Analysis of Memory Interference in Buffered Multiprocessor Systems in Presence of Hot Spots and Favorite Memories *

Sajal K. Das and Sanjoy K. Sen  
Department of Computer Sciences  
University of North Texas  
Denton, TX 76203-6886  
E-mail: {das,joy}@cs.unt.edu

Abstract  
We present a discrete Markov chain model for analyzing the effect of memory interference in processor-memory interconnections of buffered multiprocessor systems. Each module is assumed to be one of the following three types — hot memory, favorite memory and memory which is neither hot nor favorite. The analytical solutions are restricted to 2×M and N×2 systems, where N and M are respectively the number of processors and memory modules. The general case is analyzed using simulation studies and compared with the analytic results. In all cases the main criterion of the system performance are the effective bandwidth, mean queue length and mean waiting time for a memory request. It is expected that increasing the number K of hot modules will improve the performance. We also estimate the asymptotic bandwidth and propose a heuristic to find an upper bound on K beyond which the bandwidth saturates.

1 Introduction  
A shared memory multiprocessor consists of a set of N processors $\{P_0, P_1, \ldots, P_{N-1}\}$ and a set of M memory modules $\{M_0, M_1, \ldots, M_{M-1}\}$. The processor-memory communications are established either through a set of switches e.g. crossbar or multistage interconnection network (MIN), or through global shared (multiple) bus, sometimes called MBIN. In this paper, both of these types of connections between the processors and memories will be called under the generic name, interconnection mechanism.

The interconnection mechanism should allow efficient resource sharing among the processors. Conflicts arise when more than one processor attempt to access the same memory module or try to use the same link or switch in the interconnection network, leading to decrease in the acceptance rate of memory requests thereby degrading the effective memory bandwidth, BW, which is defined as the average number of busy memory modules in each memory cycle. Other performance metrics of such a processor-memory system include mean waiting time for a memory request and expected queue length.

The memory reference pattern of the processors can be uniform or non-uniform. A pattern is said to be uniform if all the processors have the same probability of accessing any memory module [1,3,9]. Otherwise the memory reference pattern will be non-uniform [4,5,8] and depend on the locality of reference in the computation requirements. From the view point of access non-uniformity, a memory module can be classified either as favorite or hot. Each processor may have a favorite memory module(s) which it accesses more frequently than the others [5]. Or, the processors might access a particular module(s) more frequently than the others (e.g. global variables) called hot-spot(s). In this paper, we deal with the problem of contention issues in accessing the shared memory modules of types hot and favorite, after a request has successfully passed through the interconnection mechanism.

2 Related Work  
Skinner and Asher [3] first used the Markov chain model to analyze multiprocessor systems with memory queues, assuming that all the processors and all the memory modules are identical. The state of the Markov chain is defined by the M-tuple $(k_1, k_2, \ldots, k_M)$, where $\sum_{i=1}^{M} k_i = N$ is the number of processors, $M$ is the number of memory modules and $k_i$ denotes the number of processors waiting (or active) in the queue of the i-th module, $M_i$. The total number of possible states is given as $\binom{N+M-1}{N}$. Clearly, the number of states become prohibitively large for $M, N \geq 3$. Bhandarkar [1] managed to reduce the number of states noting that many states of this Markov chain model are identical.

Sethi and Deo [2] proposed a discrete Markov chain model for analyzing non-uniform memory access patterns assuming the case of temporal locality of memory ref-
ferences. Analytical expressions for the average number of busy modules in a memory cycle were developed for $2 \times M, N \times 2$ systems, and also for $3 \times M$ system.

Pfister and Norton [6] introduced a separate class of memory-access non-uniformity, called hot spots, for higher access rate superimposed on the otherwise uniform traffic. Hot spots capture the effect of all the processors continuously sharing global common variables. An interesting phenomenon, called tree saturation, can be observed for MINs with finite buffers. Pfister and Norton introduced the concept of combining messages to similar destination at every node of the network to reduce the effects of tree saturation.

Mudge, Hayes, Buzzard and Winsor [7] presented a discrete stochastic model of bandwidth analysis for multiprocessor systems in which each memory module has a buffer for queuing up unsatisfied memory requests. The following assumptions are made to simplify the analysis, namely, temporal independence and spatial independence. By removing the spatial independence assumption, Mudge et al. also developed an iterative scheme to reduce the error caused by the temporal independence.

The rest of this paper is organized as follows. Our Markov model for buffered memory multiprocessors is proposed in Section 3. Section 4 presents simulation results for the general case of $N \times M$ system. It was noticed that increasing the number of hot modules $K$, leads to saturation of bandwidth. An approximate probabilistic analysis is proposed in Section 5, which leads to a heuristic for estimating this saturation value of $K$. Section 6 concludes the paper.

3 Markov Model With Hot Spots and Favorite Memories

In this section, we derive a discrete Markov model for shared memory multiprocessor systems in which each memory module has a buffer for queuing up unsatisfied memory requests. The following assumptions are made:

- There is no contention in the interconnection mechanism.
- All the processors have an identical memory request rate $\lambda$.
- The memory access of a processor may last over several memory cycles. The completion rate is given by $\mu$.
- If there are $K$ hot modules in the system, they are all identical and memory reference is uniform among them.
- The reference pattern among the ‘non-hot, non-favorite’ memory modules is also uniform.

3.1 $2 \times M$ System, $K$ Hot Spots and No Favorite Memory

There are two processors and $M$ memory modules out of which $K$ are hot modules. Let $m$ be the probability of a hot memory request by any processor. Then, $\frac{1-m}{M-K}$ is the probability of requesting any of the $M-K$ non-hot memories. If $\lambda$ is the request rate of a processor, then the probability that a processor requests a hot memory is given by $p_h = \frac{m \lambda}{K}$. The probability that a processor requests a non-hot memory is $p = \frac{\lambda (1-m)}{(M-K)}$. The states for this $2 \times M$ system are represented as two or three tuples, where each element gives the number of processor requests in the memory queue.

<table>
<thead>
<tr>
<th>state vector</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_0$ &lt; 0, 0 &gt;</td>
</tr>
<tr>
<td>$S_1$ &lt; 0, 1 &gt;</td>
</tr>
<tr>
<td>$S_2$ &lt; 0, 1, 1 &gt;</td>
</tr>
<tr>
<td>$S_3$ &lt; 0, 2 &gt;</td>
</tr>
<tr>
<td>$S_4$ &lt; 1, 0 &gt;</td>
</tr>
<tr>
<td>$S_5$ &lt; 1, 1 &gt;</td>
</tr>
<tr>
<td>$S_6$ &lt; 2, 0 &gt;</td>
</tr>
<tr>
<td>$S_7$ &lt; 1, 1, 0 &gt;</td>
</tr>
</tbody>
</table>

The state transition probabilities are computed next and a $8 \times 8$ linear system of the form $A^T \Pi = \Pi$ where $A$ is the state transition matrix and $\Pi = (\Pi_0, \Pi_1, \ldots, \Pi_6)$ is a solution vector containing the limiting probabilities such that $\Pi_i$ denotes the limiting probability of state $S_i$. For the state transition matrix and the analytical derivation of the bandwidth, refer to [10]. The bandwidth of the system is given by $BW = \Pi_1 + 2\Pi_2 + \Pi_3 + \Pi_4 + 2\Pi_5 + \Pi_6 + 2\Pi_7$. Some of the bandwidth values computed analytically for various values of the input parameters are shown in Table 1. It can be concluded from the results that increasing the service rate of the memory module (i.e., decreasing the average time of memory access by the processors) leads to a higher completion rate of jobs (memory requests), and without additional jobs to fill in ($\lambda$ remaining constant) the memory module tends to remain inactive for longer time, leading to a decrease in the bandwidth. It is also observed that the bandwidth increases as the number, $K$, of hot spots is increased because the greater memory traffic is now distributed among the additional modules.

Table 1: Memory bandwidth with varying $\lambda$ and $\mu$ for $K = 1, M = 10$ and $m = 0.75$.

<table>
<thead>
<tr>
<th>$\lambda$</th>
<th>0.1</th>
<th>0.2</th>
<th>0.3</th>
<th>0.4</th>
<th>0.5</th>
<th>0.6</th>
<th>0.7</th>
<th>0.8</th>
<th>0.9</th>
<th>1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>1.6</td>
<td>1.4</td>
<td>1.2</td>
<td>1.0</td>
<td>0.8</td>
<td>0.6</td>
<td>0.4</td>
<td>0.2</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.2</td>
<td>1.6</td>
<td>1.4</td>
<td>1.2</td>
<td>1.0</td>
<td>0.8</td>
<td>0.6</td>
<td>0.4</td>
<td>0.2</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.3</td>
<td>1.6</td>
<td>1.4</td>
<td>1.2</td>
<td>1.0</td>
<td>0.8</td>
<td>0.6</td>
<td>0.4</td>
<td>0.2</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.4</td>
<td>1.6</td>
<td>1.4</td>
<td>1.2</td>
<td>1.0</td>
<td>0.8</td>
<td>0.6</td>
<td>0.4</td>
<td>0.2</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.5</td>
<td>1.6</td>
<td>1.4</td>
<td>1.2</td>
<td>1.0</td>
<td>0.8</td>
<td>0.6</td>
<td>0.4</td>
<td>0.2</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.6</td>
<td>1.6</td>
<td>1.4</td>
<td>1.2</td>
<td>1.0</td>
<td>0.8</td>
<td>0.6</td>
<td>0.4</td>
<td>0.2</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.7</td>
<td>1.6</td>
<td>1.4</td>
<td>1.2</td>
<td>1.0</td>
<td>0.8</td>
<td>0.6</td>
<td>0.4</td>
<td>0.2</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.8</td>
<td>1.6</td>
<td>1.4</td>
<td>1.2</td>
<td>1.0</td>
<td>0.8</td>
<td>0.6</td>
<td>0.4</td>
<td>0.2</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.9</td>
<td>1.6</td>
<td>1.4</td>
<td>1.2</td>
<td>1.0</td>
<td>0.8</td>
<td>0.6</td>
<td>0.4</td>
<td>0.2</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>1.0</td>
<td>1.6</td>
<td>1.4</td>
<td>1.2</td>
<td>1.0</td>
<td>0.8</td>
<td>0.6</td>
<td>0.4</td>
<td>0.2</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>
Let $MQ_L$ and $MQ_{Lb}$ be the mean queue lengths for a hot module and non-hot module, respectively. Since we are dealing with a system of multiple queues, we actually consider the mean of the maximum queue lengths in the system. From the above state table, we obtain $MQ_L = \Pi_1 + \Pi_2 + 2\Pi_3 + \Pi_7$ and $MQ_{Lb} = \Pi_1 + \Pi_2 + 2\Pi_3 + \Pi_3 + \Pi_7$. The mean queue length for the whole system is given by $MQ_L = \Pi_1 + \Pi_2 + 2\Pi_3 + \Pi_3 + \Pi_7$. The mean waiting time is given by, $\text{MWT}_i = (1/K) \times MQ_L + (M - K) \times MQ_{Lb}$.

The variation of mean memory-queue length and the mean waiting time of memory requests with request service rate are depicted graphically in Figure 1.

![Figure 1: Variation of mean memory-queue length and mean waiting time with request service rate](image)

### 3.2 $N \times 2$ System, One Hot and One Non-Hot Non-Favorite Memory

Out of the two memory modules, let one be hot and the other module be non-hot as well as non-favorite. We also assume that the probability of request by a processor at any memory cycle is unity. That is, $\lambda = 1$. Consequently, $\mu = 1$ too. This assumption, although unrealistic, considerably simplifies the model. The total number of states in this case is $N + 1$ and the state vectors are given by $S_i = < N - i, i >$ for $0 \leq i \leq N$.

Let $a_{i,j}$ be the probability of transition from state $S_i$ to state $S_j$. Recalling that $m$ is the probability of accessing the hot module, the transition probabilities are obtained as follows.

For state $S_i$, where $i \neq 0$ or $N$: $a_{i,j} = m(1-m)$, $a_{i,i-1} = m^2$ and $a_{i,i+1} = (1-m)^2$.

For states $S_0$ and $S_N$: $a_{0,0} = m$ and $a_{0,1} = 1-m$; $a_{N,N} = 1-m$ and $a_{N,N-1} = m$.

Let $\Pi_i$ denote the limiting probability of state $S_i$. Then $\Pi_0 = \frac{1}{1 + (1-m)(1-\Pi_N)}$ and $\Pi_i = \frac{(1-m)\Pi_{i-1}}{m} \Pi_0$. For $1 \leq i \leq N$, where $l = \frac{(1-m)^2}{m}$. The bandwidth is given by $BW = 2 - \Pi_0 - \Pi_N$.

From the state table of the $N \times 2$ system, the mean queue length of the requests in the hot memory module is given as, $MQ_L = N(1 - \Pi_N) - \frac{(1-m)(1-m^2)}{m} \Pi_0$ and that for the non-hot memory module is, $MQ_{Lb} = (1-m)N - \frac{(1-m^2)}{m} \Pi_0$.

The mean waiting time for a memory request in the system is given by $\text{MWT}_i = \frac{MQ_L + MQ_{Lb}}{\lambda}$. The mean queue length of the system is given by $MQ_L = N - \frac{(1-m)(1-m^2)}{m} \Pi_0 + \frac{(1-m)}{m} \left( \frac{(1-m)^2}{m} - \Pi_0 \right)$.

The bandwidths are evaluated for various values of $m$ and $N$. As expected, the bandwidth deteriorates as the value of $m$ is increased. The results are shown in Table 2. For the same values of the input parameters, the mean queue lengths and the mean waiting times are depicted graphically in Figure 2.

![Figure 2: Mean queue length with probability of requesting hot modules](image)

### Table 2: Memory bandwidths with various values of m and N for the $N \times 2$ system

<table>
<thead>
<tr>
<th>m</th>
<th>N = 2</th>
<th>N = 5</th>
<th>N = 10</th>
<th>N = 15</th>
<th>N = 20</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.25</td>
<td>1.19</td>
<td>1.47</td>
<td>1.58</td>
<td>1.59</td>
<td>1.59</td>
</tr>
<tr>
<td>0.5</td>
<td>1.32</td>
<td>1.44</td>
<td>1.45</td>
<td>1.45</td>
<td>1.45</td>
</tr>
<tr>
<td>0.75</td>
<td>1.27</td>
<td>1.30</td>
<td>1.30</td>
<td>1.30</td>
<td>1.30</td>
</tr>
<tr>
<td>0.85</td>
<td>1.16</td>
<td>1.17</td>
<td>1.17</td>
<td>1.17</td>
<td>1.17</td>
</tr>
<tr>
<td>0.95</td>
<td>1.05</td>
<td>1.05</td>
<td>1.05</td>
<td>1.05</td>
<td>1.05</td>
</tr>
</tbody>
</table>
4 Simulation for $N \times M$ System

In the general case of an $N \times M$ system with $K$ hot memory modules, the more common practice of simulation is resorted to. An interesting phenomenon was observed from the simulation results. Keeping the values of $\lambda$ and $\mu$ constant, the variation of bandwidth with $K$ shows a definite pattern. With $M = 50$, the variation of bandwidth with $K$ for $N = 10, 20, 30$ and $40$ is observed. The bandwidth of the system seems to increase at first almost linearly with $K$ and then at a progressively lower rate, until it almost saturates with the variation of $K$. When $N$ is kept constant, and the variation of bandwidth with $K$ is observed for $M = 10, 20, 30$ and $40$, it shows similar patterns and it also decreases substantially after saturation. The value of $K$ at which the change of bandwidth is almost insignificant is called the saturation value of $K$. The saturation values of $K$ are shown in Figure 3.

\begin{equation}
\text{BW} = K \left[ 1 - \left( 1 - \frac{\lambda \cdot \mu}{K} \right)^N \right] + (M - K) \left[ 1 - \left( 1 - \frac{\lambda (1 - m)}{M - K} \right)^N \right].
\end{equation}

But this is an approximate analysis of the bandwidth since in a more realistic situation the rejected memory requests are not discarded, but stored in order to make a fresh attempt in the next cycle. We will consider this case next, following an iterative technique laid down by Mudge et al. [7] to compute the effective bandwidth of the multiple bus system. Let us assume that the rejected memory requests are resubmitted in the next cycle following a uniform distribution.

Let $\beta_h$ and $\beta_{nh}$ be the modified request rates for the hot and non-hot memory modules. Following the scheme in [7], we get the following iterative equation to evaluate the values of $\beta$’s:

\begin{equation}
\beta_{i+1} = 1 + \frac{BW^*}{Np^{i}} (1 - p)
\end{equation}

where $BW^*$ denotes either (i) $BW^h$ or (ii) $BW_{nh}$, and $p$ represents $\lambda m$ and $\lambda (1 - m)$ for Cases (i) and (ii) respectively, and $BW^*(\beta_i)$ is obtained by replacing $p$ by $\beta_h$ or $\beta_{nh}$ as the case may be. Solution for $\beta_{i+1}$ when substituted in Equation (1), gives a new value of bandwidth which is much closer to that in the real situation.

We now propose a method to find out approximately the saturation value of $K$ for given $M$ and $N$.

**Proposed Heuristic:**

1. Initialize $K$ to 1.
2. Set $M$, $N$ and $m$ to given values and set $\lambda$ and $\mu$ to 1.
3. Start with an initial estimate of $\beta_i$ in the range $0 < \beta_i < 1$.
4. Apply a non-linear equation solver to evaluate $\beta_i$ using new estimates of BW (from Equation 1) in successive stages.
5. Use this $\beta$ to get the final estimate of BW.
6. Check the relative rate of change of bandwidth.

   - If the rate $> \epsilon$, a predefined small value, then increment $K$ and go to Step 2.
   - Else print the current value of $K$.

The proposed heuristic was implemented using a Fortran non-linear equation solver package to estimate the optimum value of $K$ beyond which there is little or no change of memory bandwidth. The results obtained were quite satisfactory. The variations are also depicted in Figure 4. In all cases, the rate of change of the bandwidth progressively decreases for higher values of $K$. Our analytical model provides good estimates of memory bandwidth although it tends to overestimate bandwidths, more for lower values of $K$ than for higher values. Our analytical model also captures the decrease in bandwidth after saturation for increasing $K$, in the case for constant $N = 50$ and varying $M$ (Figure 4).

![Figure 3: Variation of memory bandwidth with number of hot modules (from simulation)](image)

5 Probabilistic Analysis of Bandwidth

Assuming only hot spots and no favorite memories, consider for the moment the case where intra-cycle uniformity of memory requests is assumed. In other words, once a processor’s memory request is failed due to memory interference, it is altogether discarded in that or the next cycle. Let $BW^h$ and $BW_{nh}$ denote the average number of active hot memory modules and active non-hot, non-favorite memory modules respectively. That is, $BW^h$ denotes the contribution to the effective bandwidth from the hot memory modules, and $BW_{nh}$ gives the contribution from non-hot memories. If $K$ is the number of hot memory modules in the system, then according to [1], $BW^h = K \left[ 1 - \left( 1 - \frac{\lambda m}{K} \right)^N \right]$, where $\frac{\lambda m}{K}$ gives the probability of requesting a particular hot memory module. Similarly, for the non-hot memory modules, $BW_{nh} = (M - K) \left[ 1 - \left( 1 - \frac{\lambda (1 - m)}{M - K} \right)^N \right]$. Hence the effective memory bandwidth for the entire system is

\begin{equation}
\text{BW} = K \left[ 1 - \left( 1 - \frac{\lambda m}{K} \right)^N \right] + (M - K) \left[ 1 - \left( 1 - \frac{\lambda (1 - m)}{M - K} \right)^N \right].
\end{equation}
Applying the heuristic, the saturation values of $K$ are found to be $K = 8, 18, 26$ and $32$, respectively for $N = 10, 20, 30$ and $40$ and $M = 50$. When $N$ is kept constant at 50, the saturation values of $K$ are $6, 14, 22$ and $30$, respectively for $M = 10, 20, 30$ and $40$. These results are comparable with the values of $K$ obtained from the experiments under similar conditions (maintaining the same values for other parameters), which are $6, 16, 26, 27$ and $8, 16, 24$, and $32$, respectively.

### 6 Conclusions

In this paper, we have studied the performance of shared memory multiprocessor systems under various types of non-uniformity in memory access patterns. A discrete Markov chain is used to model the memory references in a set of modules consisting of hot spots. Then the system with $N$ processors and two memory modules, one of them being a hot spot, is considered. Analytic expressions for bandwidth, mean queue length and mean waiting time are derived. Extensive simulation experiments are carried out for the general $N \times M$ system consisting of hot memory modules. The simulation results are found to tally well with our analytical results in the special cases. We have also proposed a heuristic to evaluate the saturation value of $K$ using an approximate probabilistic model of the memory interference. It might be concluded that in a system with hot spots, distributing the shared variables in more than one memory module is a good idea, as long as the total number of modules is below the saturation value of $K$.

### References


