PERFORMANCE OF A CROSSBAR NETWORK USING MARKOV CHAINS

D. BENAZZOUZ[1] A. FARAH[2]

1- Laboratoire LMSS, FSI, Université of Boumerdes, 35000, Algeria
2- AUST, Faculty of Computer Science and Engineering, UAE
Phone/fax (+213) 24 81 62 65 e-mail: dbenazzouz@umbb.dz

Abstract: A performance evaluation of a crossbar network is presented using discrete-time Markov chain (MC). Identical processors, totally synchronized with system clock and communicating through common memory modules. Simple MC models the behavior of each processor. We have developed state and output equations for discrete-time state space model. The transition probabilities of transition matrices are computed. The memory contention situation of the multiprocessor systems is considered. We show that the network is reliable for less then 100 processors. For network larger than 100 processors, a considerable degradation performance is observed which is due to contention.

Keywords: Crossbar - Performance analysis - Markov chains – Multiprocessor systems.

1- INTRODUCTION

Generally in multiprocessing systems, the processors can communicate and cooperate at different levels in solving a given problem, i. e., by sending messages or by sharing memory. Parallel systems are said to be tightly coupled if there is many processor interactions via shared memory. The speed of the machine is restricted by the memory bandwidth and hence by the interconnection network (IN) topology, and an IN with a dynamic topology is required. ALICE [Harrison and Reeve, 1987], designed to execute functional languages in parallel [Fiel and Harrison, 1988], and the NYU ultracomputer [Gottlieb et al, 1983] are examples of tightly coupled multiprocessor systems.
The performance studies of tightly coupled multiprocessor systems have generated a great interest [Marsan and Gerla, 1982]. The principle characteristic of a multiprocessor system is the ability of each processor to share a single main memory. The partitioning of main memory into several independent memory modules (MM), that can be in operation simultaneously, is known as memory interleaving. A memory system consisting of M memory modules (M-way interleaving) can be used to control severe performance degradation of the memory system. The interference occurs when two or more processors simultaneously attempt to access the same MM.
The mathematical models used to evaluate this class of multiprocessor systems are based on discrete-time Markov chains. A limit on the use of Markovian models of complex computer systems comes from the fact that their direct construction is practically very difficult. Various approaches were examined [Skinner and Asher, 1969; Bhandarkar 1975; Ran, 1979].
The number of states increases very rapidly with system size. The explosive growth is due to the detailed information that the state must record the exact status of the queues at each server in the system. MC approach has been very successful technique for modelling, analysis and design of various kinds of systems [Florin et al. 1991; Benazzouz and Farah, 1998].
In this correspondence we develop a discrete time MC model for crossbar multiprocessor systems. It is assumed that the processors share M-way interleaved memory and can access any MM through the network. The entire system is synchronized with a system clock whose time period is referred to as the system cycle. The operation of the system can be assumed as under. At the beginning of each system cycle the processors are permitted to make selections of a MM at random. In case more than one request is made to same MM, the information is supplied to one of the processors selected at random, and the remaining processors permitted to make a retry at the next cycle. The behavior of the processors is considered to be independent but statistically identical. The entire process is thus stochastic in nature, and permits us to use MC to represent the state transition behavior of the processors. Section II describes the multiprocessor system interconnects. A detailed description of the crossbar network is presented in this section. The proposed model and the mathematical approach were developed in section III and IV respectively. The paper concludes with section V.

2- MULTIPROCESSOR SYSTEM INTERCONNECTS

Parallel processing demands the use of efficient system interconnect for fast communication among multiprocessors and shared memory, I/O, and peripheral devices. Hierarchical buses, crossbar switches, multistage and single stage networks are often used for this purpose. Switched networks provide dynamic interconnections between the processors and MM. Many classes of switched networks may be found in literature particularly the single stage interconnection network (SSIN) and multistage interconnection network (MIN) [El-Reweni and Lewis, 1997]. The crossbar switch network is a SSIN, nonblocking permutation network.

2.1- Crossbar Network

Crossbar networks provide the highest bandwidth and interconnection capability. A crossbar network can be visualized as a single stage switch network. Each crosspoint switch can provide a dedicated connection path between a pair. The switch can be set on or off dynamically upon program demand. The crossbar switch network configuration is illustrated in Fig.1. To build a shared memory multiprocessor, one can use a crossbar network between the processors and MM (Fig.1). This is essentially a memory access network. The Cmmp multiprocessor[Wulf and Bell, 1972] has implemented a 16x16 crossbar network which connects 16 PDP 11 processors to 16 MM, each of which has a capacity of 1 million words of memory cells. The 16 MM can be accessed by at most 16 processors simultaneously. A crossbar network is cost effective only for small multiprocessors with a few processors accessing a few MM. A single stage crossbar network is not expandable one it is built. All processors can send memory requests independently and asynchronously. This poses the problem of multiple requests destined for the same MM at the same time. In such cases, only one of the requests is serviced at a time.

3- PROPOSED MODEL

One set of characteristics of a system is the states of the system. If we know all possible states of the system, then the behaviour of the system is completely described by its states. A system may have finite or infinite number of states. Here, we are concerned with only finite state systems. Suppose X(t) describes the state of the system and has n values. That is, at a given time, $X_1(t)$, $X_2(t)$,… $X_n(t)$ are the possible states of the system. $X_i(t)$ could be demand access to the MM or a process with the private memory (PM) of the processor (a PM is an interne memory within the processor). The system will move from one state to another with some random fashion. That is, there is a probability attached to this. Let us suppose that p(t) represents the probability distribution over X(t) (note: X(t) and p(t) are vectors of size nx1) i.e. $p_1(t)$ is the probability of finding the system in state $X_1(t)$. In general, the predictive distribution for X(t) is quite complicated with p(t), being a function of all previous state variables X(t-1), X(t-2) and so on. However, if p(t) depends only upon the **preceding** state then the process is called Markov process.

A Markov process is a mathematical model that describes, in probabilistic terms, the dynamic behavior of certain type of system over time. The change of state occurs only at the end of the time period and nothing happens during the time period chosen. Thus, a Markov process is a stochastic process which has the property that the probability of a transition from a given state $p_i(t)$ to a future state $p_j(t+1)$ is dependent only on the present state and not on the manner in which the current state was reached.

The multiprocessor arrangement under consideration is shown in figure1. It consists of P processors and M memory modules interconnected through crossbar. Besides an M-way interleaved shared main memory, each processor has its own PM.

Contention problem arises when a message is attempted to be written in (or read from) a common MM by more than one processors. In crossbar multiprocessor systems (Fig.1) two types of possible interference can occur;

- When more than one processor attempts to access an idle MM at the same time.
- When a processor attempts to access a busy MM, (the processor is executing in its PM).

Due to this interference, a subset of processors might be blocked, thus giving degradation in the performance. The state space X(t) of a processor in crossbar system can be a P valued random variable, taking only 3 values as a column vector;

$$X(t) = [\ X_1(t), X_2(t), X_3(t)\ ] \qquad (1)$$

$X_1(t)$ is the probability that the processor is in active state. It means that the processor is busy with its own private memory.

$X_2(t)$ is the probability that the processor is in accessing state.

$X_3(t)$ is the probability that the processor is queued at the required MM, due to nonavailability of MM.

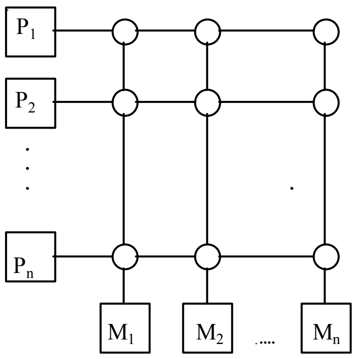


Fig. 1 Interprocessor-memory crossbar network built in C.mmp multiprocessor at Carnagie Mellon University (1972).

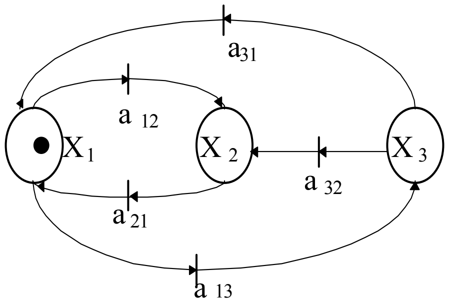


Fig. 2. Markov chain for a single processor.

For the sake of simplicity MC for the single processor is shown in Fig. 2 and can be coupled P times for P processors. The state transition behavior of each processor can be understood as follows.
The processor stays in active state for time duration equal to its processing time. It will enter the state $X_2(t)$ (accessing state) only if MM is idle and no other processor is a candidate for that module. The processor stays in this state for a time equal to the memory access time (processor-memory connection time is assumed to be zero). From accessing state it can go back to the active state after completing the memory access. Processor enter the queued state if MM has more than one simultaneous request from other processors and one of those requests is successful in getting access to the MM. In this state the processor has to wait in the queue until memory is available and then it goes back to the active state.

4- MATHEMATICAL APPROACH

A vector Markov process can be represented by the help of the discrete time state equation [Stark and Woods, 1986] provided that the input r(t) is independent of previous state X(t-1). The state and output equations are;

$$X(t+1) = AX(t) + Br(t) \qquad (2)$$

where X(t) and X(t+1) are the probabilities that the system lies in specified state of the processor during current system cycle and the next cycle respectively.
The input probability vector r(t) can be represented as a B-valued random variable and can be considered as;

$$r(t) = [\, r_1(t), r_2(t) \,] \qquad (3)$$

where the probability $r_1(t)$ represents that the requesting MM is not available and $r_2(t)$ is the probability that the requesting MM is available.
The probability of the successful request (requests for which MM is idle) can be considered as an output C(t) (which can be defined as P-valued random variable for the P processors) of the system.

$$C(t) = DX(t) + Er(t) \qquad (4)$$

The matrices A, B, D, and E are of appropriate dimensions, and their components are the transition probabilities a_{ij}, b_{ij}, d_{ij}, and e_{ij} respectively. These components are defined as follows;
a_{ij}: the transition probability that the processor currently in state i goes to state j in the next system cycle.
b_{ij} : the transition probability that MM are available for transition to next state.
d_{ij} : the transition probability that the processor currently in any one of the states, active, accessing, queued, is successful to access requested MM in the next system cycle.
e_{ij} : the transition probability that the successful request i in the next system cycle is effected by the input j.
In order to calculate the transition probabilities following terms are introduced. The probability that a processor makes a request to access a particular MM at the beginning of a bus cycle is denoted by R. This is the probability of leaving states, active, and queued to access a particular MM. Therefore R is given by;

$$R = \frac{1}{M}(g_1 + g_3) \qquad (5)$$

where g_i is the probability of leaving state i at the beginning of a system cycle. As the states of the processors are represented by irreductible ergodic MC, the g_i can be defined as the rate of leaving state i. Thus g_i can be written as $g_i = \dfrac{K_i}{T_i}$
where T_i is the average time in any one of the states which is at least one system cycle and K_i is the limiting probability of being in state i. The probability that the processor finds a MM busy at the beginning of a system cycle and is also not

available in the next system cycle is denoted by BM and calculated as follows;

$$BM = \frac{P-1}{M}(K_2 - g_2) \qquad (6)$$

That means one of the (P-1) processors is accessing that MM and is not going to release it in the next system cycle. Where K_2 is the probability that the processor is accessing MM and (K_2-g_2) is the probability that the processor is accessing and not going to release that MM in the next system cycle.

The term α is the probability that the memory request initiated by a processor finds the MM idle. The probability that a processor will not request a particular MM is (1-R), the probability that none of the P processors requests that MM is $(1-R)^P$, and therefore the probability that a particular MM is requested by at least one of the processors is $S=1-(1-R)^P$. The number of processors which request that MM at the beginning of the system cycle is PR. Then the value of α is given by;

$$\alpha = \frac{S}{PR} \qquad (7)$$

The transition probability matrix of the model is denoted as T and given as follows,

$$T = \begin{bmatrix} -2(1+BM+\alpha) & (1-BM)-\alpha & (1-\alpha)-BM \\ 1 & -1 & 0 \\ \alpha & 1-BM & (BM-1)-\alpha \end{bmatrix}$$

Using Eqs. 5, 6, and 7, Eqs. 2 and 3 can be written as follows;

$$\begin{bmatrix} X_1(t+1) \\ X_2(t+1) \\ X_3(t+1) \end{bmatrix} = T \begin{bmatrix} X_1(t) \\ X_2(t) \\ X_3(t) \end{bmatrix} + \begin{bmatrix} (1-\alpha) & 0 \\ 0 & \alpha \\ (1-\alpha) & 0 \end{bmatrix} . \begin{bmatrix} r_1(t) \\ r_2(t) \end{bmatrix} \qquad (8)$$

$$C(t) = \begin{bmatrix} (1-BM)\alpha & 1 & (1-BM) \end{bmatrix} \begin{bmatrix} X_1(t) \\ X_2(t) \\ X_3(t) \end{bmatrix} + \begin{bmatrix} 1-\alpha & \alpha \end{bmatrix} \begin{bmatrix} r_1(t) \\ r_2(t) \end{bmatrix}$$

To solve these equations let's suppose a discrete time as said earlier, which means that we have a stationary homogeneous MC, where the probabilities $X_1(t)$, $X_2(t)$, and $X_3(t)$ are independent of time. Therefore, we can say that;

$$\begin{bmatrix} X_1(t+1) \\ X_2(t+1) \\ X_3(t+1) \end{bmatrix} = \begin{bmatrix} X_1(t) \\ X_2(t) \\ X_3(t) \end{bmatrix} \qquad (9)$$

We can consider the fundamental relation to this linear system that consists of the sum of the state probabilities which is equal to 1. This relation is written as follows;

$$X_1(t) + X_2(t) + X_3(t) = 1 \qquad (10)$$

Therefore, we can rewrite Eq.(8) by considering Eq.(9) and replacing one row by Eq.(10), thus we obtain the following expression,

$$\begin{bmatrix} 1 & 1 & 1 \\ -1 & 2 & 0 \\ -\alpha & BM-1 & 2-BM+\alpha \end{bmatrix} \begin{bmatrix} X_1(t) \\ X_2(t) \\ X_3(t) \end{bmatrix} = \begin{bmatrix} 1 \\ \alpha r_2(t) \\ (1-\alpha)r_1(t) \end{bmatrix} \qquad (11)$$

We have developed an algorithm based on Gauss method to solve these equations. We calculate the probability states for one processor, then for P processors and try to see if these probabilities change as function of the number of processor. From fig. 3, we can conclude that these probability states $X_1(t)$, $X_2(t)$ and $X_3(t)$ stay almost constant. This is a very important result to proof the validity of the method since we can generalize the approach to P processors.

Two parameters were taken to evaluate the performance of a crossbar network. These parameters are the bandwidth (BW) and the probability of connection C(t).

The crossbar model will be analyzed under the same assumptions given by Agrawal [Bhuyan and Agrawal, 1983] which are:

1- The operation is synchronous; i.e., the messages begin and end simultaneously.

2- Each processor generates a random and independent request. The requests are uniformly distributed over all the memory modules.

3- At the beginning of a cycle, each processor generates a new request with a probability Pm. Thus is the average number of requests generated per cycle by each processor.

4- The requests which are not accepted are ignored. The requests issued at a cycle are independent of the requests issued in the previous cycle.

When the requests are random, it is possible for two or more processors to address the same memory module. Assumptions 1-4 are there to simplify the analysis.

The bandwidth (BW) and the probability of acceptance (Pa) of MxP crossbar network are presented by Agrawal and adapted to the model. BW is defined as the expected number of memory requests accepted per cycle. The BW and the Pa of the proposed model are given as follows;

$$BW = M - M\,(1 - C(t)\,/\,M)^P$$

where M is the number of MM and P is the number of processors.

Pa defines the probability that a request will be accepted;

$$Pa = BW/(C(t).P)$$

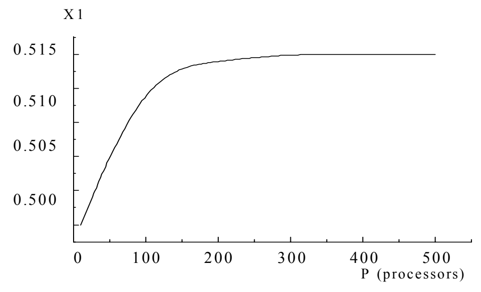
Pa is defined as the ratio of the expected BW to the expected number of requests generated by cycle.

5. RESULTS

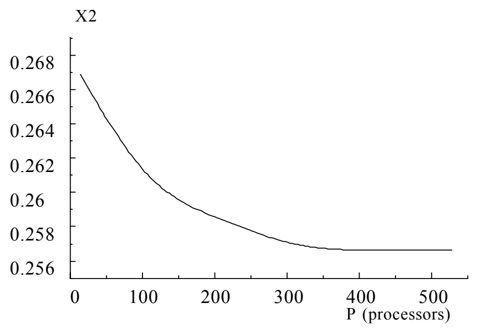
In this section, we present performance figures for a crossbar network in presence of contention. The simulation results are obtained from the proposed model where the number of processor is 500. Figures 3 shows the variation of the probability states of X_1, X_2 and X_3. For processors $P \leq 100$, we obtain an increase sharp of the state probabilities of X_1, whereby a decrease sharp of the state probabilities of X_2 and X_3 is observed. We can say that, for low processors, most of the state probabilities show that the processors are in the active states. They are busy with there private memories. Almost, with the same state probabilities, the processors are either in the accessing or in the queuing states. For processors $P>100$, the three state probabilities stay constant, $X_1=0.513$, $X_2=0.258$ and $X_3=0.229$. It is clearly shown in Fig.3d. We believe that there exits a degradation in performance and the system saturate in this range of processors. In the range of memories $M \leq 100$, a significant decrease of the probability of connection C(t) and probability of acceptance Pa are shown in Fig.5 and 6 respectively. For M >100, the memories are continuously busy and the C(t) and Pa are small and stay almost constant. In this range of memories, the processors cannot access easily the memories. This causes a degradation performance which is due to the network size.

6. CONCLUSION

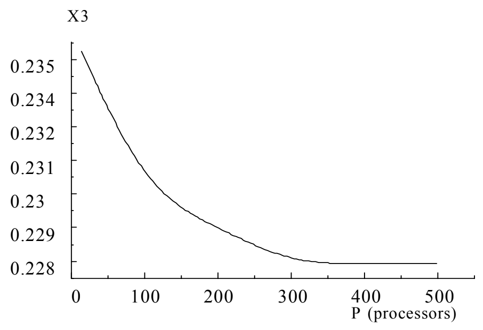
We have presented analytic model for blocking probability of crossbar network. The model is based on discrete-time MC under the assumption of random memory requests.
We believe that, the proposed analytic model of the crossbar can be used to adapt analytic models for the blocking probability of any arbitrary multistage interconnection network. The concepts developed here can later on be used to study the behavior of complex multiprocessor systems to resolve the memory contention problems under other considerations.



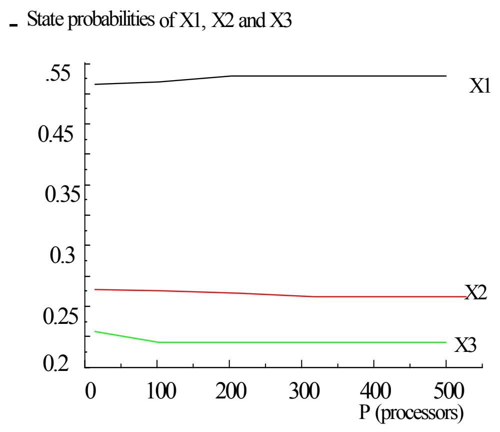
a) State probabilities of X_1



b) State probabilities of X_2



c) State probabilities of X_3



d) State probabilities of X_1, X_2, X_3

Fig 3. Variation of the probabilities states with ($K_1=K_2=0.3, K_3=0.4, R_p=D=1, r_1=0.2, r_2=0.6$)

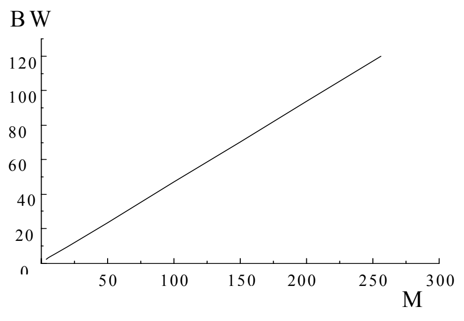


Fig. 4 Bandwidth of MxP networks

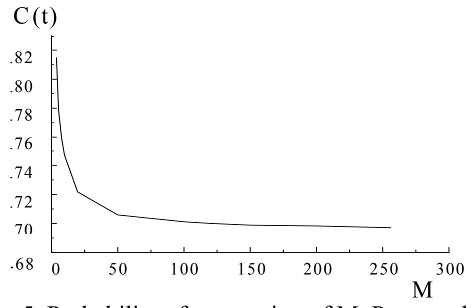


Fig. 5 Probability of connection of MxP networks

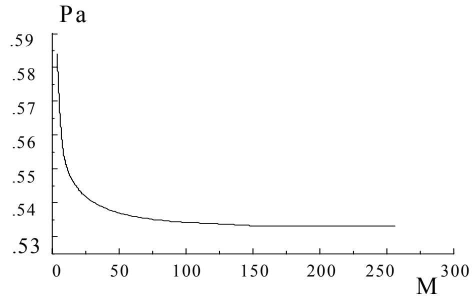


Fig. 6 Probability of acceptance of MxP networks

REFERENCES

Harrison, P. G., and Reeve, M. J. 1987, ''The parallel graph reduction machine, ALICE''. *Proc. workshop* on graph reduction, Santa Fe, LNCS, N°279. Spring-verlag New York/Berlin.

Fiel, A. J., and Harrison, P. G. 1988, ''Functional programming'', addison-wesley, New York.

Gottlieb, A. et al. 1983,''The NYU ultracomputer designing an MIMD shared memory parallel computer'', *IEEE Trans. comput*. C32, 2, pp173-189.

Marsan M.A. and Gerla M. 1982, ''Markov models for multiple bus multiprocessor systems'', *IEEE Trans. Comput.* Vol. C.31, pp239-248.

Skinner C. E. and Asher J. R.. 1969, ''Effects of storage contention on system performance'', *IBM system J.* Vol. 8, pp319-333.

Bhandarkar D. P. 1975, ''Analysis of memory interference in multiprocessors'', *IEEE Trans. Comput.,* Vol. C-34, pp897-908.

Ran B. R. 1979, ''Interleaved memory bandwidth in a model of a multiprocessor computer system'', *IEEE Trans. Comput*., Vol. C-28, N°9.

Florin G., Fraize C. and Natkin S. 1991, ''Stochastic Petri Net: Properties, Applications and Tools'', *Microelectron. reability*, vol. 31, N°4, pp669-697.

Benazzouz D.and Farah A. 1998, ''The use of Petri nets in the performance evaluation of shuffle-exchange network under uniform traffic distribution'', *the Arabian Journal for Science and Engineering, AJSE,* vol. 23, n° 2B, pp. 253-263.

Benazzouz D. and Farah A. 1998, ''Performance evaluation of baseline MIN'', *in Proc. IMACS-IEEE on Computational Engineering in Systems Application'*, Nabeul-Hammamet, Tunisia, pp.350-356.

Stark H., and Woods J.W. 1986, ''probability random processes and estimation theory for engineers'', New Jersey: *Prentice-Hall, Inc*.

Wulf W.A. and. Bell C. G 1972, ''C.mmp-A multi-miniprocessors'', *Proc. Fall joint Compt. Conf.,* pp765-777.

El-Rewini H. & Lewis T. G. 1997, ''*Distributed and parallel computing''*, *Ed. Manning,* pp. 102-103.

Bhuyan L.N. and Agrawal P. 1983, ''Design and performance of generalized interconnection networks'', *Trans. on computers* vol. C-32, no12, pp.1081-1090.

BIOGRAPHY:

Dr.Djamel BENAZZOUZ graduated in 1982 from the National Institute of Electricity and Electronics (INELEC) Boumerdes, Algeria. He joined industry as maintenance engineer in the two major Algerian Companies: Sonatrach (Petroleum Industry) and Sonelgaz (Electric Utility Company). He returned to research and Education since 1986 at the National Institute of Mechanical Engineering which became in 1998 University of Boumerdes. He received his Magister degree in applied electronics in 1991 at INELEC and his Doctorate d'Etat in 1999 at Ecole Nationale Polytechnique, Algiers. He worked as associate Professor in 1998 at the university of Boumerdes. He has been heavily involved in the field of microprocessor-based systems. His research interests include architecture digital system design, verification and test of digital circuits, hardware and software and the identification systems using neural network and fuzzy logic.