Proceedings of the 1985 Winter Simulation Conference
D. Gantz, G. Blais, S. Solomon (eds.)

GENERAL SIMULATION OF MULTIPROCESSOR INTERCONNECTION NETWORKS

Nizar Al-Holou
Electrical Engineering
University of Dayton
Dayton, OH 45469

Jay B. Ghosh
Decision Sciences
University of Dayton
Dayton, OH 45469

Dana B. Rogers
Electrical Engineering
University of Dayton
Dayton, OH 45469

ABSTRACT

Multiprocessor systems are high-performance systems that are making a rapid foray into the mainstream of computing. To realize their full potential, however, consideration must be given to their design. The design of the interconnection networks is particularly critical. Performance studies based on analytical and/or simulation models are conducted to evaluate alternate designs. The analytical models are either so complex that they are intractable, or are based on assumptions so restrictive that they are unrealistic. Simulation models, on the other hand, are simple and can capture almost any level of realism. Their problems lie in high costs for their development and the specificity of their results. The research reported here addresses these issues in specific reference to the simulation of multiprocessor interconnection networks. General simulation is proposed to counter the development cost, and simulation-based meta-models are suggested for overcoming the lack of generality in simulation results. This paper describes a general simulation model as implemented in SLAMII.

INTRODUCTION

Multiprocessor systems, with their high speed and flexibility, have been in the center of developments in the field of computing for some time now. With the availability of high speed processors and memories at lower costs, large-scale multiprocessor systems are becoming increasingly feasible. But to realize the full potential of such systems, more attention needs to be directed to their architectures. Among other things, a multiprocessor system consists of a set of processors, a set of memory modules, and a communication interface between the processors and the memory modules, which is commonly called the interconnection network (Fig. 1). As the size of the multiprocessor system increases, the interconnection network either dominates its performance or its cost [1].

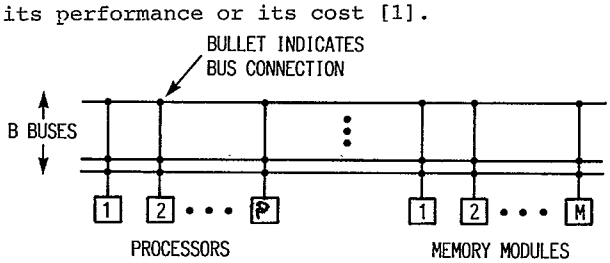

Figure 1. Multiprocessor System

Performance evaluation studies are essential to the success of a system under design. Typically, a system is evaluated based on one or more criteria which may or may not be related. Examples of such criteria include processing power and memory bandwidth. The selection of a set of criteria in a given study is usually specific to the needs of the study. There are two approaches to performance evaluation: the empirical and the analytical. The empirical, more often than not, connotes to simulation while the analytical may involve mathematical techniques such as queueing theory and Markov analysis [2,3,4].

This paper recommends the use of simulation in performance evaluation of multiprocessor systems. The following section provides the rationale for simulation. The one after that describes the requirements for a general simulator of multiprocessor systems, reviews the various elements involved, and explains how such a simulator may help in the design of multiprocessor systems. The next section details the implementation of a general simulator in SLAMII. Some results are then presented. Finally, conclusions are made on the general simulation approach.

RATIONALE FOR SIMULATION

The analytical approach to performance evaluation suffers from a series of problems. First of all, all analytical models simplify the complexities of the system to suit the applicability of their methemetical tools. This results in assumptions that are often too restrictive and may actually detract from the realism captured by the models. Even with all their simplifications, the mathematics of the analytical models can get intolerably complex to the extent of being intractable at times. In situations where the analytical models lead to solutions, asymptotic or otherwise, the solutions may only provide partial descriptions of the performance measures of interest.

Simulation, on the other hand, is very flexible and can accommodate models of arbitrary levels of complexity. It also provides more complete descriptions of the performance measures. Two charges are normally levied against simulation. The first points to the expenses involved in developing and executing simulation models. The second has to do with the accuracy and precision of the results

from the simulations. The "development" component of the first charge can be addressed by introducing the concept of a general simulation model which can be readily adapted to specific instances. "Execution" no longer remains a major concern due to the low cost and high speed of computation that are available today. One can also resort to simulation schemes that are computationally efficient. Finally, statistical procedures exist that can give the results of simulation a level of accuracy and precision that is adequate for most purposes. There is yet another drawback for simulation in that the results of simulation are not general. This lack of generality may, however, be overcome by the use of simulation-based meta-models.

The cost and the complexity of multiprocessor systems prompt proper performance evaluation, and it seems that simulation offers the ideal premise for such evaluation. But for simulation to be deemed viable, there must exist a general model for multiprocessor systems and a framework for analysis with that model.

TOWARDS A GENERAL MODEL

The use of simulation for evaluating the performance of multiprocessor systems is not entirely new. Some researchers have used simulations to validate the findings of their analytical models [e.g., 5 & 6], whereas others have used it as the means for evaluating specific systems with specific performance measures [e.g.,7]. The aim of this paper is broader as it pursues a general model which will simulate multiprocessor systems for studying any performance measure of any configuration under any operational scheme.

A general model of multiprocessor systems must satisfy an array of requirements. Specifically, it should be able to

1. accommodate any network architecture,
2. adapt to any operational assumptions,
3. effect any operational scheme,
4. estimate any performance measure, and
5. study performance under faulty conditions.

These requirements are dealt with in more detail in the later paragraphs.

Network Architecture

It is appropriate at this point to state that this paper focuses on the bus-oriented architectures that have traditionally received more attention. With this in mind it is recognized that a multiprocessor system is essentially a collection of processors, memory modules, communication links (buses), and arbiters that revolve the conflicts for the memory modules and the buses. Depending upon the number of available buses (B) with respect to the number of processors (P) and memory modules (M) and how these buses are allocated to the memory modules, one of various interconnection networks is realized.

1. For B = 1, one has a single shared bus.
2. For B > 1, one may have a multiple bus or a partial bus.
   a. In a multiple bus configuration, all the buses have access to all the memory modules. If B < min (M,P) one is said to have a bus-deficient system, while if B > = min (M,P) one is said to have a bus-sufficient (cross-bar) system.
   b. In a partial bus configuration, the buses are divided into G groups, and the buses in a given group are dedicated to a given group of memories.

It is evident that the single-shared bus and the cross-bar are the opposite ends of the performance spectrum. The performance of the single-shared bus degenerates due to contentions for both the memory modules and the buses. On the other hand, the cross-bar only experiences memory contentions. The bus-deficient and the partial bus architecture belong somewhere in the middle of the performance spectrum. The motivation behind using them is that the cross-bar may be grossly under utilized, and its performance or at least an equivalent performance may actually be obtained from using far fewer buses. The idea usually is to identify bus-deficient and partial bus architectures that exhibit cross-bar like performance. This is reasonable because the cost reduction could be substantial.

Operational Assumptions

Models of multiprocessor performance have to reasonably mimic its operations. It becomes necessary at this stage to specify the operational characteristics in the form of assumptions.

One such assumption leads to classification of multiprocessor models as being synchronous and asynchronous. A synchronous model assumes that the processing cycle and the memory cycle are controlled by the system clock, and are of fixed duration. As a result all processors may request memory at the same instant. An asynchronous model, in contrast, allows the processing cycle and the memory cycle to be of arbitrary durations. The processor requests are thus no longer synchronozed.

A second assumption classifies multiprocessor models as uniform reference models (URM) and local reference models (LRM). The uniform reference model assumes that a processor selects any memory with equal likelihood at the beginning of a memory cycle, independent of any selection it has made in the past. The local reference model reflects the situation when selection of a memory at the beginning of a cycle may have a higher likelihood as it has been selected in the last cycle.

Operational Schemes

Whereas the assumptions describe characteristics that are intrinsic to multiprocessor systems, the schemes describe characteristics that are introduced by the designer. In this case, an obvious scheme is that of arbitration. A number of arbitration schemes are

possible such as cyclical and random. Different schemes may have different effects on the performance. A designer may like to identify the scheme that is most appropriate for him before he actually incorporates it into his design.

Performance Measures

Central to multiprocessor modeling is the desire to estimate performance measures of interest. The interest in a set of performance measures is usually specific to a study. The measures that have some common appeal are:

1. memory bandwidth,
2. processing power/processing efficiency,
3. waiting time for a processor to access memory, and
4. utilization of the network.

Fault Tolerance

Multiprocessor systems are hardly failure-proof in their operations. One or more component(s) may fail from time to time. Such failures either result in some degradation in the performance of the system or in a total failure of the system. It is, therefore, important to study the effect of component failures on the performance of the system over time.

The idea behind the general model is that the designer need not be an expert in the use of the tools of performance evaluation. He should, however, be able to adjust parameters of the model to realize the model that is appropriate for him. He should also be able to study performance under various architectures and schemes, and behavior under failure.

IMPLEMENTATION IN SLAM II

To implement the concepts of the general model described above, a simulation language, particularly one with powerful modeling capabilities, is called for. SLAM II [8] and its network and discrete-event worldviews seem to be best suited for this purpose, and is consequently chosen for the research leading to this paper. The network representation in SLAM provides intuitive appeal and ease of modeling. The discrete-event feature adds to this the flexibility required to model the complexities of the multiprocessor systems.

The model has been implemented to accommodate any bus-oriented multiprocessor system with up to 32 processors, 32 buses and 32 memory modules [32 X 32 X 32] and can be adapted to reflect any assumption,effect any operational scheme, and estimate any performance measure. For the purposes of illustration, however, a 4 X 4 X 4 system will be used throughout this section.

Modeling the Processors

Figure 2 shows the model of the processors. The CREATE node (C) is used to generate one entity and release it at time zero with a maximum of N emanating activities. The ASSIGN nodes A1 through A4 are used to assign

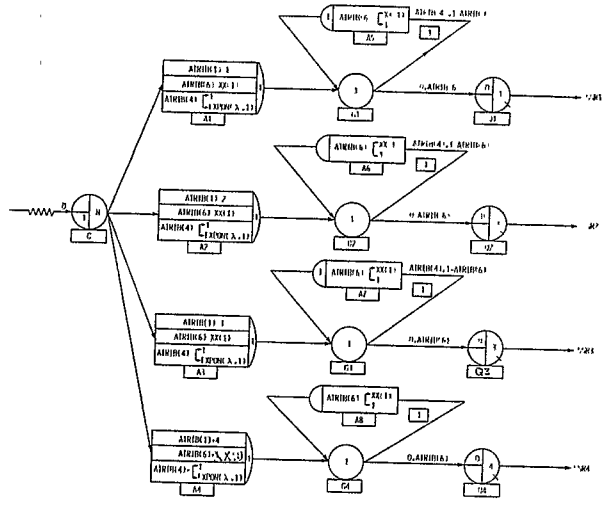

Figure 2.
Processor Model

different attributes and global variables in the following manner.

ATRIB(1) = 1,2,3, or 4: defines processor number 1,2,3, or 4.

XX(1),.....XX(4): probability of requesting shared memory R.

ATRIB(4): processing time - 1 for discrete systems and EXPON (λ) for continuous systems.

G1,.....G4 are GOON nodes. Q1,.....Q4 are QUEUE nodes where the processors wait to access memory.

In the synchronous mode, a processor either executes a program in its private memory with probability [1 - XX(N)] for time"ATRIB(4)" or request shared global memory modules with probability XX(N).

Modeling the Memory Modules

Modeling of memory modules is shown in Figure 3. Each processor may request a memory with equal probabilities (URM) or different probabilities (LRM). The SELECT nodes (SSR1.....,SSR4) select one or more of the memory modules with probabilities XX(5),.....XX(8). It is

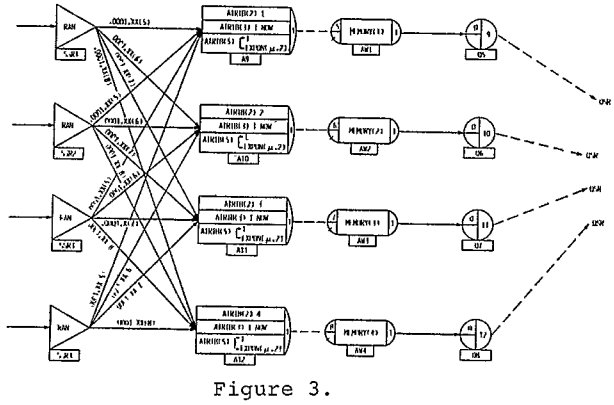

Figure 3.
Memory Model

assumed that the propagation time and the arbitration time is equal to .001 time units which is a realistic assumption. ATRIB(2) is used to define which memory has been requested. The service time at the memory is defined by ATRIB(5) - 1 time unit for discrete systems and EXPON(μ) for continuous systems. The AWAIT nodes AW1 through AW4 are used to delay requests for memory modules M1 through M4 respectively until the memory modules are free.

The QUEUES Q5 through Q8 are treated as dummies in order that the SELECT node (QSR) can select one or more of the queues which have entities waiting for service.

Modeling the Buses

The buses are modeled, as shown in Figure 4a,

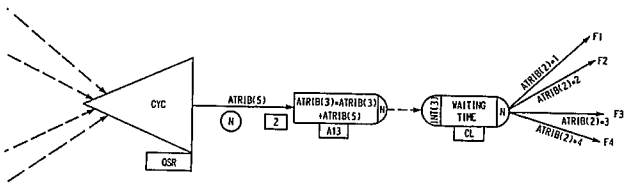

Figure 4a.
Bus Model (Multiple Bus)

as N servers. The service time is defined by ATRIB(5) on ACTIVITY number 2. The ASSIGN node (A9) is used to find the waiting time of each processor to access memory, and the COLLECT node (CL) is used to collect the waiting time statistics and develop a histogram. Figure 4b, shows modeling of partial buses

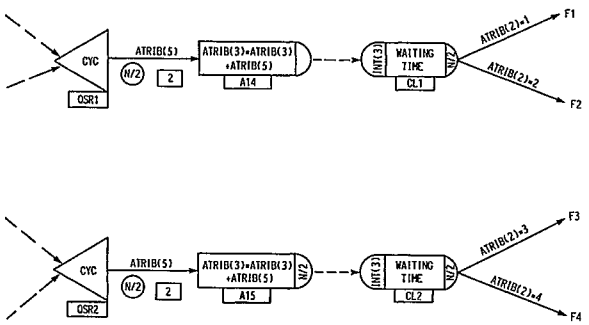

Figure 4b.
Bus Model (Partial Bus)

where the buses are divided into 2 groups (G = 2), each having a number of servers.

Modeling the Processor/Memory Cycle

Figure 5 shows the processor memory cycle model. All used resources (memories) are freed at nodes F1, F2, F3 and F4. At ASSIGN node (A11) ATRIB(2) is cleared. After each processor has been served by a memory, it cycles to ASSIGN nodes A1 through A4 in accordance with the value of ATRIB(1). There, a processor may either process a program in its private memory or request shared memory access once again.

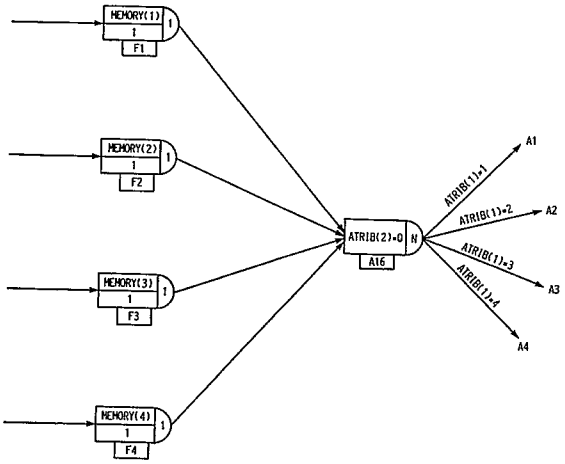

Figure 5.
Processor/Memory Cycle Model

RESULTS

The model has been validated against previously published simulation results for uniform reference and synchronous operation [7]. The results of the model have also been found to be in close agreement with the results of an approximate analytical model [6]. Table 1 below presents the results with the model for uniform reference and synchronous operation. The memory request probability (R) is equal to 0.5.

Number of Buses (B)	Number of Processors/Memories (P = M)			
	2	4	8	16
1	.425	.254	.128	.064
2	.47	.416	.241	.123
3		.44	.355	.187
4		.447	.416	.247
5			.428	.314
6			.432	.37
7			.432	.407
8			.433	.421
9				.425
10				.426
11				.4265
12				.4265
13				.427
14				.427
15				.427
16				.427

Table 1. Processing Efficiency for R = 0.5

The results conform to intuitive understanding of multiprocessor performance. The processing efficiency degrades with increase in the number of processes, and with decrease in the number of buses and memory modules. Figure 6 illustrates these facts graphically. One sees with some interest that processing efficiency changes very little as the number of buses are reduced from the cross-bar count to nearly half that count. This ovservation supports the idea that the corss-bar is under utilized and a bus-defficient or partial bus

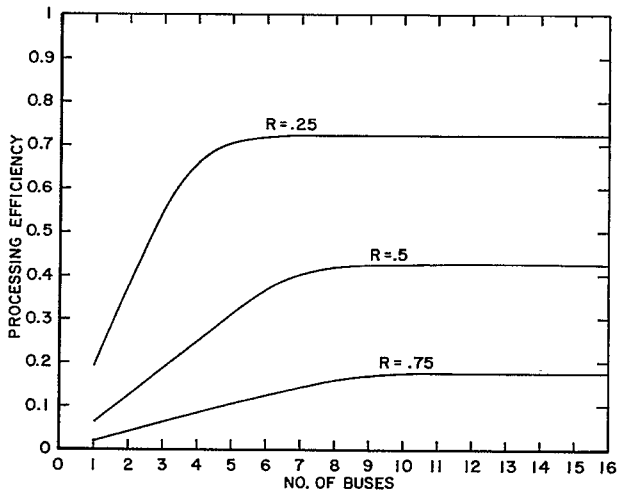

Figure 6.
Processing Efficiency Vs. No. of Buses

configuration may very well suffice for all practical purposes.

DISCUSSIONS

The major advantage of the general model is that it may allow a naive user with access to SLAM II but with minimal knowledge of the language to use it. A user's manual is being prepared to facilitate use.

It is also true that the implementation in SLAMM II is not without problems. The SLAM II code is often inefficient, and can sometimes be unpredictable in the way it handles simultaneous events. An alternative to simulations in SLAM II is simulating a Markov-chain model of the multiprocessor system. Such a simulation promises to be more efficient computationally and statistically.

A spin-off from the general model has to do with meta-models. The general model provides an experimental base for a wide range of parameter settings, thus making possible the extraction of meta-models -- models that generalize the results of simulation. A meta-model, properly identified can be as powerful as analytical formulae, only simpler. The next phase in this research is involved with building meta-models for multiprocessor systems.

REFERENCES

1. Gentleman, W. M., "Some Complexity Results for Matrix Computations," J.A.C.M., Vol. 25, No. 1, January, 1978, pp. 112-115.

2. Ajmore, M. and M. Gerla, "Markov Models for Multiple Bus Multiprocessor Systems," U.C.L.A., Los Angeles, CA., Tech. Rep. CSD 810304, Feb., 1981.

3. Ajmone, M. and G. Carra, "Bus and Memory Interference in Double Bus Multiprocessor Systems," Microprocessing and Microprogramming, North Holland Publishing Company, Vol. 13, 1984, pp. 73-96.

4. Baskett, F., and K. M. Chandy, R. R. Muntz, and F. Palacios, "Open, Closed, and Closed Mixed Networks of Queues with Different Classes of Customers," J.A.C.M., Vol. 22, April 1975.

5. Baskett, F. and A. J. Smith, "Interference in Multiprocessor Computer Systems with Interleaved Memory," C.A.C.M., Vol. 19, No. 6, June, 1976, pp. 327-334.

6. Mudge, T. N., J. P. Hayes, G. D. Buzzard and D. C. Windsor, "Analysis of Multiple-Bus Interconnection Network," Proc. 1984 Int. Conf. on Parallel Processing, Aug. 1984, pp 228-232.

7. Lang, T., M. Valero, and I. Alegre, "Bandwidth of Crossbar and Multiple-Bus Connection for Multiprocessor Systems," I.E.E.E. Trans. Comp., Vol. C-31, No. 12, Dec., 1982, pp. 1227-1233.

8. Pritsker, A. A. B., Introduction to Simulation and SLAM II, 2nd Edition, Halsted Press, New York, 1984, 612 pages.