the presence of new requests for access.

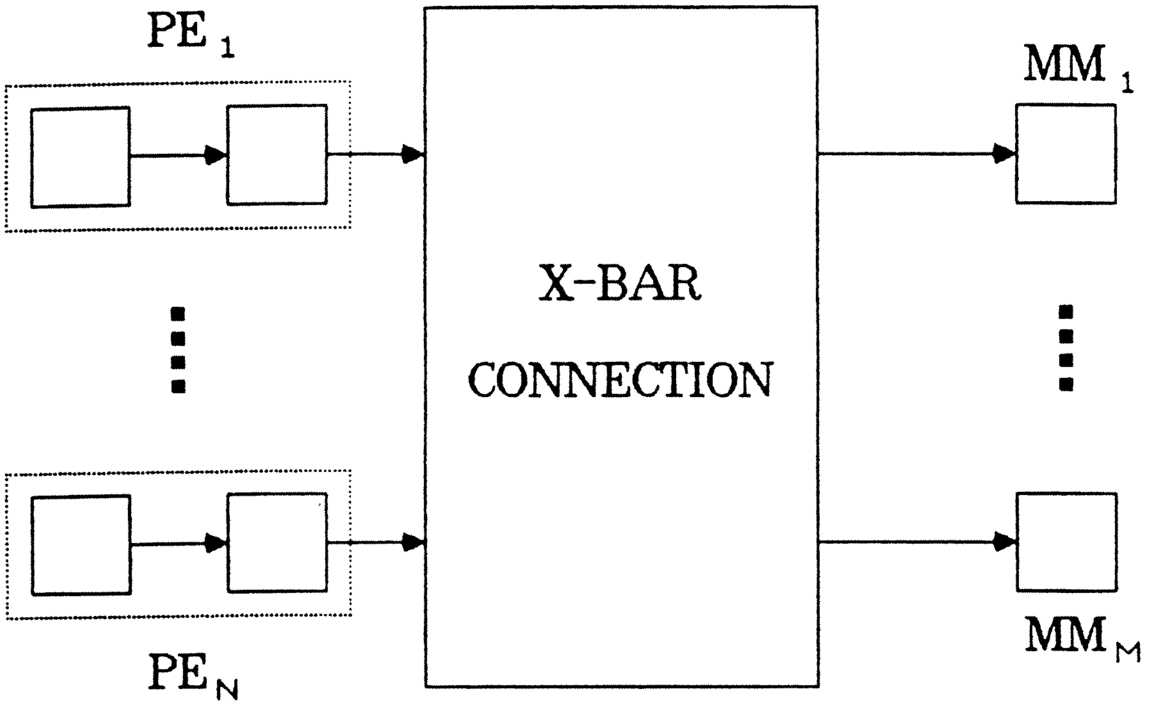


Figure 3. The multiprocessor system with private cache memories.

A study, reported in [10], models a similar multiprocessor system in which the random variable $\tilde{C}$ is deterministic and cache coherence is ignored. A later study, reported in [22], develops a more realistic model which includes the effects of coherence checks. However, this model assumes a parallel-pipelined organization of memories rather than the parallel organization of Figure 3. Nevertheless, as a result of considering cache coherency the work of [22] demonstrates the need to model the connection time as a non-deterministic random variable rather than a deterministic random variable, although it does not develop a model suitable for connection times having a high degree of non-determinism (i.e., a large coefficient of variation[4], C_v). Our results below confirm the need to consider non-zero values for C_v.

[4] $C_v = \frac{\textit{standard deviation of } \tilde{C}}{\textit{expected value of } \tilde{C}} = \sqrt{\frac{\overline{C^2}}{(\overline{C})^2} - 1}$ where $\overline{C}$ and $\overline{C^2}$ are the first and second moments of $\tilde{C}$ respectively.

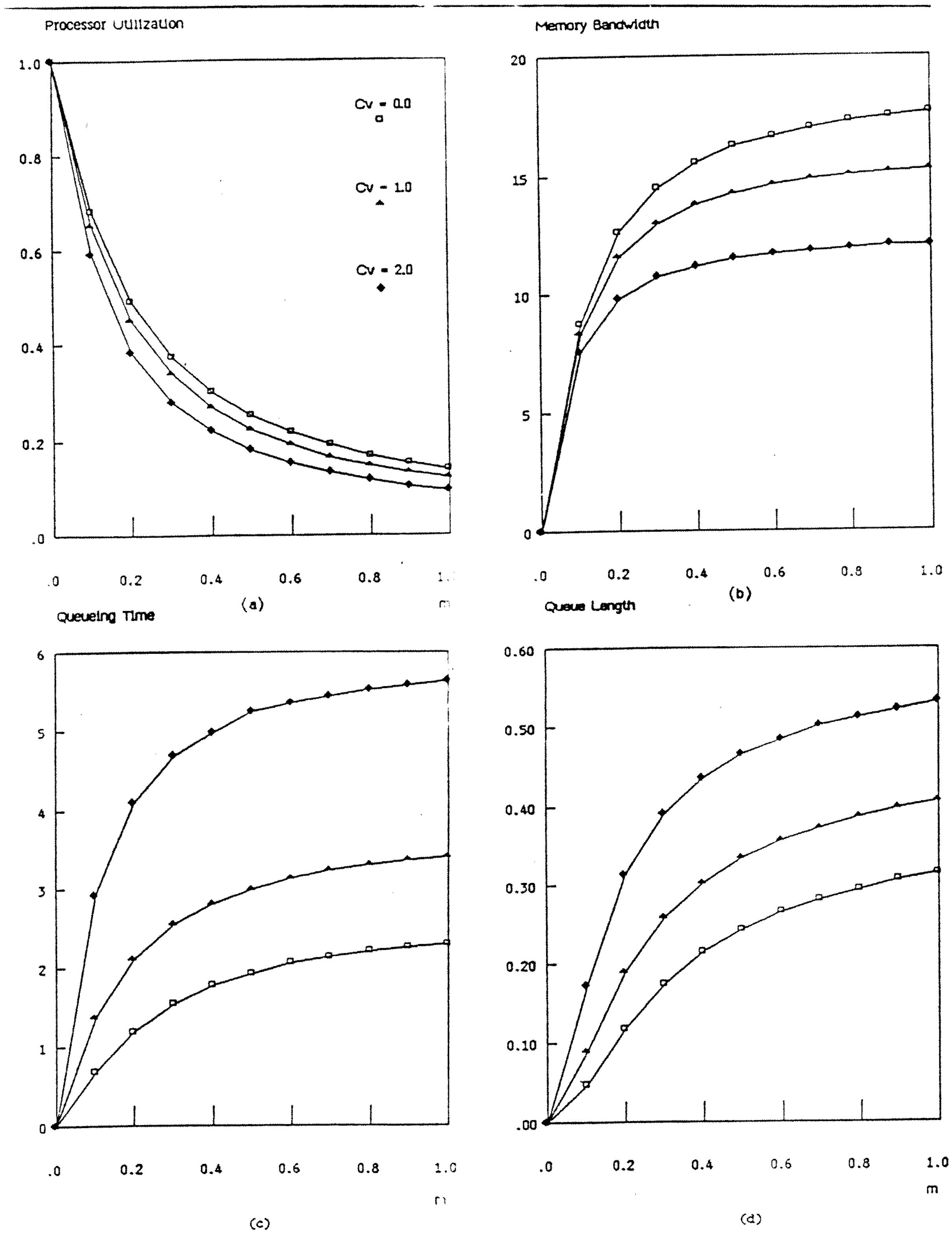


Figure 4. Simulation results for a 32x32 system.

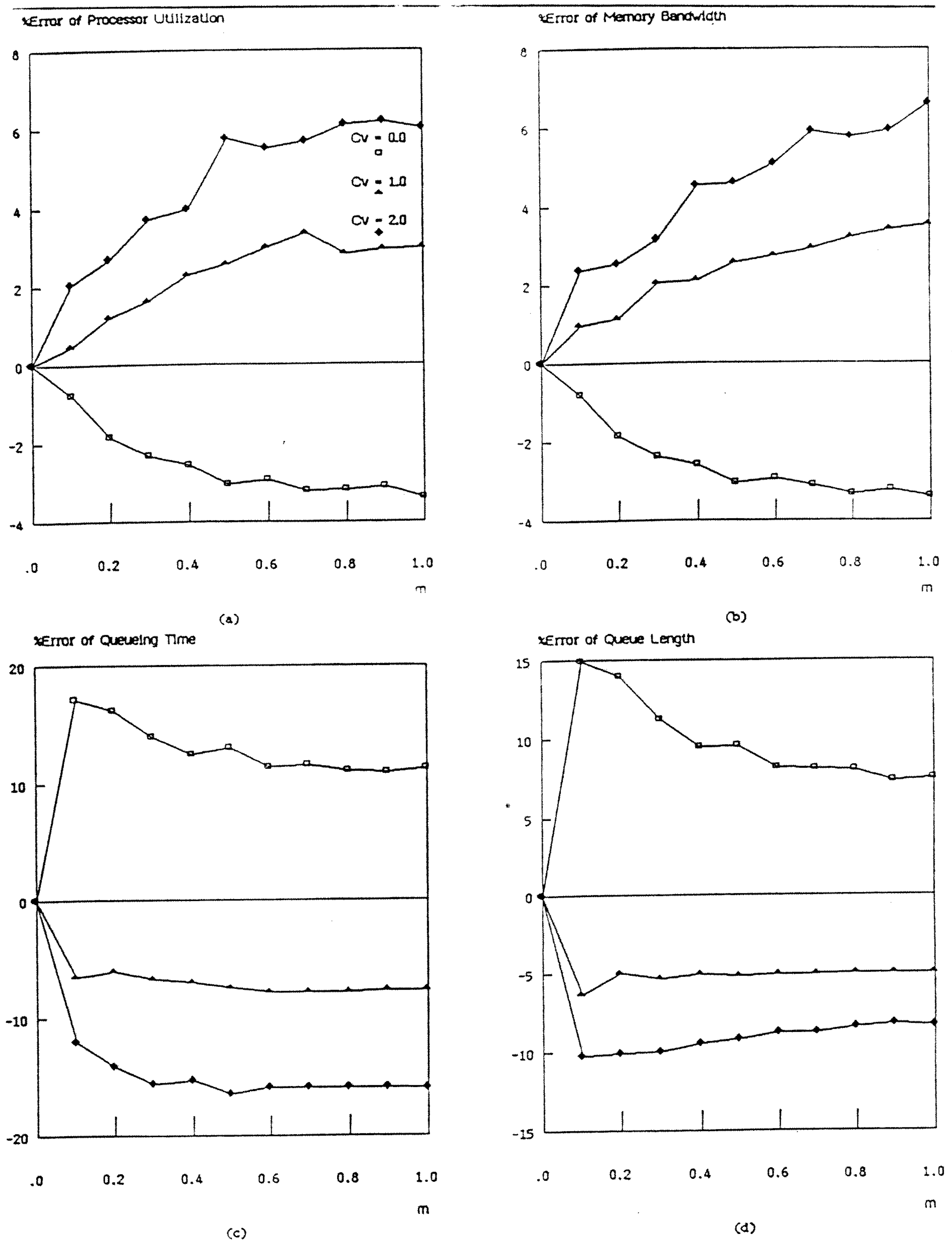
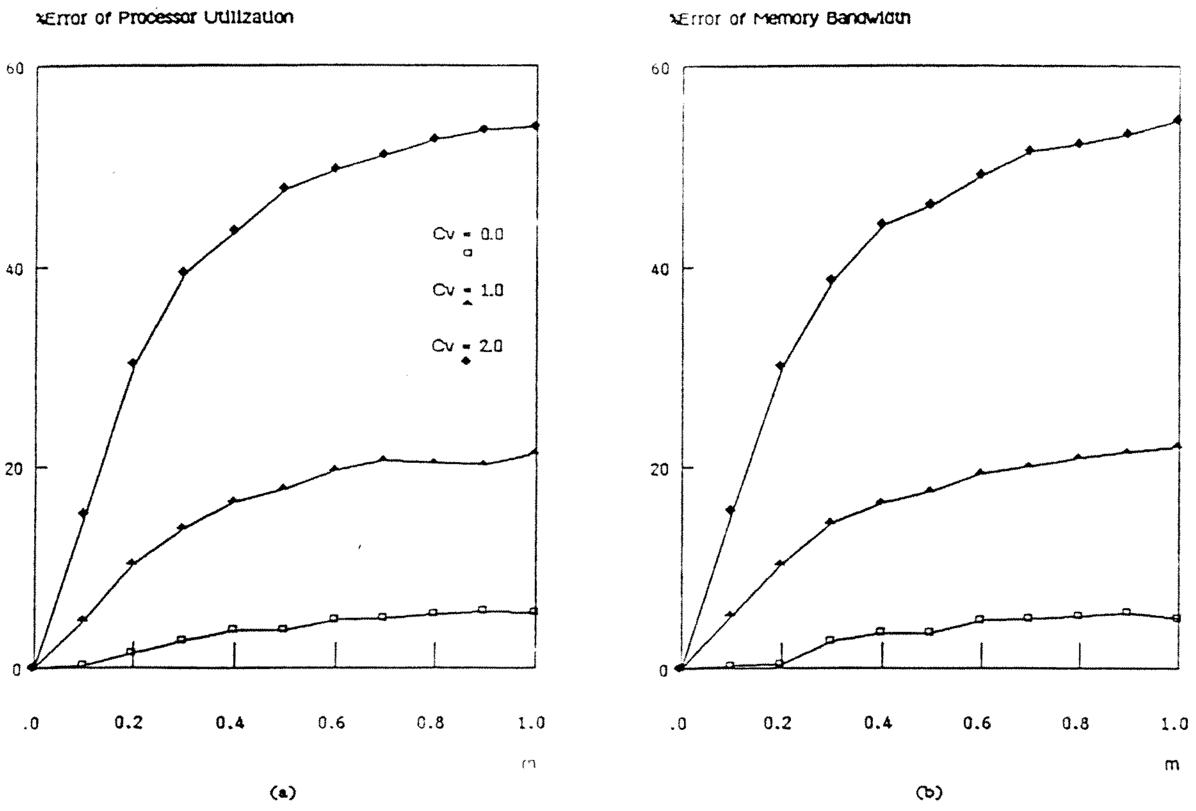


Figure 5. Percentage Error of the SMI model.



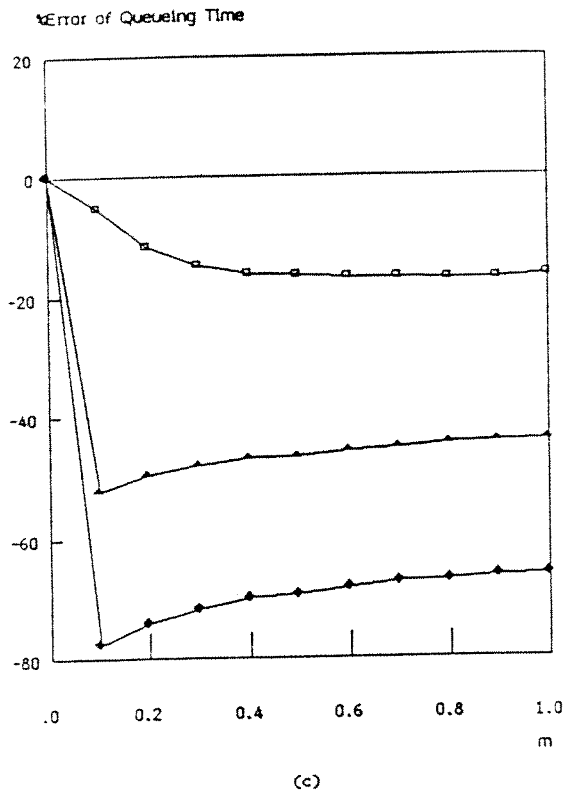


Figure 6. Percentage Error of the Simple cache model.

In this example, we compare the results of the SMI model and the results of the simple cache model reported in [10] with simulations. Furthermore, we demonstrate the effect of different connection distributions on the results of the SMI model and the simple cache model.

Figure 4 shows the simulation results for a 32×32 system for different connection distributions (all have $\overline{C} = 4.0$). It can be seen that the variation in C_v can dramatically effect BW, PU_i, W_{ij} and L_j. This effect is not captured in the simple cache model because it uses only the first moment of the connection time. Note that PU_i is defined in accordance with [10], that is to say, as the probability that PE_i is thinking when the multiprocessor system reaches steady state (see Section 3). The PU_i measure was less sensitive to variations in C_v for this reason.

Figure 5 shows the percentage difference between the simulation results and the results obtained from the SMI model. Figure 6 shows the percentage difference between the simulation results and the results obtained from the simple cache model. For the most part, it can be seen that the SMI model produces results much closer to simulation.

Example 2:

So far we have assumed that the time needed to set up the interconnection network is negligible. In this example we will develop a more realistic model that accounts for the time to traverse the interconnection network. This traversal time, t_{set}, has two components: first, there is the delay through the arbitration logic, t_a; and second, there is the delay through the interconnection network components, t_d. For example, a crossbar interconnection network can be built with multiplexer chips as described in [26]; in this case, t_d will be the delays through the multiplexer chips.

In this example we will analyze a multiprocessor system which has the same operation assumptions as the ones outlined in Section 2 plus the following assumption:

VIIa. Every time a *PE* attempts to access an idle *MM* it will spend t_{set} cycles waiting for its request to traverse the interconnection network. At the end of this traversal time the *PE* will establish the connection with the destined *MM*, or its request will be rejected due to a type one memory conflict. In the case of a type two memory conflict the *PE*'s request does not traverse the network.

A semi-Markov process that describes the *PE*'s behavior in this case is shown in Figure 7. This SMP contains an additional state in the S^{th} subset, $\bar{0}$, to represent the state in which the *PE* spends t_{set} cycles after initiating a request to access an idle *MM*. The average sojourn times in the states are now expressed as follows:

$$\begin{aligned} \eta_0 &= \overline{T} \\ \eta_{\bar{0}} &= t_{set} \\ \eta_1 &= \overline{C} \\ \eta_2 &= \overline{C} + t_{set} \\ \eta_3 &= \frac{\overline{C^2} - \overline{C}}{2\,(\overline{C} - 1)} + t_{set} \end{aligned} \tag{9}$$

In this case R can be expressed as follows:

$$R = \frac{1}{M}\left[\lambda_{\bar{0}} + \lambda_2 + \lambda_3\right]$$

The transition probabilities of the SMP are expressed as follows:

$$\alpha_j = \begin{cases} 1 & j = 0 \\ 1 - BUSY & j = \bar{0} \\ WIN & j = 1 \\ 1 - WIN & j = 2 \\ BUSY & j = 3 \end{cases}$$

$$\beta = (1 - BUSY)\,WIN$$

$$\bar{\beta} = 1 - \beta$$

The semi-Markov process, shown in Figure 7, can be solved using the same procedure outlined earlier. The SMP limiting probabilities can then be expressed as follows:

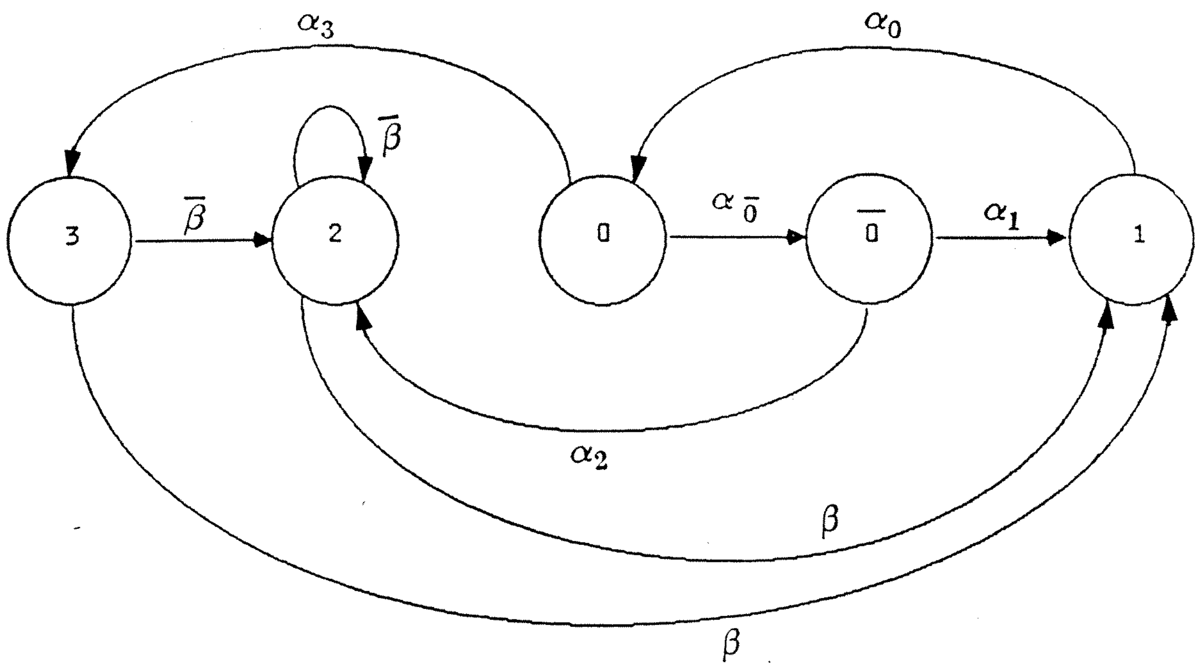


Figure 7. The SMP for the PE's behavior of Example 2.

$$P_j = \begin{cases} \dfrac{\eta_0\, M\, \beta\, R}{\beta\alpha_{\bar{0}} + \alpha_2\alpha_{\bar{0}} + \alpha_3} & j = 0 \\[2ex] \dfrac{\alpha_{\bar{0}}\, \eta_{\bar{0}}\, M\, \beta\, R}{\beta\alpha_{\bar{0}} + \alpha_2\alpha_{\bar{0}} + \alpha_3} & j = \bar{0} \\[2ex] \dfrac{\eta_1\, M\, \beta\, R}{\beta\alpha_{\bar{0}} + \alpha_2\alpha_{\bar{0}} + \alpha_3} & j = 1 \\[2ex] \dfrac{\left(\alpha_2\alpha_{\bar{0}} + \bar{\beta}\alpha_3\right)\, \eta_2\, M\, \beta\, R}{\beta\alpha_{\bar{0}} + \alpha_2\alpha_{\bar{0}} + \alpha_3} & j = 2 \\[2ex] \dfrac{\alpha_3\, \eta_3\, M\, \beta\, R}{\beta\alpha_{\bar{0}} + \alpha_2\alpha_{\bar{0}} + \alpha_3} & j = 3 \end{cases}$$

The performance measures in this case can be calculated as follows:

$$\begin{aligned} BW &= N\,P_1 \\ PU_i &= P_0 + P_{\bar{0}} + P_1 \\ MU_j &= \frac{N}{M}\;P_1 \\ L_j &= \frac{N}{M}\left[P_2 + P_3\right] \\ W_{ij} &= \alpha_{\bar{0}}\,\alpha_1\,\eta_{\bar{0}} + \left(\frac{1}{\beta} - (1 + BUSY)\right)\eta_2 + \alpha_3\,\eta_3 \end{aligned} \tag{10}$$

Figure 8 shows the simulation results for BW, PU_i, W_{ij} and L_j for a 32×32 system assuming $\bar{C} = 4.0$, $C_v = 0.0$ and $t_{set} = 1.0$. The effect of t_{set} can be deduced by comparing the simulation results in Figures 4 and 8: BW decreases, L_j changes very little, and W_{ij} increases. This is reflected in the SMI model. Equation (9) shows that η_2 and η_3 increase as t_{set} increase, and η_1 is not effected. Using equations (2), (9) and (10) BW is expressed as follows:

$$BW = N\,P_1 = N\,\frac{\pi_1\,\bar{C}}{\pi_0\,\bar{T} + \pi_1\,\bar{C} + \pi_2\,\bar{C} + \pi_3\,\dfrac{\overline{C^2} - \bar{C}}{2\,(\bar{C}-1)} + t_{set}\,(\pi_{\bar{0}} + \pi_2 + \pi_3)}$$

Clearly, BW decreases as t_{set} increase. The average queue length, L_j, is expressed as follows:

$$L_j = \frac{N}{M}\left[P_2 + P_3\right] = \frac{N}{M}\,\frac{\pi_2\,\bar{C} + \pi_3\,\dfrac{\overline{C^2} - \bar{C}}{2\,(\bar{C}-1)} + t_{set}\,(\pi_2 + \pi_3)}{\pi_0\,\bar{T} + \pi_1\,\bar{C} + \pi_2\,\bar{C} + \pi_3\,\dfrac{\overline{C^2} - \bar{C}}{2\,(\bar{C}-1)} + t_{set}\,(\pi_{\bar{0}} + \pi_2 + \pi_3)}$$

The presence of t_{set} in both numerator and denominator has a canceling effect so that changes in L_j due to t_{set} are small. Finally, an inspection of equation (10) shows W_{ij} increases with η_2 and η_3. Figure 9 shows the percentage error between the simulation results and the results obtained from the SMI model.

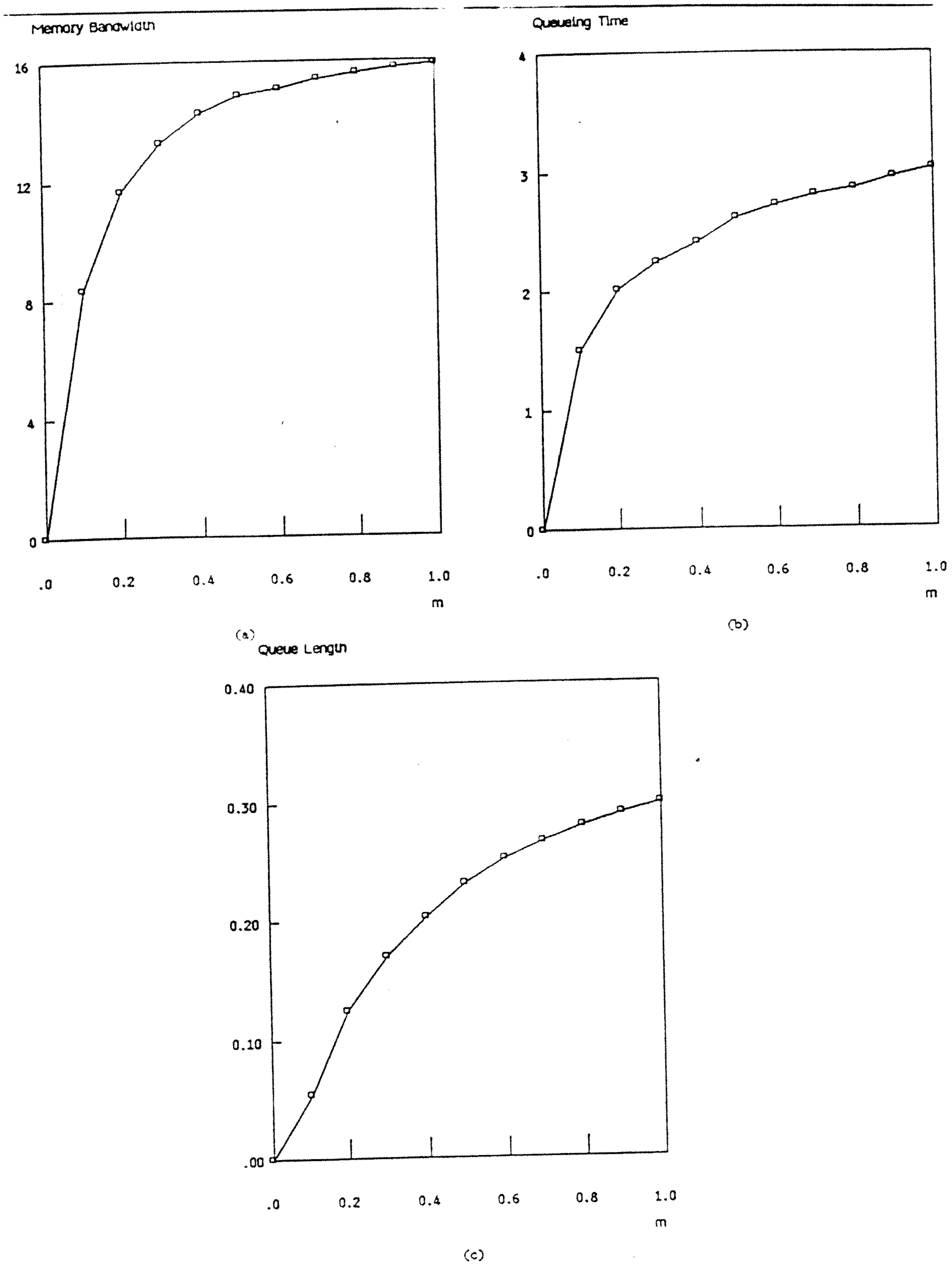


Figure 8. Simulation results using the traversal time for a 32x32 system.

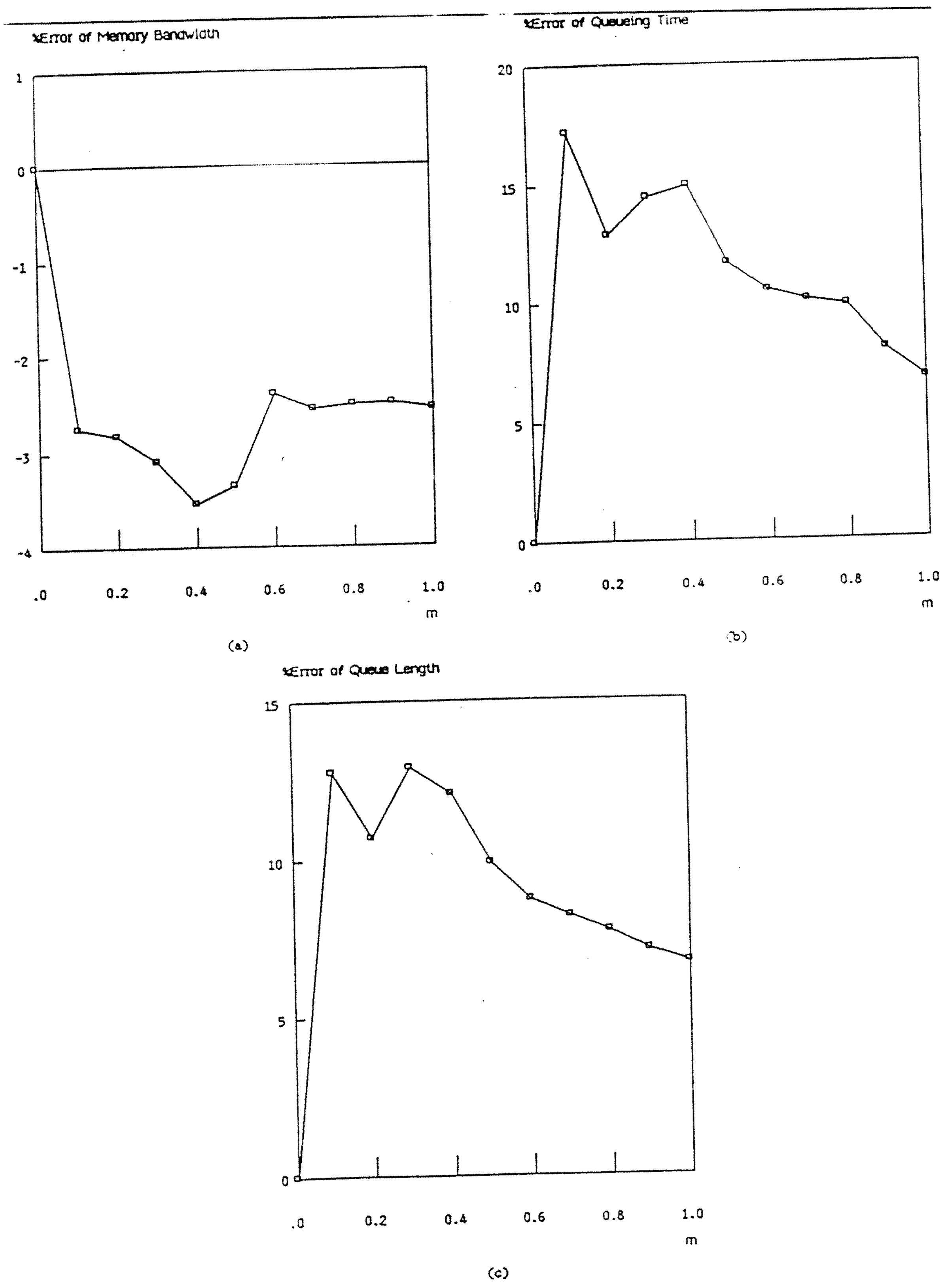


Figure 9. Percentage error of the SMI model.

5. The General Case

In the general case the random variables $\tilde{T}$, $\tilde{D}$ and $\tilde{C}$ may have any distribution. The multiprocessor operation assumptions now become:

Ib. The behavior of the PE's can be modeled as stochastic processes.

IIb. The PE's think for an integer number of system cycles. The thinking period of PE_i is characterized by a discrete independent random variable, $\tilde{T}_i$, where $1 \leq i \leq N$.

IIIb. Each PE will submit a memory request after its thinking period; requests originating from the same processing element are independent of each other. The destination memory module of the request originated from PE_i will be determined by a discrete independent random variable, $\tilde{D}_i$, where $1 \leq i \leq N$.

IVb. When the first type of memory conflict occurs, the memory module selects, equiprobably, one of the conflicting processing elements to gain access. The blocked processing element(s) wait until the connection is completed and then they resubmit their requests to the same memory module.

Vb. When the second type of memory conflict occurs, the blocked processing element(s) wait until the connection is completed and then they resubmit their requests to the same memory module.

VIb. The connection time between the i^{th} processing element, PE_i, and the j^{th} memory module, MM_j, is characterized by a discrete independent random variable, $\tilde{C}_{ij}$, where $1 \leq i \leq N$ and $1 \leq j \leq M$.

As mentioned earlier, the full distributions of $\tilde{C}_{ij}$ and $\tilde{T}_i$, where $1 \leq i \leq N$ and $1 \leq j \leq M$, are not needed. The input parameters of the SMI model are as follows:

$$\overline{T}_i \triangleq \textit{the first moment of } \tilde{T}_i$$

$$\overline{C}_{ij} \triangleq \textit{the first moment of } \tilde{C}_{ij}$$

$$\overline{C^2}_{ij} \triangleq \textit{the second moment of } \tilde{C}_{ij}$$

$$a_{ij} \triangleq Pr[\tilde{D}_i = j]$$

The SMP, shown in Figure 10, describes the behavior of PE_i. The general case requires N different SMP's to describe the system behavior ($i = 1, \cdots, N$). The state space of each of the SMP's can be divided to four disjoint subsets: $S_i^{th} = \{0\}$, $S_i^{ac} = \{1, \cdots, M\}$, $S_i^{fw} = \{M+1, \cdots, 2M\}$ and $S_i^{rw} = \{2M+1, \cdots, 3M\}$. It may be seen

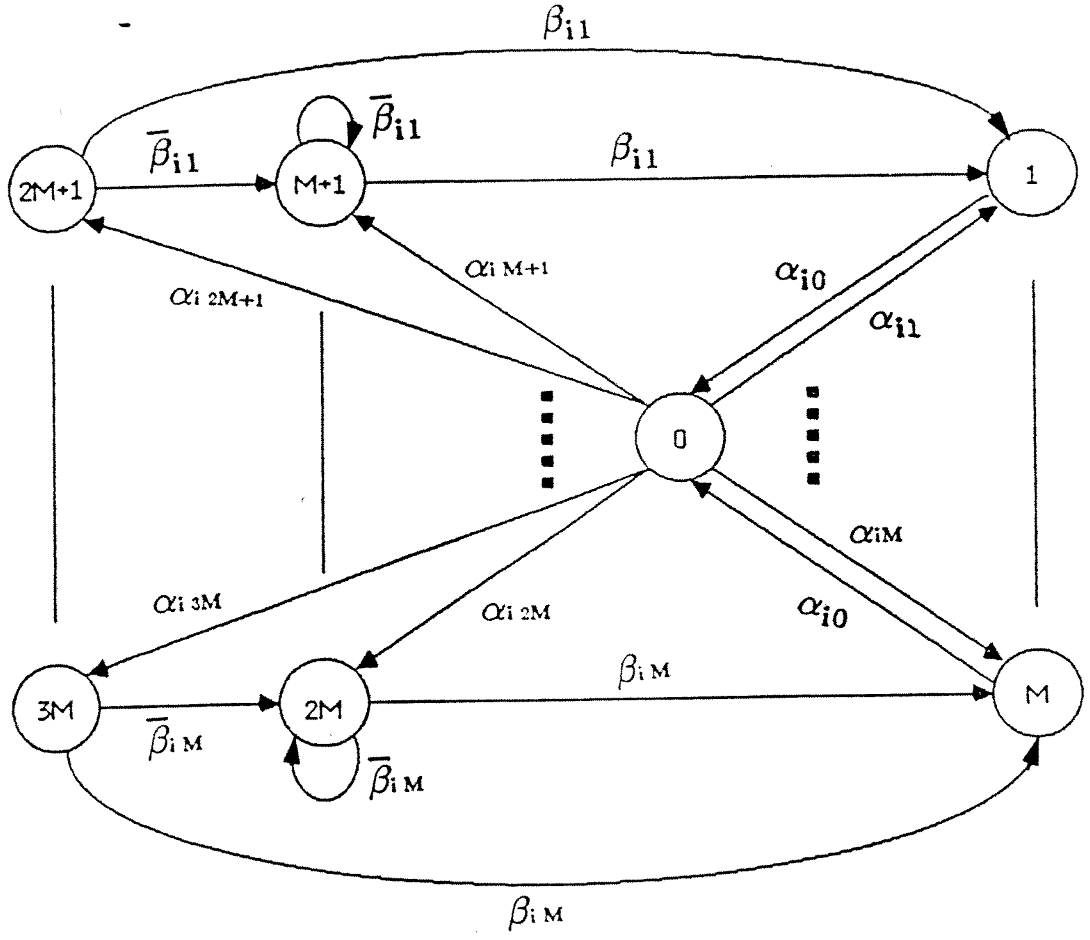


Figure 10. The SMP that describes PE behavior in the general case.

that this SMP collapses to the uniform case SMP if $a_{ij} = \frac{1}{M}$ for all i and j. The solution in the general case follows the lines of earlier cases, therefore only the outline of the solution will be presented.

The average sojourn times in the states are as follows:

$$\eta_{ij} = \begin{cases} \overline{T}_i & j \in S_i^{th} \\ \overline{C}_{ij} & j \in S_i^{ac} \\ \sum\limits_{\substack{k=1 \\ k \neq i}}^{N} \dfrac{R_{k\,j-M}}{\sum\limits_{\substack{l=1 \\ l \neq i}}^{N} R_{l\,j-M}} \; \overline{C}_{k\,j-M} & j \in S_i^{fw} \\ \sum\limits_{\substack{k=1 \\ k \neq i}}^{N} \dfrac{\left(P_{k\,j-2M} - \lambda_{k\,j-2M} \right)}{\sum\limits_{\substack{l=1 \\ l \neq i}}^{N} \left(P_{l\,j-2M} - \lambda_{l\,j-2M} \right)} \; \dfrac{\overline{C^2}_{k\,j-2M} - \overline{C}_{k\,j-2M}}{2(\overline{C}_{k\,j-2M} - 1)} & j \in S_i^{rw} \end{cases} \tag{3b}$$

Where,

$$R_{ij} = a_{ij}\,\lambda_{i\,0} + \lambda_{i\,j+M} + \lambda_{i\,j+2M}$$

The first subscript indicates the *PE* and the second indicates the state. The terms *WIN* and *BUSY* also require subscripts: the first indicates the *PE* and the second indicates the *MM*. These terms can be expressed as follows:

$$WIN_{ij} = \sum_{k=1}^{N} \frac{1}{k}\, \Psi_{ijk} \tag{4b}$$

where,

$$\Psi_{ijk} = \sum_{l=1}^{\binom{N-1}{k-1}} \prod_{\substack{h=1 \\ h \neq i}}^{N} w_{ijkl}(h)$$

and,

$$w_{ijkl}(h) = \begin{cases} R_{hj} & \text{if } PE_i\,, PE_h \text{ and } (k-2) \textit{ other PE's request } MM_j \textit{ in the } l^{th} \textit{ case} \\ 1 - R_{hj} & \textit{otherwise} \end{cases}$$

furthermore,

$$BUSY_{ij} = \sum_{\substack{k=1 \\ k \neq i}}^{N} \left(P_{kj} - \lambda_{kj} \right) = \sum_{\substack{k=1 \\ k \neq i}}^{N} \left(\overline{C}_{kj} - 1 \right) \lambda_{kj} \tag{5b}$$

The transition probabilities between the states of the SMP can be defined as follows:

$$\alpha_{ij} = \begin{cases} 1 & j \epsilon S_i^{th} \\ a_{ij}\ (1 - BUSY_{ij})\ WIN_{ij} & j \epsilon S_i^{ac} \\ a_{i\,j-M}\ (1 - BUSY_{i\,j-M})\ (1 - WIN_{i\,j-M}) & j \epsilon S_i^{fw} \\ a_{i\,j-2M}\ BUSY_{i\,j-2M} & j \epsilon S_i^{rw} \end{cases} \tag{6b}$$

$$\beta_{ij} = WIN_{ij}\ (1 - BUSY_{ij}) \qquad j \epsilon S_i^{ac}$$

$$\overline{\beta}_{ij} = 1 - \beta_{ij} \qquad j \epsilon S_i^{ac}$$

The embedded Markov chain can be solved and the π's can be represented as functions of transition probabilities. Then, from equation (2) λ_{kj} may be defined as a function of R and the transition probabilities. Therefore, using equation (5b) the term $BUSY_{ij}$ can be expressed as follows:

$$BUSY_{ij} = \sum_{\substack{k=1 \\ k \neq i}}^{N} (\overline{C}_{kj} - 1)\ \beta_{kj}\ R_{kj} \tag{7b}$$

Furthermore, the SMP limiting probabilities can be expressed as functions of R and the transition probabilities as shown below:

$$P_{ij} = \begin{cases} \eta_{ij} \sum_{k=1}^{M} \beta_{ik}\ R_{ik} & j \epsilon S_i^{th} \\ \eta_{ij}\ \beta_{ij}\ R_{ij} & j \epsilon S_i^{ac} \\ \left(\alpha_{ij} + a_{i\,j-M} \dfrac{(\overline{\beta}_{i\,j-M})^2}{\beta_{i\,j-M}} \right) \dfrac{\eta_{ij}\ \beta_{i\,j-M}}{a_{i\,j-M}}\ R_{i\,j-M} & j \epsilon S_i^{fw} \\ \dfrac{\alpha_{ij}\ \eta_{ij}}{a_{i\,j-2M}}\ \beta_{i\,j-2M}\ R_{i\,j-2M} & j \epsilon S_i^{rw} \end{cases} \tag{8d}$$

To solve the general case an iterative algorithm can be used similar to that proposed for the uniform case. In this case the algorithm iterates on the set of variables R_{ij}, where $1 \leq i \leq N$ and $1 \leq j \leq M$. The performance measures can be derived from the SMP's limiting probabilities as follows:

$$BW = \sum_{i=1}^{N} \sum_{j=1}^{M} P_{ij}$$

$$PU_i = 1 - \sum_{j=M+1}^{3M} P_{ij}$$

$$MU_j = \sum_{i=1}^{N} P_{ij}$$

$$L_j = \sum_{i=1}^{N} \left(P_{i\,j+M} + P_{i\,j+2M} \right)$$

$$W_{ij} = \eta_{i\,M+j} \left[\frac{1}{\beta_{ij}} - (1 + BUSY_{ij}) \right] + \eta_{i\,2M+j}\, BUSY_{ij}$$

Example 3:

Consider a multiprocessor system which manages a large data base. The system consists of two identical *PE*'s with private cache, two identical logical buffers and an I/O channel used for DMA between the logical buffers and a fixed head disk. The system is depicted in Figure 11. The two identical *PE*'s, PE_1 and PE_2, will request the first logical buffer with probability x, and the second logical buffer with probability $(1-x)$. The I/O channel, PE_3, requests the first logical buffer with probability y, and the second logical buffer with probability $(1-y)$.

In this example we consider the problem of deciding which is the better of two mappings from the logical buffers to the physical memory modules. In the first case, the logical buffers are mapped exactly into the physical memory modules. In the second case, each logical buffer is divided in half and, each half is placed into a physical memory module. The two mappings are illustrated in Figure 12. Notice, in the second case the $a_{ij} = 0.5$ (i=1,2,3, j=1,2) since $\frac{x}{2} + \frac{1-x}{2} = 0.5$, etc. In both cases the connection time between the *PE*'s and the *MM*'s is relatively short and equal to the time needed to transfer a line between the private cache and the *MM*. Due to the coherency checks employed by the system, the connection time between the identical *PE*'s and the

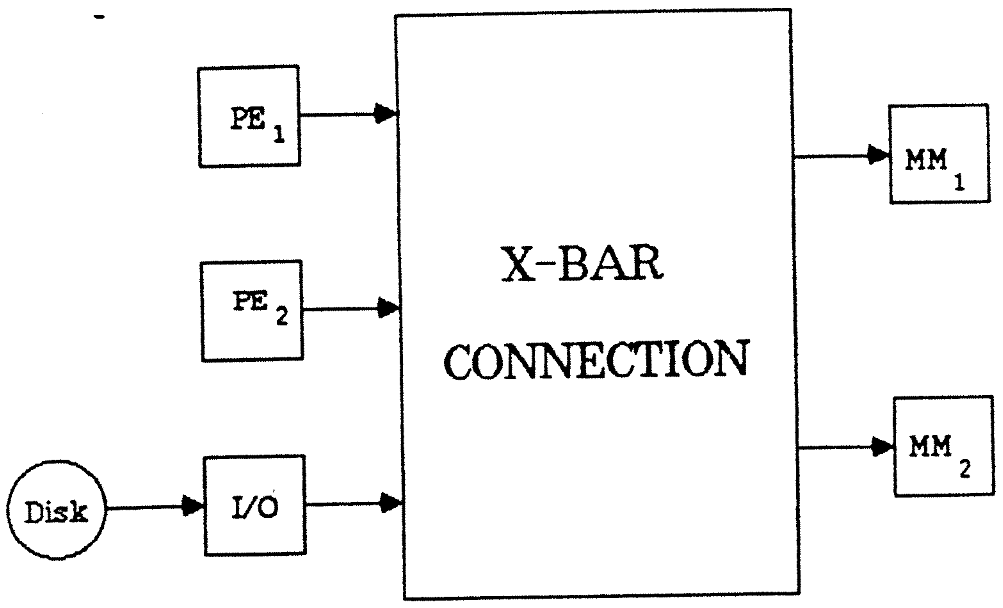


Figure 11. The multiprocessor system of Example 3.

MM's is variable. The connection time between the I/O channel and the *MM*'s is relatively long. This connection time has two components: the rotational delay and the data transfer time. The rotational delay can be characterized by a random variable which is uniformly distributed between 0 and the amount of time needed by the disk to make a full rotation. The data transfer time can be characterized as a deterministic random variable. The transfer time is the time needed to transfer a fixed block, e.g., a cylinder or a track, between the disk and the *MM*. Therefore, the connection time between the I/O channel and the *MM* can be characterized as a random variable with low C_v.

The SMI model is used to analyze the two cases mentioned above given the following operation conditions:

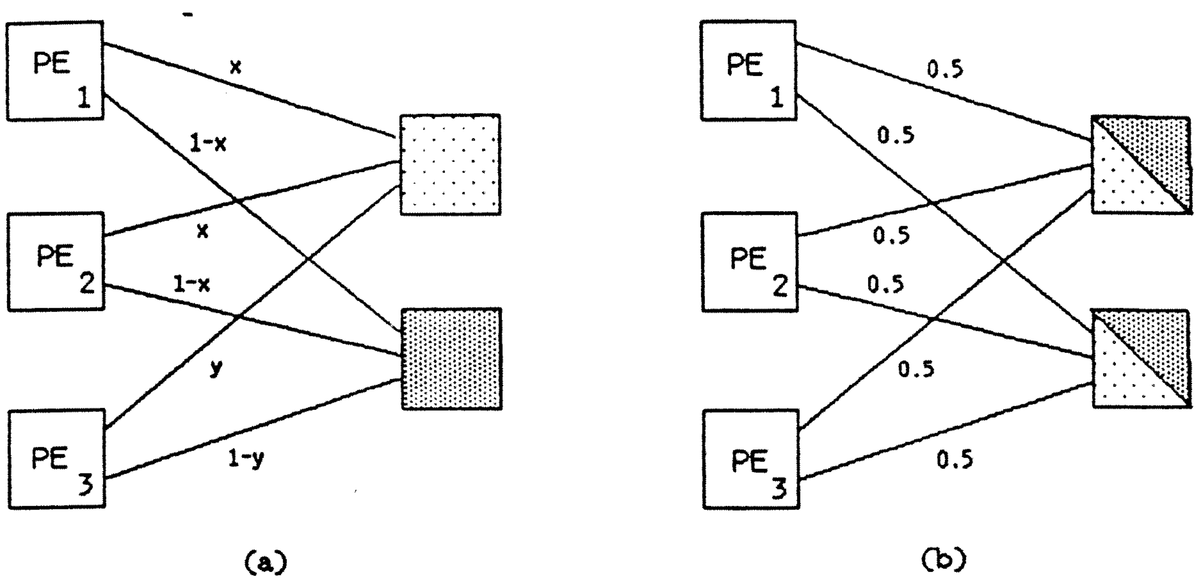


Figure 12. The two different mappings of Example 3.

$$\begin{aligned}
\overline{T}_1 &= \overline{T}_2 = 4.0 \\
\overline{T}_3 &= 12.0 \\
\overline{C}_{11} &= \overline{C}_{12} = 4.0 \\
\overline{C}_{21} &= \overline{C}_{22} = 4.0 \\
\overline{C^2}_{11} &= \overline{C^2}_{12} = 26.24 \\
\overline{C^2}_{21} &= \overline{C^2}_{22} = 26.24 \\
\tilde{C}_{31} &= \tilde{C}_{32} = UNIFORM(14,18) \\
x &= 0.9 \\
y &= 0.0
\end{aligned}$$

Where the distribution *UNIFORM*(14,18) is a uniform distribution between 14 and 18, i.e., data transfers take 14 cycles and the rotational delay takes between 0 and 4 cycles. The SMP that describes the *PE* behavior is shown in Figure 13. Notice, in the first case the behavior of PE_3 is described by an SMP with 4 states since $a_{31} = 0.0$.

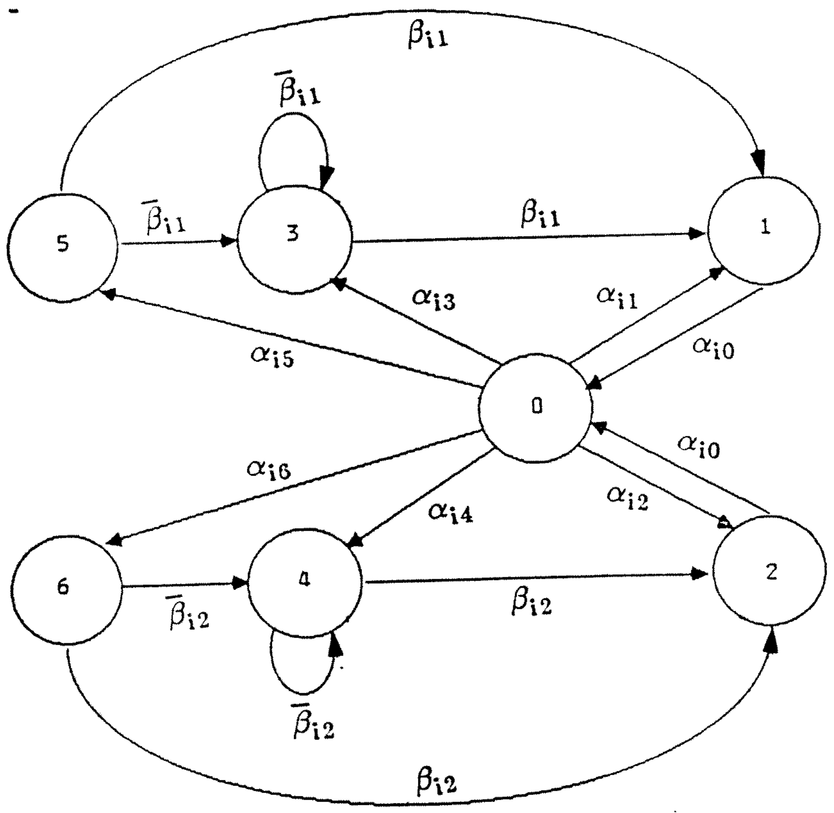


Figure 13. The SMP that describes PE behavior in Example 3.

Table 1 shows the results obtained from simulations and the SMI model. The SMI model agrees closely with simulation, and in practice would be used instead because it requires far less computation. As expected the mapping of case 1 yields better performance in most categories. The only exceptions being W_{12} and W_{22}. However, the overall waiting time for PE_i is given by:

$$W_i = \sum_{j=1}^{2} a_{ij} \quad W_{ij}$$

and a_{12}, a_{22} = 0.1 in case 1. Thus, the overall waiting times for PE_1 and PE_2 are also less in case 1.

Measure	Case # 1		Case # 2	
	Simulation	%Error	Simulation	%Error
BW	1.39415	-2.28	1.20084	-1.0
PU_1	0.8282	-2.75	0.66214	-3.33
PU_2	0.83041	-3.01	0.65778	-2.69
PU_3	0.98917	0.29	0.93723	2.47
MU_1	0.74607	-3.64	0.60502	-1.75
MU_2	0.64685	-0.53	0.59582	-0.23
L_1	0.21997	-3.51	0.38157	-0.48
L_2	0.12223	3.48	0.36931	2.82
W_{11}	1.25232	-1.59	4.15107	3.56
W_{12}	5.4316	7.18	3.9695	8.23
W_{21}	1.21373	1.54	4.15194	3.54
W_{22}	5.54553	4.98	3.9923	7.68
W_{31}	0.0	0.0	1.94831	-9.98
W_{32}	0.30673	6.09	1.87394	-6.41

Table 1. Comparisons between the two cases of Example 3.

6. Conclusion

This paper has presented a discrete time model, the SMI model, of memory interference for multiprocessor systems. The model characterizes a multiprocessor by describing the behavior of each *PE* as an independent semi-Markov process. By viewing events from the perspective of the *PE*'s, it is possible to model memory interference effects not explicitly modeled previously, such as queueing time and queue length. In addition, the effects of the coefficient of variation of the connection time and network traversal time can also be readily modeled. Their importance was shown in the examples given.

The complexity of solution for the SMI model was shown to be directly related to the number of distinct values that the discrete random variable $\tilde{D}$ can take on. For the uniform case, the SMI model requires only four states (five if traversal time is included) independent of the number of discrete values that N, M, $\tilde{T}$ and $\tilde{C}$ can have. One insight that emerged from developing the SMI model was that the simplifying assumption in the rate-adjusted models, typified by the model in [5], is the decoupling between the *PE*'s rather than the resubmission policy as suggested in [9].

Three examples were presented to illustrate how the SMI model can be used to predict the performance of multiprocessor systems. Simulations were included for comparison, and, in all cases except queueing time and queue length, the model differed by less than 6% (the queueing time and queue length differed by less than 18%). The error in the SMI model can be attributed to the decoupling between the *PE*'s behaviors. Because of this decoupling certain events are permissible in the SMI model that cannot occur in the real system. For example, the event that all *PE*'s are waiting has a non-zero probability in the SMI model. In reality, at least one *PE* would be accessing.

7. References

[1] C. E. Skinner and J. R. Asher, "Effects of Storage Contention on System Performance," *IBM Systems Journal*, vol. 8, no. 4, pp. 319-333, 1969.

[2] W. D. Strecker, in *Analysis of the Instruction Execution Rate in Certain Computer Structures* Ph.D. Thesis, Carnegie-Mellon Univ., 1970.

[3] D. P. Bhandarkar, "Analysis of Memory Interference in Multiprocessors ," *IEEE Trans. on Computers*, vol. C-24, no. 9, pp. 897-908, September 1975.

[4] F. Baskett and A. J. Smith, "Interference in Multiprocessor Computer Systems with Interleaved Memory," *Comm. of ACM*, vol. 19, no. 6, pp. 327-334, June 1976.

[5] C. H. Hoogendoorn, "A General Model for Memory Interference in Multiprocessors," *IEEE Trans. on Computers*, vol. C-26, no. 10, pp. 998-1005, October 1977.

[6] B. R. Rau, "Interleaved Memory Bandwidth in a Model of a Multiprocessors," *IEEE Trans. on Computers*, vol. C-28, no. 9, pp. 678-681, September 1979.

[7] J. H. Patel, "Processor-Memory Interconnections for Multiprocessors," *IEEE Trans. on Computers*, vol. C-30, no. 10, pp. 771-780, October 1981.

[8] T. N. Mudge and B. A. Makrucki, "Probabilistic Analysis of a Crossbar Switch," in *Proc. IEEE 9th Ann. Symp. on Computer Architecture*, pp. 311-319, April 1982.

[9] D. W. L. Yen, J. H. Patel, and E. S. Davidson, "Memory Interference in Synchronous Multiprocessor Systems," *IEEE Trans. on Computers*, vol. C-31, no. 11, pp. 1116-1121, November 1982.

[10] J. H. Patel, "Analysis of Multiprocessors with Private Cache Memories," *IEEE Trans. on Computers*, vol. C-31, no. 4, pp. 296-304, April 1982.

[11] L. N. Bhuyan and C. W. Lee, "An Interference Analysis of Interconnection Networks," in *Proc. 1983 Int'l Conf. on Parallel Processing*, pp. 2-9, August 1983.

[12] T. N. Mudge and H. B. Al-Sadoun, "Memory Interference Models with Variable Connection Time," *IEEE Trans. on Computers*, vol. C-33, no. 11, November 1984.

[13] M. A. Marsan and M. Gerla, "Markov Models for Multiple Bus Multiprocessor Systems," *IEEE Trans. on Computers*, vol. C-31, no. 3, pp. 239-248, March 1982.

[14] I. H. Onyuksel and K. B. Irani, "A Markov Queueing Network Model for Performance Evaluation of Bus-Deficient Multiprocessor Systems," in *Proc. 1983 Int'l Conf. on Parallel Processing*, pp. 437-439, August 1983.

[15] T. Lang, M. Valero, and M. Fiol, "Reduction of Connections for Multibus Organization," *IEEE Trans. on Computers*, vol. C-32, no. 8, pp. 707-716, August 1983.

[16] T. N. Mudge, J. P. Hayes, G. D. Buzzard, and D. C. Winsor, "Analysis of Multiple-Bus Interconnection Networks," in *Proc. 1984 Int'l Conf. on Parallel Processing*, pp. 228-232, August 1984.

[17] L. N. Bhuyan, "A Combinatorial Analysis of Multibus Multiprocessors," in *Proc. 1984 Int'l Conf. on Parallel Processing*, pp. 225-227, August 1984.

[18] A. Goyal and T. Agerwala, "Performance Analysis of Future Shared Storage Systems," *IBM Journal of Res. and Develop.*, vol. 28, no. 1, pp. 95-108, January 1984.

[19] D. P. Bhandarkar and S. H. Fuller, "Markov Chain Models for Analyzing Memory Interference in Multiprocessor Computer Systems," in *Proc. 1st Ann. Symp. Computer Architecture*, pp. 1-6, December 1973.

[20] S. H. Fuller, "Performance Evaluation," in *Introduction to Computer Architecture*, H. S. Stone, Ed S.R.A., Inc., pp. 474-546, 1975.

[21] B. A. Makrucki, in *A Stochastic Model of Multiprocessing* Ph.D. Thesis, The University of Michigan, 1984.

[22] F. A. Briggs and M. Dubois, "Effectiveness of Private Caches in Multiprocessor Systems with Parallel-Pipelined Memories," *IEEE Trans. on Computers*, vol. C-32, no. 1, pp. 48-59, January 1983.

[23] E. Cinlar, in *Introduction to Stochastic Processes.* Englewood Cliffs, N.J.: Prentice-Hall Inc., 1975.

[24] D. P. Heyman and M. J. Sobel, in *Stochastic Models in Operations Research,* vol. 1 McGraw-Hill Book Co., 1982.

[25] S. M. Ross, in *Applied Probability Models with Optimization Applications.* San Francisco, Calif.: Holden-Day, Inc., 1970.

[26] W. A. Wulf, R. Levin, and S. P. Harbison, in *Hydra/C.mmp: An Experimental Computer System.* New York: McGraw-Hill, 1981.