1996

Design and Analysis of Reduced-Connection Multiple Bus Systems: A Probabilistic Approach.

Najmul Karim
Louisiana State University and Agricultural & Mechanical College

DESIGN AND ANALYSIS OF REDUCED CONNECTION MULTIPLE BUS SYSTEMS: A PROBABILISTIC APPROACH

A Dissertation

Submitted to the Graduate Faculty of the
Louisiana State University and
Agricultural and Mechanical College
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

in

The Department of Electrical and Computer Engineering

by
Md. Najmul Karim
B.E., Bangladesh University of Engineering and Technology, 1983
M.S., Louisiana State University, 1990
December, 1996

UMI Number: 9720361

ACKNOWLEDGMENTS

I would like to express my sincere appreciation to Dr. A. El–Amawy for guiding me throughout my research. He made time for numerous presentations, provided valuable suggestions which helped me to construct a coherent and comprehensive piece of work.

I also thank Dr. Hegde, Dr. Kak, Dr. Ram, Dr. B. Jones and Dr. Aghazadeh for serving on my committee and for being cooperative throughout. I want to express my special gratitude to Dr. Hegde and Dr. Ram for making extra time in the course my research progress. Though not my committee members, Dr. Trahan and Dr. Oxley helped me out by making important comments for some specific problems. Very special thanks to Dr. Rai for providing invaluable information I needed for my research.

I have no way to express gratitude to my parents, brothers and sister who had always been source of encouragement during most difficult times. During this long period my wife Nahid took many responsibilities which I was supposed to share with her and she endured all the difficulties alone. The accomplishment, however small it is, though bears only my name would have never been fulfilled without the help and inspirations of my beloved ones.

TABLE OF CONTENTS

List of Tables

List of Figures

ABSTRACT

In the dissertation we have proposed the first systematic and formal approach to reduce connectivity of general purpose multiple bus systems. The approach is based on a probabilistic technique. The hypothesis on which this dissertation is based stipulates that bus connectivity in multiple bus systems can be much reduced by removing connections which are only needed for highly improbable request patterns. With this approach, performance comparable to that of the original multiple bus system could be achieved while significantly reducing memory bus connectivity. The new architecture thus obtained (Probabilistically reduced connection multiple bus system, or PRMB in short) might have different possible configurations each possibly with a different bus connectivity cost. We have studied the possible relationship among different possible configurations of PRMB systems and proposed an algorithm that determines the one with minimum memory-bus connectivity cost for a given performance level. Our analysis results strongly supported our hypothesis.

The queuing problem for PRMB systems is a complicated one because of its unique modeling requirements. An interesting and innovative modification of aggregation technique has been developed to solve queuing problem taking into account bus contention in PRMB system. We have utilized the proposed approximate technique to determine the system throughput. We have also simulated the queuing networks without applying any approximations. Comparison of analytical results with simulation data indicated that our approximate method could accurately be used to model such queuing networks. The results indicated that our hypothesis is valid when queues are utilized.

We have proposed another variant of the PRMB system which attempts to reduce the connectivity cost from both the processor side and the memory side. Our results indicated that, except under certain specific conditions, this variant of PRMB system did not offer any cost improvement over the original version. It is quite possible that PRMB system is so efficient that further reduction may not be possible without sacrificing some performance.

The technique presented in this dissertation is of very general nature and could possibly be applied to other types of networks as well.

CHAPTER 1

INTRODUCTION

Multiprocessor systems have become very popular for solving many problems in science and engineering which would run unacceptably slow in a single processor environment. A multiprocessor system is composed of a number of independent processors. The processors can execute the same code on different data sets (single instruction multiple data stream machines or SIMD in short) or they can execute different programs on different data sets (multiple instruction multiple data stream or MIMD in short). There are classes of problems which fit the SIMD model extremely well. However, some large problems cannot be organized into repetitive operations on uniformly structured data. Attaining high performance for such cases necessitates an MIMD environment [20]. This dissertation only considers MIMD mode of parallelism.

Multiprocessor systems can also be classified into shared memory systems and distributed memory systems. In a shared memory multiprocessor system, memory modules are global to all the processors and communication between two processor is achieved through the shared memory. In a distributed memory system, memory modules are local to the processors. Each processor has direct access to its own memory module. The interconnection network usually supports point–to–point communication among the processors.

Prominent examples of pint–to–point interconnection networks include meshes and hypercubes [1], [14], [20]. A mesh is a multidimensional array with each node representing a processor. If r is the dimensionality of a mesh and N is the number

of nodes in each dimension then the total number of nodes is N^r. The distance between any two farthest nodes (known as diameter) in that case is $r(N-1)$. So for a large network worst case communication latency becomes a major constraint. The hypercube network which is a special case of the mesh network, having two nodes in each dimension, reduces this problem. The diameter of a hypercube network is r, where r is the dimension of the hypercube. However for a sufficiently large network even this diameter can be high. Another problem with both of these point–to–point networks is their static topology; once the machine is built it cannot be changed anymore [1].

Communications in shared memory MIMD can be achieved by a broad spectrum of interconnection networks: crossbar networks, single or multiple bus networks and multistage interconnection networks. In a crossbar system, all possible one to one simultaneous connections are allowed between the processors and the shared memory modules. While, the crossbar provides maximum potential bandwidth, it is prohibitively expensive for large systems. Single bus interconnections are inexpensive and easy to implement but suffer from limited bandwidth. The multistage interconnection network provides a rich subset of one to one simultaneous connections between processors and memory modules and has moderate incremental cost. The main disadvantage of multistage interconnection networks is that they are not easily scalable and are subject to high latency under certain conditions [5].

Multiple bus networks which offer features like moderate cost, easy incremental expansion and fault tolerance are attractive alternatives for connecting the processors and the memory modules [5]. The standard connection scheme of an MIMD multiple bus system connects all the processors and memory modules to all the buses [10],[12]. This standard scheme is usually referred to as *full bus connection*

system. Figure 1.1 illustrates the architecture of a full bus connection system. For

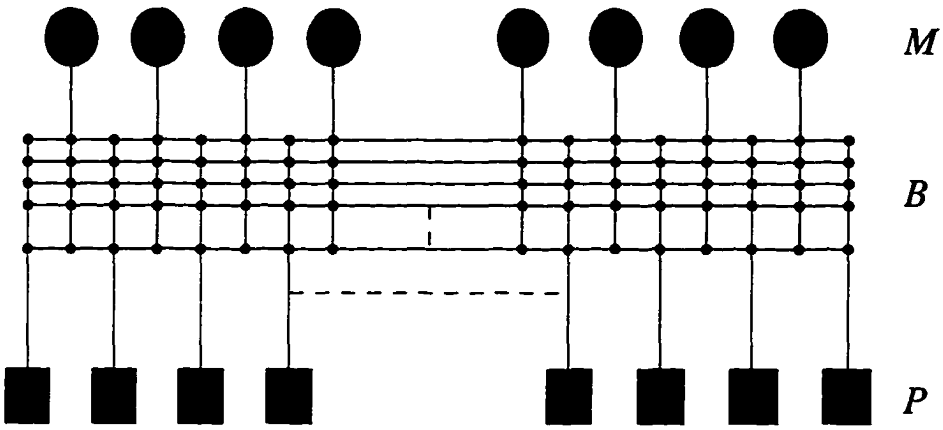


Figure 1.1: The interconnection of a full bus connected system.

a very large system a full bus connection can be too costly. The cost complexity of full connection multiple bus system is $O((P+M)B)$, where P is the number of processors, M is the number of memory modules and B is the number of buses, respectively. A few approaches have been taken by previous researchers to reduce the connectivity cost of MIMD multiple bus systems. The following section is a brief of summary of such approaches. In this dissertation we will be utilizing two connectivity cost measures each in proper context: (i) total connectivity cost (of processor and memory connections to buses); (ii) memory–bus connectivity cost (cost of memory to bus connection).

1.1 Reduced connection multiple bus systems

The major research trend in multiple bus systems has focused on reducing the cost of connections between memory modules and buses or between processors and buses [5, 9, 10, 12, 16, 24]. Thus far three different approaches have emerged. First, multiple

bus systems with improved connection styles have been proposed. Such systems keep the performance exactly the same as that of a full bus connection scheme. Second, *partial connection multiple bus systems* which offer reduced connection cost at the cost of some performance degradation, have been suggested. Third, multiple bus systems suited for special applications have been studied.

1.1.1 MULTIPLE BUS SYSTEMS WITH IMPROVED CONNECTION STYLES

In standard multiple bus systems all the processors and the memory modules are connected to all the buses. If there are P processors, M memory modules and B buses, then in such a system at most B memory modules will utilize the buses in any given cycle. Lang, Valero and Foil [12] proposed some connection styles like Rhombic, Balanced, Staircase, Cyclic, etc., in which not all the buses are connected to all the memory modules. However connection patterns are such that any B memory modules can be connected to B buses and therefore the throughput remains the same as that of a full bus connection system (processors are connected to all the buses). Arbitration mechanism for these connection styles have to follow some specific algorithms, and in general, are more complex than those used in the full bus connection system. For cyclic and balanced connection styles the arbitration procedure is difficult to implement [12], while for Rhombic and Staircase styles arbitration mechanism are somewhat simpler [12]. Figure 1.2 illustrates a Rhombic connection pattern. In this connection style each bus is connected to $M - B + 1$ memory modules. For example bus 0 is connected to memory modules $0, 1, \cdots, M - B$, bus 1 is connected to memory modules $1, 2, \cdots, M - B + 1$ and so on. The memory–bus connection complexity in the Rhombic style is $O((M-B)B)$ as opposed to $O(MB)$ in a full bus connection system.

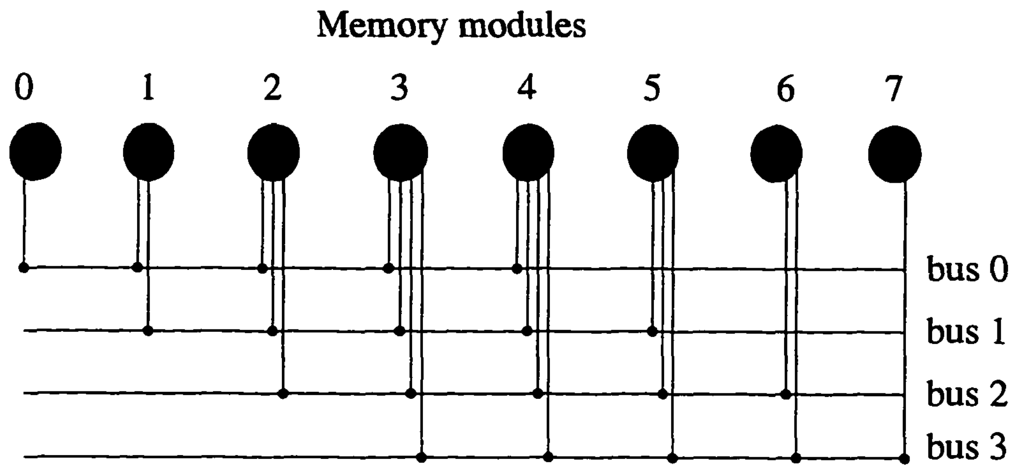


Figure 1.2: The interconnection of a multiple bus system with rhombic connection style.

The main disadvantage of these connection styles is the need for complex arbitration algorithms. The algorithms for bus allocation, even for relatively simple cases (like Rhombic, Staircase), presented in [12], seem to be complex enough to slow down the clock. This might be the reason that most of the subsequent research attempted to improve on the full bus connection system [5], [9],[16]. Even with these improved connection styles, the cost of a large multiprocessor system will still be very high. The overall cost complexity for most of these connection styles is $O(P + M - B)B)$.

1.1.2 Partial connection multiple bus systems

Some researchers proposed *partial connection multiple bus systems* [5, 9, 10] where connectivity is reduced by either connecting the memory modules or the processors to a subset of buses. The one which is traditionally known as partial connection multiple bus system was proposed by Lang, Valero and Alegre [10]. In this system

memory modules are divided into equal sized groups and each of these groups is connected to some equal but different subset of buses. The processors are connected to all the buses. Figure 1.3 illustrates the interconnection of a partial connection multiple bus system. It has been shown in [10] that degradation in performance for

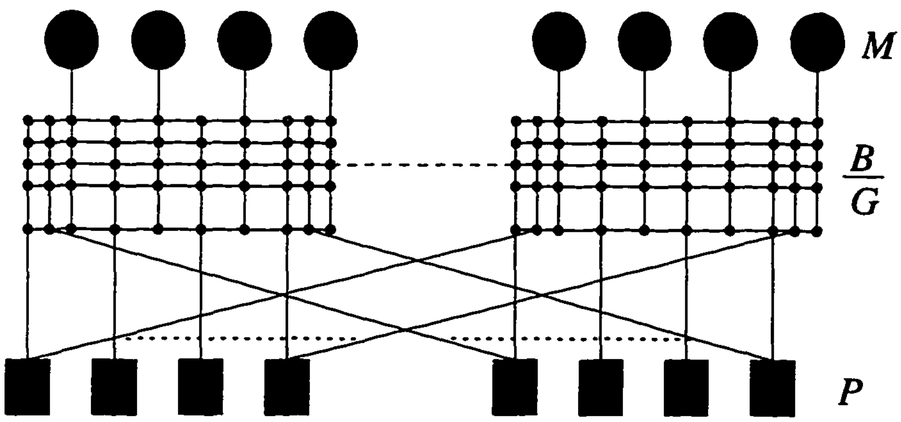


Figure 1.3: The interconnection of a partial connection multiple bus system.

a partial connection system could be reduced to at most 6% compared to a standard scheme while achieving 50 % reduction in memory–bus connections. This level of degradation is however for a system with 16 processors, 16 memory modules, 8 buses and 2 groups. In addition, for performance evaluation only a uniform request model has been considered. With non uniform request models (like the hotspot request model) and for systems with larger number of groups performance degradation is expected to be higher.

Jiang and Smith [9] proposed a *processor oriented partial connection multiple bus system* (PPMB). In this system processors are partitioned into equal sized groups with each group connected to equal number of local buses. Memory modules are connected to all the buses. The arbitration for a memory module among requesting

processors attempts to select a processor from a group whose processors make the least number of requests. With improved load balancing in the arbitration mechanism the PPMB system is claimed to improve the performance over regular partial connection bus systems. Examples presented in [9] show that up to to 20 % improvement in memory bandwidth over a regular partial multiple bus system could be achieved. Those results are based on two networks both with 32 processors and 32 memory modules, one network having 8 groups and the other having 16 groups. For this size of networks, 8 or 16 groups seem to be too many and in that case there might be significant degradation in performance compared to a full bus connection system. Besides, hardware of the arbitration mechanism described in [9] is very complicated compared to that needed in a regular partially connected multiple bus system. Bus connection complexity for this network is the same as that of a regular partial connection system.

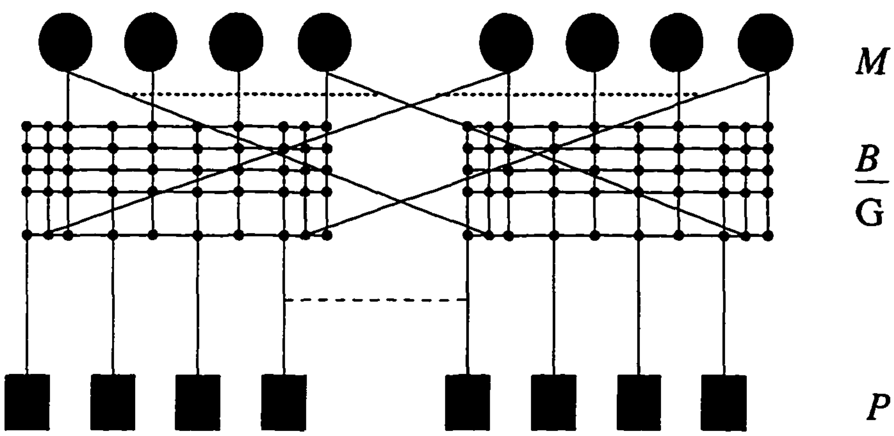


Figure 1.4: The interconnection of a PPMB system.

A third type of partial connection multiple bus system has been proposed by Chen [5]. It is called partial connection multiple bus system with K classes. It has

a cost which is intermediate between that of a regular partial connection multiple bus system and that of a full bus connection system. In this system processors are connected to all the buses. The memory modules are divided into K classes. Memory modules in class K are connected to B buses, (bus 1 through bus B), memory modules in class $K-1$ are connected to $B-1$ buses (bus 1 through $B-1$). In general memory modules in class j, where $j \in \{1, \cdots, K\}$ are connected to $j+B-K$ buses, (bus 1 through bus $j+B-K$). The architecture for this type of multiple bus system with 6 memory modules, 4 buses, and 3 classes is illustrated in Figure 1.5. Memory bandwidth for this kind of multiple bus system has been evaluated by using

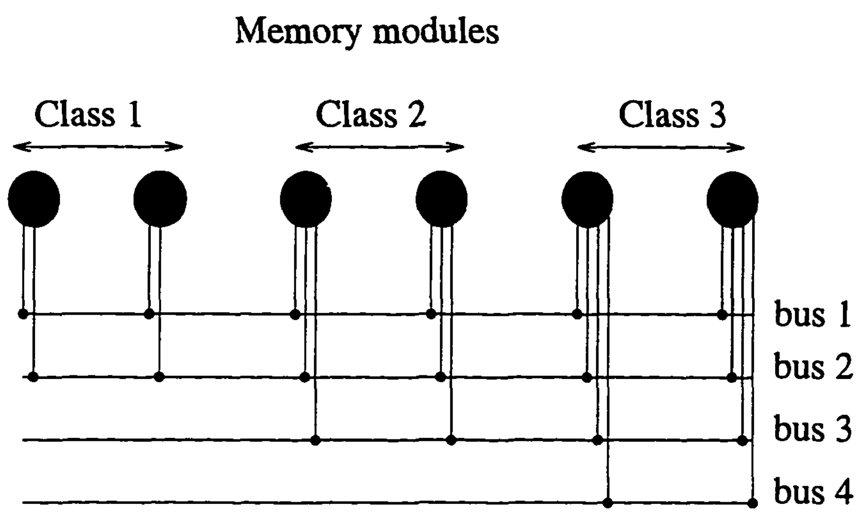


Figure 1.5: The interconnection of a partial connected multiple bus system with K class ($K = 3$).

a uniform request model and a hierarchical request model. Results presented in [5] show that memory bandwidth is comparable to that of a regular partial connection multiple bus system. Though the connectivity cost is higher than that of regular partial connection systems, this system has better fault tolerance capabilities than

other partial connection multiple bus systems [5]. A major drawback of the system is the lack of uniformity.

1.1.3 Special application multiple bus systems

Some of the proposed multiple bus systems are suited for special applications [24, 16]. Considering special situations where some kind of locality exists in the request model attempts have been made to utilize locality to reduce connectivity cost.

Wilkinson proposed a multiple bus system with overlapping connectivity [24]. Processors, memory modules and buses are divided into equal sized groups. Within a group processors are fully connected and memory modules are connected partially. However around half of the memory modules in a group are connected to buses belonging to the next group to the right and the other half are connected to buses in the group to the left. So if a processor makes a request to a memory module in its group or some of the memory modules in adjacent groups a connection can be established. The memory modules belonging to adjacent groups which can be directly reached are termed as the *sphere of influence*. Requests that are intended for memory modules neither within the group, nor within the sphere of influence have to go through one or more processors. So each processor needs bi-directional ports so that it can receive requests from processors in adjacent groups. Similar ports are needed for memory modules as well for receiving requests from two groups. Evidently this architecture is suitable for a request model with locality criteria such that with the increasing distance the probability of making requests decreases [24].

Memory bandwidth has been evaluated considering a uniform request model and a locality based request model. For the locality based model it is assumed that processors make requests only to memory modules within their group or to mem-

ory modules within the sphere of influence. Numerical results reported [24] show improvement in performance over "conventional multiple bus system" for both the uniform request model and the locality based request model. While in [24] it is not clearly mentioned what is meant by "conventional multiple bus system"; generally it is taken to mean a full bus connection system. If that is the case then the results do not seem to be correct because the architecture provides connectivity that is less than even that of a partially connected multiple bus system. On the other hand if the author implies that each group is implemented independently by a conventional multiple bus system with no interconnection between groups, then the results presented could make sense because in that case in a conventional implementation, processors in one group will not be able to reach the memory modules in another group. This will lead to some degradation for a uniform request model and more degradation for a locality based model (as presented in [24]). Under a uniform request model memory modules in different groups but within the sphere of influence will get fewer requests than what they would get under a locality based request model. Overlapping connectivity therefore will satisfy more requests directed to the sphere influence in a locality based model than in the case of a uniform request model. The difference in performance between a conventional implementation and overlapping connection scheme is due to the difference in performance in the sphere of influence. While in a conventional scheme no requests in the sphere of influence are satisfied, for the overlapping connection scheme some requests for a uniform request model and more requests for a locality based model will be satisfied. Therefore performance for conventional schemes will be further degraded in the case of a local request model than in the case of a uniform request model and the numerical results presented in [24] go along this line of argument.

Though in [24] connectivity cost has not been computed for any of the networks from the architectural features, it is evident that connectivity cost can be lower than that of a partial connection multiple bus system.

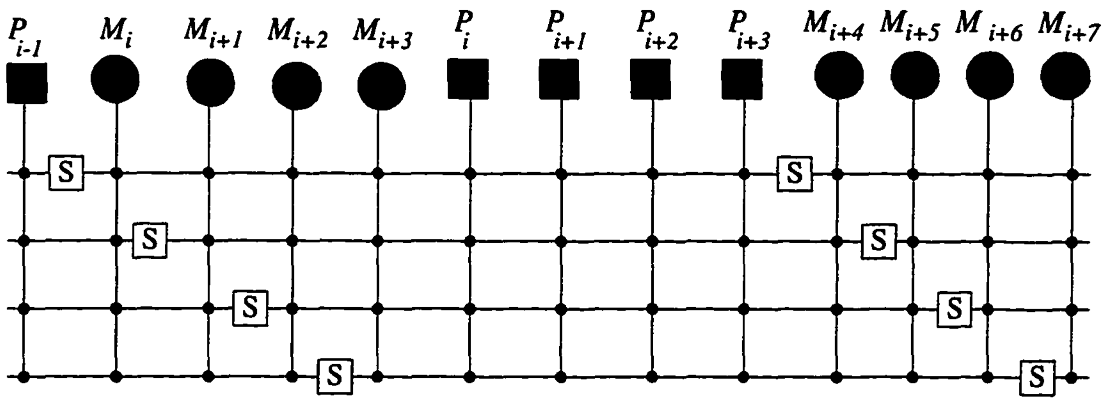


Figure 1.6: The interconnection of a multiple bus system with overlapped connectivity.

Mahmud [16] proposed a multilevel bus network suited for a hierarchical request model. In this architecture, processors and memory modules are divided into a number of multilevel clusters. At the i^{th} level, n_i processors and m_i memory modules form a cluster. In each of the clusters in the i^{th} level there will be some buses (say b_i in number) which will be connected to processors and memory modules of that cluster. In the next level, i.e., the $(i+1)^{\text{st}}$ level, some of the i^{th} level clusters form a bigger cluster and each of these bigger clusters will be connected to b_{i+1} buses. Obviously as the level number goes up, connectivity for a bus increases. If there are altogether L levels, then a bus which belongs to level L will be connected to all the processors and memory modules. The architecture is illustrated for the case of 3 levels in Figure 1.7. This Multilevel bus system is suitable for the hierarchical

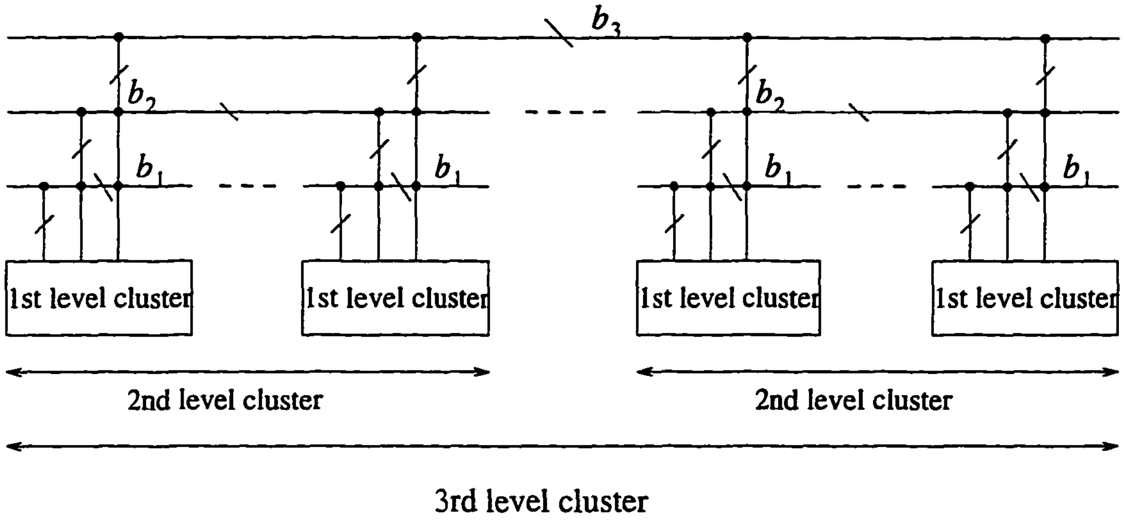


Figure 1.7: The interconnection of a multi level multiple bus system with 3 levels.

request model proposed in [16]. In this request model a processor makes a request to a memory module with probability that will depend on its position in the hierarchy compared to that of the processor. In other words, the probability that a processor makes a request to a memory module will depend on the lowest level at which the processor and the module become members of the same cluster. If this probability gradually is less for the memory modules forming clusters with the processors in the higher levels than in the lower levels, then less buses will be required at higher levels. This hierarchical request model and a uniform request model have been used to determine memory bandwidth and throughput for a multilevel bus system.

Numerical results presented [16] show that for a hierarchical request model the performance is almost the same as that of partial connection and full bus connection systems. Performance however is degraded for a uniform request model [16]. Mahmud also proposed another variant of the multilevel bus system which reduces the connectivity from the original version with additional degradation in performance. However results presented in [16] for both versions of multilevel bus systems are not

indicative enough, because only low values for processor request rates have been considered(only .05 and .1). The connectivity cost for multilevel bus systems for various networks are in most cases lower than those of partial connection multiple bus systems [16].

1.2 Motivation for the present research

The cost of connection for multiple bus systems is important for two reasons. First, as network size grows, connectivity cost grows unacceptably high. Second, the problem of bus loading becomes severe as the number of connections exceeds certain threshold [16].

The connection between a processor and a bus involves connecting all the external signals of the processor to the bus. The same is true for a connection between a memory module and a bus. The total number of bus connections grows with increase in the number of processors, memory modules and buses for all variants of multiple bus systems [5, 9, 10, 12, 16, 24]. For a large multiprocessor system the connectivity cost can constitute significant part of overall cost. This situation is depicted vividly by Mahmud [16]. We directly quote from [16]:

> "The cost of connection is significant part of the entire cost of a multiprocessor system. Because the memory bus of a 32-bit RISC or CISC processor can have as many as 80–90 signal lines, which includes address, control and bidirectional data lines. Thus, every connection (switch) will require approximately 100-120 tristate buffers. Note that two tristate buffers are necessary for every bidirectional data line. A switch can be implemented either on a single chip or on multiple chips. A single chip switch will require as many as 200–250 pins; 100–120 pins for the mem-

ory side and another additional 100–120 pins for the processor side and some additional pins for the arbitration signals. Thus, every switch may be as expensive as a processor. Hence for a system with a large number of processors it is desirable to reduce the number of connections."

Present day processors actually have even more address, data and control lines and therefore connection cost for the switches will be even higher than that depicted in [16]. For example the number of external signals in the UltraSparc processor is 144 [23]. So to build a cost effective multiprocessor system bus connectivity cost has to be reduced to the extent that is possible.

Another critical aspect of bus connectivity is the problem of bus loading. If a large number of processors and memory modules are connected to a bus then due to capacitive loading, signal quality will degrade in terms of rise and fall times as well as waveform [16], [20]. This may require reducing the speed of bus transactions and in the extreme may render the system inoperable. With future optical technology while the problem of bus loading is expected to be reduced to some extent, for a very large system bus loading will still be a problem and the cost factor may actually be more profound.

The approaches taken thus far reduced bus connectivity in some cases with degradation in performance and in other cases without degradation in performance compared to a conventional multiple bus system. Performance degradation in some cases is an acceptable trade off as in the case of partial connection multiple bus systems.

Considering the above, it is worthwhile to explore new approaches for further reduction in bus connectivity in both cases. The research reported in this dissertation is based on the important observation that part of the connectivity in any

multiple system is required only for some highly improbable request patterns. The hypothesis of our research is that further reduction in connectivity is achievable by eliminating probabilistically redundant connectivity with almost no degradation in performance for both full and partial connection systems.

The objective of this research is to investigate the possibility of reducing cost complexity considering the probability of different request patterns. We will consider a base model for an MIMD multiple bus systems and apply a probabilistic technique to reduce connectivity. We will attempt to reduce the connectivity in such a way that only some highly improbable request patterns will be allocated fewer buses than in a base model system. This technique to reduce connectivity will be referred to as *probabilistic connection reduction* (PCR) and the resulting architecture will be called *Probabilistically reduced connection multiple bus system* (PRMB). In its original form the PCR technique attempts to reduce bus connectivity from the memory side only. As will be discussed later in detail, the base model for multiple bus systems will have groups of memory modules and each group will be connected to a set of buses. In the PCR technique each group of memory modules will be further divided into subgroups and a subset of buses from the original group will be connected only within each of these subgroups. Besides, some buses known as *common buses* will be connected to all the subgroups of a group. The appropriate selection of the number of common buses will ensure that, with high probability, a request pattern will be satisfied as in a base model system. Our results will support our hypothesis and will show that the proposed technique is indeed effective in reducing bus connectivity, in some cases significantly, at almost no degradation in performance. We further study the PRMB system when memory queues are incorporated. We simplify the queuing problem by using the method of aggregation. We compare the performance of PRMB system

with memory queues with those of several traditional systems under the memory queue assumptions. Our results with queues incorporated, show that performance of PRMB system is not any way adversely affected

1.3 ORGANIZATION OF THE DISSERTATION

The organization of the rest of the dissertation is as follows. Chapter 2 will elaborate on architectural features of PRMB systems such as its interconnection pattern and arbitration mechanism. We evaluate the minimum number of common buses needed such that with high probability a request pattern will get the same number of buses as in the base model system. We introduce and formally define a general request model. This model encompasses the uniform request model, hotspot request model, locality based request model and locality based request model with local hotspots. The number of common bus thus determined will be shown to provide the desired performance for the given request models while reducing connectivity significantly. We evaluate the memory bandwidth and cost for the PRMB system and compare it with those of the base model system.

The number of common buses needed in a PRMB system to achieve certain performance level will vary depending on the number of subgroups in each group. As a result, overall connectivity will vary as the number of subgroups varies. In Chapter 3 we study the relationship between the number of common buses and the number of subgroups and attempt to reach the PRMB architecture that has minimum connectivity cost.

Queuing analysis for multiple bus systems is considered difficult because of the passive resources associated with the memory servers [6]-[8], [18]. In a PRMB system buses are associated with both the subgroup and group levels, which further

complicates the queuing analysis. In Chapter 4 we present an approximate method based on the aggregation technique to make the queuing problem tractable. We determine the throughput of the system for two different request models namely, the uniform request model and the hotspot request model. We also present simulation results for PRMB systems to determine the accuracy of the approximate method. Our results will demonstrate the accuracy of the approximating technique we propose. We also compare our results with those of corresponding base model systems. For all the networks considered, under both request models, the performance of the PRMB system is almost identical to that of the corresponding base model system.

In Chapter 5 we address the problem of reducing connectivity from both the processor side and the memory side of the base model system using our probabilistic technique. Processors and memory modules in subgroups are connected by local buses while some common buses are connected to all the processors and memory modules. The number of common buses is determined as in Chapter 2 such that a request pattern in a group gets the same number of buses as in a base model system with high probability. We consider the same general request model and study the possible cost improvement compared to the original PRMB system. Our results will show that, except for request models with unusually high locality rate, this variant does not offer any cost reduction in comparison with the original version of the PRMB system.

Finally Chapter 6 is a conclusion which summarizes the research reported in this dissertation.

CHAPTER 2

PROBABILISTICALLY REDUCED CONNECTION MULTIPLE BUS SYSTEMS

In this chapter we study the PRMB system in detail. To show that probabilistic redundancy can possibly be eliminated from most architectures proposed thus far, we will introduce a more general multiple bus system which we shall call the *base model system.*

2.1 BASE MODEL FOR A MULTIPLE BUS SYSTEM

The base model of a multiple bus system has P processors, M memory modules and B buses. The memory modules are partitioned into G groups, with each group connected to the $\frac{B}{G}$ buses. The processors are also divided into G groups but each processor is connected to all the buses. Let the number of processors, memory modules and buses in each group be denoted by M_G, P_G and B_G, respectively. Note that, for $G = 1$, the multiple bus system is a full bus connection system, otherwise it is a regular partially connected bus system.

For the base model system a request pattern for a given bus cycle is defined a G–tuple $\{i_1, i_2, \cdots, i_G\}$, where i_k is total number of memory modules in the k^{th} group that each receive at least one request in that cycle. A request pattern can thus be served by at most $\frac{B}{G}$ buses from a group and B buses altogether. The memory bus connection complexity for the base model is $O(MB_G)$.

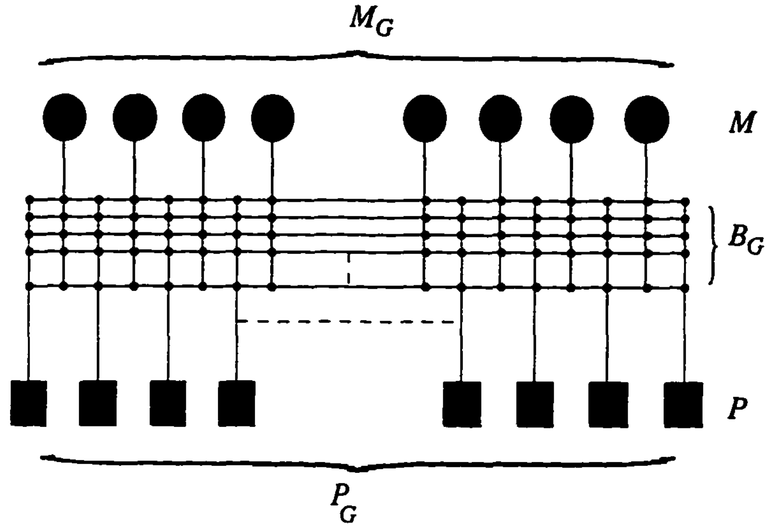
2.2 PRMB ARCHITECTURE

2.2.1 INTERCONNECTION

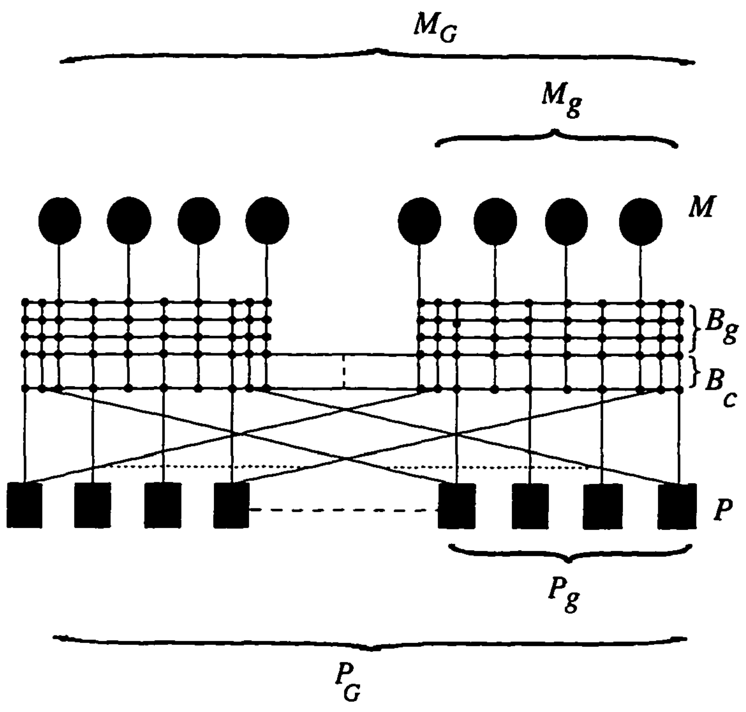
In the PRMB architecture, memory modules of each group are partitioned into g subgroups such that each subgroup will have equal number of memory modules and equal number of processors. Each subgroup within a group will be connected to some different but equal number of local buses B_g. Besides, all the memory modules within a group are connected to some common buses B_c. Notice that the common buses are common with respect to the group only and not to the entire system. This basically amounts to re–distributing B_G buses in a group (of a base model system) to different subgroups within the group with some buses being global to the group. Processors, as before, remain connected to all the buses. Let the number of memory modules and processors in each subgroup be denoted by M_g and P_g, respectively. To simplify the expressions we assume that g evenly divides both M_G and P_G. Obviously, $M_g = \frac{M_G}{g}$, $P_g = \frac{P_G}{g}$ and $B_g = \frac{B_G - B_c}{g}$. Figures 2.1 and 2.2 show two examples of PRMB architectures along with the corresponding base model systems. The base model corresponding to PRMB system in the Figure 2.1 is a full bus connection system and that corresponding to system in Figure 2.2 is a partial connection system with two groups.

2.2.2 ARBITRATION

The principle of arbitration in a PRMB system is to ensure the use of a local bus first. The arbitration mechanism is a simple modification of the two level arbitration scheme proposed by Lang and Valero [11]. A $P - user - 1 - server$ type arbiter is associated with each memory module (since there can potentially be P requests for a particular memory module). This arbiter corresponds to the first level of



(a)



(b)

Figure 2.1: (a) The interconnection of the base model system. (b) The interconnection of a PRMB system.

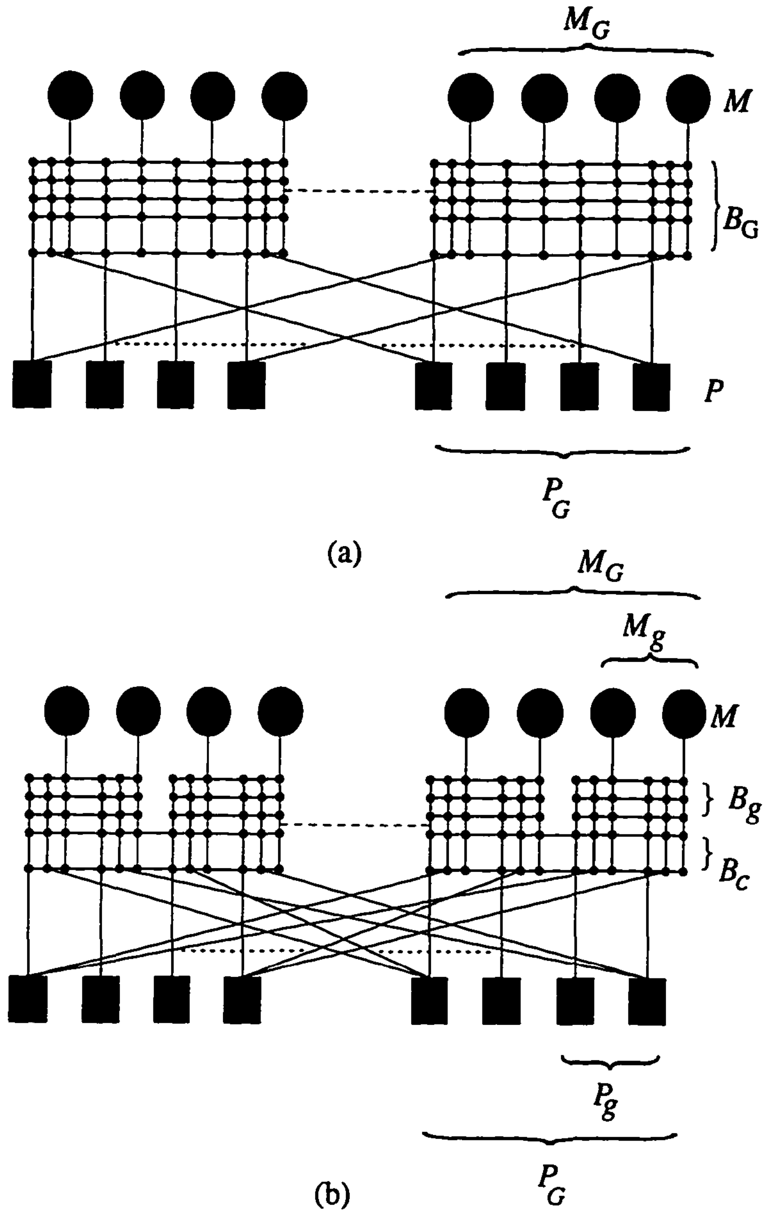


Figure 2.2: (a) The interconnection of the base model system. (b) The interconnection of a PRMB system.

arbitration. The first level arbiter selects one request for each of the requested memory modules. Winner of the arbitration proceeds to the second level. The arbitration in the second level involves arbitrating among M_g customers competing for B_g local buses and also involves arbitrating among the requests in each subgroup which cannot obtain local buses; (i.e., arbitration among $gM_g - gB_g = M_G - B_G + B_c$ customers) for B_c buses. Therefore, in second level arbitration, for each group we need g, $M_g - user - 1 - server$ arbiters and one $(M_G - B_G + B_c) - user - B_c\ server$ arbiter. In this level, first an attempt is made to find a free bus from the local buses connected to the particular memory subgroup to which the request is directed. If a local bus is unavailable, only then an attempt is made for a common bus. This is done by the B_c server arbiter mentioned above. At each level, the arbitration is done on random basis.

2.3 Bandwidth Analysis

For multiple bus systems memory bandwidth is widely used as a performance measure [3, 5, 7, 9, 10, 12, 16, 24]. Memory bandwidth is defined as the expected number of busy memory modules (or buses in this context) in a cycle. To evaluate the memory bandwidth for a PRMB system first we have to develop a request model. Thus far most researchers have considered either a uniform request distribution model or some locality based request model. For bandwidth analysis and (later for evaluation of the number of common buses) we will consider a generalized request model.

2.3.1 Request Model

To define the generalized request model we first introduce the concept of *class*.

Definition 2.1 Processors and memory modules are divided into some arbitrary number of sets with each set having equal number of adjacent processors and memory

modules. Each of these sets is said to form a *class.* For a memory module in a given class all the processors within the class are *local* and all other processors are *non local.* Similarly, for a processor, all memory modules within the same class are *local* and the rest of the memory modules are *non local.* Processors in a class may favor one local memory module over the other local memory modules. In that case such a memory module is called a *hot module.* A module that is not a hot module is called a *regular module.*

Note that while for a given PRMB architecture the number of groups and the number of subgroups per group are fixed, the number of classes might vary depending on the application. In other words groups and subgroups are architectural features whereas classes are features of the request model. Classes are intended to reflect the locality behavior of different subtasks assigned to subsets of processors and memory modules. Now, our generalized request model is described as follows.

1. Processors operate in synchronous MIMD mode. Each processor generates requests independently of others. Request generation can take place only at the beginning of a cycle.

2. A memory request generated by a processor has certain probability of being directed to its local class, otherwise it is equally distributed to all the memory modules. If the request is directed to the local class then there is a certain probability that it is directed to the local *hot module*, otherwise the probability is equally distributed to all the modules in the local class.

3. Requests generated by a processor at successive cycles are independent.

4. Propagation delay and arbitration time are included in memory access time.

5. Requests which are blocked (those that do not pass the arbitration) are ignored.

Assumptions 3 and 5 will later be relaxed while considering our queuing model. The general request model presented, encompasses four specific request models: *uniform request model, hot spot request model, locality based computation model* and *locality based computation model with local hotspots.* If there is only one class in the request model and a processor request has no bias to any memory module then the request model is a "uniform request model". If the request model has only one class and a processor generates requests with some bias to one particular memory module (hot memory module) then the request model is a "hotspot request model". If the request model has more than one class and all the memory modules within a class are treated equally then the request model is a "locality based request model". Finally, if there is more than one class and within each class there is a favorite memory module then the request model is a "locality based request model with local hotspots".

2.3.2 Notation

We introduce the following notation for the analysis of memory bandwidth in a PRMB system

C: number of classes in the request model.

r: probability that a processor makes a request in any given cycle.

l: probability that a processor makes a request to its local class.

h: probability that a request to a memory module in the local class is a hotspot request.

P_l: number of processors in a class.

P_n: number of processors in non local classes of a memory module.

M_l: number of memory modules in the local class of a processor.

M_n: number of memory modules in non local classes of a processor.

H: number of hotspot memory modules in a group.

H_k: number of hotspot memory modules in the k^{th} subgroup of a group.

q_r: probability that a regular (non hotspot) memory module gets at least one memory request.

q_h: probability that a hot memory module gets at least one memory request.

$(i_1, i_2, \cdots, i_g)$: request pattern in a group where i_k is the number of memory modules in the k^{th} subgroup receiving requests in the memory cycle under consideration. Notice that i_k does not reflect the number of requests to individual modules. If a module receives at least one request, that module is included as one in the i_k count.

$Pr(i_1, i_2, \cdots, i_g)$: probability of occurrence of the request pattern $(i_1, i_2, \cdots, i_g)$.

$bus(i_1, i_2, \cdots, i_g)$: number of buses that will be available to request pattern $(i_1, i_2, \cdots, i_g)$.

bw: memory bandwidth of a group for the base model system.

$\tilde{bw}$: memory bandwidth of a group for a PRMB system.

BW: overall bandwidth in a base model system.

$\widetilde{BW}$: overall bandwidth of a PRMB system.

2.3.3 BANDWIDTH

Remark: Unless otherwise stated, throughout the analysis a *memory cycle* means any arbitrary memory cycle. A *request pattern* implies a random request pattern with some probability distribution associated with it. The analysis that follow is based on one memory cycle.

The memory bandwidth in a PRMB system can be determined by determining bandwidths of individual groups. Memory bandwidth in a group is the expected number of busy memory modules in a group per cycle. To evaluate the memory bandwidth in a group we have to evaluate the probability of a request pattern and determine the bus allocation for that request pattern in the group.

The probability of a request pattern can be evaluated as follows. We start by evaluating the probability that a memory module (regular or hot) gets a request in a memory cycle. The probability that a processor P_i will make a request to a local and regular memory module M_j is $\frac{rl(1-h)}{M_l} + \frac{r(1-l)}{M}$. The probability that it will make a request to nonlocal memory M_k is $\frac{r(1-l)}{M}$. Further, the probability that it will make a request to a local hotspot memory module is $rlh + \frac{rl(1-h)}{M_l} + \frac{r(1-l)}{M}$. Therefore the probability that a regular memory module will not get a request from any of the processors in its local class is $(1 - \frac{rl(1-h)}{M_l} - \frac{r(1-l)}{M})^{P_l}$. The probability that it will not get a request from any of the non-local processors is $(1 - \frac{r(1-l)}{M_n})^{P_n}$. Therefore,

$$q_r = 1 - (1 - \frac{rl(1-h)}{M_l} - \frac{r(1-l)}{M})^{P_l}(1 - \frac{r(1-l)}{M_n})^{P_n}$$

Similarly, the probability that a hotspot module will not get a local request is $(1 - rlh - \frac{rl(1-h)}{M_l} - \frac{r(1-l)}{M})^{P_l}$. The probability that it will not get a request from any of the non-local processors is $(1 - \frac{r(1-l)}{M_n})^{P_n}$. Therefore,

$$q_h = 1 - (1 - rlh - \frac{rl(1-h)}{M_l} - \frac{r(1-l)}{M})^{P_l}(1 - \frac{r(1-l)}{M_n})^{P_n}$$

Now that we have evaluated the probability that a memory module gets a request, we can determine the probability of a request pattern $(i_1, i_2, \cdots, i_g)$. To evaluate

this probability we have to consider the number of hotspot requests in a subgroup. Assume that out of i_k modules receiving requests in the k^{th} subgroup, j_k are hot spot modules. Obviously, $j_k = \min(H_k, i_k)$. Therefore, the probability of a request pattern $(i_1, i_2, \cdots, i_g)$ is given by,

$$
\begin{aligned}
Pr(i_1, i_2, \cdots, i_g) &= \sum_{j_1=0}^{\min(H_1,i_1)} \sum_{j_2=0}^{\min(H_2,i_2)} \cdots \sum_{j_g=0}^{\min(H_g,i_g)} \prod_{k=1}^{g} \binom{M_g - H_k}{i_k - j_k} \binom{H_k}{j_k} \\
&\quad q_r^{\sum(i_k - j_k)} \times (1 - q_r)^{M-H-\sum(i_k-j_k)} \times q_h^{\sum j_k} \times (1 - q_h)^{H-\sum j_k}
\end{aligned}
$$

Request pattern $(i_1, i_2, \cdots, i_g)$ will take $\min(i_k, B_g)$ local buses from the k^{th} subgroup. The number of requests in the k^{th} subgroup that cannot be satisfied by the local buses is $i_k - \min(i_k, B_g)$. Therefore, the total number of buses that will be available to the request pattern $(i_1, i_2, \cdots, i_g)$ is

$$
bus(i_1, i_2, \cdots, i_g) = \sum_{k=1}^{g} \min(i_k, B_g) + \min(B_c, \sum_{k=1}^{g}(i_k - \min(i_k, B_g)))
$$

We can now determine the memory bandwidth due to a group as follows

$$
\tilde{bw} = \sum_{i_1=0}^{M_g} \sum_{i_2=0}^{M_g} \cdots \sum_{i_g=0}^{M_g} Pr(i_1, i_2, \cdots, i_g) \times bus(i_1, i_2, \cdots, i_g)
$$

Overall memory bandwidth can be determined by adding the memory bandwidths due to individual groups. Memory bandwidth due to each group will be determined by the above formula. However, depending on the number of classes in the request model the number of hotspots in different groups might be different, which would affect the probability of request pattern $(i_1, i_2, \cdots, i_g)$. This may hap-

pen if C is not evenly divisible by G or vice versa. If the number of hotspots in different groups is not the same, then the k^{th} subgroup, $k \in \{1, 2, \cdots, g\}$ in group i might have different number of hotspot memory modules from that of the k^{th} subgroup in group j, $j \neq i$. This will affect the value of $P(i_1, i_2, \cdots, i_g)$ and thus the memory bandwidth in a group. Let $\breve{bw}_i$ denote the memory bandwidth due to group i. We evaluate $\breve{bw}_i$ for $i \in \{1, \cdots, G\}$ and then calculate the overall memory bandwidth by summing the bandwidths in individual groups. Therefore,

$$\widetilde{BW} = \sum_{i=1}^{G} \breve{bw}_i$$

Note that the memory bandwidth for the base model system can be derived by considering the special case of PRMB system where there is one subgroup in each group and where all the buses in a group are common buses.

2.4 Evaluation of the Number of Common Buses for Certain Performance

As mentioned earlier, our hypothesis for this work is that some connections in traditional systems are probabilistically redundant and can thus be removed with negligible effect on performance. Such connections, according to our hypothesis are used only when some highly improbable request patterns occur. Hence we attempt to find the number of common buses (in a group) in a PRMB system such that bus assignment in a group will be, with high probability, the same as that in a corresponding group of the base model system. Since the connectivity cost for a common bus is much higher compared than that of a local bus, the number of common buses should be kept as small as possible. So, we have to evaluate the minimum num-

ber of common buses that ensures with high probability the same level of request satisfaction as in a base model system.

Let $(I_1, I_2, \cdots, I_g)$ be a random vector denoting the number of requests to distinct modules in different subgroups of a given group. The distribution of this random vector will depend on the request model being chosen. In a memory cycle, the minimum number of common buses needed for allocating the same number of buses as in a base model system is a random variable which is a function of the random vector $(I_1, I_2, \cdots, I_g)$. Let this random variable be denoted by $f(I_1, I_2, \cdots, I_g)$. Let the cumulative distribution function (cdf) for the random variable $f(I_1, I_2, \cdots, I_g)$ be denoted by $P_{f(I_1,I_2,\cdots,I_g)}(.)$. Therefore, $P_{f(I_1,I_2,\cdots,I_g)}(n)$ denotes the probability that the minimum number of common buses required in a memory cycle is less than or equal to n. The number of common buses should be chosen such that this probability is high. Therefore evaluating the number of common buses is essentially equivalent to evaluating the cumulative distribution function $P_{f(I_1,I_2,\cdots,I_g)}(.)$. Once we derive the cdf function, the number of common buses will be a value n for which the value of this cdf function is a chosen high value.

The value of $P_{f(I_1,I_2,\cdots,I_g)}(n)$ can be determined by adding the probabilities of the request patterns for which the number of common buses needed will be less than or equal to n. We have a lemma that will identify those request patterns. Following is a definition that will be used in that lemma.

Definition 2.2 *excess requests* in a request pattern is the number of requests to distinct memory modules of a group that cannot be satisfied by the local buses. For a request pattern $(i_1, i_2, \cdots, i_g)$ excess requests is therefore $\sum_{k=1}^{g} \max(i_k - Bg, 0)$

Lemma 1 *Consider a given group. If for a request pattern the number of excess requests is more than B_c and at least one subgroup in that group has less than B_g*

requests, then the request pattern will not get the same number of buses as it would get in the corresponding group of a base model system, otherwise it will get the same number of buses as in the corresponding group of a base model system

Proof: If a request pattern has more than B_c excess requests, it will not get enough buses to satisfy all the requests. Also if in this case one or more subgroups has fewer requests (to distinct memory modules) than B_g, then there will be some unused buses while other requests (for modules in other subgroups of the same group) are unsatisfied. In a base model system since all the buses in a group are available to all the requests in that group, for the same request pattern there will not be any unused buses. Hence in this case the request pattern will not be satisfied in the PRMB system as it would be in the corresponding base model system.

Now consider the situation where the number of excess requests that cannot be satisfied by local buses is less than B_c. In that case the excess requests will be satisfied by the common buses and there will be no unsatisfied requests in the request pattern. Obviously the request pattern in this case is satisfied in the same way as in a base model system.

Finally consider the case where the number of excess requests is more than B_c and no subgroup has unused buses. In this case though some requests in the request pattern will not be satisfied in the PLMB, there will be no unused buses in the group. So the request pattern gets all the buses in the group, as it would be the case in the corresponding group of a base model system. ■

Consider an arbitrary group. If a request pattern $(i_1, i_2, \cdots, i_g)$ is satisfied with n common buses, then the request patterns obtained by permuting $(i_1, i_2, \cdots, i_g)$ will also be satisfied by n common buses. Similarly, if a request pattern $(i_1, i_2, \cdots, i_g)$ is not satisfied with n common buses then the request patterns obtained by permuting

$(i_1, i_2, \cdots, i_g)$ will not be satisfied either. Let $\hat{Pr}(i_1, i_2, \cdots, i_g)$ be the probability that one subgroup (no matter which subgroup) gets i_1 requests, one subgroup receives i_2 requests and so on. In other words this is the portability of all request pattern obtained by permuting all the tuples in the request pattern $(i_1, i_2, \cdots, i_g)$. Let the tuples in the request pattern $(i_1, i_2, \cdots, i_g)$ be denoted by a set R and suppose that this set can be partitioned into k subsets, where $1 \leq k \leq g$, such that each subset has identical members. By identical members we imply that every member of a subset is the same. Also suppose that the number of members in the j^{th} subset is η_j. If the request model is uniform or is a locality based model with classes equally distributed among all the subgroups, then

$$\hat{Pr}(i_1, i_2, \cdots, i_g) = Pr(i_1, i_2, \cdots, i_g) \times \frac{g!}{\eta_1! \eta_2! \cdots \eta_k!}$$

For other request models,

$$\hat{Pr}(i_1, i_2, \cdots, i_g) = \frac{1}{\eta_1! \cdots \eta_k!} \sum_{n_1 \in R} \sum_{n_2 \in R - n_1} \cdots \sum_{n_g \in R - n_1 - \cdots - n_{g-1}} Pr(n_1, \cdots, n_g)$$

So while considering request patterns for evaluating $P_{f(I_1, I_2, \cdots, I_g)}(n)$ we need to check only one permutation of the pattern $(i_1, i_2, \cdots, i_g)$. In order not to be satisfied as in a base model system a request pattern has to have more than B_c excess requests. For this to take place certain minimum number of subgroups has to contribute a given number of excess requests, because one subgroup can have at most $M_g - B_g$ excess requests. We now introduce the following additional notation.

Notation:

$g_{min}(n)$: minimum number of subgroups which have excess requests such that sum of the excess requests is $\geq n+1$. Let these subgroups belong to a set called S.

$\sigma(n)$: minimum number of requests a subgroup has to have if that subgroup is to belong to set S defined above.

$\lambda(i_1, i_2, \cdots, i_g : n)$: A function whose value is 1 if the sum of excess requests in group for request pattern $(i_1, i_2, \cdots, i_g) > n$, otherwise its value is zero.

Thus,

$$g_{min}(n) = \left\lceil \frac{n+1}{M_g - B_g} \right\rceil$$

$$\sigma(n) = B_g + n + 1 - \left\lfloor \frac{n+1}{M_g - B_g} \right\rfloor (M_g - B_g)$$

$$\begin{aligned} \lambda(i_1, i_2, \cdots, i_g : n) &= 1 \text{ if } \sum_{k=1}^{g} \max(i_k - B_g, 0) > n \\ &= 0 \text{ otherwise} \end{aligned}$$

We can now determine the request patterns which are not satisfied as in the base model system. According to Lemma 1 a request pattern is not satisfied as in a base model system if at least one subgroup has less than B_g requests to distinct memory modules and the total number of excess requests in the group is more than n, where n is the number of common buses. If a request pattern has more than n excess requests, then this excess must be coming from a set S of at least $g_{min}(n)$ subgroups and a subgroup in S must have at least $\sigma(n)$ requests. Having determined $g_{min}(n)$ and $\sigma(n)$ we can determine which request patterns will have at least one

subgroup receiving less than B_g while producing $> n$ excess requests in the group. Also among all such request patterns we need to check only for unique patterns (i.e., patterns which are not permutations of a request pattern under consideration).

$$\begin{aligned} P_{f(I_1,I_2,\cdots,I_g)}(n) &= 1 - \sum_{i_1=0}^{B_g-1} \sum_{i_2=i_1}^{M_g} \cdots \sum_{i_\beta=max(\sigma(n),i_{\beta-1})}^{M_g} \sum_{i_{\beta+1}=i_\beta}^{M_g} \cdots \sum_{i_g=i_{g-1}}^{M_g} \\ & \quad \hat{Pr}(i_1, i_2, \cdots, i_g)\lambda(i_1, i_2, \cdots, i_g : n) \\ \text{where } \beta &= g - g_{min}(n) + 1 \end{aligned}$$

In the above expression the limit for i_1 ensures one subgroup gets less than B_g buses. The limits for subgroup β ensures that if the excess request $> n$ has to come through $g_{min}(n)$ subgroups then one subgroup gets at least $\sigma(n)$ requests. By making lower limit of each index equal to the value of previous index we ensure that we consider only unique request patterns.

From the distribution function $P_{f(I_1,I_2,\cdots,I_g)}(n)$, we can determine the probability that a PRMB system with n common buses per group will allocate the same number of buses as in the corresponding group of the base model system counterpart. In the following subsection we will consider specific examples with different network configurations under different request models.

2.5 Numerical Results

Here we consider some base model networks and for each network evaluate the number of common buses per group needed for the corresponding PRMB system such that with high probability (to defined) the PRMB system will offer the same

bus allocation (for a random request pattern) as that in the corresponding base model system.

First consider a network with $P = M = 64, B = 32$, and $G = 1$. Notice that this network is a full bus connection system. We consider also a PRMB system where the group is further divided into 4 subgroups. We evaluate the cdf function $P_{f(I_1,I_2,\cdots,I_g)}(n)$ considering different possible values for n and different possible request rates under uniform request distribution. Figure [1] 2.3 shows the probability that the PRMB system will allocate the same number of buses as in a base model system for different possible numbers of common buses and different possible request rates. From the figure it is clear that for any request rate, with 16 common buses, the probability will be more than .99 that a request pattern will be satisfied in the same way as in a base model system. Also note that for any number of common buses, the probability that a request pattern gets the same number of buses as in the base model system does not change monotonically with the request rate and that its value is lowest around r =.7. This is because at $r = .7$, the mean number of requests a subgroup gets is approximately $\frac{B_{G}}{g}$ and obviously fluctuations around this mean will require more common buses than for other mean values. For the same network we considered other request models.

Figure 2.4, Figure 2.5 and Figure 2.6 show, respectively, the results for a hotspot request model, a locality based computation model and a locality based computation model with local hotspots with locality rate of l=.3 and a hotspot rate of h=.1. Note that the hotspot request model assumes a single hotspot. For the locality based model we consider $C = 4$. The non-monotone nature of the cdf function with

[1] In the figure caption throughout the dissertation we denote the network size by $P \times M \times B$ where they are respectively the number of processors, memory modules and total number of buses.

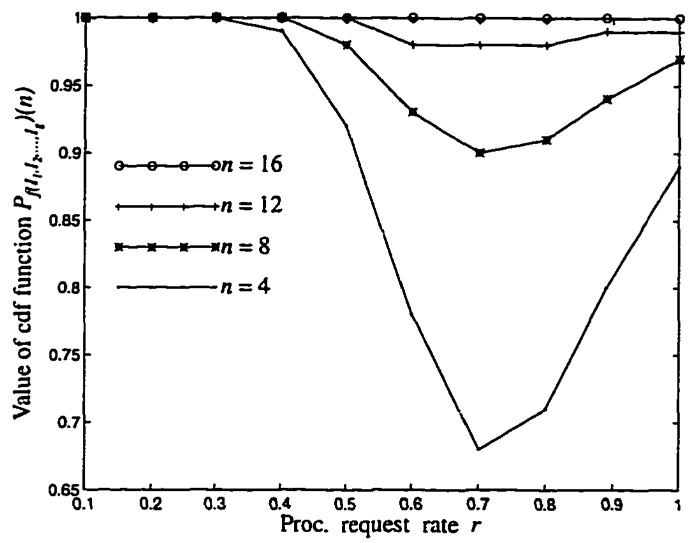


Figure 2.3: Probability that a request pattern will get the same number of buses as in a base model system under uniform request model for a network of size $64 \times 64 \times 32$ with 1 group and 4 subgroups in each group.

respect to the request rate is clear for all these request models. For the hotspot model the minimum value for the cdf function is at $r \approx .8$, whereas for other request models it is at $r \approx .7$ (Figures 2.4–2.6) An important observation from these figures is that for all these request models we need at least 16 common buses so that the probability of a request pattern being satisfied in the same way as in a base model system is at least .99. This can be explained as follows.

First consider the cases of uniform and locality based request models. The value of $P_{f(l_1,l_2,\ldots,l_g)}(n)$ function for different request rates under these two request models are practically the same. This is because for both the request models the probability that a memory module gets a request in a memory cycle is the same. For both the request models this probability is obtained by assuming zero hotspot rate in the

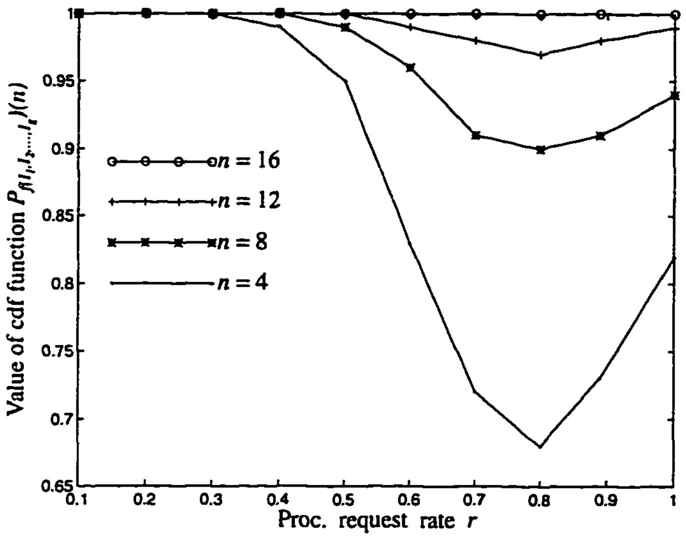


Figure 2.4: Probability that a request pattern will get the same number of buses as in a base model system under hotspot request model for a network of size $64 \times 64 \times 32$ with 1 group and 4 subgroups in each group ($h = .1$).

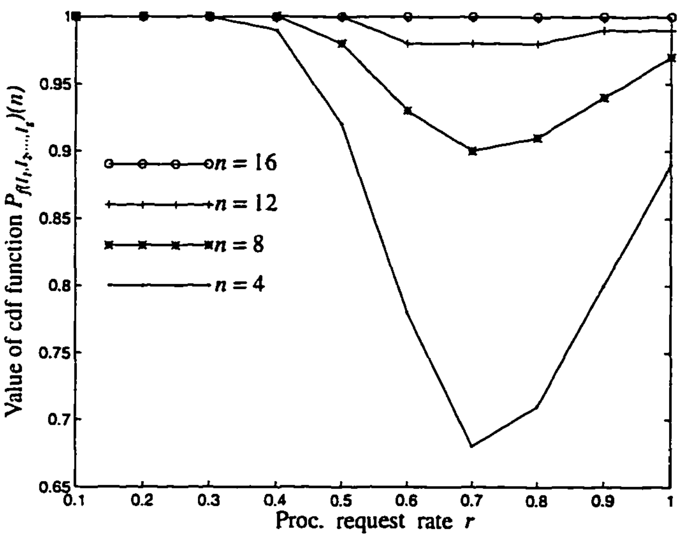


Figure 2.5: Probability that in a PRMB system a request pattern will get the same number of buses as in a base model system under locality based request model for a network of size $64 \times 64 \times 32$ with 1 group and 4 subgroups in each group ($C = 4, l = .3$).

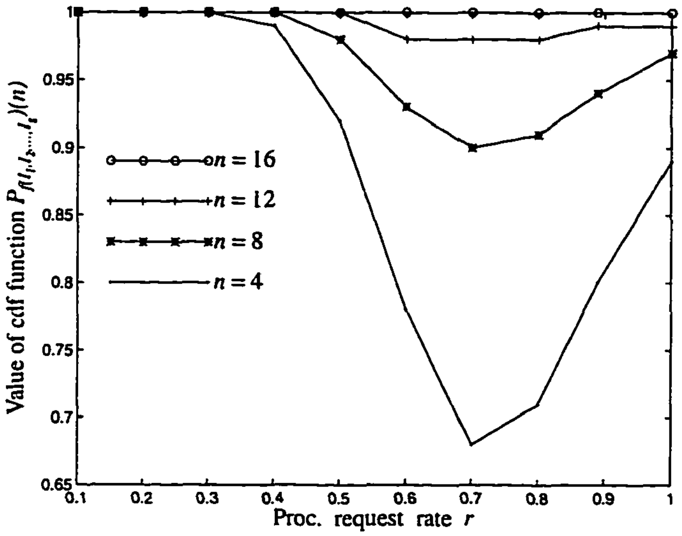


Figure 2.6: Probability that in a PRMB system a request pattern will get the same number of buses as in a base model system under locality based request model with local hotspot for a network of size $64 \times 64 \times 32$ with 1 group and 4 subgroups in each group $(C = 4, l = .3, h = .1)$.

generalized request model. Therefore,

$$
\begin{aligned}
q_r &= 1 - (1 - \frac{rl}{M_l} - \frac{r(1-l)}{M})^{P_l}(1 - \frac{r(1-l)}{M})^{P_n} \\
&= 1 - (1 - \frac{Crl + r(1-l)}{M})^{\frac{P}{C}}(1 - \frac{r(1-l)}{M})^{P-\frac{P}{C}} \qquad (2.1)
\end{aligned}
$$

Using the above formula we evaluate the value of q_r for the uniform request model and for the locality based model with a locality rate of $l = .3$. Note here that the uniform request model is a special case of the locality based model with one class and and a locality rate of unity. The values of q_r for different request rates are reported in Table 2.1. By comparing the first two columns of this table it is evident that for all request rates the value of q_r is almost the same for both request models. This can be explained mathematically as well. In Equation 2.1 the quantities $\frac{Crl+r(1-l)}{M}$ and

$\frac{r(1-l)}{M}$ are very small, because $M >> Crl + r(1-l)$ and $M >> r(1-l)$. Therefore we can write q_r as

$$\begin{aligned} q_r &\approx 1 - \exp(-\frac{Crl + r(1-l)}{M} \times \frac{P}{C}) \exp(-\frac{r(1-l)}{M} \times \frac{PC - P}{C}) \\ &= 1 - \exp(-\frac{rP}{M}) \end{aligned} \tag{2.2}$$

Equation 2.2 shows that q_r is independent of the number of classes C and the locality rate l. So the value of q_r should not vary for different locality based request models (which include the uniform request model as a special case). In Table 2.1 we report the values of q_r evaluated by using Equation 2.2. It is clear that values of q_r obtained this way are almost identical to those obtained for the uniform and the locality based model.

Now the fact that q_r is about the same for both request models directly implies that a given request pattern will have the same probability under both request models. Since $P_{f(I_1, I_2, \ldots, I_g)}(n)$ is the sum of probabilities of some request patterns (those satisfied by n common buses in the same way as in the base model system), the value of this function for both request models remains almost the same. Therefore, the number of common buses needed for a desired performance level is the same for both request models. Note that a request pattern for a PRMB system only considers the total number of memory modules in each subgroup (of a group) receiving requests. It does not take into account the source of requests (because bus assignment does not depend on that). If for bus assignment the source of requests had to be taken into account, then the probability of a request pattern will differ for a uniform request model from that of a locality based request model and the value of the cdf function in that case will be different for these two request models. This argument

will be better supported in Chapter 6 when we consider another variant of PRMB systems, where the source of requests plays an important role in bus assignment.

In the case of a hotspot request model the value of q_r is different from that of a uniform request model. Besides, in a hotspot request model the hot memory module gets requests at a different rate than q_r (it gets request at a rate of q_h). From Figure 2.4 it is clear that for the hotspot request model the value of $P_{f(I_1,I_2,\cdots,I_g)}(n)$ is different from that in a uniform request model for any given request rate. However this difference is not very substantial and 16 common buses are needed so that the value of the cdf function $\geq .99$. This is explained as follows. The hotspot memory module gets requests at a higher rate than a regular memory module. For the hotspot rate of $h = .1$, considered in the example, bias towards the hotspot is not significant. Note that here we don't assume queuing, where significant number of processors in any memory cycle can be blocked at the hotspot module. Under the present request model (non queue situation) and with the present hotspot rate, the hotspot module gets only slight bias over other modules. So the probability distribution that a given number of memory modules in the subgroup containing the hotspot module will get requests in any memory cycle does not vary significantly from that of other subgroups. The request situation therefore will not change much from that of a uniform request model.

With a higher hotspot rate bias towards hotspot memory module can be increased to a great extent. This can be seen if we consider a hotspot rate of, $h = .5$. In that case the value of the function $P_{f(I_1,I_2,\cdots,I_g)}(n)$ changes substantially from that for a uniform request model. This is illustrated in Figure 2.7. In that case we need only 12 common buses to make the value of the cdf function $\geq .99$. The reason for the drop in the needed number of common buses is that with high bias towards the

hotspot module, excess requests will tend to be less evenly distributed among subgroups and statistically the surplus buses would tend to come from more subgroups than in the case of a uniform request model. If the same number of surplus buses comes from more subgroups, the number of common buses will be obviously less.

The case of a locality based computation model with local hotspot is analogous to the hotspot case. However since the hotspots are distributed in different classes and a hot module is favored only by processors in the local class, the impact of hotspots is even less significant. If the rate of locality and the hotspot rates are increased sufficiently there will be substantial changes in $P_{f(I_1,I_2,\cdots,I_g)}(n)$. The locality rate has to be increased because the hotspot modules are favored only by local processors. For the same network, if we change the locality rate l to .7 and the hotspot rate h to .7, as illustrated in Figure 2.8, the value of the function $P_{f(I_1,I_2,\cdots,I_g)}(n)$ will vary significantly from that for a locality based request model. The number of common buses needed for the cdf function to be $\geq .99$ in this case will change to 12. The reason for the drop in the number of common buses is analogous to that given in the case of a hotspot request model.

Now that we have evaluated the number of common buses required for the PRMB system, we can numerically evaluate the memory bandwidth for the PRMB system and compare it with the corresponding value for the base model system. Table 2.2 lists the memory bandwidths for both the PRMB system and the corresponding base model system for different request rates. It also shows memory bus connectivity cost for both systems. For simplicity memory bus connectivity cost is assumed to equal to the number of interfaces between all the memory modules and all the buses. From the table it is clear that while there is almost no degradation in performance, there is significant reduction in connectivity cost. We evaluate the memory bandwidth for

Table 2.1: Comparison of q_r under uniform request model, locality based computation model with $C = 6$ and $l = .3$ and the value from Equation 2.2.

Request rate r	q_r uniform request model	locality based model	value from Eqn 2.2
.1	.095	.095	.095
.2	.182	.182	.181
.3	.260	.260	.259
.4	.331	.331	.330
.5	.395	.395	.393
.6	.453	.453	.451
.7	.505	.506	.503
.8	.553	.554	.551
.9	.596	.597	.593
1.0	.635	.636	.632

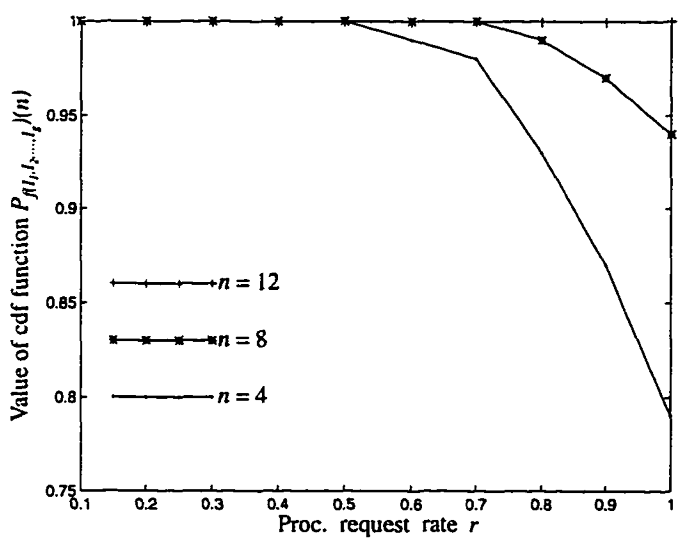


Figure 2.7: Probability that a request pattern will get the same number of buses as in a base model system under hotspot request model for a network of size $64 \times 64 \times 32$ with 1 group and 4 subgroups in each group ($h = .5$).

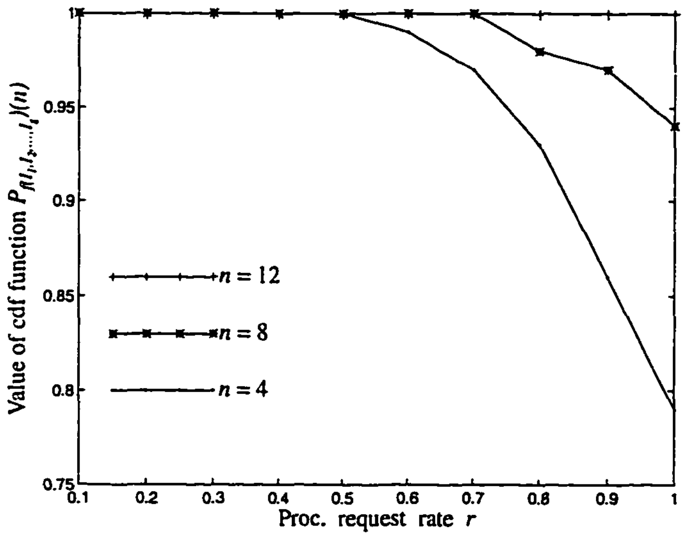


Figure 2.8: Probability that in a PRMB system a request pattern will get the same number of buses as in a base model system under locality based request model with local hotspot for a network of size $64 \times 64 \times 32$ with 1 group and 4 subgroups in each group ($C = 4, l = .7, h = .7$).

other request models as well. Table 2.3, Table 2.4 and Table 2.5 show the results, respectively, for the hotspot request model, the locality based computation model and the locality based computation model with local hotspot. As can be seen, for all these request models the performance of the two systems are almost identical.

We consider some more networks for numerical illustration. For all these networks we consider a moderate locality rate of $l = .3$ and a hotspot rate of $h = .1$ (Note that the hotspot rates of .5 or .7, considered earlier for illustration purpose are not realistic). For locality based request models number of classes considered is different for different networks. Figures 2.9 through 2.24 show the cdf function for theses networks. The results for memory bandwidth are illustrated in Tables 2.6–2.21. The results obtained for these networks are similar to those obtained for the previously considered networks. The results show improvement in memory bus

Table 2.2: Memory bandwidths under a uniform request model for a network of size $64 \times 64 \times 32$, 1 group and 4 subgroups.

Request rate r	Memory bandwidth for base model system cost=2048	Memory bandwidth for PRMB system ($B_c = 16$) cost=1288
.1	6.095	6.095
.2	11.618	11.618
.3	16.621	16.621
.4	21.150	21.150
.5	25.188	25.187
.6	28.464	28.461
.7	30.575	30.570
.8	31.565	31.561
.9	31.899	31.896
1.0	31.981	31.980

Table 2.3: Memory bandwidths of multiple under a hotspot request model for a network of size $64 \times 64 \times 32$, 1 group and 4 subgroups with $h = .1$.

Request rate r	Memory bandwidth for base model system cost=2048	Memory bandwidth for PRMB system ($B_c = 16$) cost=1288
.1	5.946	5.946
.2	11.163	11.163
.3	15.827	15.827
.4	20.040	20.040
.5	23.845	23.8433
.6	27.132	27.127
.7	29.592	29.585
.8	31.046	31.039
.9	31.699	31.694
1.0	31.923	31.921

Table 2.4: Memory bandwidths under a locality based request model for a network of size $64 \times 64 \times 32$, 1 group and 4 subgroups with $C = 4, l = .3$.

Request rate r	Memory bandwidth for base model system cost=2048	Memory bandwidth for PRMB system ($B_c = 16$) cost=1288
.1	6.097	6.097
.2	11.626	11.626
.3	16.637	16.637
.4	21.177	21.177
.5	25.224	25.223
.6	28.502	28.499
.7	30.603	30.599
.8	31.579	31.575
.9	31.903	31.901
1.0	31.983	31.982

Table 2.5: Memory bandwidths under a locality based request model with local hotspot for a network of size $64 \times 64 \times 32$, 1 group and 4 subgroups with $C = 4, l = .3$ and $h = .1$.

Request rate r	Memory bandwidth for base model system cost=2048	Memory bandwidth for PRMB system ($B_c = 16$) cost=1288
.1	6.084	6.084
.2	11.599	11.599
.3	16.596	16.596
.4	21.121	21.121
.5	25.157	25.156
.6	28.438	28.435
.7	30.561	30.556
.8	31.560	31.556
.9	31.897	31.895
1.0	31.981	31.980

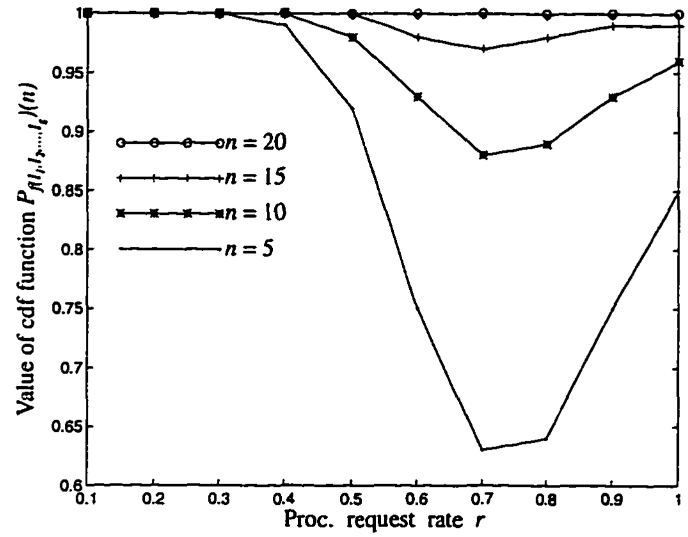


Figure 2.9: Probability that in a PRMB system a request pattern will get the same number of buses as in a base model system under uniform request model for a network of size $70 \times 70 \times 35$ with 1 group and 5 subgroups in each group.

connectivity cost with almost no change in memory bandwidth. The memory bus connectivity cost can be improved further (in some cases) by developing an optimal PRMB network which as we discuss in the next chapter.

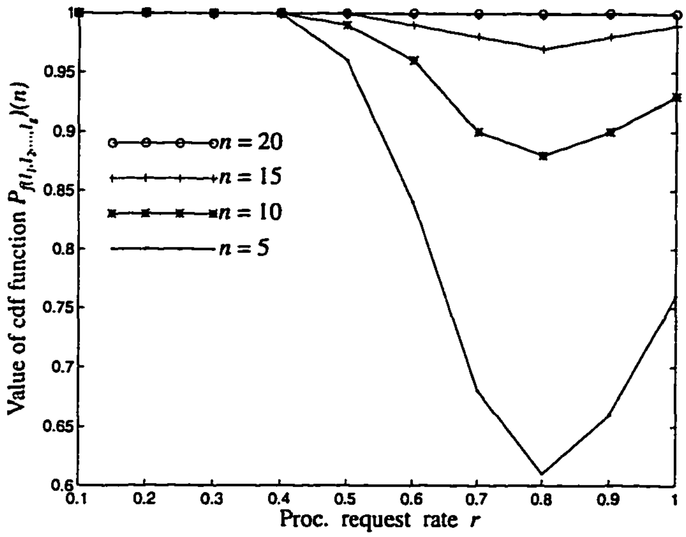


Figure 2.10: Probability that in a PRMB system a request pattern will get the same number of buses as in a base model system under hotspot request model for a network of size $70 \times 70 \times 35$ with 1 group and 5 subgroups in each group ($h = .1$).

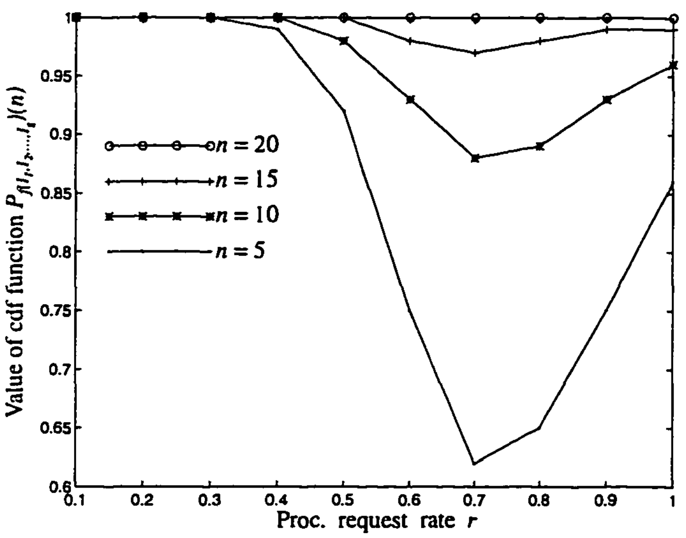


Figure 2.11: Probability that in a PRMB system a request pattern will get the same number of buses as in a base model system under locality based request model for a network of size $70 \times 70 \times 35$ with 1 group and 5 subgroups in each group ($C = 10, l = .3$).

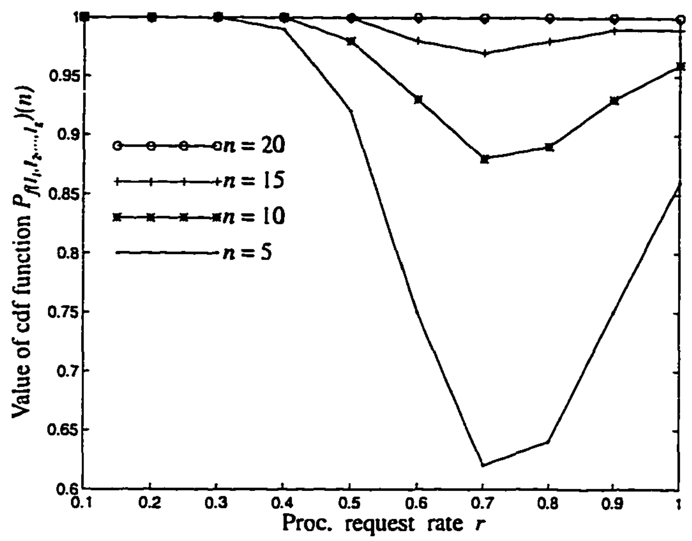


Figure 2.12: Probability that in a PRMB system a request pattern will get the same number of buses as in a base model system under locality based request model with local hotspots for a network of size $70 \times 70 \times 35$ with 1 group and 5 subgroups in each group ($C = 10, l = .3$).

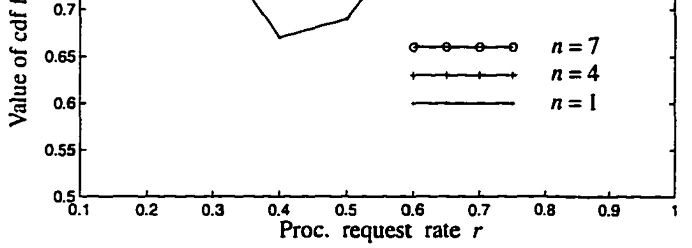


Figure 2.13: Probability that in a PRMB system a request pattern will get the same number of buses as in a base model system under uniform request model for a network of size $90 \times 90 \times 30$ and 3 groups with 3 subgroups in each group.

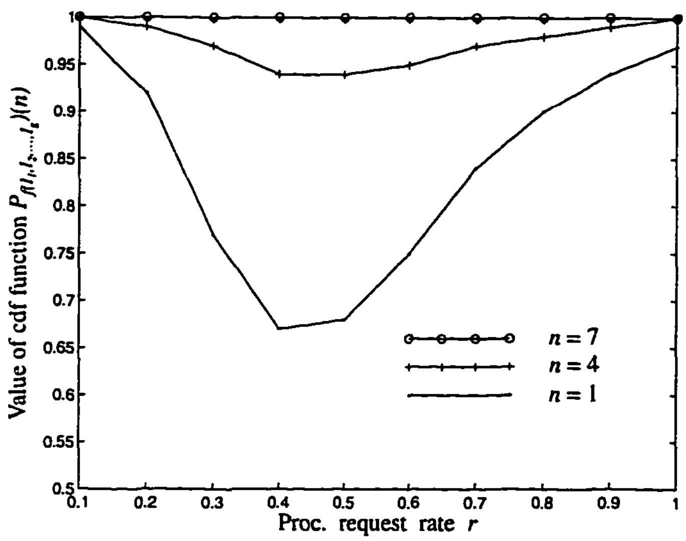


Figure 2.14: Probability that in a PRMB system a request pattern will get the same number of buses as in a base model system under hotspot request model for a network of size $90 \times 90 \times 30$ and 3 groups with 3 subgroups in each group ($h = .1$).

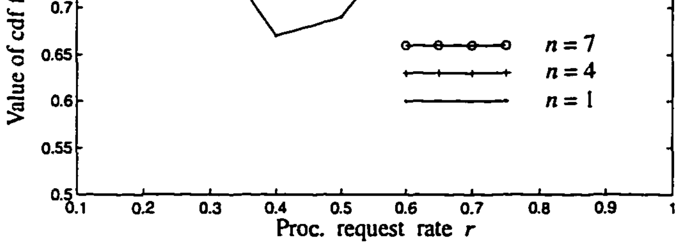


Figure 2.15: Probability that in a PRMB system a request pattern will get the same number of buses as in a base model system under locality based request model for a network of size $90 \times 90 \times 30$ with 3 groups and 3 subgroups in each group ($C = 6, l = .3$).

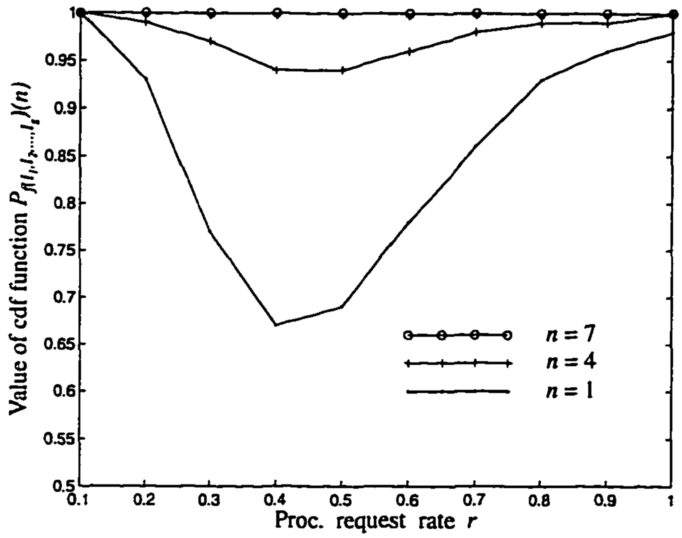


Figure 2.16: Probability that in a PRMB system a request pattern will get the same number of buses as in a base model system under locality based request model with local hotspots for a network of size $90 \times 90 \times 30$ with 3 groups and 3 subgroups in each group ($C = 6, l = .3, h = .1$).

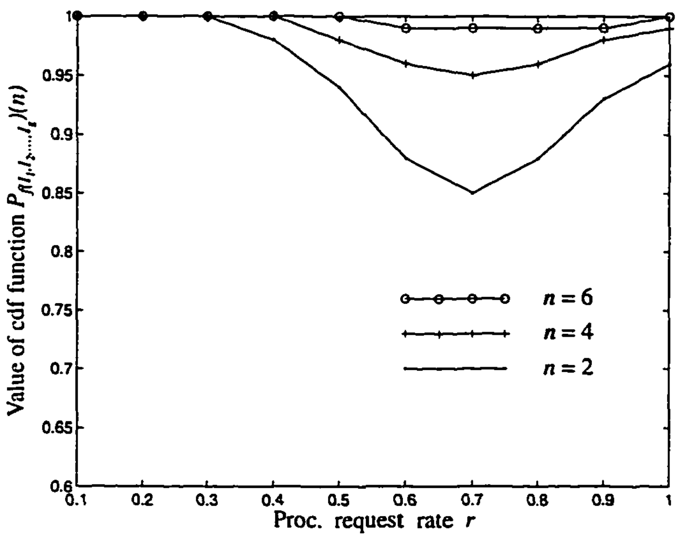


Figure 2.17: Probability that in a PRMB system a request pattern will get the same number of buses as in a base model system under uniform request model for a network of size $96 \times 96 \times 48$ with 2 groups and 2 subgroups in each group.

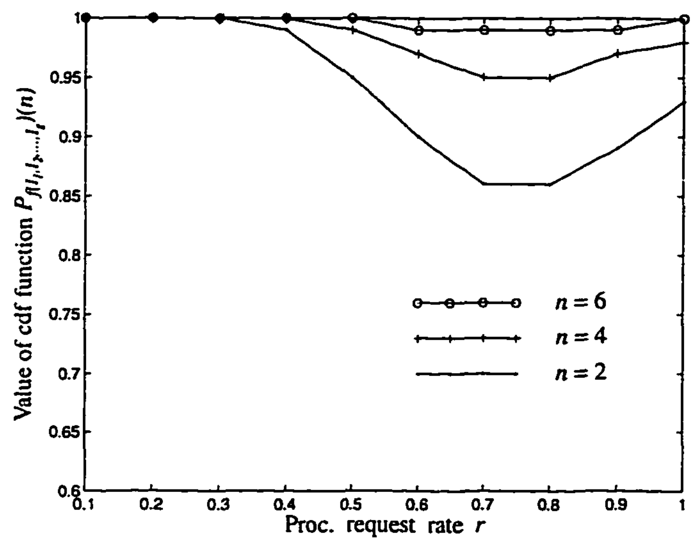


Figure 2.18: Probability that in a PRMB system a request pattern will get the same number of buses as in a base model system under hotspot request model for a network of size $96 \times 96 \times 48$ with 2 groups and 2 subgroups in each group ($h = .1$).

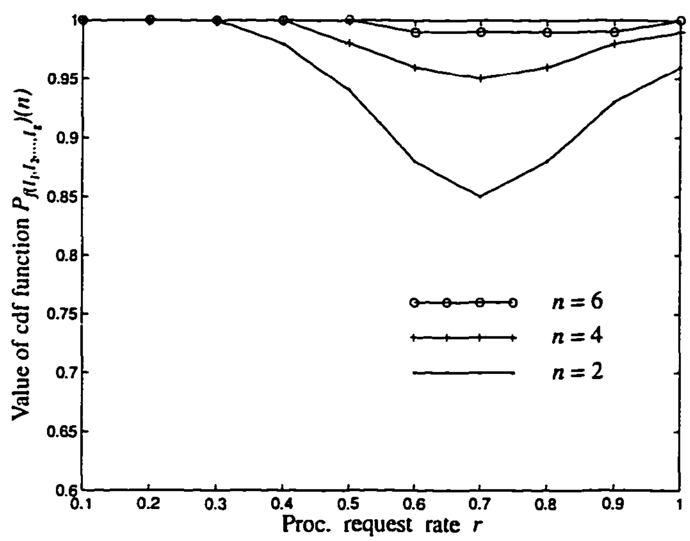


Figure 2.19: Probability that in a PRMB system a request pattern will get the same number of buses as in a base model system under locality based request model for a network of size $96 \times 96 \times 48$ with 2 groups and 2 subgroups in each group ($C = 4, l = .3$).

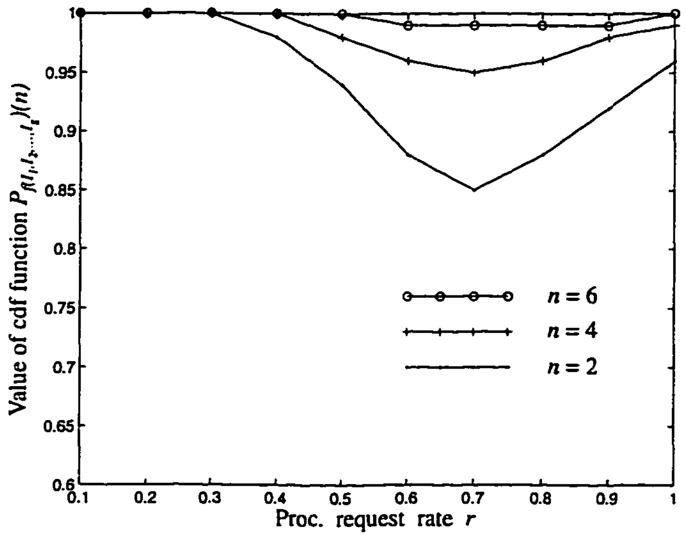


Figure 2.20: Probability that in a PRMB system a request pattern will get the same number of buses as in a base model system under locality based request model with local hotspots for a network of size $96 \times 96 \times 48$ with 2 groups and 2 subgroups in each group ($C = 4, l = .3, h = .1$).

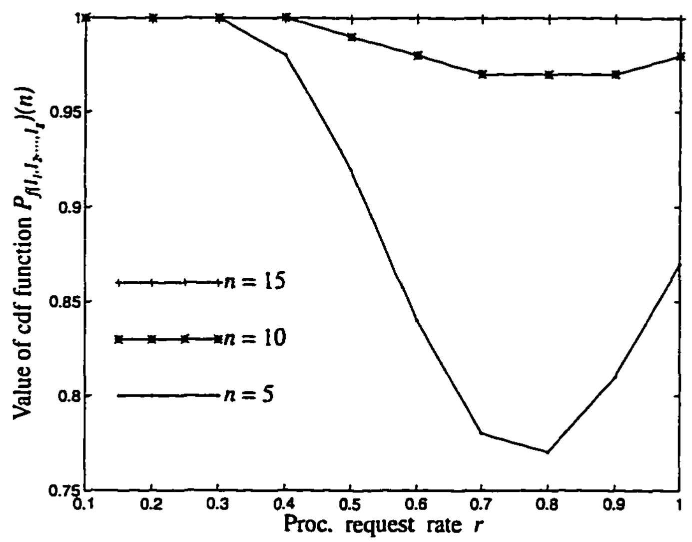


Figure 2.21: Probability that in a PRMB system a request pattern will get the same number of buses as in a base model system under uniform request model for a network of size $120 \times 120 \times 60$ with 3 groups and 5 subgroups in each group.

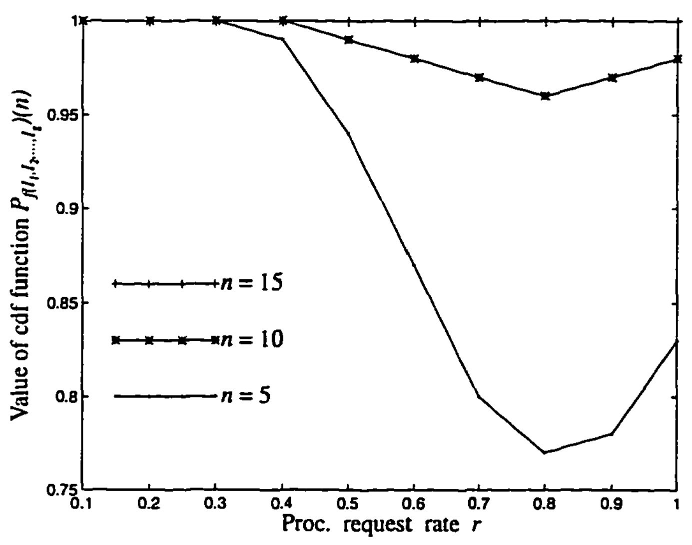


Figure 2.22: Probability that in a PRMB system a request pattern will get the same number of buses as in a base model system under hotspot request model for a network of size $120 \times 120 \times 60$ with 3 groups and 5 subgroups in each group ($h = .1$).

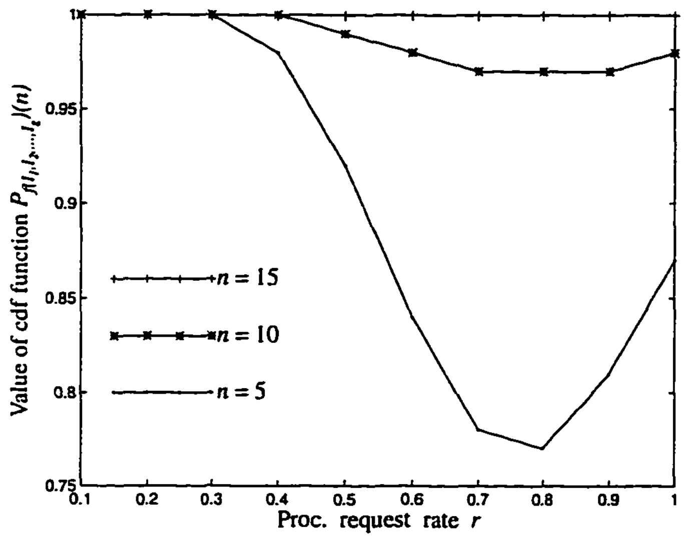


Figure 2.23: Probability that in a PRMB system a request pattern will get the same number of buses as in a base model system under locality based request model for a network of size $120 \times 120 \times 60$ with 3 groups and 5 subgroups in each group $(C = 6, l = .3)$.

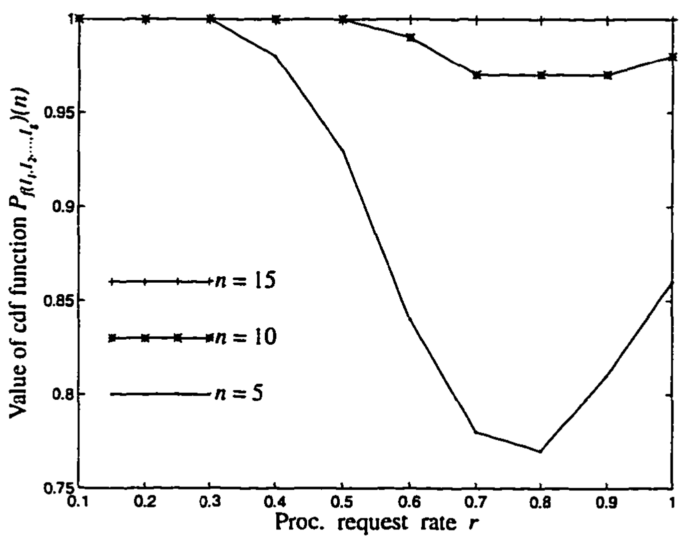


Figure 2.24: Probability that in a PRMB system a request pattern will get the same number of buses as in a base model system under locality based request model with local hotspots for a network of size $120 \times 120 \times 60$ with 3 groups and 5 subgroups in each group $(C = 6, l = .3, h = .1)$.

Table 2.6: Memory bandwidth under a uniform request model for a network of size $70 \times 70 \times 35$ with 1 group and 5 subgroups in a group.

Request rate r	Memory bandwidth for base model system cost=2450	Memory bandwidth for PRMB system ($B_c = 20$) cost=1610
.1	6.666	6.666
.2	12.705	12.705
.3	18.176	18.176
.4	23.129	23.129
.5	27.559	27.558
.6	31.180	31.178
.7	33.512	33.507
.8	34.573	34.570
.9	34.909	34.907
1.0	34.985	34.984

Table 2.7: Memory bandwidth under a hotspot request model for a network of size $70 \times 70 \times 35$ with 1 group and for the PRMB system and 5 subgroups in a groupwith $h = .1$.

Request rate r	Memory bandwidth for base model system cost=2450	Memory bandwidth for PRMB system ($B_c = 20$) cost=1610
.1	6.491	6.491
.2	12.177	12.177
.3	17.265	17.265
.4	21.865	21.865
.5	26.031	26.031
.6	29.659	26.657
.7	32.398	32.395
.8	34.005	34.001
.9	34.705	34.701
1.0	34.931	34.929

Table 2.8: Memory bandwidth under a locality based request model for a network of size $70 \times 70 \times 35$ with 1 group and 5 subgroups in the group with $C = 10, l = .3$.

Request rate r	Memory bandwidth for base model system cost=2450	Memory bandwidth for PRMB system ($B_c = 20$) cost=1610
.1	6.670	6.670
.2	12.719	12.719
.3	18.204	18.204
.4	23.174	23.174
.5	27.619	27.619
.6	31.245	31.242
.7	33.558	33.554
.8	34.596	34.592
.9	34.916	33.265
1.0	34.987	34.986

Table 2.9: Memory bandwidth under a locality based request model with local hotspot for a network of size $70 \times 70 \times 35$ with 1 groups and 5 subgroups in each group with $C = 10, l = .3$ and $h = .1$

Request rate r	Memory bandwidth for base model system cost=2450	Memory bandwidth for PRMB system cost=1610
.1	6.668	6.668
.2	12.713	12.713
.3	18.193	18.193
.4	23.156	23.156
.5	27.595	27.595
.6	31.220	31.218
.7	33.542	33.538
.8	34.588	34.585
.9	34.914	34.912
1.0	34.986	34.986

Table 2.10: Memory bandwidth under a uniform request model for a network of size $90 \times 90 \times 30$ with 3 groups and 3 subgroups in each group.

Request rate r	Memory bandwidth for base model system cost=900	Memory bandwidth for PRMB system ($B_c = 7$) cost=720
.1	8.569	8.569
.2	16.275	16.273
.3	22.633	22.625
.4	26.805	26.791
.5	28.870	28.856
.6	29.664	29.655
.7	29.914	29.909
.8	29.980	29.978
.9	29.996	29.995
1.0	29.999	29.999

Table 2.11: Memory bandwidth under a hotspot request model for a network of size $90 \times 90 \times 30$ with 3 groups and 3 subgroups in each group with $h = .1$.

Request rate r	Memory bandwidth for base model system cost=900	Memory bandwidth for PRMB system ($B_c = 7$) cost=720
.1	8.294	8.294
.2	15.503	15.502
.3	21.549	21.543
.4	25.845	25.832
.5	28.276	28.261
.6	29.385	29.373
.7	29.807	29.799
.8	29.945	29.942
.9	29.986	29.984
1.0	29.997	29.996

Table 2.12: Memory bandwidth under a locality based request model for a network of size $90 \times 90 \times 30$ with 3 groups and 3 subgroups in each group with $C = 6, l = .3$.

Request rate r	Memory bandwidth for base model system cost=900	Memory bandwidth for PRMB system $(B_c = 7)$ cost=720
.1	8.571	8.571
.2	16.282	16.280
.3	22.645	22.637
.4	26.817	26.803
.5	28.878	28.864
.6	29.668	29.659
.7	29.915	29.910
.8	29.981	29.979
.9	29.996	29.995
1.0	29.999	29.999

Table 2.13: Memory bandwidth under a locality based request model with local hotspot for a network of size $90 \times 90 \times 30$ with 3 groups and 3 subgroups in each group with $C = 6, l = .3$ and $h = .1$.

Request rate r	Memory bandwidth for base model system cost=900	Memory bandwidth for PRMB system $(B_c = 7)$ cost=720
.1	8.566	8.566
.2	16.266	16.264
.3	22.620	22.612
.4	26.795	26.681
.5	28.865	28.851
.6	29.662	29.653
.7	29.913	29.908
.8	29.980	29.978
.9	29.996	29.995
1.0	29.993	29.999

Table 2.14: Memory bandwidth under a uniform request model for a network of size $96 \times 96 \times 48$ with 2 groups and 2 subgroups in each group.

Request rate r	Memory bandwidth for base model system cost=2304	Memory bandwidth for PRMB system ($B_c = 6$) cost=1440
.1	9.140	9.140
.2	17.418	17.418
.3	24.915	24.914
.4	31.687	31.685
.5	37.645	37.634
.6	42.375	42.352
.7	45.470	45.442
.8	47.065	47.043
.9	47.713	47.700
1.0	47.926	47.920

Table 2.15: Memory bandwidth under a hotspot request model for a network of size $96 \times 96 \times 48$ with 2 groups and 2 subgroups in each group with $h = .1$.

Request rate r	Memory bandwidth for base model system cost=2304	Memory bandwidth for PRMB system ($B_c = 6$) cost=1440
.1	8.832	8.832
.2	16.543	16.543
.3	23.466	23.466
.4	29.746	29.745
.5	35.401	35.395
.6	40.221	40.204
.7	43.836	43.810
.8	46.098	46.071
.9	47.258	47.238
1.0	47.751	47.739

Table 2.16: Memory bandwidth under a locality based request model for a network of size $96 \times 96 \times 48$ with 2 groups and 2 subgroups in each group with $C = 4, l = .3$.

Request rate r	Memory bandwidth for base model system cost=2304	Memory bandwidth for PRMB system ($B_c = 6$) cost=1440
.1	9.141	9.141
.2	17.423	17.423
.3	24.924	24.923
.4	31.701	31.699
.5	37.664	37.653
.6	42.295	42.372
.7	45.485	45.458
.8	47.074	47.052
.9	47.717	47.705
1.0	47.927	47.921

Table 2.17: Memory bandwidth under a locality based request model with local hotspot for a network of size $96 \times 96 \times 48$ with 2 groups and 2 subgroups in each group with $C = 4, l = .3$ and $h = .1$.

Request rate r	Memory bandwidth for base model system cost=2304	Memory bandwidth for PRMB system ($B_c = 6$) cost=1440
.1	9.133	9.133
.2	17.392	17.392
.3	24.863	24.863
.4	31.605	31.605
.5	37.543	37.532
.6	42.274	42.252
.7	45.396	45.369
.8	47.024	47.002
.9	47.696	47.683
1.0	47.920	47.914

Table 2.18: Memory bandwidth under a uniform request model for a network of size $120 \times 120 \times 60$ with 3 groups and 5 subgroups in each group.

Request rate r	Memory bandwidth for base model system cost=2400	Memory bandwidth for PRMB system ($B_c = 15$) cost=1920
.1	11.424	11.424
.2	21.769	21.769
.3	31.134	31.134
.4	39.574	39.574
.5	46.922	46.920
.6	52.686	52.682
.7	56.499	56.492
.8	58.568	58.562
.9	59.495	59.490
1.0	59.844	59.841

Table 2.19: Memory bandwidth under a hotspot request model for a network of size $120 \times 120 \times 60$ with 3 groups and 5 subgroups in each group with $h = .1$.

Request rate r	Memory bandwidth for base model system cost=2400	Memory bandwidth for PRMB system ($B_c = 15$) cost=1920
.1	11.417	11.417
.2	21.745	21.745
.3	31.086	31.086
.4	39.501	39.500
.5	46.831	46.829
.6	52.597	52.593
.7	56.432	56.426
.8	58.529	58.523
.9	59.476	59.471
1.0	59.837	59.833

Table 2.20: Memory bandwidths under a locality based request model for a network of size $120 \times 120 \times 60$ with 3 groups and 5 subgroups in each group with $C = 6, l = .3$.

Request rate r	Memory bandwidth for base model system cost=2400	Memory bandwidth for PRMB system ($B_c = 15$) cost=1920
.1	11.426	11.426
.2	21.776	21.776
.3	31.149	31.149
.4	39.598	39.598
.5	46.954	46.952
.6	52.719	52.715
.7	56.525	56.519
.8	58.585	58.579
.9	59.503	59.499
1.0	59.848	59.845

Table 2.21: Memory bandwidth of multiple under a locality based request model with local hotspot for a network of size $120 \times 120 \times 60$ with 3 groups and 5 subgroups in each group with $C = 6, l = .3$ and $h = .1$.

Request rate r	Memory bandwidth for base model system cost=2400	Memory bandwidth for PRMB system ($B_c = 15$) cost=1920
.1	9.133	9.133
.2	17.392	17.392
.3	24.863	24.863
.4	31.605	31.605
.5	37.543	37.532
.6	42.274	42.252
.7	45.396	45.369
.8	47.024	47.002
.9	47.696	47.683
1.0	47.920	47.914

CHAPTER 3

OPTIMAL PRMB ARCHITECTURE

In this chapter we attempt to find the optimal design for the PRMB architecture. A PRMB architecture is said to be *optimal* if for certain *performance level*, it incurs minimum memory bus connectivity cost. Performance level in this context is the probability that the PRMB system will provide the same number of buses to a request pattern as that provided by the corresponding base model system. Memory -bus connectivity [1] in a PRMB system has two components–connectivity due to common buses and connectivity due to local buses. In the previous chapter we evaluated the minimum number of common buses for a PRMB system that will ensure certain level of performance. However the number was evaluated considering the number of subgroups in a group to be fixed and given. If the number of subgroups within a group is allowed to vary, then the number of common buses may also vary thus resulting in different bus connectivity cost. If the number of common buses increases then the connectivity cost due to local buses is likely to decrease and connectivity cost due to common buses will increase. The overall memory–bus connectivity cost might increase or decrease depending on the size of the subgroups. If the number of common buses is known for a certain number of subgroups in a group, then that information might be useful in determining the number of common buses for some different number of subgroups.

[1] Recall the memory–bus connectivity cost is the cost of the connections between the memory modules and the buses only

It is possible that for a given performance level there may be some relationship between the required number of common buses for two different PRMB systems. In that case knowing the number of common buses in one architecture may make finding the number of common buses in another architecture easy. Consider two architectures one with g_1 subgroups in each group and the other with g_2 subgroups in each group. Consider an arbitrary request pattern $(i_1, i_2, \cdots, i_{g_1})$ in the first architecture. In this request pattern i_k memory modules in the k^{th} subgroup could be any i_k memory modules in that subgroup. We can consider the requested memory modules in the entire group and from that determine the request pattern in the second architecture. Let the request pattern in the second architecture determined this way be denoted by $(\tilde{i}_1, \tilde{i}_2, \cdots, \tilde{i}_{g_2})$. We call the request pattern $(\tilde{i}_1, \tilde{i}_2, \cdots, \tilde{i}_{g_2})$ in the second architecture "the *mapped request pattern*" of $(i_1, i_2, \cdots, i_{g_1})$ from first architecture. These two request patterns obviously have the same probability under any request model. We will explore possible relationship between the minimum number of common buses required for a request pattern $(i_1, i_2, \cdots, i_{g_1})$ in the first architecture and the number of common buses required by the mapped request pattern $(\tilde{i}_1, \tilde{i}_2, \cdots, \tilde{i}_{g_2})$ in the second architecture for the same bus assignment as in the base model system. Since the choice of request pattern $(i_1, i_2, \cdots, i_{g_1})$ is arbitrary, the relationship will hold for any particular request pattern in the first configuration and the corresponding request pattern in the second configuration. We will then use this general and deterministic relationship to probabilistically evaluate the minimum number of common buses needed in the second architecture given the minimum number of common buses needed in the first architecture. First we determine the number of common buses required to satisfy a request pattern in the PRMB system with the same number of buses as in the base model system.

3.1 NUMBER OF COMMON BUSES NEEDED FOR A GIVEN REQUEST PATTERN

To evaluate the minimum number of common buses required to satisfy a request pattern with the same number of buses as in the base model system we introduce the following definitions

Definition 3.1 *The number of memory requests in excess of* $\left\lfloor \frac{B_G}{g} \right\rfloor$ *to the* k^{th} *subgroup, is denoted by* e_k *and the number of requests short of* $\left\lfloor \frac{B_G}{g} \right\rfloor$ *to the* k^{th} *subgroup is denoted by* s_k.

Thus,

$$\begin{aligned} e_k &= \max(i_k - \left\lfloor \frac{B_G}{g} \right\rfloor, 0) \\ s_k &= \max(\left\lfloor \frac{B_G}{g} \right\rfloor - i_k, 0) \end{aligned}$$

where i_k is the number of requests directed at distinct memory modules in the k^{th} subgroup. We also define the following quantities

$$\begin{aligned} \Delta &= B_G - g\left\lfloor \frac{B_G}{g} \right\rfloor \\ \xi &= \sum_{k=1}^{g} e_k - \Delta \\ \zeta &= \sum_{k=1}^{g} s_k \end{aligned}$$

Obviously a subgroup may have at most $\left\lfloor \frac{B_G}{g} \right\rfloor$ local buses and thus Δ is the number of remaining buses after each subgroup is assigned $\left\lfloor \frac{B_G}{g} \right\rfloor$ buses. Therefore, a PRMB system has to have a minimum of Δ common buses in each group. ξ is the total

number of excess requests in all the subgroups of a group and ζ is the number of surplus buses in all the subgroups of a group, provided that each memory subgroup is exclusively connected to $\left\lfloor \frac{B_G}{g} \right\rfloor$ buses.

Definition 3.2 *$h(N : \alpha_1, \alpha_2, \cdots, \alpha_n)$ is a recursive function where N is its argument and the α_i's are the parameters. Let γ_i be the indicator function for the i^{th} parameter α_i. The function is defined as follows*

$$h(N : \alpha_1, \alpha_2, \cdots, \alpha_n) = \begin{cases} 0, & \text{if either } N \leq 0 \text{ or } \forall i, 1 \leq i \leq n, \alpha_i = 0 \\ n + h(N - \sum_{k=1}^{n} \gamma_k : \min(\alpha_1 - 1, 0), \cdots, \min(\alpha_n - 1, 0)) \\ , & \text{otherwise} \end{cases}$$

In each recursion step parameter α_i is decremented by 1 as long as $\alpha_i \geq 1$. The recursion argument is decremented by the sum of the decrements of the recursion parameters. Each step of the recursion adds n, the total number of recursion parameters to the function $h(.)$. Finally the recursion terminates either when all the parameters are zeros or when the recursion argument becomes ≤ 0 for the first time.

The following theorem makes use of the above definitions to determine the minimum number of common buses in a PRMB system so that a request pattern $(i_1, i_2, \cdots, i_g)$ gets the same number of buses as in a corresponding base model system

Theorem 1 *The minimum number of common buses required for a given request pattern $(i_1, i_2, \cdots, i_g)$ such that in the PRMB system the request pattern will be assigned the same number of buses as that assigned in a base model system is given*

by:

(i) $f(i_1, i_2 \cdots, i_g) = h(\zeta : s_1, s_2, \cdots, s_g) + \Delta$, if $\xi \geq \zeta$

(ii) $f(i_1, i_2, \cdots, i_g) = h(\xi : s_1, s_2 \cdots, s_g) + \Delta$, if $\xi < \zeta$

Proof:(i) When $\xi \geq \zeta$, the number of excess requests is more than the number of surplus buses, if as many buses as possible are connected locally within subgroups. In this case the subgroups with surplus buses have to give up all the extra buses. Therefore, those subgroups have to supply ζ buses to the subgroups which have bus shortages. Since in a PRMB architecture all the memory subgroups will be connected to the same number of local buses, if a certain number of buses has to be taken away from a subgroup, then the same number has to be taken away from each of the other subgroups. Suppose that a particular subgroup say the k^{th} subgroup, has the largest number of surplus buses and suppose that the amount of surplus is δ_k. Since all the surplus has to be taken away, the number of buses that would be taken away from the k^{th} subgroup is δ_k. So, altogether $g \times \delta_k$ buses will be taken away from all the subgroups and the number of common buses will be $g \times \delta_k + \Delta$. Now we evaluate the expression $h(\zeta : s_1, s_2, \cdots, s_g)$. Since $\zeta = \sum_{i=1}^{g} s_i$, the argument ζ of the recursive expression $h(\zeta : s_1, s_2, \cdots, s_g)$ cannot become zero before all the parameters become zeros. Since each step of the recursion decrements a parameter by 1, the number of steps needed to make ζ equal to zero is $Max(s_1, s_2, \cdots, s_g) = \delta_k$. Since the number of parameters in the recursive expression is g, then $h(\zeta : s_1, s_2, \cdots, s_g) = g \times \delta_k$ and the minimum number of common buses needed is $h(\zeta : s_1, s_2, \cdots, s_g) + \Delta$.

(ii) When $\xi < \zeta$, the number of excess requests is less than the number of surplus buses. Therefore we need to take surplus buses only to the extent needed to meet ξ excess requests. Since there are more surplus buses in different subgroups than there

are bus shortages, there may exist different possible ways of taking surplus buses from subgroups, all of which might not result in the same number of common buses. Suppose to satisfy the excess requests of ξ, the maximum number of surplus buses a subgroup contributes is δ_{max}. Now depending on how we take surplus buses from different subgroups, the value of δ_{max} may vary. We can take surplus buses from different subgroups in an even fashion or we can take relatively more buses from some subgroups than others. Depending on our choice, the value of δ_{max} may be small or large. We want to make δ_{max} as small as possible because the number of common buses finally will be $g\delta_{max} + \Delta$. The recursive relation $h(\xi : s_1, s_2, \cdots, s_g)$ takes one surplus bus from a subgroup at each step as long as the subgroup has a surplus and the argument of the recursive expression (which at any step represents the number of excess requests at that step) is > 0. This ensures that a subgroup is not favored over another in the sense that two buses will not be taken from a subgroup without taking at least one surplus bus from every other subgroup. This ensures that the largest number of surplus buses taken from any subgroup is minimum. Therefore, the minimum number of common buses needed is $h(\xi : s_1, s_2, \cdots, s_g) + \Delta$ ■

3.2 Relation between the number of common buses in two different PRMB architectures

In this section we will explore the relationship between the number of common buses needed for two PRMB systems derived from the same base model system. Let the number of subgroups in each of the groups in the first PRMB system be denoted by g_1 and and the same number for the second system be denoted by g_2. We introduce the following notation which will be subsequently used for the rest of the discussion in this chapter.

Notation

ϵ : the intended performance level for a PRMB system under design. The PRMB system must assign the same number of buses to a random request pattern as the base model system with probability $\geq \epsilon$ for it to achieve a performance level of ϵ.

B_{c_1}: the minimum number common buses, per group needed in the first configuration for a performance level of ϵ

B_{c_2}: the minimum number common buses, per group in the second configuration to be considered for a performance level of ϵ

We will use subscripts to distinguish between the parameters for the two systems under consideration. For example ξ_1 and ξ_2 will denote the number of excess requests in a group in the first and second configurations, respectively.

Now if the value of B_{c_1} is known and there is some relationship between B_{c_1} and B_{c_2} then that relationship can be used to determine the value for B_{c_2}. We first look at how bus surplus and excess requests in a group for a given request pattern vary among the two configurations. The following is a lemma to that end.

Lemma 2 *If $g_1 = kg_2$, where k is an integer > 1 and if the number of buses in a group B_G is divisible by both g_1 and g_2, then for any request pattern $(i_1, i_2, \cdots, i_{g_1})$ in the first configuration and the mapped request pattern $(\tilde{i}_1, \tilde{i}_2, \cdots, \tilde{i}_{g_2})$ in the second configuration, $\xi_1 \leq \xi_2$ and $\zeta_1 \leq \zeta_2$*

Proof: A subgroup in the first configuration corresponds to k subgroups in the second configuration. Since B_G is divisible by both g_1 and g_2, $\Delta_1 = \Delta_2 = 0$. Therefore

the number of local buses in the bigger sized subgroup of the first configuration is equal to the sum of the number of local buses in the corresponding smaller sized subgroups in the second configuration. Since, $(i_1, i_2, \cdots, i_{g_1})$ and $(\tilde{i}_1, \tilde{i}_2, \cdots, \tilde{i}_{g_2})$ correspond to a specific set of processor requests to distinct memory modules, bus shortage in a subgroup of the first configuration will be at most equal to the sum of bus shortages in the smaller sized subgroups in the second configuration. This is because some of the shortages in some of the smaller sized subgroups might be satisfied by the surplus buses in some other smaller subgroups when they merge and thus $\xi_1 \leq \xi_2$. The same argument applies for bus surpluses in the two configurations and thus $\zeta_1 \leq \zeta_2$. ∎

The following theorem uses Lemma 2 to establish a lower bound on the minimum number of common buses to consider (per group) for a PRMB system under design given that the minimum number of common buses needed to achieve a desired performance level in another configuration is known.

Theorem 2 (i) *If* $g_2 = kg_1$*, where* k *is a positive integer, then* $B_{c_2} \geq B_{c_1}$ (ii) *If* $g_2 = kg_1$*, where* $k > 1$ *is not necessarily an integer and* $n = \lfloor k \rfloor$*, then* $B_{c_2} \geq \frac{k}{n} B_{c_1} - \frac{k-n}{n} B_G$

proof:(i) There could be two possible scenarios: First, the number of common buses in a group, B_G, is divisible by both g_1 and g_2. Second, B_G is not evenly divisible by either g_1 or g_2 or by both.

Consider a request pattern $(i_1, i_2, \cdots, i_{g_1})$ in the first architecture and let the mapped request pattern in the second architecture be denoted by $(\tilde{i}_1, \tilde{i}_2, \cdots, \tilde{i}_{g_2})$. Consider the case when B_G is divisible by both g_1 and g_2. A subgroup in the first configuration corresponds to k subgroups in the second configuration. By Lemma 2 the number of excess requests in a group in the first configuration will be at

most equal to that in the corresponding group of the second configuration (if we consider request patterns $(i_1, i_2, \cdots, i_{g_1})$ and $(\tilde{i}_1, \tilde{i}_2, \cdots, \tilde{i}_{g_2})$ as defined above). So the number of surplus buses that need to be taken from a group in the first configuration will be at most equal to that taken from the corresponding group in the second configuration. The number of surplus buses a subgroup in the first configuration needs to contribute should be at most equal to that contributed by the corresponding k subgroups in the second configuration. Let us consider an arbitrary subgroup, say, the p^{th} subgroup, in the first configuration. This subgroup corresponds to the $(pk+1)^{\text{st}}, \cdots, (pk+k)^{\text{th}}$ subgroups in the second configuration. Suppose that the maximum number of surplus buses any one of these subgroups (in the second configuration) provides is δ_{max}. Thus the number of surplus buses provided by all these subgroups in the second configuration will be $\leq k\delta_{max}$. Therefore, the p^{th} subgroup in the first configuration has to provide $\leq k\delta_{max}$ of its surplus buses. The number of iteration steps needed in the recursive relation of Theorem 1 to achieve that many buses will be at most k times those needed in the case of the second configuration. Since each step for the first architecture generates $1/k^{\text{th}}$ the number of buses of the second architecture, the number of common buses in the first architecture will be at most equal to that in the second architecture.

Now let us consider the case when B_G is not divisible by either g_1 or g_2. Since $g_1 < g_2$, then $\Delta_1 \leq \Delta_2$. In other words, the number of common bus to start with in the first configuration is less than or equal to the corresponding number in the second configuration. Suppose that, for the first configuration, instead of starting with Δ_1 common buses, we start with Δ_2 common buses. In that case the total bus shortage in a subgroup of the first configuration will be at most equal to the sum of bus shortages in the corresponding k subgroups in the second configuration. The

same argument applies for bus surplus in a subgroup in the first configuration. So if we start with Δ_2 common buses in the first configuration, we have the relations $\xi_1 \leq \xi_2$ and $\zeta_1 \leq \zeta_2$. As shown in the above case, the number of buses that would be taken from each subgroup in the first configuration is at most equal to that taken form the corresponding subgroups in the second configuration. Had we started with Δ_1 common buses instead, the final number of common buses would not be greater than that obtained by starting with Δ_2.

So for both cases, the number of common buses required to satisfy a request pattern $(i_1, i_2, \cdots, i_{g_1})$ in the first configuration will be less than or equal to that needed to satisfy the mapped request pattern $(\tilde{i}_1, \tilde{i}_2, \cdots, \tilde{i}_{g_2})$ in the second configuration. Note that our choice of the first request pattern is arbitrary. Therefore, this relationship will hold for every request pattern in the first configuration and the mapped one in the second configuration. Let the set of request patterns which are satisfied in the second configuration with B_{c_2} common buses, in the same way as in the base model system, be denoted by S. Clearly, the minimum number of common buses required in the first configuration for each of the request patterns in set S will be less than or equal to B_{c_2}. Therefore $B_{c_2} \geq B_{c_1}$.

(ii) First we consider the case when B_G is divisible by both g_1 and g_2. In that case a subgroup in the first configuration will contribute a number of surplus buses at most equal to those supplied by the corresponding k subgroups in the second configuration. Since k is not an integer, the corresponding set of k subgroups in the second configuration consists of $n = \lfloor k \rfloor$ full subgroups and a fraction of a subgroup; the value of the fraction being $(k - n)$. Let A be a subgroup in the first configuration. There will be n full subgroups and a fraction of a subgroup in the second configuration corresponding to subgroup A. Let B be the subgroup in the

second configuration that is split. Let the fraction of B in the second configuration that becomes part of the subnetwork corresponding to A be denoted by SG_B and the remainder of the fraction of be denoted by $\tilde{SG_B}$. It is possible that for a given request pattern there will be fewer requests (may be none) for the memory modules belonging to SG_B and more requests (may be all) for the memory modules in $\tilde{SG_B}$. In that case the surplus buses in A (in the first configuration) correspond to the surplus buses in n full subgroups in the second configuration and some buses (may be all) belonging to SG_B. The number of buses connected to memory modules in SG_B is $(k-n)\frac{B_G}{g_2}$. Let the maximum number of surplus buses that is taken from a subgroup in the second configuration be denoted by δ_{max}. In the first configuration a subgroup has to contribute a number of surplus buses which is at most equal to that contributed by the corresponding n full subgroups in the second configuration and those coming from the split subgroup, which is $\leq (k-n)\frac{B_G}{g_2}$. Therefore, the total number of surplus buses that needs to be taken away from a subgroup in the first configuration is

$$\leq n\delta_{max} + (k-n)\frac{B_G}{g_2}$$

$$= \frac{n}{k}k\delta_{max} + (k-n)\frac{B_G}{g_2}$$

Hence the number of common buses per group required in the first configuration is

$$\leq g_1\frac{n}{k}k\delta_{max} + g_1(k-n)\frac{B_G}{g_2}$$

$$= \frac{n}{k}g_2\delta_{max} + (k-n)\frac{B_G}{k}$$

Since this relation holds for any given request pattern , we can write,

$$B_{c_1} \leq \frac{n}{k} B_{c_2} + \frac{k-n}{k} B_G$$

In other words,

$$B_{c_2} \geq \frac{k}{n} B_{c_1} - \frac{k-n}{n} B_G$$

Thus for this case the theorem is true.

Now consider the case when the number of buses in a group is not evenly divisible by either g_1 or g_2 or by both. As before, starting with Δ_2 common buses for the first configuration the situation becomes analogous to that when B_g is divisible by both g_1 and g_2 and the above relation follows. Hence the theorem is true for all cases. ■

Theorem 2 gives the minimum value for the number of common buses that should be considered for a PRMB architecture if the minimum number of common buses is known for another architecture derived from the same base model system. If the number of common buses thus determined for the second architecture does not ensure the desired performance level then we have to consider a higher value for the number of common buses. It is possible that while increasing the number of common buses for the architecture under consideration we could exceed the connectivity cost of the first architecture. Even if the desired performance level is achieved we clearly will not accept an architecture with higher cost. Therefore we have to have an upper bound on the acceptable number of common buses for an architecture under consideration. The following lemma establishes such an upper bound.

Lemma 3 *The upper bound on the acceptable number of common buses in the second configuration is given by*

$$B_{c_2} < \frac{g_2(g_1 - 1)B_{c_1} + (g_2 - g_1)B_G}{g_1(g_2 - 1)}$$

Proof: Memory bus connectivity cost for a group in the first architecture is given by $\frac{M_G}{g_1}B_{g_1}g_1 + B_{c_1}M_G = M_G B_{g_1} + B_{c_1}M_G$. Similarly memory bus connectivity for a group in the second architecture is $M_G B_{g_2} + B_{c_2}M_G$. Therefore, the second architecture is less costly than the first architecture if $M_G B_{g_2} + B_{c_2}M_G < M_G B_{g_1} + B_{c_1}M_G$ which after simplification reduces to

$$B_{c_2} < \frac{g_2(g_1 - 1)B_{c_1} + (g_2 - g_1)B_G}{g_1(g_2 - 1)}$$

■

In the following section we present an algorithm to determine the optimal PRMB system based on the relationships established by Theorem 2 and Lemma 3.

3.3 Algorithm for Determining the Optimal PRMB Architecture

Thus far we have determined the lower bound and the upper bound on the number of common buses to consider for a PRMB system given the minimum number of common buses needed to achieve the desired performance level in another PRMB system, derived from the same base model. A closed form expression for B_{c_2} from the knowledge of B_{c_1} does not seem to be possible. As such, we could not find a direct analytical solution for an optimal PRMB architecture. However to determine the optimal PRMB system we have taken the following approach. We assume here

that a base model system is given (i.e. M, P, B and G are known) and that we are trying to design a lower connectivity cost system but with the same basic parameters (M, P, B and G).

First we determine the factors of M_G, the number of memory modules in a group. We then consider the factors which are less than B_G, the total number of buses in a group of the base model system. Now the number of subgroups in a group could only be one of these factors. Let the minimum factor be denoted by g_{min} and the maximum factor (lower than B_G) be denoted by g_{max}. The number of subgroups in a group could only be a factor of M_G in the interval $[g_{min}, g_{max}]$. We start with an architecture with the minimum number of subgroups in a group (the lowest factor of M_G) and evaluate the number of common buses which ensures the performance level of ϵ. Then using Theorem 2 we can determine the minimum number of buses that would be required for the next possible architecture. By next architecture we mean the architecture with the next higher number of subgroups in a group. Though we know the minimum value for the number of common buses for the next architecture, we still don't know whether or not that minimum value will ensure a performance level of ϵ. We have to check starting with the determined minimum number of common buses and gradually increase that number till the desired performance level is achieved. However it is possible that the number of common buses thus determined will incur higher cost than the previously considered architecture. So while trying to find the number of common buses stating with the minimum value and increasing successively, we must not exceed the upper bound given by Lemma 3. . Let the minimum number of common buses that need to be considered for an architecture be denoted by b_{c_l} and the maximum number that could be allowed by cost consideration be denoted by b_{c_h}. For the first architecture these values are g_{min}

Algorithm 1

- INPUTS : M, P, B, G
- Determine the factors of M_G. Denote them by $g_1, g_2 \cdots, g_n$ in ascending order, where $\forall k, 2 \leq g_k < B_G$.

For($k = 1; k \leq n; k++$)
{
$g = g_k$
Determine bc_l and bc_h applying Theorem 2 and Lemma 3 respectively
For($B_c = bc_l; B_c < bc_h; B_c += g_k$)
{
$P_{f((I_1,J_1),(I_1,J_1)\cdots(I_{g_k},J_{g_k}))}(B_c) = 1.0$
$B_g = \frac{B_G - B_c}{g}$
For($i_1 = 0; i_1 < B_g; i_1 ++$)
For($i_2 = i_1; i_2 \leq M_g; i_2 ++$)
$\vdots$
For($i_{g-g_{min}} = i_{g-g_{min}-1}; i_{g-g_{min}} \leq M_g; i_{g-g_{min}} ++$)
For($i_{g-g_{min}+1} = \max(i_{g-g_{min}}, \alpha(B_c)); i_{g-g_{min}+1} \leq M_g; i_{g-g_{min}+1} ++$)
$\vdots$
For($i_g = i_{g-1}; i_2 \leq M_g; i_g ++$)
{
$P_{f((I_1,J_1),(I_1,J_1)\cdots(I_{g_k},J_{g_k}))}(B_c)$
$-= \hat{Pr}(i_1, i_2, \cdots, i_g) u(i_1, i_2, \cdots, i_g)$
}
If $P_{f(I_1,I_2,\cdots(I_g)}(B_c) \geq \epsilon$
break
}
}

Figure 3.1: Algorithm to determine optimal PLMB architecture

and B_G respectively. For all subsequent architectures these values are determined by applying Theorem 2 and Lemma 3.

Algorithm 1, listed in Figure 3.1, starts with the architecture with minimum number of subgroups in a group. It then checks subsequent configurations with increasing number of subgroups in a group. For each configuration it looks for the minimum number of common buses within the interval of $[b_{c_l}, b_{c_h}]$ that will ensure a performance level of ϵ.

The complexity of the algorithm is $O(\alpha\beta^{g_{max}-1})$ where $\alpha = \frac{B_G B_{g_{max}}}{g_{min}}(\frac{C}{G} + g_{max})$ and $\beta = M_{g_{max}}(\frac{C}{G} + g_{max})$. The complexity of the algorithm is determined in Appendix A. This is an exponential time algorithm which needs to be executed for different request rates and request models. Though the algorithm has high complexity for some networks, it needs to be run only once during the design phase.

3.4 Numerical Results

By applying Algorithm 1 we attempt to determine the optimal architectures for the networks considered earlier. We choose ϵ to be .99; that is the probability that a request pattern will be satisfied in the same way as in the base model system is $\geq .99$. For probabilistic connectivity reduction, the performance level ϵ is chosen to be very high.

For our results we consider the four request models utilized earlier. For the hotspot rate we assume $h = .1$ and for the locality based request models we assume $l = .3$. We consider a broad range of request rates for processors (from .1 to 1.0). The results of our evaluation are illustrated in Table 3.1. The table lists relevant parameters for optimal PRMB networks corresponding to each of the base model systems considered earlier. For the optimal PRMB network, the number of

subgroups in each group, the number of groups and the number of common buses needed per group are reported. For both the base model system and the optimal PRMB systems memory bus connectivity costs are also listed in the table. If we compare these connectivity costs we find that the optimal PRMB achieves significant reduction in memory bus connectivity (35%–42%) while providing performance within 1% of the corresponding (and more costly) base model system. Notice that base model system with $G = 1$ is a full bus connection system and for other values of G it is a traditional partial connection bus system.

Table 3.1: Cost comparison of optimal PRMB architectures and corresponding base model system

Base model system	Optimal PRMB system		cost of base model	cost of PRMB system
	number of subgroups g	number of common buses B_c		
$P, M = 64, B = 32, G = 1$	2	8	2048	1288
$P, M = 70, B = 35, G = 1$	2	9	2450	1540
$P, M = 90, B = 30, G = 3$	2	4	900	540
$P, M = 96, B = 48, G = 2$	3	9	2304	1344
$P, M = 120, B = 60, G = 3$	2	6	2400	1560

To see the effect of relaxing the requirement on performance level we have evaluated the minimum number of common buses and memory-bus connectivity cost for the same network with $\epsilon = .95$. The results are illustrated in Table 3.2. Clearly requiring *epsilon* to be only .95 results in further cost reduction form most networks. In this case cost reduction is from 40%–44% compared to the base model system. Notice that in two cases the cost remained the same as in the case of $\epsilon = .99$.

Table 3.2: Cost comparison of optimal PRMB architecture and base model system

Base model system	Optimal PRMB system		cost of base model	cost of PRMB system
	number of subgroups g	number of common buses B_c		
$P, M = 64, B = 32, G = 1$	2	6	2048	1216
$P, M = 70, B = 35, G = 1$	2	9	2450	1400
$P, M = 90, B = 30, G = 3$	2	4	900	540
$P, M = 96, B = 48, G = 2$	3	9	2304	1344
$P, M = 120, B = 60, G = 3$	2	6	2400	1440

As opposed to relaxing the requirement for the performance level, if we increase the requirement for performance level the number of common buses needed is likely to increase and so will the memory–bus connectivity cost. However increasing the performance level $\epsilon > .99$ does seem to be justified at the cost of possibly accepting a higher cost.

CHAPTER 4

QUEUING ANALYSIS OF PRMB SYSTEMS

Performance of multiple bus systems has been studied in the literature both in the presence of memory queues and without memory queues [3],[5]–[19], [21]– [24]. While involving memory queues is likely to add to the hardware complexity, queues tend to improve utilization of system resources. If memory queues are present then a processor does not have to resubmit a rejected request. While waiting on a memory queue a processor can be busy with other activities. In the design of a multiprocessor system whether or not queues should be involved depends on the expected set of applications. Nevertheless, performance of multiprocessor systems should be studied in presence of memory queues. In the case of the PRMB system it is important that performance be evaluated considering memory queues to check any possible adverse affect on system performance due to connectivity reduction. So the performance of the PRMB system in presence of memory queue should be compared with that of the corresponding base model system also in the presence of memory queues.

Queuing analysis for multiple bus systems is generally considered a difficult problem. Thus far most research in this area has focused on full bus connection systems and partial connection systems. These two classes have been incorporated in our base model. The contention for resources in these systems is simpler than that in PRMB systems. Nonetheless, even with full and partial connection systems, the queuing problem becomes complicated for several reasons. First, unlike conventional queuing networks multiple bus systems have passive resources, like buses. A memory server functions only if a bus is available. Second, with the increase in

the number of processors or memory modules the number of states in the queuing network increases exponentially [18]. A third difficulty stems from the distribution of the memory service time which is constant.

The approaches taken by the researchers in the past are as follows. Boudec [13] proved that if each processor is an infinite server with exponential service rate and there are M classes of customers (corresponding to M memory modules) and B exponential servers (corresponding to B buses) then the multiple bus system has a product form solution. The assumptions adopted by Boudec [13] for the processors are restrictive and those for the memory–bus subsystems are unrealistic. Chiola, Marsan and Balbo [6] claimed to provide exact solutions for multiple bus systems with the assumption of general service rate for the processors and exponential service time for memory servers. They assumed the multiple bus system to be a product form network based on Boudec's proof, though their model is not exactly the same as that of Boudec (different service time distribution for processors). Mudge and Al-Sadoun [19] proposed a semi–markov model for the analysis of a multiple bus system under a uniform request model. While their model has limited number of states, it requires solving a semi–markov process describing the behavior of a processing element individually. This model is applicable, however, only in a situation where passive resources are global and the request model is uniform [19]. Irani and Onyuksel [8] provided a closed form solution for the multiple bus system considering a uniform request model. They too considered exponential service time for the memory servers. Marsan [18] gave an approximate solution which reduces the number of states by a lumping technique and several other approximations. This way he eliminated the numerical difficulties posed by exponentially increasing number of states for an exact model. His technique is also limited by the fact that memory

service time has been considered exponential. Mahmud [16] used the approach of mean value analysis for multilevel multiple bus system. He assumed exponential service time for both processors and memory modules. Towsley [21], [22] provided an approximate solution techniques based on the method of aggregation for multiple bus systems with both exponential and constant service times. While for the case of constant service time the multiprocessor system has to be homogeneous in the sense that all the processors behave identically, with exponential service time heterogeneous systems can be handled to some extent. The method of aggregation has been utilized by other researchers as well [6]. The aggregation technique to simplify a queuing network has originally been proposed by Chandy, Herzog and Woo [4]. They have shown that for a product form closed queuing network the aggregation technique gives exact solution [4].

The queuing problem in the PRMB system is more complicated than in other multiple bus systems studied thus far. While for the base model system allocation of passive resources does not depend on the distribution of non–empty queues within a group, for a PRMB system resource allocation depends on that distribution. So the number of states in the queuing networks for PRMB systems increases with the increase in the number of subgroups as well. So an exact solution for the PRMB system does not seem feasible, especially if we consider a realistic distribution for memory service time. We will be using an approximate technique for solving the queuing problems of PRMB systems based on the method of aggregation. It has been shown in [4] that the aggregation technique gives an exact solution for closed product form networks. For full bus connection systems Towsley [21], [22] has applied the aggregation technique and obtained results that were very close to those obtained by simulations. We will be adapting this technique for PRMB systems. However

due to bus contention which depends on the the number of non–empty queues in different subgroups of a group, we cannot directly aggregate memory queues in a group as it can be done in a full bus connection system. While memory queues at the subgroup level can directly be aggregated, aggregation from the subgroup level to the group level has to be done in a taking bus contention into account. We modified the aggregation technique in a very innovative way at this level. We used the probability distribution evaluated by method of aggregating two queue at a time and used those information to aggregate multiple number of queues simultaneously.

4.1 Model for PRMB queuing systems

The model for the queuing network of a multiple bus system depends on its architecture and the request model. The architecture defines how the passive resources are allocated to different requests in non-empty memory queues. The request model defines the arrival process and the distribution of memory service time. It also defines whether the network corresponds to an open queue model or a closed queue model. In previous chapters we have discussed the architecture of PRMB systems. To develop the queuing model for a PRMB system, we need a request model. We will make more realistic assumptions for both processor and memory service times. We will assume that a non–queued processor has a certain probability to generate a request every cycle and that memory service time is constant. The request model will encompass the uniform request distribution and the hot spot request distribution. The request model we consider is outlined below.

REQUEST MODEL

1. Processors operate in synchronous MIMD mode. Each processor generates requests independently of others. Request generation can take place only at the beginning of a cycle.

2. a memory request generated by a processor has certain probability to be directed to the hotspot module, otherwise it is equally likely to be directed to any memory module.

3. a processor is either *active* or in a *waiting* state. A processor is active when it is not queued to any memory module, otherwise it is in a waiting state. In the waiting state a processor does not generate new requests. At the beginning of each cycle an active processor has certain probability of generating a memory request.

4. Propagation delay and arbitration time are included in memory access time.

5. memory service time is constant and is equal to memory cycle time.

From the request model just described, the PRMB system can be modeled by a closed queuing network with P customers (one associated with each processor), and M queues (one associated with each memory module). When a processor makes a request, it joins the targeted memory queue. A memory server works if the associated memory queue is non-empty and a bus is available. The availability of a bus depends on the distribution of the non-empty memory queues among different subgroups. Figure 4.1 illustrates the closed queue model for a PRMB system

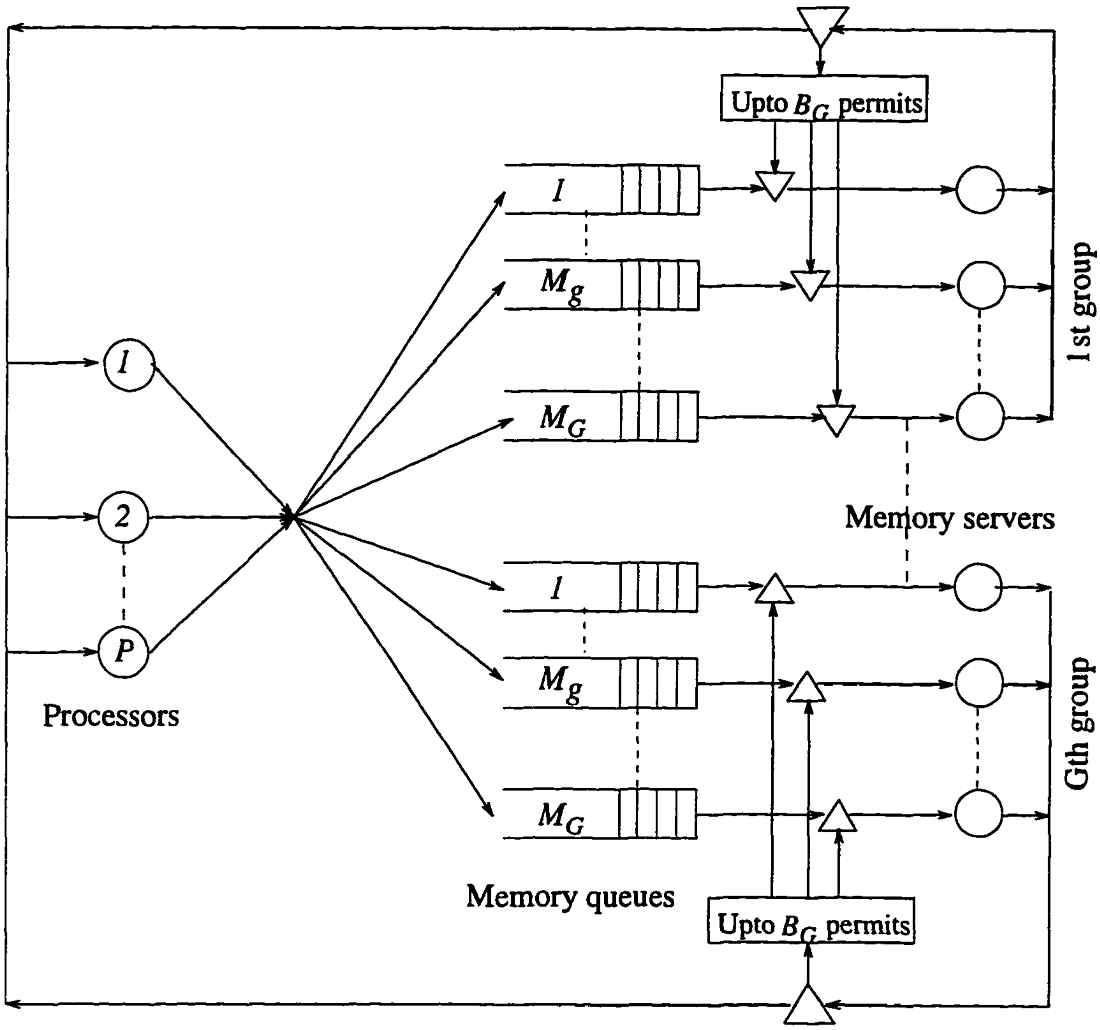


Figure 4.1: A closed queue model for a PRMB architecture. The number of permits in each group depends on the distribution of nonempty queues in subgroups within a group.

4.2 Approximate Analysis

We will simplify the PRMB queuing network by applying the method of aggregation [4]. In the aggregation technique, a subset of queues in a network is replaced by an equivalent composite queue. The equivalent queue is intended to behave in a very similar way to the replaced subset in its interaction with the rest of the network. The simplified network is much easier to solve than the original network. Chandy, Herzog and Woo [4] have been shown that the aggregation technique gives accurate results for closed product form networks. For non–product form networks the aggregation technique is used only as an approximating technique [4],[25]. The PRMB system as we modeled is in general a non–product form network. This has been illustrated through an example in Appendix B.

Suppose that we want to aggregate a subnetwork N of a queuing network Θ . Suppose that N contains the queues $Q_1, Q_2, \cdots, Q_i$ from queuing network Θ. We take these queues out of the original network, short the entry points and exit points of N and form a closed queue network. The probability that a customer from any of the queue outputs goes to a particular queue is made equal to the conditional probability that a customer will go to that queue in the original network given that the customer goes to subnetwork N. We solve this closed queue network and evaluate the combined service rate. In a continuous time model the combined service rate is the throughput of the network and in a discrete time model it is the distribution for different possible number of customers that can be served in a unit time. The subset of queues $(Q_1, Q_2, \cdots, Q_i)$ is then replaced by a single queue having a service rate equal to the combined service rate thus determined. In the discussion that follows we outline the aggregation procedure for a PRMB system.

We will simplify the PRMB queue system by replacing memory queues by an aggregated queue. We will then solve the two queue system consisting of a processor queue and the combined memory queue to determine the overall throughput. The aggregation of memory queues in a subgroup can be done without taking bus contention into account. However bus allocation within a group depends on the distribution of the number of non–empty queues in different subgroups and does not depend on how the non–empty memory queues are distributed within a subgroup. Therefore, memory queues within a subgroup can be directly aggregated. The service parameter of the aggregated queue (henceforth termed as subgroup queue) will have a one–to–one correspondence with the probability of a given number of non empty queues in a subgroup. The aggregation of subgroup queues into a single group queue has to be done taking bus contention into account because the combined service parameter depends on bus allocation. We discuss the aggregation procedure in the following subsection. The steps taken in the procedure are summarized as follows.

- Replace M_g memory queues in each subgroup by a single aggregate queue whose service rate reflects the behavior of M_g memory queues when there is no bus contention.
- Replace g aggregated queues in each group by a single composite queue including the effect of bus contention.
- Replace G aggregated queues, each representing the memory queues in a corresponding group, with a single queue.
- Represent all processors by a single queue.

- Solve a two queue network consisting of a processor queue and a combined memory queue to determine the throughput of the overall system.

4.2.1 Aggregating memory queues in a subgroup

In each subgroup there are M_g queues. These queues can be aggregated in one step or in $M_g - 1$ steps. In the latter approach at each step two queues are aggregated. Let us suppose that within a subgroup memory queues are numbered $1, 2, \cdots, M_g$ following any arbitrary order. At step 1, queue 1 and queue 2 are aggregated. At the i^{th} step, the $(i+1)^{\text{st}}$ queue is aggregated with the aggregated queue obtained in step $(i-1)$. The result of aggregation in the i^{th} step is an aggregated queue representing queues $1, 2, \cdots, i+1$. If we attempt to aggregate M_g queues in one step, the number of states will be very large even for a moderately sized network. It has been shown by Zahorjan [26] that aggregating two queues at a time provides more accurate results for first–in–first–out queues with non–exponential service times. Towsley used the approach of aggregating two queues at a time for a multiple bus system with constant memory service time [21]–[22] and obtained results which are very close to those of simulations. We also take the approach of aggregating two queue at a time and aggregate the M_g queues in a subgroup in $M_g - 1$ steps.

As mentioned above, at any step of aggregation we consider that the first queue is the queue aggregate produced in the previous step and the second queue is the new queue being aggregated. So at any step we have to solve a parallel closed queue network as shown in Figure 4.2. We define the state of a two queue network by the number of customers in queue 1. If there are k customers, then queue 1 can have a minimum of 0 customers and a maximum of k customers resulting in $k+1$ states. Therefore at any step of aggregation we have to deal with at most $P+1$ states. Had we chosen to aggregate all the M_g queues in a in a single step, then we would

have had to consider all possible combinations of distributing P customers in M_g queues. This is equivalent to find all M_g partitioning of P distinct objects and the number of such partitions is $\binom{P+M_g-1}{P}$. Such a high number of states is certain to pose numerical difficulties even for a moderate sized network.

For solving a closed queue network of two queue system we introduce the following notations.

Notation

r_m = probability that a processor makes a request to one particular regular memory module.

r_h = probability that a processor makes a request to the hot memory module.

ρ_i = probability that upon completion, a customer will enter queue i, where $i \in \{1,2\}$.

$s_i(j:n)$ = probability that exactly j customers complete service at a given memory cycle when there are n customers in queue i, where $i \in \{1,2\}$.

$s_{aggrt}(j:k)$ = probability that exactly j customers complete service from the aggregated queue when there are k customers in the two queue system being aggregated.

P_n = steady state probability that there are n customers in queue 1.

p_{ij} = transition probability from state i to state j.

The probability that a customer after finishing in either queue joins queue 1(2) is the conditional probability that a customer joins queue 1(2) in the original network given that it enters the subnetwork represented by queue 1 and queue 2. Therefore these probabilities are given by

$$\rho_1 = \frac{i}{i+1}, \quad \text{if neither queue associates the hotspot queue}$$

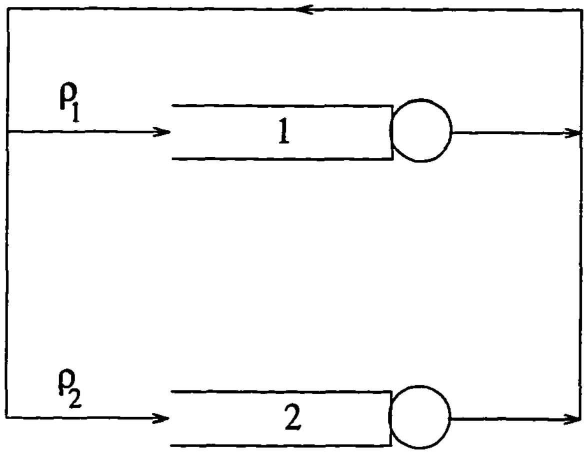


Figure 4.2: A parallel two queue closed network

$$= \frac{ir_m}{r_h + ir_m}, \quad \text{if queue 2 associates the hotspot queue}$$

$$= \frac{(i-1)r_m}{r_h + ir_m}, \quad \text{otherwise}$$

$$\rho_2 = \frac{1}{i+1}, \quad \text{if neither queue associates the hotspot queue}$$

$$= \frac{r_h}{r_h + ir_m}, \quad \text{if queue 2 associates the hotspot queue}$$

$$= \frac{r_m}{r_h + ir_m}, \quad \text{otherwise}$$

In order to solve the two queue system we have to evaluate the transition probability ρ_{ij} from state i to state j, where $i, j \in \{0, 1, \cdots, k\}$. This is evaluated as follows. The number of customers in the first queue is i and in the second queue is

$k-i$. The maximum number of customers these queues can serve is therefore i and $k-i$, respectively. In order to go from state i to state j, the number of customers in the first queue might have to be changed and this change depends on whether $i > j$ or $i \leq j$. In the first case, the first queue has to serve at least $i-j$ customers. In the second case, the second queue has to serve at least $j-i$ customers. Let the number of customers served by the first queue be m and let the number served by the second queue be l. Out of these $l+m$ customers who finished service, $j-(i-m)$ have to join the first queue so that the number of customers in the first queue becomes j. Therefore the transition probability is given by

$$\begin{aligned} p_{ij} &= \sum_{m=\max(0,i-j)}^{i} \sum_{l=\max(0,j-i)}^{k-i} s_1(m:i)s_2(l:k-i) \\ &\quad \binom{l+m}{j-i+m} \rho_1^{j-i+m} \rho_2^{l-j+i} \text{ , where } \quad 0 \leq i \leq k \end{aligned}$$

From the expression above it is evident that to determine the transition probability we have to have the service parameters of both queues. Now, the first queue represents the aggregated queue in step $i-1$, if i is the current step of aggregation. Let $s^i_{aggrt}(j:k)$ be the service parameter obtained by aggregation at step i. The service parameter of the first queue for any step can be found by the following recursive formula.

At step 1

$$\begin{aligned} s_1(j:n) &= 1, \text{ if } j=1 \text{ and } \forall n, 1 \leq n \leq k \\ &= 0, \text{otherwise} \end{aligned}$$

At step i,

$$s_1(j:n) = s^{i-1}_{aggrt}(j:n)$$

Since the second queue represents a single memory queue, its service parameter at any step is given by

$$\begin{aligned} s_2(j:n) &= 1, \text{ if } j=1 \text{ and } \forall n, 1 \le n \le k \\ &= 0, \text{otherwise} \end{aligned}$$

Since at any step the service parameter of the aggregated queue from the previous step is known, we have the service parameters of both queues under aggregation. We need to evaluate the steady state distribution for the state space of the two queue system. For that purpose we have to solve the following set of linear equations

$$\begin{aligned} P_i &= \sum_{j=0}^{k} P_j p_{ji}, \forall i, 0 \le i \le k \\ \sum_{i=0}^{k} P_i &= 1 \end{aligned}$$

Once we have evaluated the steady state distribution for the state space we can determine the aggregated service parameter which is nothing but the probability that certain number of customers will be served by the two queue system in a cycle. Therefor the combined service parameter is given by

$$s_{aggrt}(j:n) = \sum_{i=0}^{n} \sum_{m=\max(0,j+i-n)}^{\min(j,i)} P_i\, s_1(m:i) s_2(j-m:n-i)$$

4.2.2 Aggregating subgroup queues

The aggregation of subgroup queues into a group queue involves taking bus contention within a group into account. While for local buses in a subgroup there is no contention with other subgroups in the group, there is contention for the common buses in the group. Because of this contention aggregating two queue at a time will not be applicable going from subgroup level to the group level. All the subgroup queues have to be aggregated together taking into account the effect of bus contention. However that involves solving a g–queue system with P customers. The total number of states in such a network will be $\binom{g+P-1}{P}$. Solving a queuing network with such a high number of states will not be numerically feasible. We will use a technique which utilize two queue aggregation method to determine the probability distribution of the number nonempty queues in different subgroups for a given number of customers. With this distribution we can directly aggregate all the subgroup queues into a group queue in one step.

Contention in the original PLMB network (before aggregating memory queues within a subgroup) depends on the number of non-empty memory queues in each subgroup. In the simplified queue model obtained by aggregation at the subgroup level, the same contention is reflected by the number of customers each subgroup queue intends to serve. The total number of customers served by all the subgroups of a group depends on the distribution of the non-empty queues in all the subgroups in that group. Determining this distribution in the presence of bus contention might not be computationally feasible even for a moderate size network. We make the simplifying assumption that this distribution for a given number of customers in all the queues is independent of bus contention. Intuitively this assumption seems logical because the assumption is not about the total number of customers present

in all queues. The effect of this assumption along with other assumptions will later be checked by means of simulations. For aggregating subgroup queues into a single group queue we introduce the following notation. We use the term "queue" here to imply a subgroup queue.

NOTATION:

- $P\{n_1, n_2, \cdots, n_g : k\}$ = probability that there are n_m customers in the m^{th} queue given that there are k customers in all the queues, where $0 \leq n_m \leq k$ and $k = n_1 + n_2 + \cdots + n_g$, considering no bus contention.
- $S\{j_1, j_2, \cdots, j_g : k\}$ = probability that j_m customers are served in the m^{th} queue given that there are k customers in all the queues, considering no bus contention.
- $S_i(j : k)$ = probability that j customers will be served by the i^{th} queue if there are k customers in that queue and there is no bus contention.
- $S_c(j : k)$ = probability that j customers will be served by all the queues given that there are k customers in all the queues, considering bus contention

We define the following recursive relation

$$\begin{aligned} \psi_0 &= 0 \\ \psi_1 &= k \\ \psi_m &= \psi_{m-1} - n_{m-1} \end{aligned}$$

In the recursive relation above ψ_i signifies the maximum number of customers that can be in queue i, given that there are $n_1, n_2, \cdots, n_{i-1}$ customers present in

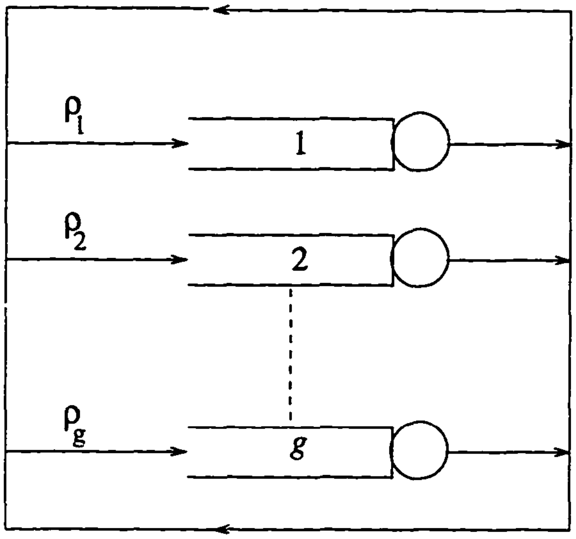


Figure 4.3: A parallel g queue closed network

queues $1, 2, \cdots, i-1$, respectively, and given that there is a total of k customers in all the queues.

Now, the service parameter $S\{j_1, j_2, \cdots, j_g : k\}$ depends on the distribution of the number of customers in different queues, i.e. $(P\{n_1, n_2, \cdots, n_g : k\})$ and on the service parameter of each of the subgroup queues, in the absence of bus contention. In other words,

$$S\{j_1, j_2, \cdots, j_g : k\} = \sum_{n_1=j_1}^{\psi_1} \sum_{n_2=j_2}^{\psi_2} \cdots \sum_{n_g}^{\psi_g} P\{n_1, n_2, \cdots, n_g : k\} \\ S_1(j_1 : n_1) \cdots S_g(j_g : n_g)$$

Determining the service parameter $S\{j_1, j_2, \cdots, j_g : k\}$ depends on finding the probability distribution $P\{n_1, n_2, \cdots, n_g : k\}$. This is essentially equivalent to solving a parallel g closed queue model as illustrated in Figure 4.3.

We determine $P\{n_1, n_2, \cdots, n_g : k\}$ in $g-1$ steps by successive aggregation as outlined below. In step 1, we determine the probability distribution for the number of customers in queue 1 given the total number of customers in queue 1 and queue 2. We then aggregate queue 1 and queue 2. At step 2, we determine the probability distribution for the number of customers in the aggregated queue from step 1 given the total number of customers in this queue and queue 3. We use this probability as the probability of the number of customers in queues 1 and 2 given the total number of customers in all the queues (i.e. queue 1, queue 2 and queue 3). Aggregation in step 1 allows us to determine this probability by solving only a two queue system. We then aggregate the previously aggregated queue with queue 3. In general at step i, where $i < g-1$, we get the probability distribution for the number of customers in queues $1, 2, \cdots, i$ given the total number of customers in queues $1, 2, \cdots, i, i+1$, and aggregate queue $i+1$ with the aggregated queue from step $i-1$. At step $g-1$, we simply determine the probability distribution for the number of customers in queues $1, 2, \cdots, g-1$ given the total number of customers in all the queues. At this step we don't do any aggregation. We will use the probability distributions thus determined at each step to determine $P\{n_1, n_2, \cdots, n_g : k\}$. We first address the general aggregation procedure in any step i, $1 \leq i < g-1$.

Aggregating two queues at any of the above mentioned steps is similar to aggregating two queues at the subgroup level as we discussed earlier. Let ρ_1 be the probability that a customer joins queue 1 after finishing and let ρ_2 be the probability that it joins queue 2. Also, we use the notation p_{ij} to denote the transition probability from state i to state j, with the meaning of the state being the number of customers in queue 1. At any step, queue 1 is the aggregated queue of some subgroup queues and queue 2 is a single subgroup queue. Accordingly, the probabilities

ρ_1 and ρ_2 will depend on the specific step of aggregation. At step i of aggregation these probabilities are given by

$$
\begin{aligned}
\rho_1 &= \frac{i}{i+1}, \quad \text{if neither queue associates the hotspot queue} \\
&= \frac{iM_g r_m}{r_h + (M_g - 1)r_m + iM_g r_m}, \quad \text{if queue 2 associates the hotspot queue} \\
&= \frac{r_h + (M_g - 1)r_m + (i-2)M_g r_m}{r_h + (M_g - 1)r_m + iM_g r_m}, \quad \text{otherwise} \\
\rho_2 &= \frac{1}{i+1}, \quad \text{if neither queue associates the hotspot queue} \\
&= \frac{(M_g - 1)r_m + r_h}{r_h + (M_g - 1)r_m + ir_m}, \quad \text{if queue 2 associates the hotspot queue} \\
&= \frac{M_g r_m}{r_h + (M_g - 1)r_m + ir_m}, \quad \text{otherwise}
\end{aligned}
$$

The transition probabilities at any step of aggregation can be evaluated in the same way as before. Finally we solve the following set of linear equations to evaluate the distribution of the state space in steady state.

$$
\begin{aligned}
P_i &= \sum_{j=0}^{k} P_j p_{ji}, \forall i, 0 \leq i \leq k \\
\sum_{i=0}^{k} P_i &= 1
\end{aligned}
$$

Once we determine the steady state probability distribution for the state space at any step, we can aggregate the two queues in that step by determining the combined

service parameter in exactly the same manner we followed earlier at the subgroup level of aggregation.

As explained above, in the i^{th} step of successive aggregation, we obtain the probability distribution that there is a certain number of customers in queues $1, 2, \cdots, i$ for a given number of customers in queues $1, 2, \cdots, i, i+1$. We use this probability to evaluate the final probability $P\{n_1, n_2, \cdots, n_g : k\}$. Let us denote the the number of customers in queue i by γ_i. Then we can write,

$$\begin{aligned} P\{n_1, n_2, \cdots, n_g : k\} &= P(\gamma_1 = n_1 : \gamma_2 = n_1 + n_2) \\ &\quad P(\gamma_2 = n_1 + n_2 : \gamma_3 = n_1 + n_2 + n_3) \\ &\quad \vdots \\ &\quad P(\gamma_{g-1} = n_1 + \cdots + n_{g-1} : \gamma_g = k) \end{aligned}$$

The probability distribution $P(\gamma_i = n_1 + \cdots + n_{i-1} : \gamma_i = k)$ is determined at step i. So, once we are done with $g - 1$ steps we have all the information required to evaluate $P\{n_1, n_2, \cdots, n_g : k\}$. As we determine this probability we can determine the service parameter $S\{j_1, j_2, \cdots, j_g : k\}$. Now, $S\{j_1, j_2, \cdots, j_g : k\}$ is the probability that j_m customers will be served in queue m given that there are k customers in all the queues, in the absence of bus contention. In the absence of bus contention if queue m serves j_m customers, then there are j_m non-empty queues in the m^{th} subgroup. Therefore, the service parameter $S\{j_1, j_2, \cdots, j_g : k\}$ is the probability distribution for the number of non–empty queues in different subgroups. Since we assumed that this distribution is independent of bus contention, we can use this probability distribution to determine the combined service parameter for g

subgroup queues with bus contention. Therefore, we can write

$$
\begin{aligned}
S_c(j:k) &= \sum_{n_1=0}^{\psi_1}\sum_{n_2=0}^{\psi_2}\cdots\sum_{n_g=0}^{\psi_g} S\{n_1,n_2,\cdots,n_g:k\} \\
&\quad u(n_1,n_2,\cdots,n_g:j) \\
\text{where } u(n_1,n_2,\cdots,n_g:j) &= 1, \text{ if } bus(n_1,n_2,\cdots,n_g)=j \\
&= 0, \text{ otherwise}
\end{aligned}
$$

The evaluation of $S_c(j:k)$ leads to aggregation of all subgroup queues into a group queue with bus contention included. Now we have G queues, one for each group and we can further aggregate these queues into a single queue in $G-1$ steps. Since contention for passive resources at the group level has been already taken care of, the aggregation of G queues can be done in a straightforward manner. This final queue thus obtained represents the memory–bus subsystem of the PRMB network.

4.2.3 Solving two queue system of the processor queue and memory queue

After replacing the memory queues by a combined queue with bus contention included we have a much simpler model to deal with. The processors in the PRMB system can be represented by a single queue with a service parameter that reflects the behavior of active processors in the system. In other words, the distribution of the number of requests generated by active processors in a cycle will be the service parameter of the processor queue. The number of active processors is represented by the number of customers present in the processor queue. So the processors and the combined memory queue can be modeled by a series two queue system as shown in Figure 4.4. Note in the figure that a customer after receiving service in the memory queue can immediately rejoin the memory queue, whereas a customer after receiving

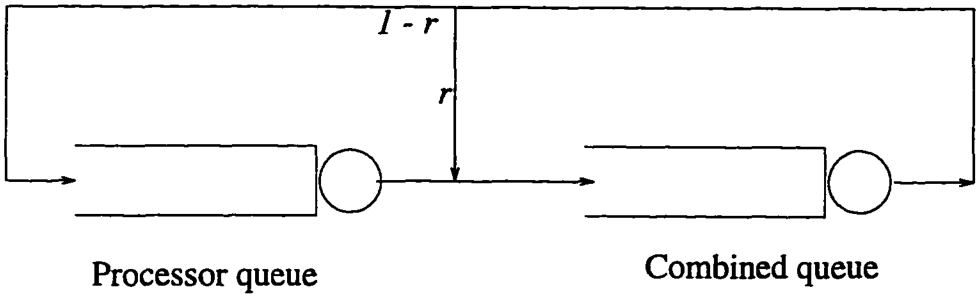


Figure 4.4: Two queue network for processor and memory queue.

service in the processor queue must join the memory queue. The first case is analogous to a processor generating a request immediately after its previous request has been served. The second case reflects the situation where an active processor generating a request to one of the memory modules. To solve the two queue network for the evaluation of throughput, (which is the average number of busy memory modules) we introduce some more notation.

NOTATION

P_n : probability that there are n customers in the processor queue.

p_{ij} : transition probability from state i to state j.

$S_p(j : k)$: service parameter for the processor queue denoting the probability that j customers will be served if there are k customers in the processor queue.

The service parameter of the processor queue is the probability that a certain number of customers are served for a given number of customers in the processor

queue. Therefore,

$$S_p(j:k) = \binom{k}{j} \times r^j(1-r)^{k-j}$$

The transition probability p_{ij} is evaluated as follows. Let the number of customers served by the processor queue be m and the number of customers served by the memory queue be l. At state i, the processor queue can serve up to i customers. However if $i > k-j$ it cannot serve more than $k-j$ customers since in that case even if all the customers served by the memory queue join the processor queue the processor queue will still have less than j customers. Therefore the maximum number of customers the processor queue can serve is $min(i, k-j)$. If $i \geq j$, the processor queue has to serve at least $i-j$ customers, otherwise state j cannot be reached. So, the minimum number of customers the processor queue has to serve is $max(0, i-j)$. For the memory queue, the maximum number of customers it can serve is $P-i$, the number of customers it has. If the processor queue serves m customers (those will join the memory queue with probability one), the number of customers left in the processor queue would be $i-m$. Therefore, to reach state j, the processor queue needs $j-(i-m)$ more customers and the minimum number of customers served by the memory queue will have to be $j-i+m$. Thus we have,

$$\begin{aligned} p_{i,j} &= \sum_{m=\max(0,i-j)}^{\min(i,k-j)} \sum_{l=j-i+m}^{P-j} S_p(m:i)S_c(l:P-i) \\ &\quad \binom{l}{j-i+m} r^{l+i-j-m}\,(1-r)^{j-i+m} \end{aligned}$$

To evaluate the throughput we have to determine the probability distribution for the state space of the processor-memory queue system which can be evaluated

by solving the following set of linear equations

$$\begin{aligned} P_i &= \sum_{j=0}^{P} P_j p_{ji} \\ \sum_{i=0}^{P} P_i &= 1 \end{aligned}$$

Finally Throughput could be derived as

$$\text{Throughput} = \sum_{i=1}^{P} \sum_{j=0}^{P-i} P_{P-i}\, j\, S_c(j : P - i)$$

4.3 Numerical Results

We evaluate the throughput of different PRMB queuing networks using the approximate method described above . We consider the same set of networks we considered earlier and consider both a uniform request model and a hotspot model. For both models we assume several possible processor request rates and for the hotspot model we choose $h = .1$. The results of our evaluations are listed in Tables 4.1–4.10.

In the approximate model developed for the queuing analysis we considered some simplifying assumptions. First, we applied the method of aggregation for our network which is not a product form network. Second, we assumed that the distribution of the number of non–empty memory queues in different subgroups within a group is independent of bus contention.

To verify the effect of these approximations, we simulated the same networks without any simplifying assumptions. This allows us to compare the simulation results with those obtained by the approximate analysis. The simulation program was written using the C programming language and it was driven by a non–linear additive feedback random number generator. The probabilistic request generation

of a processor was achieved by generating a random number uniformly distributed between 0 and 1. To generate such a random number Unix system function random() was employed. Depending on the outcome of the random number generator, a processor will either make a request or be simply idle in a memory cycle. If the outcome of the random number calls for a memory request, then another random number is generated to determine whether the request is a hotspot request. Without loss of generality any memory module could be chosen as hot module. In our case we considered memory module 1 as the hot module. If the outcome of the second random number did not indicate a hotspot request, then a third random number was generated, which was uniformly distributed between 1 and M (total number of memory modules). Depending on the outcome, a memory module was selected as the one to which the request was to be directed. We repeated the entire process for each processor and carried out the simulation for 20000 memory cycles. The program was run ten different times, each time with different seeds (different seeds were obtained by calling system function srandom() with different arguments). The results were evaluated by using a confidence interval of 99%. Simulation results are presented along with analytical results in Tables 4.1–4.10. It is evident that the throughput of the systems is almost insensitive to the simplifying assumptions adopted in our queuing analysis.

The first approximation of applying the aggregation method in a non-product form network (PRMB system) does not (as expected) make significant difference. Since in the method of aggregation the combined service rate of queues under aggregation is determined by keeping routing among these queues the same as in the original network, the approximation should not affect the overall system through-

put. Comparison of simulation results with those of analytical ones validates this approximation.

The second approximation is concerned with the distribution of non empty memory queues in different subgroups of a group. The total number of customers present in all the memory queues should depend on bus contention. If bus contention is low, the probability will be low that a high number of customers are present in the memory queues and vice versa. However if it is given that a certain number of customers is present in all the memory queues, then how they are distributed among different queues should not depend on bus contention (because resources are uniform across all the subgroups). The distribution of the number of customers in different queues (given the total number of customers in all the queues) will determine the distribution of the number of nonempty memory queues in all the subgroups. In that case the distribution of nonempty memory queues in different subgroups given the total number of customers in these queues should not depend on bus contention.

We also simulated the base model system corresponding to each of the PRMB networks considered. This is especially important because it needs to be checked whether reduced connectivity affects the performance of a PRMB system in the presence of memory queues. This can only be achieved by comparing the throughput of each PRMB network with that of the corresponding base model network. The simulation results for these base model systems are also presented in Tables 4.1–4.10. Comparison of system throughputs for each PRMB system considered and the corresponding base model system reveals that reduced connectivity does not have any adverse affect on the performance of PRMB systems.

Table 4.1: Throughput of a PRMB system and that of the corresponding base model system under uniform request model for a network of size $64 \times 64 \times 32$ with 1 group and 2 subgroups in each group (for PRMB system).

Request rate r	PRMB system		base model system
	appx. analysis	simulation	simulation
.1	6.365	6.364±.0113	6.364±.0113
.2	12.502	12.502±.0136	12.502±.0136
.3	18.145	18.141±.0166	18.141±.0166
.4	23.064	23.060±.0194	23.060±.0198
.5	27.098	27.096 ±.0194	27.098 ±.0172
.6	29.969	29.922 ±.017	29.927 ±.0168
.7	31.386	31.290 ±.007	31.302 ±.008
.8	31.832	31.767 ±.008	31.774 ±.006
.9	31.949	31.918 ±.002	31.923 ±.004
1.0	31.982	31.968 ±.002	31.972 ±.001

Table 4.2: Throughput of a PRMB system and that of corresponding base model system under hotspot request model for a network of size $64 \times 64 \times 32$, with 1 group and 2 subgroups in each group (for PRMB system) with $h = .1$.

Request rate r	PRMB system		base model system
	appx. analysis	simulation	simulation
.1	6.239	6.239±.0284	6.293±.0284
.2	8.763	8.7±.0718	8.791±.0718
.3	8.767	8.780±.0780	8.780±.0782
.4	8.767	8.790±.0645	8.790±.0647
.5	8.767	8.782 ±.0689	8.782 ±.0690
.6	8.767	8.760 ±.0639	8.760 ±.0636
.7	8.767	8.783±.0667	8.783 ±.0671
.8	8.767	8.790 ±.0882	8.790 ±.0880
.9	8.767	8.790 ±.0881	8.790 ±.0881
1.0	8.767	8.886±.0699	8.886±.0699

Table 4.3: Throughput of a PRMB system and that of corresponding base model system under uniform request model for a network of size 70 × 70 × 35 with 1 group and 2 subgroups in each group (for PRMB system).

Request rate r	PRMB system		base model system
	appx. analysis	simulation	simulation
.1	6.962	6.961±.0125	6.961±.0125
.2	13.674	13.672±.0161	13.672±.0161
.3	19.846	19.839±.0186	19.839±.0186
.4	25.224	25.219±.0225	25.219±.0223
.5	29.640	29637±.0132	29.638±.0137
.6	32.811	32.763 ±.0119	32.769 ±.0116
.7	34.371	34.273±.007	34.277±.009
.8	34.842	34.778±.007	34.782 ±.004
.9	34.956	34.928 ±.002	34.931 ±.002
1.0	34.986	34.973 ±.001	34.975 ±.002

Table 4.4: Throughput of a PRMB system and corresponding base model system under hotspot request model for a network of size 70 × 70 × 35 with 1 group and 2 subgroups in each group (for PRMB system) with $h = .1$.

Request rate r	PRMB system		base model system
	appx. analysis	simulation	simulation
.1	6.785	6.856±.0127	6.856±.0127
.2	8.861	8.876±.0474	8.876±.0474
.3	8.861	8.860±.0348	8.860±.0348
.4	8.861	8.878±.0243	8.878±.0243
.5	8.861	8.855±.0471	8.855±.0471
.6	8.861	8.847±.0325	8.847 ±.0325
.7	8.861	8.860±.0422	8.860 ±.0423
.8	8.861	8.870±.0557	8.871 ±.0557
.9	8.861	8.875±.0438	8.875 ±.0437
1.0	8.861	8.954±.0257	8.954±.0257

Table 4.5: Throughput of a PRMB system and corresponding base model system under uniform request model for a network of size $90 \times 90 \times 30$ with 3 groups and 2 subgroups in each group (for PRMB system).

Request rate r	PRMB system		base model system
	appx. analysis	simulation	simulation
.1	8.951	8.947±.0163	8.949±.0210
.2	17.556	17.552±.0215	17.546±.0288
.3	24.844	24.797±.0223	24.824±.0258
.4	28.279	28.190±.0246	28.219±.0311
.5	29.027	28.977±.0203	28.994±.0325
.6	29.258	29.239±.0289	29.246±.0314
.7	29.368	29.379±.0348	29.386 ±.0373
.8	29.431	29.454±.0308	29.464 ±.0368
.9	29.472	29.492±.0213	29.504 ±.0202
1.0	29.501	29.494±.0221	29.502 ±.0176

Table 4.6: Throughput of a PRMB system and that of corresponding base model system under hotspot request model for a network of size $90 \times 90 \times 30$ with 3 groups and 2 subgroups in each group (for PRMB system).

Request rate r	PRMB system		base model system
	appx. analysis	simulation	simulation
.1	8.381	8.527 ±.0272	8.531 ±.0304
.2	9.090	9.095 ±.0417	9.118 ±.0582
.3	9.090	9.101 ±.0399	9.105 ±.0557
.4	9.090	9.111 ±.0451	9.104 ±.0536
.5	9.090	9.098 ±.0662	9.094 ±.0654
.6	9.090	9.086 ±.0323	9.079 ±.0265
.7	9.090	9.088 ±.0489	9.100 ±.0555
.8	9.090	9.110 ±.0645	9.115 ±.0852
.9	9.090	9.098 ±.0447	9.108 ±.0551
1.0	9.090	9.202 ±.0382	9.210 ±.0618

Table 4.7: Throughput of a PRMB system and that of corresponding base model system under uniform request model for a network of size $96 \times 96 \times 48$ with 2 groups and 3 subgroups in each group (for PRMB system).

Request rate r	PRMB system appx. analysis	PRMB system simulation	base model system simulation
.1	9.548	9.545±.0171	9.545 ±.0171
.2	18.751	18.749±.0200	18.749 ±.0200
.3	27.210	27.199±.0237	27.199 ±.0238
.4	34.557	34.547±.0190	34.549 ±.0199
.5	40.415	40.383±.0188	40.393 ±.0184
.6	44.234	44.132±.0195	44.159 ±.0206
.7	46.042	45.933±.0150	45.962 ±.0131
.8	46.755	46.686±.0269	46.713 ±.0294
.9	47.064	47.013±.0288	47.038 ±.0270
1.0	47.224	47.163±.0178	47.184 ±.0143

Table 4.8: Throughput of a PRMB system and that of corresponding base model system under hotspot request model for a network of size $96 \times 96 \times 48$ with 2 groups and 3 subgroups in each group (for PRMB system).

Request rate r	PRMB system appx. analysis	PRMB system simulation	base model system simulation
.1	8.737	8.876 ±.0394	8.876 ±.0394
.2	9.143	9.151 ±.0433	9.151 ±0433
.3	9.143	9.157 ±.0448	9.157 ±.0448
.4	9.143	9.167 ±.0457	9.167 ±.0457
.5	9.143	9.151 ±.0670	9.151 ±.0670
.6	9.143	9.143 ±.0323	9.143 ±.0322
.7	9.143	9.145 ±.0449	9.145 ±.0450
.8	9.143	9.164 ±.0652	9.164 ±.0652
.9	9.143	9.157 ±.0467	9.157 ±.0468
1.0	9.143	9.263 ±.0341	9.263 ±.0340

Table 4.9: Throughput of a PRMB system and corresponding base model system under a uniform request model for a network of size 120 × 120 × 60 with 3 groups and 2 subgroups in each group (for a PRMB system).

Request rate r	PRMB system		base model system
	appx. analysis	simulation	simulation
.1	11.935	11.933±.0178	11.933 ±.0178
.2	23.437	23.435±.0278	23.435 ±.0278
.3	34.006	33.995±.0257	33.996 ±.0259
.4	43.156	43.140±.0234	3.185 ±.0224
.5	50.305	50.249±.0252	50.588 ±.0202
.6	54.791	54.647±.0259	55.463 ±.0274
.7	56.925	56.793±.0307	57.757 ±.0259
.8	57.850	57.798±.0314	58.653 ±.0333
.9	58.300	58.259±.0306	58.998 ±.0342
1.0	58.554	58.465±.0225	59.165 ±.0250

Table 4.10: Throughput of a PRMB system and that of corresponding base model system under hotspot request model for a network of size 120 × 120 × 60 with 3 groups and 2 subgroups in each group (for PRMB system)

Request rate r	PRMB system		base model system
	appx. analysis	simulation	simulation
.1	9.298	9.314 ±.0559	9.314 ±.0559
.2	9.302	9.326 ±.0433	9.326 ±.0433
.3	9.302	9.327 ±.0459	9.327 ±.0459
.4	9.302	9.340 ±.0473	9.340 ±.0473
.5	9.302	9.323 ±.0634	9.323 ±.0634
.6	9.302	9.310 ±.0328	9.310 ±.0328
.7	9.302	9.317 ±.0439	9.318 ±.0440
.8	9.302	9.334 ±.0658	9.334 ±.0659
.9	9.302	9.331 ±.0555	9.331 ±.0557
1.0	9.302	9.435 ±.0350	9.435 ±.0352

CHAPTER 5

REDUCING CONNECTIVITY FROM BOTH SIDES

The PRMB systems we have considered thus far take the approach of reducing connectivity form the memory side while keeping the processor side fully connected (the other way around is also possible). In an attempt to explore further possible cost reduction, here we attempt to reduce the connectivity form both sides. For this case we propose an architecture in which the processors and memory modules in a group are divided into equal sized subgroups. In each subgroup processors are connected to the memory modules by some local buses. However, unlike the systems considered in this dissertation thus far there will be some common buses to which all the processors and memory modules are connected. Thus, unlike in an original PRMB system, processors are not connected to all buses. They are connected to the local buses within their own subgroups and to some common buses (which are connected to all processors and all memory modules). The number of common buses will be determined such that, with high probability, bus assignment in a group will be the same as in the corresponding base model system. The proposed new architecture is shown in Figure 5.1.

Unlike our original PRMB system, bus assignment in a group does not depend only on request distribution in the subgroups within that group. It will depend on the request distribution in all the subgroups of all groups. This is so because requests in all the subgroups which cannot be satisfied by the local buses have to be satisfied by the common buses, to the extent possible. Since the objective is to eliminate redundant connectivity in the base model system, we need the number

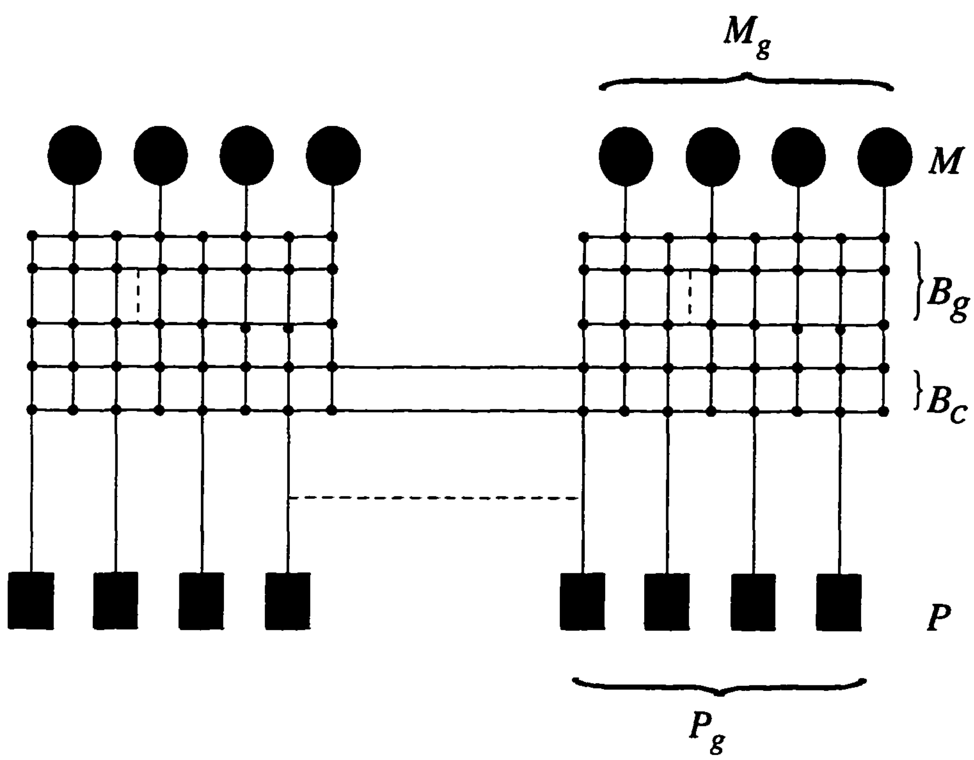


Figure 5.1: Proposed architecture with connectivity reduced from both sides

of common buses to be the minimum possible such that bus assignment in a group is, with high probability, the same as that in the base model system. Another important difference with PRMB systems considered earlier is that, here we need to know the source of a request a subgroup receives. More precisely, we need to know if the request is from a processor in the same subgroup as the memory module or it is from some other subgroup. The source of a request was irrelevant in the original PRMB system because processors were connected to all the buses. In the new scheme requests to memory modules in a particular subgroup from processors belonging to other subgroups can only be met by the common buses. Also, simply knowing the number of memory modules receiving requests from the processors within the same subgroup and the number of memory modules receiving requests

from processors in other subgroups is not sufficient. Arbitration plays an important role in this new architecture. Suppose a memory module receives some requests from processors in the same subgroup and some other requests from processors in other subgroups. If a request from a processor in some other group wins memory arbitration, a common bus is definitely needed, otherwise if a local bus is available the request can be satisfied. Therefore, while considering a request pattern, for each subgroup, we have to keep information about the number of memory modules for which processors from the same subgroup win memory arbitration and the number of memory modules for which requests from processors in some other subgroups win memory arbitration.

One final point about the new architecture is that it cannot be justified only by its cost being simply less than that of the base model system; the cost has to be less than that of the original PRMB system. For convenience of reference, we will refer to the original PRMB system as *OPLMB* (original PLMB) and the architecture with connectivity reduced from both sides as *MPLMB* (modified PRMB system).

5.1 Bandwidth Analysis

In this section we will evaluate the memory bandwidth for a MPRMB system. We will use the general request model we considered earlier. We will use the same notations to denote the numbers of processors, memory modules, buses, groups and subgroups . We denote the total number of subgroups in MPRMB system by τ. Obviously, $\tau = G \times g$. With the change in architecture the number of local buses in each subgroup is given by, $B_g = \frac{B-B_c}{\tau}$. To take care of the modified architectural features and the extra information which needs to be included with respect to a

request pattern we introduce some definitions and some more notation.

5.1.1 Some Preliminaries

Definition 5.1 For a memory module, a processor in the same subgroup is a *near processor* and a processor in the same class is a *local processor.* A processor in a different subgroup is a *far processor* and a processor in a different class is a *non local processor.*

Definition 5.2 In a given cycle, a request from a processor is said to *pass the arbitration* for a memory module if it wins the memory arbitration for the memory module over all the processors requesting access to the same module in that cycle. Note that passing memory arbitration does not guarantee access since another level of arbitration (for a bus) must be passed for access to occur.

Notation

Q_{rl} : probability that, for a regular memory module, a request from a near processor passes memory arbitration.

Q_{rn} : probability that, for a regular memory module, a request from a far processor passes memory arbitration

Q_{hl} : probability that, for a hot memory module, a request from a near processor passes memory arbitration

Q_{hn} : probability that, for a hot memory module, a request from a far processor passes memory arbitration.

$((\hat{i}_1, \check{i}_1), (\hat{i}_2, \check{i}_2), \cdots, (\hat{i}_\tau, \check{i}_\tau))$: a request pattern where $\hat{i}_k$ is the number of memory modules in the k^{th} subgroup for which requests from near processors passed memory arbitration and where $\check{i}_k$ is the number of memory modules in the k^{th} subgroup for

which requests from far processors passed memory arbitration. Obviously, $\hat{i}_k + \check{i}_k \leq M_g$.

$Pr((\hat{i}_1, \check{i}_1), (\hat{i}_2, \check{i}_2), \cdots, (\hat{i}_T, \check{i}_T))$: probability of occurrence of the request pattern $((\hat{i}_1, \check{i}_1), (\hat{i}_2, \check{i}_2), \cdots, (\hat{i}_T, \check{i}_T))$.

$bus((\hat{i}_1, \check{i}_1), (\hat{i}_2, \check{i}_2), \cdots, (\hat{i}_T, \check{i}_T))$: bus assignment in an MPRMB system for a given request pattern$((\hat{i}_1, \check{i}_1), (\hat{i}_2, \check{i}_2), \cdots, (\hat{i}_T, \check{i}_T))$

5.1.2 Memory Bandwidth

Memory bandwidth as defined earlier, is the expected number of busy memory modules per bus cycle. To determine this expected value we need to determine the probability of a request pattern and its corresponding bus assignment.

For a given request pattern in the MPRMB system we must consider the number of memory modules in each subgroup for which requests from near processors pass memory arbitration and the number of memory modules for which requests from far processors pass memory arbitration. This is necessary because a request from a near processor can be served by a local bus, whereas a request from a far processor can only be served by a common bus. The number of regular (hot) memory modules for which near processors pass memory arbitration depends on Q_{rl} (Q_{hl}). The number of regular (hot) memory modules for which far processors pass memory arbitration depends on Q_{rn} (Q_{hn}). Under fair arbitration policy (which we assume here) the probabilities Q_{rl}, Q_{rn}, Q_{hl} and Q_{hn} depend on the number of requests a memory module receives from near processors and the number of requests it receives from far processors. These numbers depend on the request model under consideration. For example the probability that a memory module will get a request from a near processor will depend on whether the processor belongs to the local class or to a

non local class. The probability that a processor will make a request to a memory module in the local class is different from the probability that it will make a request to a memory module in a non local class. Requests from near processors could come from the same class or from different classes. That depends on the total number of subgroups in the MPRMB system and the total number of classes in the request model. There are two possible scenarios: (i) $C \leq \tau$, i.e the number of classes is less than or equal to the total number of subgroups and (ii) $C > \tau$, i.e. the number of classes is greater than the number of subgroups. In the first case each class will encompass one or more subgroups with the possibility of a subgroup being split between two classes; and in the latter case each subgroup will have more than one class with the possibility of a class being split between two subgroups. We will consider the two cases separately as follows.

CASE I: $C \leq \tau$

This case is illustrated in Figure 5.2. In this case each class will encompass at least one subgroup. Note that according to our definition of a class (Section 2.3.1), all the classes are of equal size. Also subgroups are assumed to be of equal size. It is possible that in the present case a subgroup is split into two classes. Thus requests

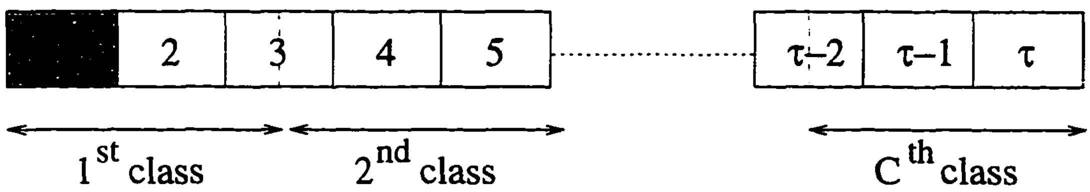


Figure 5.2: Illustration of the situation where the number of classes is less than or equal to total number of subgroups.

from local processors may come from different subgroups. Let i denote the number

of requests a regular memory module receives from processors which are both local and near, let j denote the number of requests it receives from processors which are far but local, and let k denote the number of requests it receives from processors which are both far and non local. Since i and j both correspond to requests from the same class, they will be distributed identically while k will be distributed in a different way. As far as arbitration is concerned the conditional probability (for given i, j, k) that a request from a near processor passes memory arbitration is $\frac{i}{i+j+k}$ and the conditional probability that a request from a far processor passes memory arbitration is $\frac{j+k}{i+j+k}$.

Now, there are P_l processors in a class. So the probability that out of these processors some particular $i+j$ processors will make a request to a memory module in the same class is $(\frac{rl(1-h)}{M_l})^{i+j}$. Since i processors can be chosen from P_g processors in the same subgroup in $\binom{P_g}{i}$ ways, and j processors can be chosen from $P_l - P_g$ processors in the same class but in different subgroups in $\binom{P_l-P_g}{j}$ ways, the probability that there will be requests from any $i+j$ processors in the same class, with any i requests from the same subgroup and any j requests from different subgroups is

$$\binom{P_g}{i}\binom{P_l - P_g}{j}(\frac{rl(1-h)}{M_l})^{i+j}(1 - \frac{rl(1-h)}{M_l})^{P_l-i-j}$$

There are $P - P_l$ processors in the non local class of a memory module. In the present case processors in non local classes of a memory module also belong to subgroups other than the one to which the memory module belongs. The probability that a regular memory module will get requests from k non local processors is

therefore,

$$\binom{P-P_l}{k}(\frac{r(1-l)}{M})^k$$

Considering all possible values for i, j and k, the probability that a request from a near processor passes memory arbitration is

$$\begin{aligned} Q_{rl} &= \sum_{i=0}^{P_g}\sum_{j=0}^{P_l-P_g}\sum_{k=0}^{P-P_l}\binom{P_g}{i}\binom{P_l-P_g}{j}\binom{P-P_l}{k}(\frac{rl(1-h)}{M_l})^{i+j}(\frac{r(1-l)}{M})^k \\ & (1-\frac{rl(1-h)}{M_l})^{P_l-i-j}(1-\frac{r(1-l)}{M})^{P-P_l-k} \times \frac{i}{i+j+k} \end{aligned}$$

Determination of the probability that a request from a far processor passes memory arbitration is similar. This is true because the probability that there will be i requests from processors which are both near and local, j requests from processors which are far but local and k requests from processors which are both far and non local remains the same. Therefore we can also write,

$$\begin{aligned} Q_{rn} &= \sum_{i=0}^{P_g}\sum_{j=0}^{P_l-P_g}\sum_{k=0}^{P-P_l}\binom{P_g}{i}\binom{P_l-P_g}{j}\binom{P-P_l}{k}(\frac{rl(1-h)}{M_l})^{i+j}(\frac{r(1-l)}{M})^k \\ & (1-\frac{rl(1-h)}{M_l})^{P_l-i-j}(1-\frac{r(1-l)}{M})^{P-P_l-k} \times \frac{j+k}{i+j+k} \end{aligned}$$

In the case of a hotspot module, the probability that it will get i requests from near and local processors and j requests from far but local processors will change to

$$\binom{P_g}{i}\binom{P_l-P_g}{j}(\frac{rl(1-h)}{M_l}+rlh)^{i+j}(1-\frac{rl(1-h)}{M_l}-rlh)^{P_l-i-j}$$

The probability that a hot module will receive k requests from far and non local processors remains the same as in the case of a regular memory module. Therefore

we can write,

$$Q_{hl} = \sum_{i=0}^{P_g} \sum_{j=0}^{P_l - P_g} \sum_{k=0}^{P-P_l} \binom{P_g}{i} \binom{P_l - P_g}{j} \binom{P - P_l}{k} \left(\frac{rl(1-h)}{M_l} + rlh\right)^{i+j} \left(\frac{r(1-l)}{M}\right)^k$$
$$\left(1 - \frac{rl(1-h)}{M_l} - rlh\right)^{P_l - i - j} \left(1 - \frac{r(1-l)}{M}\right)^{P-P_l-k} \times \frac{i}{i+j+k}$$

$$Q_{hn} = \sum_{i=0}^{P_g} \sum_{j=0}^{P_l - P_g} \sum_{k=0}^{P-P_l} \binom{P_g}{i} \binom{P_l - P_g}{j} \binom{P - P_l}{k} \left(\frac{rl(1-h)}{M_l} + rlh\right)^{i+j} \left(\frac{r(1-l)}{M}\right)^k$$
$$\left(1 - \frac{rl(1-h)}{M_l} - rlh\right)^{P_l - i - j} \left(1 - \frac{r(1-l)}{M}\right)^{P-P_l-k} \times \frac{j+k}{i+j+k}$$

CASE II: $C > \tau$

This case is illustrated in Figure 5.3. In this case each subgroup will encompass more than one class. It is possible that a subgroup will include a fraction of a class.

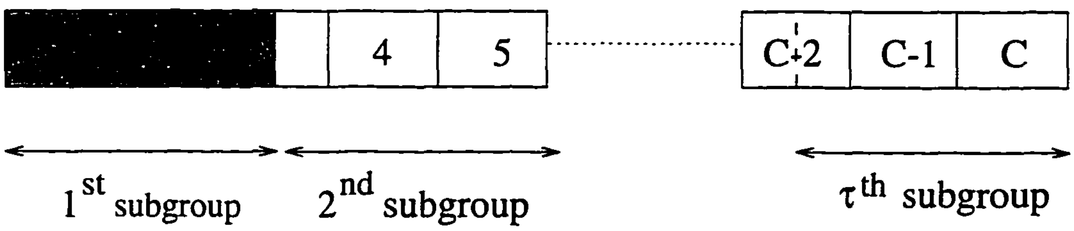


Figure 5.3: Illustration of the situation where the number of classes is more than the total number of subgroups.

Let i denote the number of requests a memory module receives from processors which are both near and local; let j denote the number of requests it receives from processors which are near but non local, and let k denote the number of requests from processors which are both far and non local. The conditional probability (for a given

i, j, k) that a request from a near processor will pass memory arbitration is $\frac{i+j}{i+j+k}$ and the conditional probability that a request from a far processor will pass memory arbitration is $\frac{k}{i+j+k}$. i requests might come from P_g processors, j requests come from $P_g - P_l$ processors and finally k requests could come from $P - P_g$ processors. In this case it can be easily shown that,

$$Q_{rl} = \sum_{i=0}^{P_l} \sum_{j=0}^{P_g-P_l} \sum_{k=0}^{P-P_g} \binom{P_l}{i} \binom{P_g - P_l}{j} \binom{P - P_g}{k} (\frac{rl(1-h)}{M_l})^i (\frac{r(1-l)}{M})^{j+k}$$
$$(1 - \frac{rl(1-h)}{M_l})^{P_l - i} (1 - \frac{r(1-l)}{M})^{P-P_l-j-k} \times \frac{i+j}{i+j+k}$$

$$Q_{rn} = \sum_{i=0}^{P_l} \sum_{j=0}^{P_g-P_l} \sum_{k=0}^{P-P_g} \binom{P_l}{i} \binom{P_g - P_l}{j} \binom{P - P_g}{k} (\frac{rl(1-h)}{M_l})^i (\frac{r(1-l)}{M})^{j+k}$$
$$(1 - \frac{rl(1-h)}{M_l})^{P_l - i} (1 - \frac{r(1-l)}{M})^{P-P_l-j-k} \times \frac{k}{i+j+k}$$

$$Q_{hl} = \sum_{i=0}^{P_l} \sum_{j=0}^{P_g-P_l} \sum_{k=0}^{P-P_g} \binom{P_l}{i} \binom{P_g - P_l}{j} \binom{P - P_g}{k} (\frac{rl(1-h)}{M_l} + rlh)^i (\frac{r(1-l)}{M})^{j+k}$$
$$(1 - \frac{rl(1-h)}{M_l} - rlh)^{P_l - i} (1 - \frac{r(1-l)}{M})^{P-P_l-j-k} \times \frac{i+j}{i+j+k}$$

$$Q_{hn} = \sum_{i=0}^{P_l} \sum_{j=0}^{P_g-P_l} \sum_{k=0}^{P-P_g} \binom{P_l}{i} \binom{P_g - P_l}{j} \binom{P - P_g}{k} (\frac{rl(1-h)}{M_l} + rlh)^i (\frac{r(1-l)}{M})^{j+k}$$
$$(1 - \frac{rl(1-h)}{M_l} - rlh)^{P_l - i} (1 - \frac{r(1-l)}{M})^{P-P_l-j-k} \times \frac{k}{i+j+k}$$

Once the probabilities Q_{rl}, Q_{rn}, Q_{hl} and Q_{hn} are determined the probability of the request pattern $((\hat{i}_1, \check{i}_1), (\hat{i}_2, \check{i}_2), \cdots, (\hat{i}_\tau, \check{i}_\tau))$ can be determined. The quantity $\hat{i}_k$ is the number of memory modules in the k^{th} subgroups for which requests from near processors pass memory arbitration. Suppose that out of these $\hat{i}_k$ memory modules, $\hat{j}_k$ are hotspot modules. Obviously $\hat{j}_k \leq H_k$. $\hat{j}_k$ hot modules can be chosen from H_k memory modules in $\binom{H_k}{\hat{j}_k}$ ways and the remaining $\hat{i}_k - \hat{j}_k$ memory modules can be chosen from the remaining $M_g - H_k$ modules in the k^{th} subgroup in $\binom{M_g - H_k}{\hat{i}_k}$ ways. This will leave $M_g - H_k - \hat{i}_k$ regular memory modules and $H_k - \hat{j}_k$ hot memory modules in the k^{th} subgroup. Suppose that out of $\check{i}_k$ memory modules (for which requests from far processors passed memory arbitration) the number of hotspot modules is $\check{j}_k$. $\check{j}_k$ hotspot memory modules can be chosen in $\binom{H_k - \hat{j}_k}{\check{j}_k}$ ways and the remaining $\check{i}_k - \check{j}_k$ regular memory modules can be chosen in $\binom{M_g - H_k - \hat{i}_k}{\check{i}_k - \check{j}_k}$ ways. Considering all possible values for $\hat{j}_k$ and $\check{j}_k$ we can write the probability of the request pattern $((\hat{i}_1, \check{i}_1), (\hat{i}_2, \check{i}_2), \cdots, (\hat{i}_\tau, \check{i}_\tau))$ as follows

$$Pr((\hat{i}_1, \check{i}_1), (\hat{i}_2, \check{i}_2), \cdots, (\hat{i}_\tau, \check{i}_\tau)) =$$

$$\sum_{\hat{j}_1=0}^{\min(H_1, \hat{i}_1)} \cdots \sum_{\hat{j}_\tau=0}^{\min(H_\tau, \hat{i}_\tau)} \sum_{\check{j}_1=0}^{\min(H_1 - \hat{j}_1, \check{i}_1)} \cdots \sum_{\check{j}_\tau=0}^{\min(H_\tau - \hat{j}_\tau, \check{i}_\tau)}$$

$$\prod_{k=1}^{\tau} \binom{M_g - H_k}{\hat{i}_k - \hat{j}_k} \binom{M_g - H_k - \hat{i}_k}{\check{i}_k - \check{j}_k} \binom{H_k}{\hat{j}_k} \binom{H_k - \hat{j}_k}{\check{j}_k}$$
$$Q_{rl}^{\sum \hat{i}_k - \hat{j}_k} Q_{rn}^{\sum \check{i}_k - \check{j}_k} Q_{hl}^{\sum \hat{j}_k} Q_{hn}^{\sum \check{j}} (1 - q_r)^{M - C - \sum \hat{i} + \check{i}} (1 - q_h)^{C - \sum \hat{j} + \check{j}}$$

Once the probability of a request pattern is determined, we need to determine the bus assignment for it. Consider a request pattern $((\hat{i}_1, \check{i}_1), (\hat{i}_2, \check{i}_2), \cdots, (\hat{i}_\tau, \check{i}_\tau))$.

The local buses in the k^{th} subgroup, where $k \in \{1, \cdots, \tau\}$, will provide $\min(\hat{i}_k, B_g)$ buses to this request pattern. The number of requests in the k^{th} subgroup which cannot be satisfied by local buses equals the number of requests from far processors passing memory arbitration plus the number of requests from near processors passing memory arbitration minus the number of local buses. Stated another way this number is $\check{i}_k + \max(\hat{i}_k - B_g, 0)$. Such requests, for all the subgroups, can be satisfied only to the extent allowed by the number of common buses in the MPRMB system. Therefore, the number of buses that will be available to request pattern $((\hat{i}_1, \check{i}_1), (\hat{i}_2, \check{i}_2), \cdots, (\hat{i}_\tau, \check{i}_\tau))$ is given by,

$$bus((\hat{i}_1, \check{i}_1), (\hat{i}_2, \check{i}_2), \cdots, (\hat{i}_\tau, \check{i}_\tau)) = \sum_{k=1}^{\tau} \min(\hat{i}_k, B_g) + \min(B_c, \sum_{k=1}^{\tau} \max(\hat{i}_k - B_g, 0) + \check{i}_k)$$

Notice that the second term corresponds to the number of common buses allocated to the request pattern and the first term corresponds to the number of local buses allocated to the request pattern. Since the request pattern and the bus assignment are based on consideration of all the subgroups, we can express the memory bandwidth of MPRMB system as follows,

$$\begin{aligned} \tilde{BW} \;=\; & \sum_{\hat{i}_1=0}^{M_g} \sum_{\check{i}_1=0}^{M_g} \cdots \sum_{\hat{i}_\tau=0}^{M_g} \sum_{\check{i}_\tau=0}^{M_g} Pr((\hat{i}_1, \check{i}_1), (\hat{i}_2, \check{i}_2), \cdots, (\hat{i}_\tau, \check{i}_\tau)) \times \\ & bus((\hat{i}_1, \check{i}_1), (\hat{i}_2, \check{i}_2), \cdots, (\hat{i}_\tau, \check{i}_\tau)) \end{aligned}$$

5.2 Number of common buses

The number of common buses in a MPRMB system is determined such that, with high probability, a request pattern in a group will have the same bus assignment as that in the corresponding base model system. As described earlier, in the MPRMB

system, common buses are connected to all the memory modules and also to all the processors. In some sense the common buses are simply contributions from the subgroups with each subgroup providing an equal number. As an architectural feature the group loses its identity. However the group is still a useful concept in the analysis because our objective remains the same as before.

In the MPRMB system, bus shortages in all the subgroups with shortages are met by utilizing the bus surpluses from all the subgroups with surpluses. While taking bus surplus from different subgroups, we take them only to the extent that, in a group, a request pattern does not get more buses than it would get in the base model system. It is possible in a base model system that there would be bus shortages in some groups and bus surpluses in some other groups. In the MPRMB system, since all the subgroups contribute to the pool of common buses, it might be possible to improve bus availability in comparison to that in the corresponding base model system. However since the objective is to eliminate probabilistic redundancy, we will not attempt to improve the bus assignment beyond that of a base model system. So while taking surplus buses from all the subgroups, we take them only to the extent that bus assignment, in a group, will be at best the same as that in a corresponding group of the base model system.

Let e_k be the total number of excess requests in different subgroups of the k^{th} group and let s_k be the total number of surplus buses in the subgroups of the same group. In the base model system excess requests in subgroups (conceptual) of a group will be balanced by surplus buses of other subgroups (conceptual) of the same group. Excess requests in some subgroups of a group will not get the surplus buses of the subgroups in other groups. In other words in the base model system e_k will be met only up to extent of s_k. . Excess requests in the k^{th} group in MPRMB

would be met only to the extent of bus surplus in the k^{th} group. However excess requests for all the groups will be met by the surplus buses of all the subgroups. Stated in other way the excess requests would be met up to $\sum_{k=1}^{G} \min(e_k, s_k)$ by using bus surpluses in all the subgroups. On one hand this makes more efficient use of the surplus buses, but on the other hand since processors are no longer connected to all the buses, the demand for common buses is likely to increase. To determine the number of common buses in the new MPRMB system first we introduce some notation.

Notation

$\alpha((\hat{i}_1\check{i}_1), (\hat{i}_2, \check{i}_2) \cdots, (\hat{i}_g, \check{i}_\tau) \;:\; n)$: a function whose value is 1 if request pattern $((\hat{i}_1, \check{i}_1), (\hat{i}_2, \check{i}_2), \cdots, (\hat{i}_\tau, \check{i}_\tau))$ produces $> n$ excess requests, else its value is zero.

$\hat{P}r((\hat{i}_1\check{i}_1), (\hat{i}_2, \check{i}_2) \cdots, (\hat{i}_g, \check{i}_\tau))$: probability that one subgroup (no matter which subgroup) has $(\hat{i}_1, \check{i}_1)$ requests, one subgroup has $(\hat{i}_2, \check{i}_2)$ requests and so on.

$((\hat{I}_1, \check{I}_1), (\hat{I}_2, \check{I}_2), \cdots, (\hat{I}_\tau, \check{I}_\tau))$: random vector denoting a request pattern.

$f((\hat{I}_1, \check{I}_1), (\hat{I}_2, \check{I}_2), \cdots, (\hat{I}_\tau, \check{I}_\tau))$: random variable representing the minimum number of common buses needed in a memory cycle to satisfy a request pattern with the same number of buses as that assigned in a base model system.

$P_{f((\hat{I}_1, \check{I}_1), (\hat{I}_2, \check{I}_2), \cdots, (\hat{I}_\tau, \check{I}_\tau))}(n)$: The cumulative distribution function for the random variable $f(((\hat{I}_1, \check{I}_1), (\hat{I}_2, \check{I}_2), \cdots, (\hat{I}_\tau, \check{I}_\tau)))$

We basically have to determine the cumulative distribution function for the random variable $f((\hat{I}_1, \check{I}_1), (\hat{I}_2, \check{I}_2), \cdots, (\hat{I}_\tau, \check{I}_\tau))$ and ensure a value for that function that is $\geq \epsilon$, where ϵ is the desired performance level. In order to determine that we first determine $g_{min}(n)$ and $\sigma(n)$, where these quantities carry the same meaning as defined earlier. In an MPRMB system a subgroup may have up to M_g excess requests that cannot be fulfilled by local buses. This is because for all memory modules in a

subgroup requests from far processors may pass memory arbitration. Therefore,

$$
\begin{aligned}
g_{min}(n) &= \left\lceil \frac{n+1}{M_g} \right\rceil \\
\sigma(n) &= n+1-\left\lfloor \frac{n+1}{M_g} \right\rfloor M_g
\end{aligned}
$$

The number of excess requests of the m^{th} subgroup in the k^{th} group is the sum of two components. The first component is the number of memory modules in excess of B_g for which requests from near processors passed memory arbitration. The second component is the number of memory modules for which requests from far processors passed memory arbitration. Requests from far processors cannot be met by the local buses of a subgroup. So the number of excess requests of the m^{th} subgroup in the k^{th} group is given by,

$$\min(\hat{\imath}_{(k-1)g+m} - B_g, 0) + \check{\imath}_{(k-1)g+m}$$

The total number of excess requests in all subgroups of the k^{th} group is therefore

$$e_k = \sum_{m=1}^{g} \min(\hat{\imath}_{(k-1)g+m} - B_g, 0) + \check{\imath}_{(k-1)g+m}$$

A subgroup will have bus surplus if the number of local buses is more than the number of memory modules for which near processors pass memory arbitration. Proceeding as above, the number of surplus buses in the k^{th} group can be determined as

$$s_k = \sum_{k=1}^{g} \min(B_g - \hat{\imath}_{(k-1)g+m}, 0)$$

In the base model system the maximum number of excess requests that could be satisfied in a group is $\min(s_k, e_k)$. In order to have the same bus assignment in all the groups, the number of surplus buses that need to be taken away is $\sum_{i=1}^{G} \min(e_i, s_i)$, and therefore we can write

$$
\begin{aligned}
\alpha((\hat{i}_1\check{i}_1),(\hat{i}_2,\check{i}_2)\cdots,(\hat{i}_g,\check{i}_\tau):n) &= 1, \quad \text{if } \sum_{i=1}^{G}\min(e_i,si) > n \\
&= 0, \quad \text{otherwise}
\end{aligned}
$$

In order to determine $P_{f((\hat{i}_1,\check{i}_1),(\hat{i}_2,\check{i}_2),\cdots,(\hat{i}_\tau,\check{i}_\tau))}(n)$, we need to determine $\hat{P}r((\hat{i}_1\check{i}_1),(\hat{i}_2,\check{i}_2)\cdots,(\hat{i}_g,\check{i}_\tau))$. Let the request pattern $((\hat{i}_1,\check{i}_1),(\hat{i}_2,\check{i}_2),\cdots,(\hat{i}_\tau,\check{i}_\tau))$ be denoted by a set of tuples R and assume that this set can be partitioned into k subsets, where $1 \le k \le \tau$, such that each subset has identical members. Also let the number of members in subset j be denoted by η_j.

If the request model is uniform or, in the case of locality based models if either subgroups are equally distributed among classes or classes are equally distributed among the subgroups, then

$$
\begin{aligned}
\hat{P}r((\hat{i}_1,\check{i}_1),(\hat{i}_2,\check{i}_2),\cdots,(\hat{i}_\tau,\check{i}_\tau)) &= Pr((\hat{i}_1,\check{i}_1),(\hat{i}_2,\check{i}_2),\cdots,(\hat{i}_\tau,\check{i}_\tau)) \\
&\quad \times \frac{\tau!}{\eta_1!\eta_2!\cdots\eta_k!}
\end{aligned}
$$

For other request models,

$$
\begin{aligned}
\hat{P}r((\hat{i}_1,\check{i}_1),(\hat{i}_2,\check{i}_2),\cdots,(\hat{i}_\tau,\check{i}_\tau)) &= \frac{1}{\eta_1!\cdots\eta_k!}\sum_{(\hat{n}_1,\check{n}_1)\in R}\;\sum_{(\hat{n}_2,\check{n}_2)\in R-(\hat{n}_1,\check{n}_1)}\cdots \\
&\quad \sum_{(\hat{n}_\tau,\check{n}_\tau)\in R-(\hat{n}_1,\check{n}_1)-\cdots-(\hat{n}_{\tau-1},\check{n}_{\tau-1})}
\end{aligned}
$$

$$Pr((\hat{i}_1, \check{i}_1), (\hat{i}_2, \check{i}_2), \cdots, (\hat{i}_\tau, \check{i}_\tau))$$

Lemma 1 which identifies the request patterns that will not be satisfied in the same way as in the base model system, does apply to the MPRMB system with slight modification. In order for a request pattern not to be assigned the same number of buses as in a base model system, at least for one subgroup the number of requests from near processors passing memory arbitration has to be less than B_g. Thus we can write,

$$\begin{aligned} P_{f((\hat{I}_1,\check{I}_1),(\hat{I}_2,\check{I}_2),\cdots,(\hat{I}_\tau,\check{I}_\tau))}(n) &= 1.0 - \sum_{\hat{i}_1=0}^{B_g-1} \sum_{\check{i}_1=0}^{M_g-\hat{i}_1} \sum_{\hat{i}_2=\hat{i}_1}^{M_g} \sum_{\check{i}_2=0}^{M_g-\hat{i}_2} \cdots \sum_{\hat{i}_\beta=max(\hat{i}_{\beta-1},\sigma(n))}^{M_g} \sum_{\check{i}_\beta=0}^{M_g-\hat{i}_\beta} \cdots \\ & \sum_{\hat{i}_g=\hat{i}_{g-1}}^{M_g} \sum_{\check{i}_g=0}^{M_g-\hat{i}_g} \hat{P}r((\hat{i}_1, \check{i}_1), (\hat{i}_2, \check{i}_2), \cdots, (\hat{i}_\tau, \check{i}_\tau)) \times \\ & \alpha(((\hat{i}_1, \check{i}_1), (\hat{i}_2, \check{i}_2), \cdots, (\hat{i}_\tau, \check{i}_\tau)) : n) \end{aligned}$$

Using the above expression we determine the minimum number of common buses that will achieve, with high probability, the same level of request satisfaction as in a base model system.

5.3 Numerical Results

We will consider some of the networks considered earlier for the evaluation of the number of common buses in MPRMB systems. We first, consider the base model system with $P = M = 64, B = 32, G = 1$. We also consider a MPRMB system where each group is divided into two subgroups under each of the four request

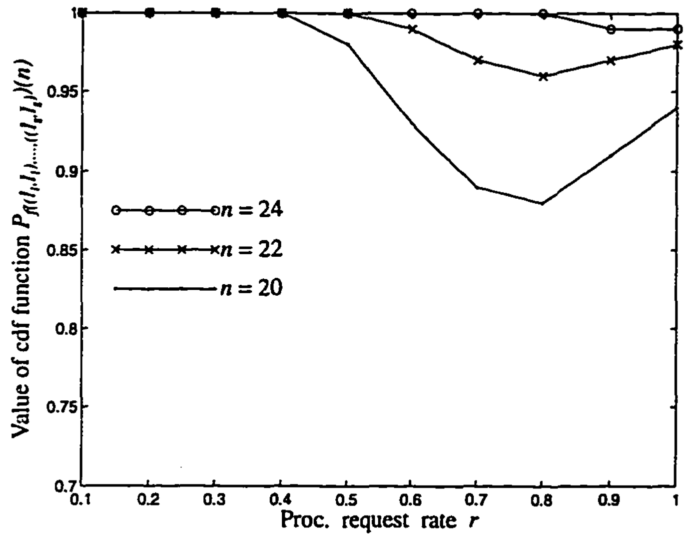


Figure 5.4: Probability that in a MPRMB system a request pattern will get the same number of buses as in a base model system under uniform request model for a network of size 64 × 64 × 32 with 1 group and 2 subgroups in each group.

models we considered earlier. For this system results are shown in Figures 5.4–5.11.

The figures illustrate that for a probability of $\geq .99$ that bus allocation in a group will remain the same as in the base model system, we need 24 common buses for either of the uniform request model or the hotspot request model. This number is much higher than the corresponding number for an OPRMB system. This higher number is expected because in a MPRMB system processors are not connected to all the buses. An MPRMB system will be acceptable only if it has less connectivity cost than the corresponding OPRMB configuration. Another important point to note here is that the number of common buses required is smaller for locality based request models (with and without hotspots). This result is reasonable because with locality present in the request model, processors will request memory modules in the same class more frequently. By definition processors and memory modules in the

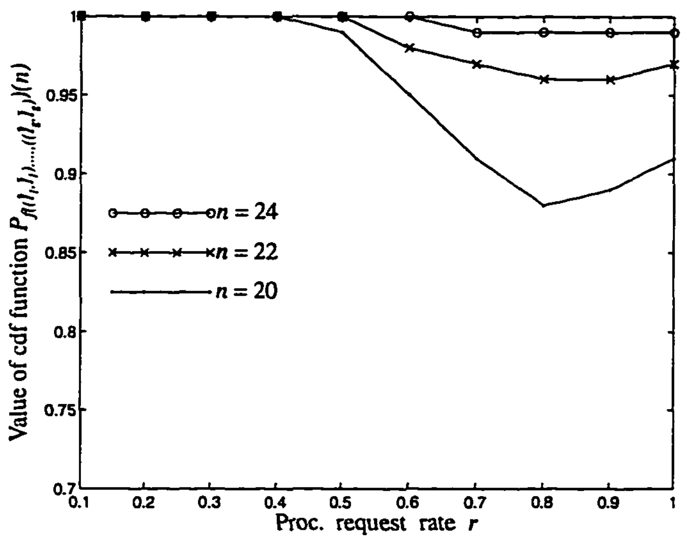


Figure 5.5: Probability that in a MPRMB system a request pattern will get the same number of buses as in a base model system under hotspot request model for a network of size 64 × 64 × 32 with 1 group and 2 subgroups in each group ($h = .1$).

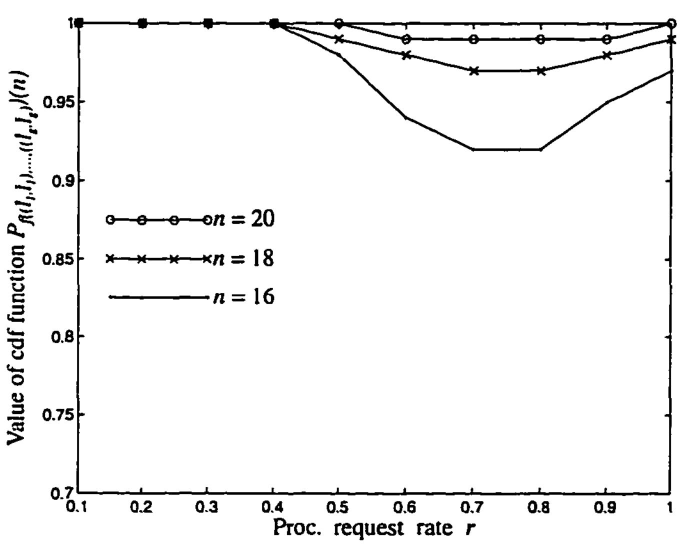


Figure 5.6: Probability that in a MPRMB system a request pattern will get the same number of buses as in a base model system under locality based request model for a network of size 64 × 64 × 32 with 1 group and 2 subgroups in each group ($C = 4, l = .3$).

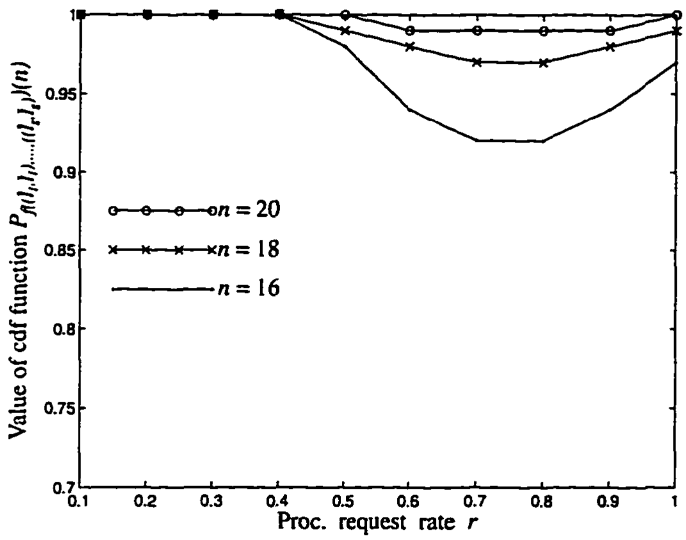


Figure 5.7: Probability that in a MPRMB system a request pattern will get the same number of buses as in a base model system under locality based request model with local hotspot for a network of size $64 \times 64 \times 32$ with 1 group and 2 subgroups in each group ($C = 4, l = .3, h = .1$).

same class are contiguous, and in the present example they also belong to the same subgroup. So locality based request models, in a way, ensure that processors from the same subgroup make requests to the memory modules in the same subgroup more frequently. In that case the minimum number of common buses needed will be smaller because requests from non local subgroups which cannot be met without common buses are expected to be fewer. If a class is split between two subgroups, then this advantage might be diminished to some extent and the number of needed common buses might increase. With the above explanation, it seems likely that with the increase of locality the needed number of common buses should reduce.

To verify this we evaluated the needed number of common buses for the same network using a higher locality rate of $l = .5$. The results are illustrated in Figures 5.8–5.9 which show that for the same performance level, the minimum number of

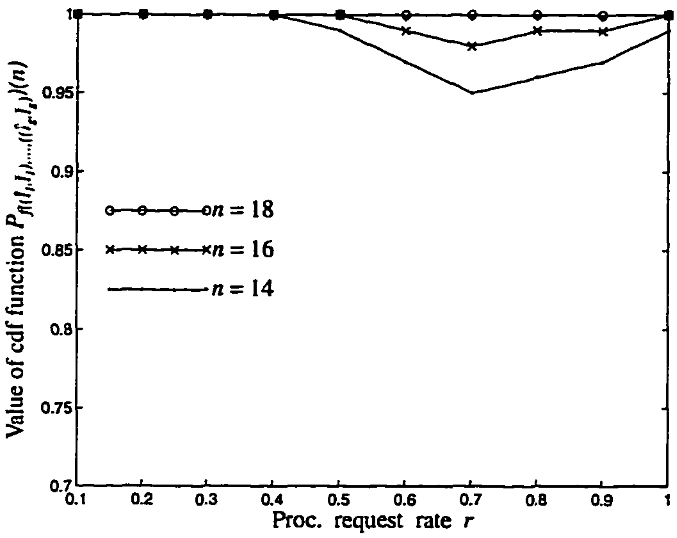


Figure 5.8: Probability that in a MPRMB system a request pattern will get the same number of buses as in a base model system under locality based request model for a network of size 64 × 64 × 32 with 1 group and 2 subgroups in each group ($C = 4, l = .5$).

common buses required reduces to 18. If the locality is increased even further to $l = .7$, then the number of common buses needed is only 14, as illustrated in Figures 5.10–5.11. The decrease in the needed number of common buses with the increase of locality makes the MPRMB system a good candidate for applications with high locality. However the final judgment depends on whether with reduced number of common buses an MPRMB system is less costly than the corresponding OPRMB system or not.

We consider one more network with $P = M = 70, B = 35, G = 1$ for illustration. For the MPRMB system we consider $g = 2$. For all four request models the results of the evaluation of the minimum needed number of common buses are illustrated in Figures 5.12–5.19. For locality based request models, both a moderate rate of $l = .3$ and higher rates like $l = .5$ and $l = .7$ have been considered. In this case also

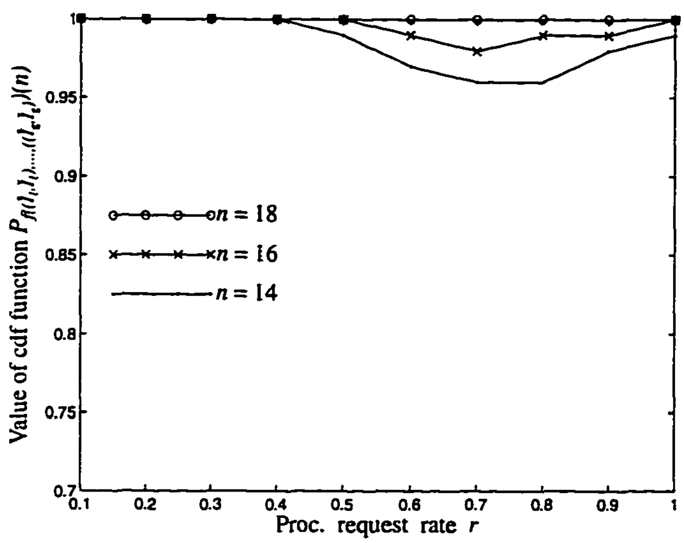


Figure 5.9: Probability that in a MPRMB system a request pattern will get the same number of buses as in a base model system under locality based request model with local hotspot for a network of size 64 $\times$ 64 $\times$ 32 with 1 group and 2 subgroups in each group ($C = 4, l = .5, h = .1$).

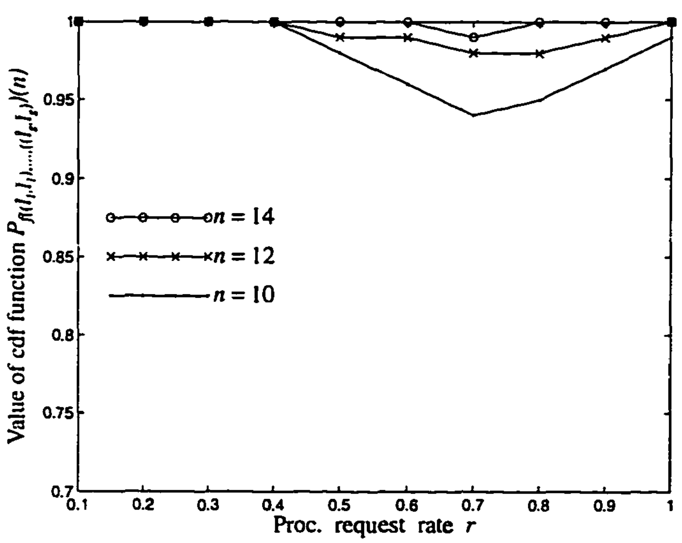


Figure 5.10: Probability that in a MPRMB system a request pattern will get the same number of buses as in a base model system under locality based request model for a network of size 64 $\times$ 64 $\times$ 32 with 1 group and 2 subgroups in each group ($C = 4, l = .7$).

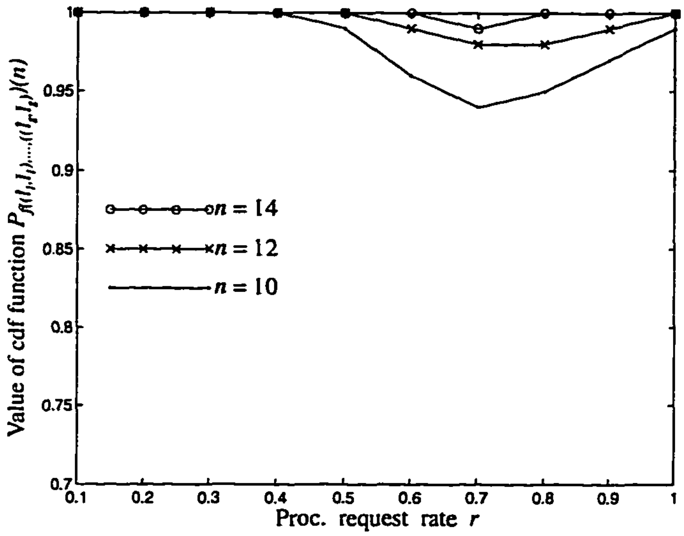


Figure 5.11: Probability that in a MPRMB system a request pattern will get the same number of buses as in a base model system under locality based request model with local hotspot for a network of size 64 × 64 × 32 with 1 group and 2 subgroups in each group ($C = 4, l = .7, h = .1$).

the number of buses required for a locality based request model is less than that for other models; and, the higher the locality rate is the smaller the number of common buses needed will be.

For the networks considered above we evaluated memory bandwidths under all four request models. The results are shown in Tables 5.1-5.16. The connectivity costs for MPRMB systems and the corresponding OPRMB systems are also reported in the tables. Since processor–bus connectivity cost in an MPRMB system and that in an OPRMB system are not the same, we computed the overall connectivity cost. An MPRMB system can be justified only if it has lower connectivity cost than the corresponding OPRMB system. This is because cost connectivity from the base model system has been already reduced in the OPRMB system. From the tables it is clear that while memory bandwidths for both the OPRMB system and MPRMB

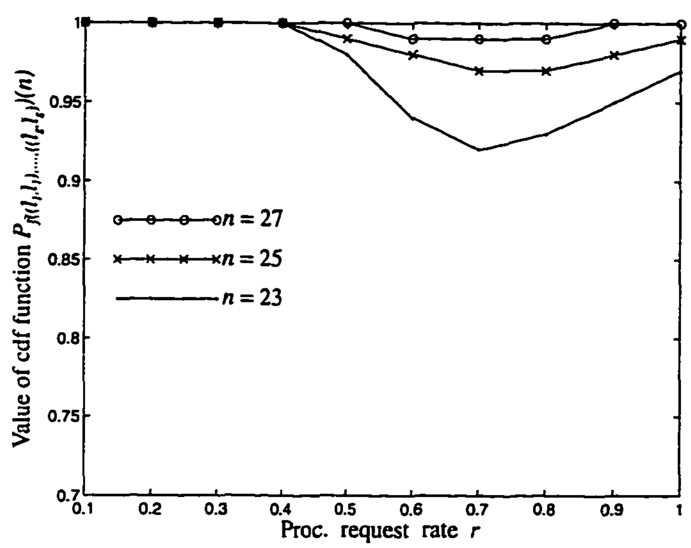


Figure 5.12: Probability that in a MPRMB system a request pattern will get the same number of buses as in a base model system under uniform request model for a network of size 70 × 70 × 35 with 1 group and 2 subgroups in each group.

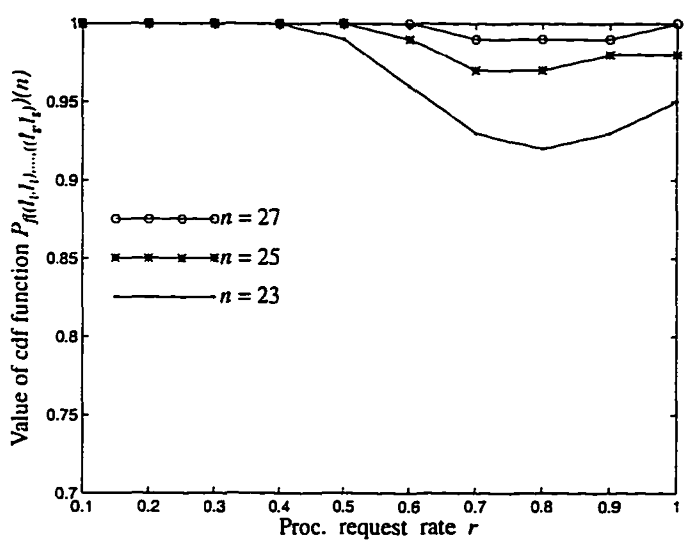


Figure 5.13: Probability that in a MPRMB system a request pattern will get the same number of buses as in a base model system under hotspot request model for a network of size 70 × 70 × 35 with 1 group and 2 subgroups in each group ($h = .1$).

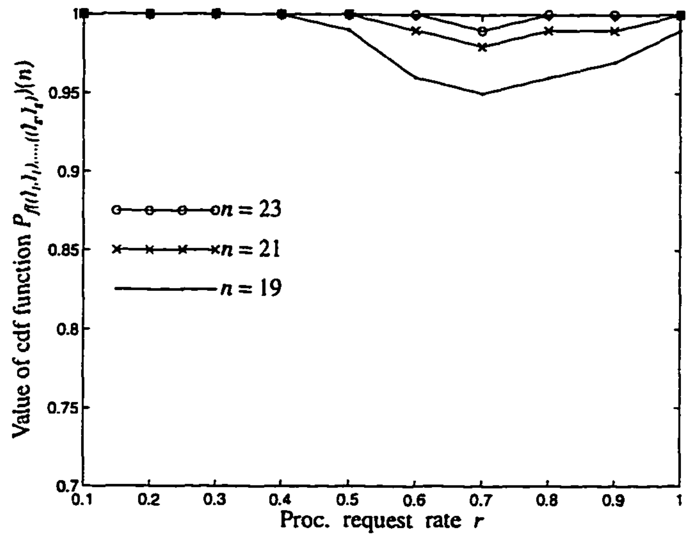


Figure 5.14: Probability that in a MPRMB system a request pattern will get the same number of buses as in a base model system under locality based request model for a network of size $70 \times 70 \times 35$ with 1 group and 2 subgroups in each group $(C = 4, l = .3)$.

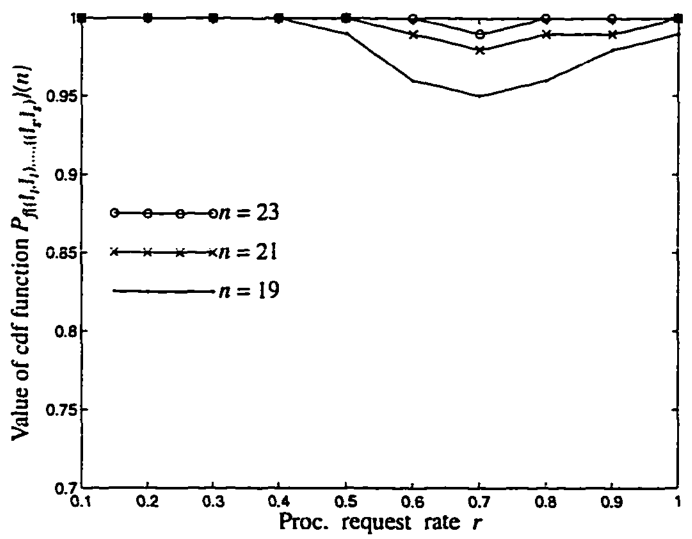


Figure 5.15: Probability that in a MPRMB system a request pattern will get the same number of buses as in a base model system under locality based request model with local hotspot for a network of size $70 \times 70 \times 35$ and 1 group and 2 subgroups in each group $(C = 4, l = .3, h = .1)$.

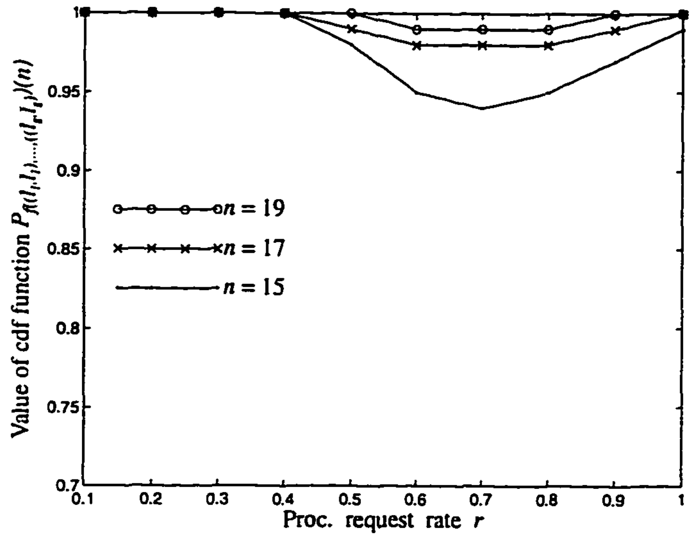


Figure 5.16: Probability that in a MPRMB system a request pattern will get the same number of buses as in a base model system under locality based request model for a network of size $70 \times 70 \times 35$ with 1 group and 2 subgroups in each group ($C = 4, l = .5$).

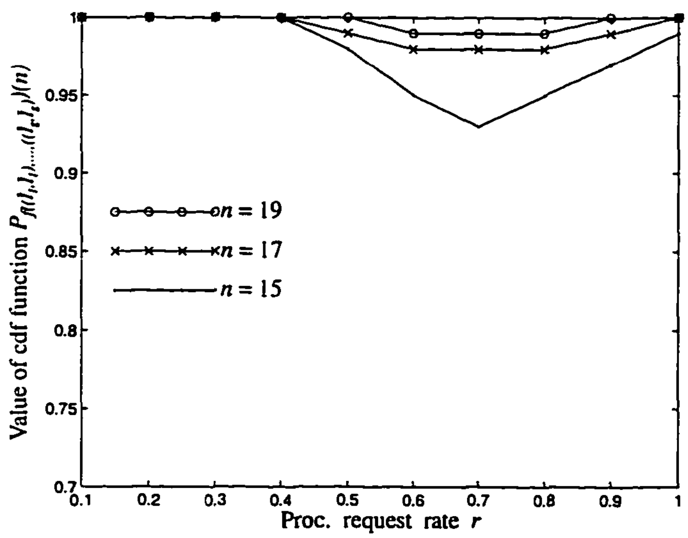


Figure 5.17: Probability that in a MPRMB system a request pattern will get the same number of buses as in a base model system under locality based request model with local hotspot for a network of size $70 \times 70 \times 35$ with 1 group and 2 subgroups in each group ($C = 4, l = .5, h = .1$).

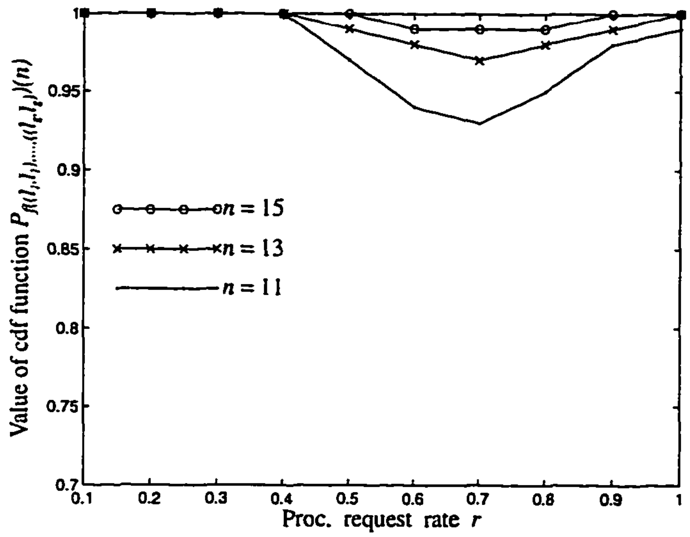


Figure 5.18: Probability that in a MPRMB system a request pattern will get the same number of buses as in a base model system under locality based request model for a network of size 70 × 70 × 35 with 1 group and 2 subgroups in each group ($C = 4, l = .7$).

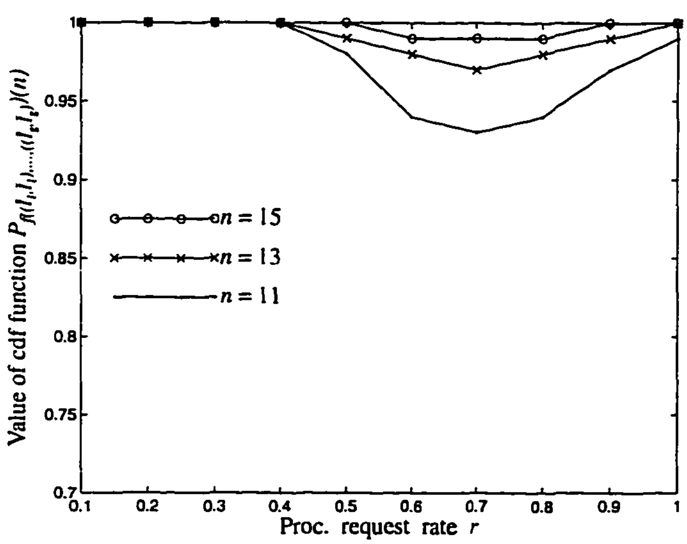


Figure 5.19: Probability that in a MPRMB system a request pattern will get the same number of buses as in a base model system under locality based request model with local hotspot for a network of size 70 × 70 × 35 with 1 group and 2 subgroups in each group ($C = 4, l = .7, h = .1$).

system are almost identical for all the request models, overall cost for MPRMB system deteriorates for uniform and hotspot request models. For locality based request models the cost situation improves. For a modest locality rate like, $l = .3$, the cost of an OPRMB system is very similar to that of an MPRMB system. If there is high locality (like $l = .5$ or $l = .7$), then the cost of an MPRMB system is less than that of an OPRMB system. With a locality rate of $l = .7$, there is cost improvement of up to 9%. This is due to the decrease in the needed number of common buses with locality based request models. If it is known that some particular application executes algorithms which exhibit high locality then MPRMB system might be more economical than the corresponding OPRMB system. Otherwise, in general, it does not seem that the MPRMB system can offer any significant advantages over the OPRMB system. This might be because OPRMB system is so efficient that it might not be possible, in general, to achieve further cost reduction without sacrificing performance.

Table 5.1: Memory bandwidths of MPRMB and OPRMB systems under uniform request model for a network of size 64 × 64 × 32, 1 group and 2 subgroups.

Request rate r	Memory bandwidth OPRMB system ($B_c = 8$) cost=3336	Memory bandwidth MPRMB system ($B_c = 24$) cost=3594
.1	6.095	6.095
.2	11.618	11.618
.3	16.621	16.621
.4	21.150	21.150
.5	25.186	25.186
.6	28.458	28.455
.7	30.568	30.560
.8	31.560	31.550
.9	31.896	31.887
1.0	31.981	31.975

Table 5.2: Memory bandwidths of MPRMB and OPRMB systems under hotspot request model for a network of size 64 × 64 × 32, 1 group and 2 subgroups with $h = .1$.

Request rate r	Memory bandwidth OPRMB system ($B_c = 8$) cost=3336	Memory bandwidth MPRMB system ($B_c = 24$) cost=3594
.1	5.946	5.946
.2	11.163	11.163
.3	15.827	15.827
.4	20.040	20.040
.5	23.843	23.843
.6	27.127	27.126
.7	29.585	29.580
.8	31.039	31.030
.9	31.694	31.684
1.0	31.921	31.913

Table 5.3: Memory bandwidths of MPRMB and OPRMB systems under locality based request model for a network of size 64 × 64 × 32, 1 group and 2 subgroups with $C = 4, l = .3$.

Request rate r	Memory bandwidth OPRMB system ($B_c = 8$) cost=3336	Memory bandwidth MPRMB system ($B_c = 20$) cost=3328
.1	6.096	6.096
.2	11.622	11.622
.3	16.630	16.630
.4	21.165	21.165
.5	25.206	25.206
.6	28.479	28.477
.7	30.583	30.578
.8	31.568	31.562
.9	31.899	31.894
1.0	31.981	31.978

Table 5.4: Memory bandwidths of MPRMB and OPRMB systems under locality based request model with local hotspot for a network of size 64 × 64 × 32, 1 group and 2 subgroups with $C = 4, l = .3$ and $h = .1$.

Request rate r	Memory bandwidth OPRMB system ($B_c = 8$) cost=3336	Memory bandwidth MPRMB system ($B_c = 20$) cost=3328
.1	6.093	6.084
.2	11.609	11.599
.3	16.609	16.596
.4	21.123	21.121
.5	25.150	25.156
.6	28.422	28.435
.7	30.544	30.556
.8	31.549	31.556
.9	31.892	31.895
1.0	31.980	31.980

Table 5.5: Memory bandwidths of multiple bus systems under locality based request model for a network of size $64 \times 64 \times 32$, 1 group and 2 subgroups with $C = 4, l = .5$.

Request rate r	Memory bandwidth OPRMB system ($B_c = 8$) cost=3336	Memory bandwidth MPRMB system ($B_c = 18$) cost=3200
.1	6.098	6.097
.2	11.630	11.626
.3	16.646	16.637
.4	21.191	21.177
.5	25.241	25.223
.6	28.516	28.499
.7	30.611	30.599
.8	31.581	31.575
.9	31.904	31.901
1.0	31.983	31.982

Table 5.6: Memory bandwidths of MPRMB and OPRMB systems under locality based request model with local hotspot for a network of size $64 \times 64 \times 32$, 1 group and 2 subgroups with $C = 4, l = .5$ and $h = .1$.

Request rate r	Memory bandwidth OPRMB system ($B_c = 8$) cost=3336	Memory bandwidth MPRMB system ($B_c = 18$) cost=3200
.1	6.088	6.084
.2	11.594	11.599
.3	16.574	16.596
.4	21.077	21.121
.5	25.091	25.156
.6	28.364	28.435
.7	30.504	30.556
.8	31.530	31.556
.9	31.886	31.895
1.0	31.978	31.980

Table 5.7: Memory bandwidths of multiple bus systems under locality based request model for a network of size $64 \times 64 \times 32$, 1 group and 2 subgroups with $C = 4, l = .7$.

Request rate r	Memory bandwidth OPRMB system ($B_c = 8$) cost=3336	Memory bandwidth MPRMB system ($B_c = 14$) cost=2944
.1	6.102	6.097
.2	11.642	11.626
.3	16.671	16.637
.4	21.230	21.177
.5	25.294	25.223
.6	28.572	28.499
.7	30.651	30.599
.8	31.601	31.575
.9	31.910	31.901
1.0	31.984	31.982

Table 5.8: Memory bandwidths of MPRMB and OPRMB systems under locality based request model with local hotspot for a network of size $64 \times 64 \times 32$, 1 group and 2 subgroups with $C = 4, l = .7$ and $h = .1$.

Request rate r	Memory bandwidth OPRMB system ($B_c = 8$) cost=3336	Memory bandwidth MPRMB system ($B_c = 14$) cost=2944
.1	6.082	6.084
.2	11.572	11.599
.3	16.532	16.596
.4	21.012	21.121
.5	25.008	25.156
.6	28.282	28.435
.7	30.448	30.556
.8	31.504	31.556
.9	31.878	31.895
1.0	31.976	31.980

Table 5.9: Memory bandwidths of MPRMB and OPLMB systems under uniform request model for a network of size 70 × 70 × 35, 1 group and 2 subgroups in each group.

Request rate r	Memory bandwidth OPRMB system ($B_c = 9$) cost=4060	Memory bandwidth MPRMB system ($B_c = 27$) cost=4340
.1	6.666	6.666
.2	12.705	12.705
.3	18.176	18.176
.4	23.129	23.129
.5	27.558	27.558
.6	31.177	31.178
.7	33.507	33.507
.8	34.570	34.570
.9	34.908	34.907
1.0	34.985	34.984

Table 5.10: Memory bandwidths of MPRMB and OPRMB systems under hotspot request model for a network of size 70 × 70 × 35, 1 group and 2 subgroups in the group with $h = .1$.

Request rate r	Memory bandwidth OPRMB system ($B_c = 9$) cost=4060	Memory bandwidth MPRMB system ($B_c = 27$) cost=4340
.1	6.491	6.491
.2	12.177	12.177
.3	17.265	17.265
.4	21.865	21.865
.5	26.031	26.031
.6	29.659	26.657
.7	32.398	32.395
.8	34.005	34.001
.9	34.705	34.701
1.0	34.931	34.929

Table 5.11: 3Memory bandwidths of MPRMB and OPRMB systems under locality based request model for a network of size $70 \times 70 \times 35$, 1 group and 2 subgroups in the group with $C = 4, l = .3$.

Request rate r	Memory bandwidth OPRMB system ($B_c = 9$) cost=4060	Memory bandwidth MPRMB system ($B_c = 23$) cost=4060
.1	6.667	6.670
.2	12.710	12.719
.3	18.186	18.204
.4	23.145	23.174
.5	27.580	27.619
.6	31.203	31.242
.7	33.528	33.554
.8	34.581	34.592
.9	34.912	33.265
1.0	34.986	34.986

Table 5.12: Memory bandwidths of MPRMB and OPRMB systems under locality based request model with local hotspot for a network of size $70 \times 70 \times 35$, 1 groups and 2 subgroups in the group with $C = 4, l = .3$ and $h = .1$.

Request rate r	Memory bandwidth OPRMB system ($B_c = 9$) cost=4060	Memory bandwidth MPRMB system ($B_c = 23$) cost=4060
.1	6.658	6.668
.2	12.685	12.713
.3	18.142	18.193
.4	23.081	23.156
.5	27.497	27.595
.6	31.118	31.218
.7	33.468	33.538
.8	34.553	34.585
.9	34.903	34.912
1.0	34.984	34.986

Table 5.13: Memory bandwidths of MPRMB and OPRMB systems under locality based request model for a network of size 70 x 70 x 35, 1 group and 2 subgroups in the group with $C = 4, l = .5$.

Request rate r	Memory bandwidth OPRMB system ($B_c = 9$) cost=4060	Memory bandwidth MPRMB system ($B_c = 19$) cost=3780
.1	6.669	6.670
.2	12.718	12.719
.3	18.202	18.204
.4	23.172	23.174
.5	27.617	27.619
.6	31.238	31.242
.7	33.551	33.554
.8	34.591	34.592
.9	34.915	33.265
1.0	34.986	34.986

Table 5.14: Memory bandwidths of MPRMB and OPRMB systems under locality based request model with local hotspot for a network of size 70 x 70 x 35, 1 groups and 2 subgroups in the group with $C = 4, l = .5$ and $h = .1$.

Request rate r	Memory bandwidth OPRMB system ($B_c = 9$) cost=4060	Memory bandwidth MPRMB system ($B_c = 19$) cost=3780
.1	6.649	6.668
.2	12.661	12.713
.3	18.098	18.193
.4	23.015	23.156
.5	27.414	27.595
.6	31.036	31.218
.7	33.413	33.538
.8	34.528	34.585
.9	34.895	34.912
1.0	34.982	34.986

Table 5.15: Memory bandwidths of MPRMB and OPRMB systems under locality based request model for a network of size $70 \times 70 \times 35$, 1 group and 2 subgroups in the group with $C = 4, l = .7$.

Request rate r	Memory bandwidth OPRMB system ($B_c = 9$) cost=4060	Memory bandwidth MPRMB system ($B_c = 15$) cost=3500
.1	6.673	6.670
.2	12.730	12.719
.3	18.228	18.204
.4	23.212	23.174
.5	27.671	27.619
.6	31.297	31.242
.7	33.593	33.554
.8	34.611	34.592
.9	34.921	33.265
1.0	34.988	34.986

Table 5.16: Memory bandwidths of MPRMB and OPRMB systems under locality based request model with local hotspot for a network of size $70 \times 70 \times 35$, 1 groups and 2 subgroups in the group with $C = 4, l = .7$ and $h = .1$.

Request rate r	Memory bandwidth OPRMB system ($B_c = 9$) cost=4060	Memory bandwidth MPRMB system ($B_c = 15$) cost=3500
.1	6.638	6.668
.2	12.629	12.713
.3	18.039	18.193
.4	22.928	23.156
.5	27.303	27.595
.6	31.927	31.218
.7	33.338	33.538
.8	34.495	34.585
.9	34.885	34.912
1.0	34.980	34.986

Chapter 6

Conclusion

In this dissertation we have introduced the first systematic and formal approach for reducing connectivity in general purpose multiple bus systems such that performance degradation is negligible. Our work has been based on the hypothesis that many connections in traditional multiple bus systems are only needed to satisfy some highly improbable request patterns. We have defined a generalized base model system which encompasses the traditional full bus connection system and the partial connection bus systems. We have also introduced a new architecture called probabilistically reduced connection multiple bus system; PRMB for short. In a PRMB system groups of memory modules are divided into subgroups. Within a subgroup a number of local buses is utilized to connect the memory modules within the subgroup. In addition some buses, called common buses, are connected to the all memory modules within the group (i.e., they are common to all subgroups in the group). Further, we have introduced a very general request model which encompasses the following well–known models: 1) uniform request model, 2) hotspot request model 3) locality based request model; and 4) locality based request model with local hotspots.

We have determined the memory bandwidth for the PRMB system assuming a given number of subgroups in a group, a given number of common buses per group and a given number of local buses per subgroup. We then attempted to evaluate the minimum number of common buses that would be needed to achieve a given performance level, which would typically be chosen to be very close to that of the

corresponding base model system (a system with the same total number of groups, memory modules, processors and buses). In the process we have considered request patterns that are not permutations of already considered patterns. Our evaluations were conducted using all four request models. The results of our evaluations have shown that memory–bus connectivity cost can be significantly reduced without any significant performance degradation for a range of system configurations. A typical cost reduction range is 20%–37%.

An attempt has been made to determine the optimal PRMB architecture corresponding to a given base model system. Although it did not seem that the problem is tractable based on theory alone, it has been found that a simple algorithmic approach could be utilized to find the solution. To that end we have established lower and upper bounds on the number of common buses needed in a PRMB system such that a certain performance is achieved, given that the parameters of another PRMB (derived from the same base model system) that meets the performance criterion are known. These bounds limit the search space considerably. We have then introduced an algorithm which uses the established bounds to find the optimal PRMB architecture. Although the algorithm has an exponential time complexity it only needs to be run during the design phase of the system. We have then utilized the algorithm to determine the optimal architectures for several previously analyzed networks. Our results have shown that for negligible performance degradation (at most 1%), we could achieve significant cost reduction. For instance memory–bus connectivity cost could be reduced by 35%–42% with at most 1% performance degradation.

To study the effect of reduced connectivity on system performance in the presence of memory queues, we have developed queuing analysis for PRMB systems. The problem at hand has been found to be very difficult if an exact solution is de-

sired. Thus we have resorted to an approximation technique based on the method of aggregation. The approach is approximate because our queuing network is not a product form network. In this approach we first aggregate memory queues in a subgroup into a single queue by taking two queues at a time. The aggregated subgroup queues are then aggregated to form an aggregated group queue by the method of successive aggregation. The method of successive aggregation is a creative application of two queue aggregation method. It utilizes probability information from each step of the two queue aggregation for aggregating all the subgroup queues in single step taking bus contention into account. The aggregated group queues are then aggregated to form a combined memory queue. At the same time request generation of active processors is modeled by a single queue. The final queues are utilized to form a closed two queue system. Our simplifying technique utilized another approximation which is concerned with the distribution of non-empty memory queues in different subgroups of a group. We assumed that this distribution for a given number of customers in all the queues is independent of bus contention.

Using the aggregation technique we have solved the queuing problem and evaluated the performance of different PRMB systems under the uniform and hotspot request models. We compared the performance of PRMB systems with those of base model system counterparts. The comparison of the two systems has shown that reduced bus connectivity has no adverse affect on performance in the presence of memory queues.

To check the effect of the approximations introduced in the queuing analysis we have run some extensive set of simulations without the approximations. Our simulation results closely matched those obtained from the queuing analysis thereby justifying our intuition that the assumptions had little effect on the accuracy of the

approximate queuing approach. The results show that error due to the approximating technique, in all the cases, is less than 1%.

We have also studied the possibility of further reduction in connectivity cost by suggesting another architecture in which processors are not connected to all buses as in the original PRMB system. Both memory modules and processors are divided into equal sized subgroups. Processors and memory modules in the same subgroup are connected by some local buses. Besides there are some common buses which connect all the processors and memory modules in all the groups. In this modified version of PRMB system sources of requests and arbitration play important roles in determining both the memory bandwidth and the needed minimum number of common buses. This is because requests from processors passing arbitration but belonging to subgroups different from those of the requested memory modules can be met only by common buses. So, while considering a request pattern in the modified system one has to take into account, for each subgroup, the number of memory modules for which near processors pass memory arbitration and the number of memory modules for which far processors pass memory arbitration. We have evaluated both memory bandwidth and the needed minimum number of common buses for this architecture. Our numerical results indicated that reducing connectivity beyond that achieved by the original PRMB system may not be possible except when the request model involves a high degree of locality. Even in this case the reduction is rather insignificant and does not justify using this last approach.

Future Research

Future research could study whether connectivity cost of the modified PRMB cost can be significantly improved by modifying the arbitration mechanism. An unfair

arbitration scheme which favors near processor requests would make better use of local buses and might possibly reduce the number of common buses needed. This might make it possible to achieve cost reduction beyond that is achieved by the original PRMB system. Another issue that should be addressed in future research is the question of to solve the queuing problem of the PRMB system under locality based models. Note that in that case we have to deal with multiple classes of customers and the aggregation technique as we adopted, by itself, will not be sufficient to handle the problem.

An important aspect of our research is the probabilistic connection reduction technique itself. Though in this dissertation the technique been applied in the context of general purpose multiple bus systems only, the approach can possibly be extended in the domain of other interconnection networks such as other type of multiple bus systems, crossbar networks and perhaps multistage interconnection networks. Probabilistically redundant connectivity, most possibly, exist in all of these networks. A creative approach needs to be developed to modify the architectural features of these networks and apply this new technique.

Bibliography

[1] G. S. Almasi and A. Gottelib, "Highly parallel computing," The Benjamin/Cumming Publishing Company, Inc, 1989.

[2] Forest Basket, K. Mani Candy, Richard R. Muntz and Fernando G. Palacios, "Open, closed, and mixed networks of queues with different classes of customers," *Journal of the Association for Computing Machinary*, Vol. 22, No. 2, pp. 248–256, April 1975

[3] L. N. Bhuyan, "An analysis of processor–memory interconnection networks," *IEEE Trans. on Comput*, vol. C-34, no. 3, pp 279-283, March 1985

[4] K. M. Chandy, U. Herzog, L. Woo, "Parameteric analysis of queuing networks," *IBM J. RES. DEVELOP.*, pp 36–42, January 1975

[5] W. Chen, "Performance analysis of a multiple bus interconnection networks with hierarchal requesting model," *IEEE Trans. on Comput*, vol. 40, no. 7, July 1991

[6] G. Chiola, M. A. Marsan and G. Balbo, "Product–form solution technique for the performance analysis of multiple–bus multiprocessor systems with nonuniform memory references," *IEEE Trans. on Comput*, vol. 37, no 5, pp 532–540, May 1988

[7] C. R. Das, and L. N. Bhuyan, "Bandwidth availability of multiple–bus multiprocessors," *IEEE Trans. on Comput*, vol. C-34, no. 10, pp 918-926, Oct 1985

[8] K. B. Irani and I. H. Onyuksil, "A closed form solution for the performance analysis of multiple–bus systems," *IEEE Trans. on Comput*, vol. C-33, no 11, pp 1004-1012, Nov, 1984.

[9] H. Jiang and K. C. Smith, "PPMB: A partial–multiple–bus multiprocessor architecture with improved cost–effectiveness," *IEEE Trans. on Comput*, vol. 41, no. 3, pp 361-366, March 1992.

[10] T. Lang, M.Valero, and I. Algere "Bandwidth of crossbar and multiple bus connections for multiprocessors," *IEEE Trans. on Comput*, vol. D-31, pp 1227-1234, Dec. 1982.

[11] T. Lang, M.Valero, "M–users-B–servers arbiter for multiple buses multiprocessors," *Microprocessors and Microprogramming*, North Holland Publishing Company, 10, 1982, pp. 11-18

[12] T. Lang, M.Valero, and M. A. Foil, "Reduction of connections for multiple buses multiprocessors," *IEEE Trans. on Comput*, vol. C-32, pp 707-715, Aug 1983.

[13] J. Y. Le Boudec"A BCMP extension to multiserver stations with concurrent classes of customers," *Proc. ACM SIGMETRICS*, Raleigh, NC, May 1986

[14] F. Thomas Leighton, "Introduction to parallel algorithm and architectures: arrays. trees. hypercubes," Morgan Kaufmann Publishers, 1992.

[15] Y. -C. Liu and C. -J. Jou, "Effective memory bandwidth and processor blocking probability in multiple–bus systems," *IEEE Trans. on Comput*, vol. C-34, pp 761-764, June 1987

[16] S. M. Mahmud, "Performance Analysis of Multilevel Bus Networks for Hierarchal Multiprocessors," *IEEE Trans. on Comput*, vol 43, no. 7, pp 789-795, July 1994.

[17] M. A. Marsan, G. Balbo, G. Chiola and S. Donatelli, "On the Product–Form Solution of a class of Multiple–Bus Multiprocessor System Models," *The journal of Systems and Software*, vol. 6, pp 117–124, May 1986

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

[19] T. N. Mudge and H. B. Al-Sadoun, "A semi–markov models for the performance of multiple-bus systems," *IEEE Trans. on Comput*, vol. C-34, no 10, pp 934-942, Oct, 1985

[20] H. Stone, "High performance computer architecture," Addison–Wesely Publishing company, 1994.

[21] D. Towsley, "Approximate models for multiple bus multiprocessor Systems," *IEEE Trans. on Comput*, vol. C-35, no 3, pp 220-228, March 1986

[22] D. Towsley, "An approximate analysis of multiprocessor systems," *Proc. SIGMETRICS*, pp 207–213, August 1983

[23] M. Tremblay, "UltraSparc I: A four–issue processor supporitng multimedia," *IEEE Micro*, pp 42-49, April 1996.

[24] B. Wilkinson, "On crossbar switch and multiple bus interconnection networks with overlapping connectivity," *IEEE Trans. on Comput*, vol. 41, no. 6, pp 738-746, June 1992.

[25] N. Viswanadhan and V. Narahain, "Performance modelling of automated manufacturing system," Prentic Hall, 1992

[26] J. L. Zahorjan"Iterative aggregation with global balance," *Project SAM Notes, Computer Science Research Group, University of Toronto*, Feb. 1977

Appendix A. Complexity of the Algorithm to determine optimal PRMB system

The complexity for the algorithm to find the optimal PRMB architecture is derived as follows. The outermost loop of the algorithm, indexed by k, considers all possible numbers of subgroups within a group. Since each subgroup has equal number of memory modules, the number of subgroups in a group is a factor of M_G. Also, the number of subgroups in a group is less than B_G, otherwise there cannot be any common buses. Thus k is the number of factors of M_g in the interval $[g_{min}, g_{max}]$ and is therefore $\leq \frac{g_{max}}{g_{min}}$.

The second outer loop, indexed by B_c, considers different possible values for the number of common buses for a given number of subgroups g. The value of B_c varies between b_{c_l} and b_{c_h}, where b_{c_l} is determined by applying Theorem 2 and b_{c_h} is determined by applying Lemma 3. The value of b_{c_l} cannot be less than g and the value of b_{c_h} cannot be more than B_G. So B_c varies at most between g and B_G and the number of iterations due to the second outer loop is at most $\left\lfloor \frac{B_G}{g} \right\rfloor \leq \frac{B_G}{g}$.

The loop indices $i_1, i_2, \cdots, i_g$ are used to consider different request patterns which will not be satisfied by the considered number of common buses in the same way as in the base model systems. The maximum number of iterations in this case is $\leq B_g \times (M_g + 1) \times M_g \times \cdots \times (M_g - g) \approx B_g M_g^{(g-1)}$, since $M_g >> g$.

The number of iterations before the evaluation of $\tilde{P}r(i_1, i_2, \cdots, i_g)$ is $\frac{g_{max}}{g_{min}} \times \frac{B_G}{g} B_g M_g^{g-1}$. Evaluation of $\tilde{P}r(i_1, i_2, \cdots, i_g)$ involves considering different permutations of the request pattern $(i_1, i_2, \cdots, i_g)$. The maximum number of request pattern permutations that we might have to consider is $g!$. For each of these request pat-

terns we have to evaluate the probability of the request pattern $P(i_1, i_2, \cdots, i_g)$. The number of iterations involved in evaluating $P(i_1, i_2, \cdots, i_g)$ is $(\frac{C}{Gg}+1)^g$. Therefore the total number of steps in the algorithm

$$\leq \frac{g_{max}}{g_{min}} \times \frac{B_G}{g} \times B_g M_g^{g-1} \times g! \times (\frac{C}{gG}+1)^g$$

Consider the maximum possible value for g, which is g_{max}. This will correspond to the maximum number of iteration steps in the algorithm which is

$$\begin{aligned}
&\leq \frac{B_G}{g_{min}} B_{g_{max}} M_{g_{max}}^{g_{max}-1} g_{max}! \times (\frac{C}{g_{max}G}+1)^{g_{max}} \\
&= \frac{B_G B_{g_{max}}}{g_{min}} M_{g_{max}}^{g_{max}-1} \times (\frac{C+g_{max}G}{Gg_{max}})^{g_{max}} g_{max}! \\
&\leq \frac{B_G B_{g_{max}}}{g_{min}} M_{g_{max}}^{g_{max}-1} \times (\frac{C}{G}+g_{max})^{g_{max}} \\
&= \frac{B_G B_{g_{max}}}{g_{min}} (\frac{C}{G}+g_{max}) \times \{M_{g_{max}}(\frac{C}{G}+g_{max})\}^{g_{max}-1} \\
&= \alpha \beta^{g_{max}-1}
\end{aligned}$$

where $\alpha = \frac{B_G B_{g_{max}}}{g_{min}}(\frac{C}{G}+g_{max})$ and $\beta = M_{g_{max}}(\frac{C}{G}+g_{max})$. Therefore the complexity of the algorithm is $O(\alpha\beta^{g_{max}-1})$.

APPENDIX B. PRMB SYSTEM: A NON–PRODUCT FORM NETWORK

In the queuing analysis of the PRMB (Chapter 4)system we used the method of aggregation to simplify the queuing model. In this case aggregation is used as an approximating technique because PRMB queuing network is not a product form network. Proving that PRMB is not a product form network is analytically very difficult. However we illustrate this fact by considering a simple example.

Consider a network with $P = M = 2, G = 1, B = 1$. We assume two subgroups per group so that $M_g = 1, B_g = 0$ and $B_c = 1$. The closed queue model for this system with 2 customers is shown in Figure B.1. In the figure queue 1 is the processor queue while the queues 2 and 3 are two memory queues. The state of the network is defined by the number of customers present in different queues and is denoted with 3–tuples (i, j, k), where i, j and k correspond to the number of customers in queue 1, queue 2 and queue 3, respectively. Obviously $i, j, k \in \{0, 1, 2\}$ and $i + j + k = 2$.

STATES OF THE NETWORK

There are altogether six states and for convenience we denote them as follows

state 1: $\{2, 0, 0\}$

state 2: $\{1, 0, 1\}$

state 3: $\{1, 1, 0\}$

state 4: $\{0, 1, 1\}$

state 5: $\{0, 2, 0\}$

state 6: $\{0, 0, 2\}$

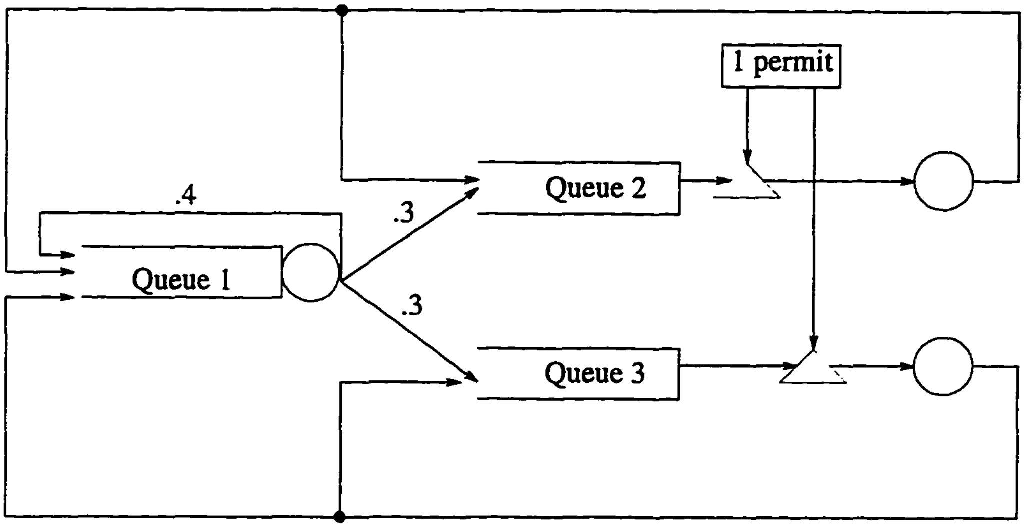


Figure B.1: The example of a simple PRMB network

The transition probability from a state i to another state j is denoted by ρ_{ij}. In the following we determine the different transition probabilities. We assume that in a memory cycle a customer in the processor queue has a probability of .6 to leave the processor queue. We further assume that if a customer leaves the processor queue, then there is an equal probability of .5 that it will join either of the memory queues. When a customer completes service in queue 2 or queue 3 it might encounter the problem of bus contention. If two customers are looking for a bus, then there will be a probability of .5 for each of them to get the bus. Now we can determine all the transitional probabilities straightforwardly as follows.

$$\begin{aligned}
\rho_{11} &= .4 \times .4 = .16 \\
\rho_{12} &= (.6 \times .5 \times .4) \times 2 = .24 \\
\rho_{13} &= (.6 \times .5 \times .4) \times 2 = .24 \\
\rho_{14} &= (.6 \times .5 \times .6 \times .5) \times 2 = .18
\end{aligned}$$

$$
\begin{aligned}
\rho_{15} &= (.6 \times .5 \times .6 \times .5) = .09 \\
\rho_{16} &= (.6 \times .5 \times .6 \times .5) = .09 \\
\rho_{21} &= \rho_{31} = (.4 \times .4) = .16 \\
\rho_{22} &= \rho_{33} = (.4 \times .3) + .6 \times .5 \times .4 = .24 \\
\rho_{23} &= \rho_{32} = .4 \times (.6 \times .5) + (.6 \times .5) \times .4 = .24 \\
\rho_{24} &= \rho_{34} = (.6 \times .5) \times (.6 \times .5) \times 2 = .18 \\
\rho_{25} &= \rho_{35} = .6 \times .5 \times .6 \times .5 = .09 \\
\rho_{26} &= \rho_{36} = .09 \\
\rho_{41} &= 0 \\
\rho_{42} &= .5 \times .4 = .2 \\
\rho_{43} &= .5 \times .4 = .2 \\
\rho_{44} &= (.5 \times .6 \times .5) \times 2 = .3 \\
\rho_{45} &= .5 \times .6 \times .5 = .15 \\
\rho_{46} &= .5 \times .6 \times .5 = .15 \\
\rho_{51} &= 0 \\
\rho_{52} &= 0 \\
\rho_{53} &= .4 \\
\rho_{54} &= .6 \times .5 = .3 \\
\rho_{55} &= .6 \times .5 = .3 \\
\rho_{56} &= 0 \\
\rho_{61} &= 0 \\
\rho_{62} &= .4
\end{aligned}
$$

$$\begin{aligned}
\rho_{63} &= 0 \\
\rho_{64} &= .3 \\
\rho_{65} &= 0 \\
\rho_{66} &= .3
\end{aligned}$$

Let P denote the transition matrix and let Π denotes the probability vector for steady state space. Also let the probability that the network is in state i be denoted by P_i. The solution for the probability distribution for steady state space can be obtained by solving the following set of linear equations

$$\begin{aligned}
P\Pi &= P \\
\sum_{i=1}^{6} P_i &= 1.0
\end{aligned}$$

Solving these linear equations we get,

$$P_1 = P_2 = P_3 = P_4 = P_5 = P_6 = .166667$$

Now assume that the network is a product form network and under that assumption we will attempt to evaluate steady state distribution for state space. Under the product form assumption, we use a different notation to denote the probability of a state. Let $\tilde{P}_i$ be the probability that the network is in state i. We use $\tilde{p}_{ij}$ to denote the routing probability that a customer after finishing in queue i goes to queue j, where $i, j \in \{1, 2, 3\}$. Also, let u_i, be the relative utilization of queue i. Let $\tilde{P} = [\tilde{p}_{ij}]$ denote the routing matrix.

While the probability that a customer goes from queue 1 to any of the queues is straightforward, because of bus contention, the probability that a customer will leave either queue 2 or queue 3 to another queue is state dependent. Let us first evaluate the probability that the customer goes from queue 2 to any of the other queues.

Let us first introduce the following events.

$A \equiv$ queue 2 is non-empty

$B_1 \equiv$ a customer goes from queue 2 to queue 1

$B_2 \equiv$ a customer goes from queue 2 to queue 2

$B_3 \equiv$ a customer goes from queue 2 to queue 3

Now the probability that a customer goes from queue 2 to other queues are evaluated as follows:

$$\begin{aligned}
Pr\{A\} &= P_3 + P_4 + P_5 = .5 \\
Pr\{AB_1\} &= P_3 \times .4 + P_4 \times .5 \times .4 + P_5 \times .4 = .166667 \\
Pr\{AB_2\} &= P_3 \times .6 \times .5 + P_4 \times (.5 + .5 \times .6 \times .5) + P_5 \times .6 \times .5 = .21 \\
Pr\{AB_3\} &= P_3 \times .6 \times .5 + P_4 \times .5 \times .6 \times .5 + P_5 \times .6 \times .5 = .13
\end{aligned}$$

Since $\tilde{p}_{21} = Pr\{B_1|A\}, \tilde{p}_{22} = Pr\{B_2|A\}$ and $\tilde{p}_{23} = Pr\{B_3|A\}$, we can easily evaluate these probabilities as follows

$$\begin{aligned}
\tilde{p}_{21} &= .33 \\
\tilde{p}_{22} &= .42 \\
\tilde{p}_{23} &= .25
\end{aligned}$$

Because of the symmetry of the network, it is obvious that $\tilde{p}_{31} = .33$, $\tilde{p}_{32} = .25$ and $\tilde{p}_{33} = .42$. The routing probabilities from queue 1, which is the processor queue are $\tilde{p}_{11} = .4$, $\tilde{p}_{12} = .3$ and $\tilde{p}_{13} = .3$. Now we can evaluate the relative utilization of the queues as follows

$$\begin{aligned} u_2 &= u_1 \times .3 + u_2 \times .42 + u_3 \times .25 \\ u_3 &= u_1 \times .3 + u_2 \times .25 + u_3 \times .42 \end{aligned}$$

Using $u_1 = 1$, we solve the above equations and obtain $u_2 = .6977$ and $u_3 = .6977$. Since we assumed that the network is a product form network, probability distribution for the state space is given by

$$Pr\{i,j,k\} = \frac{u_1^i u_2^j u_3^k}{G}$$

where i, j, k are, respectively, the number of customers in queue 1, queue 2 and queue 3, and G is the normalizing constant given by

$$G = \sum_{\forall i,j,k \in \{0,1,2\} \text{ and } i+j+k=2} u_1^i u_2^j u_3^k = 3.8558$$

Now the probabilities of individual states under the product form network assumption are evaluated as follows

$$\begin{aligned} \tilde{P}_1 &= \frac{u_1^2}{G} = .2593 \\ \tilde{P}_2 &= \frac{u_1 u_2}{G} = .1809 \\ \tilde{P}_3 &= \frac{u_1 u_3}{G} = .1809 \end{aligned}$$

$$
\begin{aligned}
\tilde{P}_4 &= \frac{u_2 u_3}{G} = .1262 \\
\tilde{P}_5 &= \frac{u_2^2}{G} = .1262 \\
\tilde{P}_6 &= \frac{u_3 u^2}{G} = .1262
\end{aligned}
$$

Obviously the probability of state space under the product form network assumption is not the same as those obtained by direct solution of the Markov chain. Therefore the network is not a product form network. The above example clearly shows that our queuing network for PRMB system are not, in general, product form network.

VITA

Md. Najmul Karim was born in 1961 in Dhaka, Bangladesh. He received his bachelor in Electrical and Electronic Engineering in 1983. He worked for several years in Bangladesh in both academic and research oriented environment. He had his Masters degree from Louisiana State University in 1990. Currently he is a Ph.d candidate in the same school, the degree for which will be conferred in December, 1996.

DOCTORAL EXAMINATION AND DISSERTATION REPORT

Candidate: Md. Najmul Karim

Major Field: Electrical Engineering

Title of Dissertation: Design and Analysis of Reduced Connection Multiple Bus Systems: A Probabilistic Approach

Approved:

A. Elmaghraby
Major Professor and Chairman

Dean of the Graduate School

EXAMINING COMMITTEE:

J. B. Jones

Subhash Kak

J. R

Date of Examination:

October 31, 1996