Imbedded Markov Chains Model of Multiprocessor with Shared Memory

Angel Vassilev Nikolov,

National University of Lesotho, Roma 180, Lesotho

Summary
This paper addresses the problem of evaluating the performance of multiprocessor with shared memory and private caches executing Invalidate Coherence Protocols. The model is grounded in queuing network theory and includes bus interference, cache interference, and main memory interference. The method of the Imbedded Markov Chains is used. The highest and lowest performance characteristics are calculated for both equilibrium and transient states.

Key words:
Invalidate Cache Protocols, Markov Chains, Multiprocessor, Queuing Network

1. Introduction

Shared bus shared memory multiprocessors are a suitable platform to speed up execution of general purpose workload. Shared bus architecture is the straightforward approach to connect several processor having private caches by means of a simple interconnection network [2]. Coherency units have to be added in order to maintain a consistent view of the shared memory for each processor. The traffic induced by cache coherency contributes to the degradation of the overall performance, so that its adverse effect should be recognized and evaluated.

A commonly used modeling technique is execution driven simulation in which a parallel program is executed on top of a simulated memory system [4]. These simulations can be extremely accurate but incur very long execution times and are highly prone to logical errors in coding. Less detailed simulation can be used when the target of the performance evaluation is the memory hierarchy and the shared bus. The method of trace -driven simulation is based on the collection of traces [4, 10] and on the utilization of the trace as input for the simulator of the memory hierarchy. For memory structures relatively accurate analytical models were developed [3] through extensive use of various queuing systems. Even though those models are not very accurate and fewer architectural details can be incorporated into, they can be used successfully at the early design stage. The most popular analytical technique for this purpose is the Mean Value Analysis (MVA) [3, 5, 7, 8]. It allows the total number of the customers to be fixed (closed queue system), and this seems to be adequate representation of the processes of self-blocking requestors [5]. Calculations of output parameters such as residency times, waiting times and utilization are shown in [3, 7, 8]. MVA is based on the forced flow that means in equilibrium output rate equals input rate. However, instantaneously, we can have input rate different from output rate, so that the instantaneous probabilities could be different from equilibrium [5]. MVA offers no possibility to study transient effects. Moreover, the assumption of exponential service times is not realistic, in fact all bus access times and memory access times are constants. The technique of continuous time Markov processes, which produces both the stationary and the instantaneous probabilities, can also be used to describe the behavior of multiprocessor implementing cache coherence [6]. Solutions for the instantaneous metrics, however, are too complex and very difficult to follow and require use of special software tools to obtain the Inverse Laplace transform.

Alternative to the approach described above is presented by considering both the processors and the memories as discrete machines, so that there is no need to approximate times by continuous distributions. We employ discrete time Markov chains to obtain the performance characteristics of the system for blocking and nonblocking caches.

2 Definition of the model

A multiprocessor consists of several processors connected together to a shared main memory by a common complete transaction bus. Each processor has a private cache. When a processor issues a request to its cache, the cache controller examines the state of the cache and takes suitable action, which may include generating bus transaction to access main memory. Coherence is maintained by having all cache controllers "snoop" on the bus and monitor the transaction. Snoopy cache-coherence protocols fall in two major categories: Invalidate and Update [9]. Invalidating protocols are studied here but the concepts can be applied with some modifications to updating protocols too. Transactions may or may not include the memory block and the shared bus. Typical transaction that does not include memory block is Invalidate Cache Copy which occurs when a processor requests writing in the cache. All other processors simply change the status bit(s) of their on copies to Invalid. If the

Manuscript received April 5, 2009
Manuscript revised April 20, 2009

memory block is uncached or not clean it can be uploaded from the main memory, but in todays multiprocessors it is rather uploaded from another cache designated as Owner (O) (cache-to cache transfer). Memory-to-cache transfer occurs when the only clean copy is in the main memory. A cache block is written back (WB) in the main memory (bus is used) when a dirty copy is evicted [6]. Apparently the bus can be considered as the bottleneck of the system.

In terms of the queuing theory processors can be viewed as customers (clients) and the bus can be viewed as a server in a closed queuing network.

The customer (processor) spends time t_{int} doing internal processing (computing, thinking) during which all memory requests are satisfied from its private cache, then it issues a blocking request, and cannot proceed unless this request is satisfied. Requests are served from another cache-cache to cache transfer- or from the main memory. After completion of the blocking request it resumes internal processing with probability *p* or resumes processing and generates a new request with probability *q (p+q=1)*. This new request corresponds to WB transaction. It does not block the customer but the server is held until completion of WB transaction therefore adding to the queue. Details on how to obtain the input parameters are given in [3, 8].

The system can be in one of the following states: 1) *N*: all *N* customers are doing internal processing; 2)*i,bl* : *i* customers are doing internal processing $(N-i$ are blocked respectively), the server is serving blocking request $(0 \le i \le N-1)$, 3) *i,wb: i* customers are doing internal processing, the server is serving WB request, and $N-i$ customers are waiting in the queue $(1 \le i \le N)$. Transfer times from/to a cache and from/to a main memory are quite different, so all possible orderings (combinations of the customers at the server should be distinguished. If the number of customers at the server (queued and served) is *m* and *i* customers are requesting cache to cache transfer the number of the combinations is $\binom{m}{i}$. The total of combinations for a system with *N* customers then will be $2x\sum_{m=0}^{N}\sum_{i=0}^{m}\binom{m}{i}$. For N=16 this number and hence the number of equations is 262,142 and this is very difficult to be handled. On the other hand we would need to know whether the requested block is available in other caches or in the main memory only. This parameter is heavily dependent on the applications running on the processors and their interactions and is difficult to be determined using trace simulation [3]. At the early phase of the design, however, precise estimation of the performance is not even needed. It would be satisfactory to predict the lowest and highest limits for the performance [10]. Below we shall refer to the case of maximal performance as HL (high limit), and to the case of minimal performance as LL (low limit). Apparently maximal performance is achieved when all blocks are supplied from other caches and minimal performance corresponds to the case of all request served by the main memory. A suitable model for these two cases is defined below.

We introduce the following notations:

t_c -cache to cache transfer time including bus delays,

t_b -bus cycle time,

t_m -memory to cache and cache to memory transfer time; write back time is equal to t_m

t_w - waiting time at the server

c_c -number of bus cycles needed for cache to cache transfer,

c_{int} -number of bus cycles needed for internal processing,

c_m -number of bus cycles needed for memory to cache and cache to memory transfer

c_w -number of bus cycles the customer waits at the server

P-state transition matrix

$\Pi^{(n)}$ -vector of the systems state probabilities after *n* transactions

$$\Pi = \lim_{n\to\infty} \Pi^{(n)} \quad .$$

Obviously $t_c = c_c t_b \quad t_m = c_m t_b$ and $t_w = c_w t_b$ (1).

We assume that requests are independent and identically distributed and apply the imbedded markov chains technique. The imbedded points are chosen to be completion of the bus cycle when all processors do internal processing, and completions of service when the server is busy serving coherence transfer or write back request. Let *x* be the probability that a particular processor is requesting a service in a bus cycle, and *y* the probability that such request is not generated (*x+y=1).* From our definition follows that $x = t_b/(t_{\text{int}} + t_w)$ (2). After substitution of the first equation of (1) in (2) we get

$x = 1/(c_{\text{int}} + c_w)$ (3).

We also introduce x_{cc} and x_{cm} as probabilities that a customer (processor) requests service during cache to cache transfer or cache to memory transfer, respectively, and y_{cc} and y_{cm} as probabilities that such request is not generated ($x_{cc} + y_{cc} = 1; x_{cm} + y_{cm} = 1$). Similarly

$x_{cc} = c_c/(c_{\text{int}} + c_w)$, and $x_{cm} = c_m/(c_{\text{int}} + c_w)$ (4).

Then the nonzero probabilities of the state transition matrix **P** for HL are given by

$$p_{N\to i,bl} = \binom{N}{i} x^{N-i} y^{i} \quad \text{for } i=0,\ldots,N \qquad (5)$$

$$p_{j,bl\to i,wb} = p\binom{j}{i-1} x_{cc}^{\;j-i+1} y_{cc}^{\;i-1} \quad \text{for } j=1,\ldots,N-1 \qquad (6)$$

[1] $\binom{n}{m} = 0$ *if* $m<0$ *or* $m<n$

$$p_{j,bl\to i,wb} = q\binom{j}{i-1}x_{cc}^{\,j-i+1}y_{cc}^{\,i-1} \quad \text{for } j=1,\ldots,N-1 \qquad (7)$$

$$p_{j,wb\to i,bl} = \binom{j}{i}x_{cm}^{\,j-i}y_{cm}^{\,i} \quad \text{for } j=1,\ldots,N \qquad (8).$$

For LL case the probabilities in (6) and (7) are

$$p_{j,bl\to i,wb} = p\binom{j}{i-1}x_{cm}^{\,j-i+1}y_{cm}^{\,i-1} \quad \text{for } j=1,\ldots,N-1 \qquad (9)$$

$$p_{j,bl\to i,wb} = q\binom{j}{i-1}x_{cm}^{\,j-i+1}y_{cm}^{\,i-1} \quad \text{for } j=1,\ldots,N-1 \qquad (10).$$

In equilibrium

$$\pi_N + \sum_{i=0}^{N-1}\pi_{i,bl} + \sum_{i=1}^{N}\pi_{i,wb} = 1, \qquad (11)$$

and Π **(P-I)**=0 (12),

where **I** is identity matrix.

The number of cycles in waiting c_w can be expressed in terms of Π**:**

$$c_w = \sum_{i=0}^{N-1}\pi_{i,bl}(N-i)c_c + \sum_{i=1}^{N}\pi_{i,wb}\left[c_c(N-i)+c_w\right] \qquad (13).$$

c_w and Π can be determined from the set of equations (3), (11), (12), and (13), which is solved numerically.

3 Numerical Results

In this section, we present the numerical results for the system with blocking caches (BL) and fully nonblocking caches (NBL) [2]. All performance characteristics are expressed as a function of the bus cycles.

The bus utilization is defined as the fraction of bus cycles that are used to service a memory. Since the bus is in use unless all processors do internal processing and no processors generate new request, the bus utilization *U* is given by:

$$U = 1-\pi_N y^N \qquad (14)$$

We set $c_{\text{int}} = 300$, $c_c = 5$, and $c_m = 15$. Plots in Fig. 1 are obtained from (14). Since nonblocking caches do not change the traffic the bus utilization has the same value for BL and NBL.

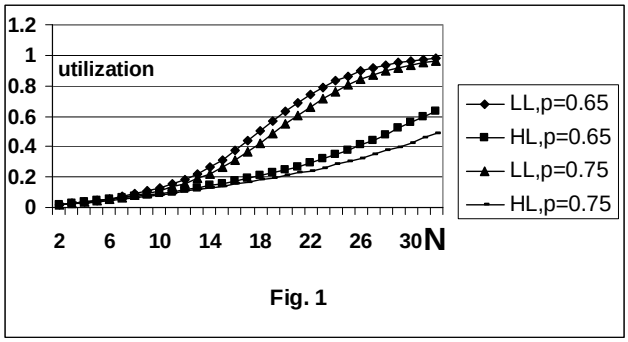

Fig. 1

For BL case the waiting time is given by (13).

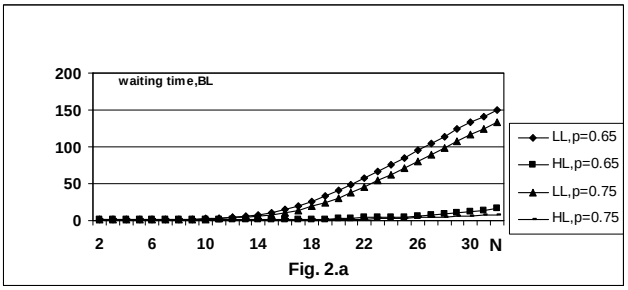

Fig. 2.a

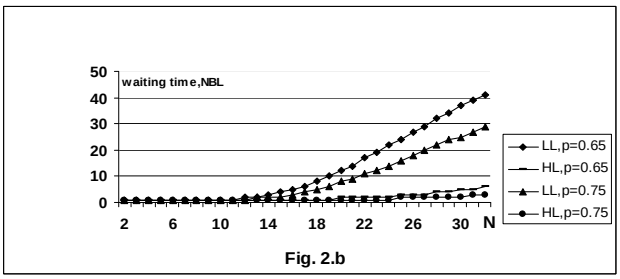

Fig. 2.b

If the processor utilizes nonblocking caches, however, it consumes data while transfer is in progress, so the waiting

$$c_w = \sum_{i=0}^{N-1}\pi_{i,bl}(N-i-1)c_c + \sum_{i=1}^{N}\pi_{i,wb}\left[c_c(N-i)+c_w\right]$$

time will be:

(15)

It is assumed that during consumption of the incoming traffic, the processor does not make coherency request.

Speedup is computed using the formula: $Nc_{\text{int}}/(c_{\text{int}}+c_w)$. It is graphically illustrated in Fig. 3.a and Fig. 3.b.

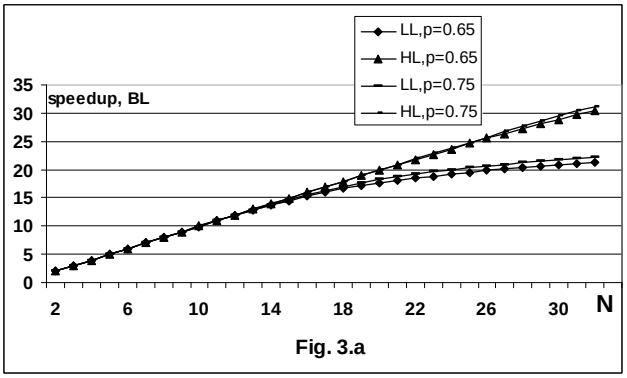

Fig. 3.a

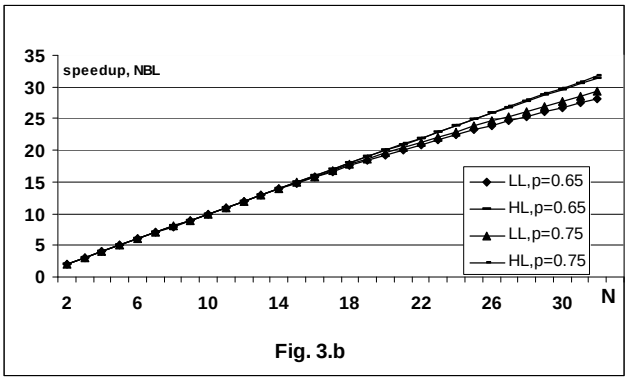

Fig. 3.b

The difference between maximal and minimal performance could be significant. For the worst case scenario, saturation might set in, while the maximal performance still increases linearly (Fig. 3.a). Use of nonblocking caches leads to essential improvement of the overall performance. Even when the bus utilization is close to 1, the system is still scalable (Fig. 1 and Fig. 3.b).

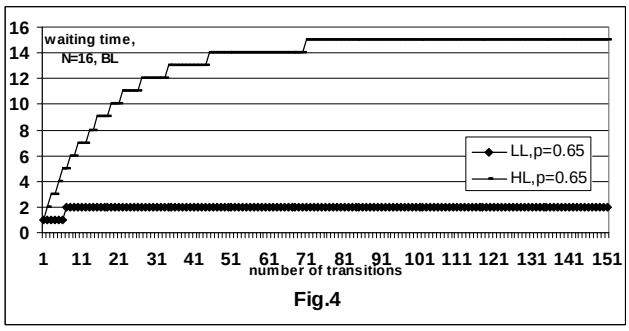

Fig.4

The vector of the state probabilities after *n* transitions $\Pi^{(n)}$ can be expressed as $\Pi^{(n)} = \Pi^{(0)} \mathbf{P}^n$ (16), where $\Pi^{(0)}$ = (1, 0,…, 0) is the initial vector. Transient waiting time now can be obtained by replacing the values of Π with $\Pi^{(n)}$ in (13) and (14). The number of transitions needed to reach equilibrium strongly depends on the transfer time (Fig. 4).

4 Concluding Remarks

This work presented a model for a shared bus, shared memory multiprocessor with private caches and captures the whole spectrum of Invalidate type cache coherence protocols. We focused on two extreme cases: all cache requests are satisfied from the main memory, and all cache requests are satisfied from another cache. We used the technique of the Imbedded Markov Chains to construct the state transition matrix in terms of the architectural and applications parameters with the waiting time as unknown. This system of nonlinear equations is solved numerically in a meaningless time. The model eliminates the need for approximating times by continuous distributions. Another merit is that it gives insight both into the stationary and transient behavior.

The model can be put to good use for evaluating the protocols more thoroughly and efficiently at the early design phase.

ACKNOWLEDGMENT

This work was supported in part by a grant from the National University of Lesotho.

References

[1] S. K. Bose, Introduction to Queuing Systems, Kluwer/Plenum Publishers, 2001

[2] J. L. Hennessy, D. A. Patterson; Computer Architecture: A Quantitative Approach, Pearson Publishers, 2003

[3] M. C. Chiang, Memory System Design for Bus Based Multiprocessor, *PhD Thesis*, University of Wisconsin, 1991

[4] P.Foglia, R. Giorgi, C.A. Prete, Simulation Study of Memory Performance of SMP Multiprocessor running a TPC-W workload, *IEE Proc. Comput. Digit. Tech.*, Vol. 151, March 2004, pp.93-109

[5] R. E. Matick, Comparison of analytic performance models using closed mean-value analysis versus open-queuing theory for estimating cycles per instruction of memory hierarchies, *IBM Journal of Research and Development*, Jul 2003

[6] A. V. Nikolov, Analytical Model For a Multiprocessor With Private Caches And Shared Memory, *Int. J. of Computers, Communications & Control*, Vol. III (2008), No. 2, pp. 172-182

[7] D. J. Sorin et. al., A customized MVA model for ILP multiprocessors, *Technical report No.1369*, University of Wisconsin-Madison, 1998

[8] D. J. Sorin et. al., Evaluation of shared-memory parallel system with ILP processors, *Proc. 25th Int'l Symp. On Computer Architecture*, June 1998, pp. 180-191

[9] J. Sustersic, A. Hurson, Coherence protocol for bus-based and scalable multiprocessors, Internet, and wireless distributed computing environments: a survey, *Advances in Computers*, vol.59, 2003, pp. 211-278

[10] T. Suh, D.M. Blough, Hsien-Hsin S. Lee, Supporting Cache Coherence in Heterogeneous Multi processor System, *Proc. Of Conference of Design, Automation and test in Europe*, Vol.2, February 16-20, 2004

Angel Vassilev Nikolov received the BEng degree in Electronic and Computer Engineering from the Technical University of Budapest, Hungary in 1974 and the PhD degree in Computer Science from the Bulgarian Academy of Sciences in 1982 where he worked as a Research Associate. In 1989 he was promoted to Associate Research Professor in Bulgaria. Dr Nikolov also served as a Lecturer of Computer Science at the National University of Science and Technology, Bulawayo, Zimbabwe and at the Grande Prairie Regional College, Alberta, Canada and as an Associate Professor at Sharjah College, United Arab Emirates. Currently he works for the National University of Lesotho, Roma, Lesotho. His research interests include computer architecture, performance evaluation of multiprocessors, and reliability modeling.