

T
✓
D65-88
PRA

**ON SOME DESIGN ASPECTS OF INTERCONNECTION NETWORKS
FOR
TIGHTLY COUPLED MULTIPROCESSOR SYSTEMS**

A THESIS

submitted in fulfilment of the
requirements for the award of the degree
of
DOCTOR OF PHILOSOPHY
in
ELECTRONICS AND COMPUTER ENGINEERING

by

E. VENKATESWARA PRASAD



DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING
UNIVERSITY OF ROORKEE
ROORKEE-247667 (INDIA)

AUGUST, 1988

Dedicated

TO

MY PARENTS

Smt. E. KAMESWARI

AND

Sri E. SRIRAMA MURTI

CANDIDATE'S DECLARATION

I here by certify that the work which is being presented in the thesis entitled 'ON SOME DESIGN ASPECTS OF INTERCONNECTION NETWORKS FOR TIGHTLY COUPLED MULTIPROCESSOR SYSTEMS' in fulfilment of the requirement for the award of the Degree of Doctor of Philosophy, submitted in the Department of Electronics & Computer Engineering of the University is an authentic record of my own work carried out during a period from August 1985 to August 1988 under the supervision of Dr.Suresh Rai and Dr.A.K.Sarje.

The matter embodied in this thesis has not been submitted by me for the award of any other Degree.

Date:

(E.V.PRASAD)

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Suresh Rai
(Dr.SURESH RAI)
Reader

Sarje
30/8/88
(Dr.A.K.SARJE)
Professor

The candidate has passed the viva-voce examination held on _____ at _____. The thesis is recommended for award of the Ph.D.Degree.

1. Suresh Rai
(Guide)

1. _____
(External Examiner)

2. _____
(Guide)

ABSTRACT

Tightly connected multiprocessor systems (MPSs) are characterized by the presence of several autonomous processors sharing multiple memory modules via some interconnection network. Since both the basic elements, processors and memory modules, are available as standard integrated circuits, the key design problem is how to put them together so that the system is efficient and reliable. The present work is a study of the interconnecting structures covering both design and analysis aspects. The interconnection networks (INs) considered are a) multiple bus b) crossbar c) multiport memory and d) multistage interconnection network. Methodologies are presented for the design of MPs by considering a variety of performance measures as no single measure gives a truly accurate estimate of the system performance. The criteria considered are : processor - memory interference, fault tolerance and some fundamental measures such as waiting time and hardware utilization.

In the design of an MPS there exists enormous number of alternative decisions. In this thesis, the major design parameters that are allowed to vary for a given architecture are number of processors, memory modules, interconnection links, and the parameters of the computation being executed, such as the memory request probability (MRP) and memory access probabilities (MAP's).

The performance of multiple-bus IN for MPS is analyzed taking into account conflicts arising from memory and bus interference. Given the number of processors, the number of memory modules, the MRP and the MAP's, the model produces as output the memory bandwidth, processor utilization, memory utilization, channel utilization and waiting time of a processor while waiting to access a memory module.

Using this model it is possible to analyze the effect of input parameters on the system performance. The model presented differs from other models in its ability to allow generalized processor demand patterns for memory access with processors of equal memory request probabilities. This model also examines the following three situations : when a memory module is equally likely to be addressed by all processors, when each processor has a different favourite memory and when all processors have the same favourite memory.

Crossbar is a special case of this analysis. The closed form solutions are compared with simulation results. This analysis is extended to partially connected bus and also to Delta Network, a multistage interconnection network.

The modeling technique adapted for the analysis is based on t-out-of-s system principle. Algorithms for computing the exact system reliability of t-out-of-s systems are proposed. These are simple, easy to implement, fast and memory and time efficient.

The two types of real time systems, failure-critical and non failure-critical are considered. Using failure-critical models expressions for multiple-bus, crossbar and multiport memory INs are derived for the criterion, multiprocessing and terminal reliabilities. A technique for computing multiprocessor reliability through explicit path enumeration is also proposed. Non failure - critical models for the analysis of multiprocessing reliability and bandwidth availability are made, where coverage factors take into account the reconfiguration process for a graceful degradation in persence of failure of system components. The INs considered for analysis are multiple-bus and crossbar.

ACKNOWLEDGEMENT

I wish to express my deep appreciation and profound gratitude for the continuous interest taken and encouragement given by Dr.Suresh Rai and Dr.A.K.Sarje, who supervised this work. Dr.Rai's active association in the initial phase before he left to USA and regular dispatches of research material, comments and suggestions through correspondence after, had been a great help and it gives me pleasure to acknowledge him. My heartfelt thanks are due to Prof. Sarje, for his invaluable guidance, painstaking supervision, keen personal interest and meticulous scrutiny which have immensely contributed towards the successful completion of this thesis.

I express my sincere thanks to Dr.N.K.Nanda, the former Head of the Department and Dr.R.Mitra, the present Head of the Department of Electronics and Computer Engineering for informal working conditions and their encouragement.

It won't be out of place to acknowledge the financial support from the QIP scheme of Govt. of India, Ministry of Human Resources and the JNT University, Hyderabad for sponsoring the author to pursue the work. My sincere thanks go to my colleagues and Dr. M. Subbarayudu, the Head of the Department of Electronics and Communication Engineering, J.N.T.U. College of Engineering, Anantapur (AP) for sharing my load during my absence.

It is a pleasure to acknowledge a great deal of assistance received from my friends, academic or otherwise, particularly A.U.Ravi Sankar, J.Ravi Sankar, N.S.Krishna Mohan and P.Vinod Kumar at the time of preparation of this Thesis. Thanks are due to Y.R.Singh, S.Maheswari, A. K. Singh and UPPER GANGES SYSTEMS for neat typing of the manuscript.

Task of undertaking a research involves a sacrifice on the part of many. In this connection, I would like to express my deep gratitude to my life partner, Lakshmi, for extending constant encouragement, moral support and complete cooperation throughout the course of this work. I owe a debt of gratitude to my daughters Sri Vidya, Sri Vani and Sri Valli for their patient understanding and for granting lot of time that was due to them.

As always, I owe more to my beloved parents than words can express.

E. V. PRASAD

CONTENTS

	Title	Page No
	ABSTRACT	i
	ACKNOWLEDGEMENT	iv
	TABLE OF CONTENTS	vi
	LIST OF TABLES	ix
	LIST OF FIGURES	x
	NOTATIONS AND ACRONYMS	xii
CHAPTER I	INTRODUCTION	1
1.1	PARALLEL PROCESSING -AN IDEA	1
	1.1.1 Device Technology	2
	1.1.2 Processing Technology	2
	1.1.3 Taxonomics	4
	1.1.4 Multiprocessing systems	5
1.2	PROBLEM FORMULATION	8
1.3	STATEMENT OF THE PROBLEM	9
1.4	OUTLINE OF THE THESIS	10
CHAPTER II	DESIGN OF MPS- AN OVERVIEW	11
2.1	Interconnection Networks	11
	2.1.1 Time shared Multiple- Bus	12
	2.1.2 Crossbar Switch	15
	2.1.3. Multiport Memory System	15
	2.1.4 Multistage Interconnection Networks (MINs)	15
	2.1.5. Operational Characteristics	17

2.2	SYSTEM EVALUATION	19
	2.2.1 Evaluation Techniques	19
	2.2.2 t-out-of-s System:Modeling	23
2.3	PERFORMANCE EVALUATION CONSIDERATION	23
2.4	FAULT TOLERANCE SYSTEM CONSIDERATION	24
	2.4.1 Reliability	25
	2.4.2 Bandwidth Availability	27
	2.4.3 Evaluation Techniques-Review	27
2.5	CONCLUSION	29
CHAPTER III	MULTIPROCESSOR SYSTEM RELIABILITY	30
3.1	MULTIPROCESSING RELIABILITY	34
	3.1.1 Primary Method	34
	3.1.2 Construction of Connection Matrix	34
	3.1.3 Reduction of the Connection Matrix	38
	3.1.4 Enumeration of Events	39
	3.1.5 Reliability Computation	43
3.2	t-out-of-s SYSTEMS	44
	3.2.1 Previous Works: A Review	45
	3.2.2 Theory	47
	3.2.3 Algorithm Development	49
	3.2.4 An Efficient Recursive Algorithm	55
	3.2.5 Non-Recursive Approach	59
	3.2.6 Comparisons	62
3.3	RELIABILITY MODELING	66
	3.3.1 Multiprocessing Reliability	66

	3.3.2 Terminal Reliability	74
	3.4 CONCLUSION	76
CHAPTER IV	PERFORMANCE EVALUATION	78
4.1	CHARACTERISTICS OF MULTIPROCESSORS	78
4.2	REVIEW OF EXISTING MEMORY INTERFERENCE MODELS	82
4.3	ASSUMPTIONS	85
4.4	MEMORY INTERFERENCE- ANALYSIS	86
	4.4.1 Uniform Reference Model	88
	4.4.2 Local Reference Model	97
	4.4.3 Partial Bus System	107
	4.4.4 Errors in Bandwidth Analysis	113
4.5	FUNDATMENTAL MEASURES	114
4.6	DELTA NETWORK	120
4.7	CONCLUSION	127
CHAPTER V	FAULT TOLERANCE	128
5.1	MULTIPLE-BUS SYSTEM	130
5.2	CROSSBAR SYSTEM	132
5.3	PARTIAL-BUS System	132
5.4	CONCLUSION	134
CHAPTER VI	CONCLUSIONS AND SCOPE OF FUTURE WORK	135
6.1	SUMMARY AND CONCLUSIONS	135
6.2	SUGGESTIONS FOR FUTURE INVESTIGATION	138
BIBLIOGRAPHY		141

LIST OF TABLES

Table 3.1	Variants of conservative policy	48
Table 3.2	Computation of G_j and $H(s,t)$ of Example 3.7	60
Table 4.1	BW of $n \times k \times z$ multiple-bus MPs obtained by simulation (URM)	90
Table 4.2	BW of $n \times k \times z$ multiple bus calculated from equation 4.7(URM)	91
Table 4.3	Percentage difference between calculated (Table 4.2) and simulated values (Table 4.1) of BW of a multiple-bus MPS (URM)	94
Table 4.4	BW of $n \times k$ crossbar MPS obtained from Eq. 4.8 with $r = 1.0$ (URM)	95
Table 4.5	BW of $n \times k$ crossbar MPS obtained from Eq. 4.8 with $r = 0.5$ (URM)	96
Table 4.6	BW of $n \times k$ crossbar with $r = 1.0$, URM (Theoretical, simulation and % error)	96
Table 4.7	BW of $n \times k \times z$ multiple-bus with $\alpha = 0.8$ (UBRM)	100
Table 4.8	BW of $n \times k$ crossbar MPS with $\alpha=0.8$ (UBRM)	101
Table 4.9	BW of $n \times k \times z$ multiple-bus MPS with $m = 0.8$ (NURM)	104
Table 4.10	BW of $n \times k$ crossbar MPS with $m = 0.8$ (NURM)	105
Table 4.11	Bandwidth of $n \times k \times z$ partial-bus MPS obtained from Eq. 4.15 with $r = 1.0$	111
Table 4.12	BW of partial bus: Comparison of analytical results with simulation results with $r=1.0$ (URM)	112
Table 4.13	Bus utilization in $8 \times 8 \times z$ configuration (URM)	119
Table 4.14	Inputs to switches of two stage Delta network	125

LIST OF FIGURES

Fig 1.1	A general structure of MPS	7
Fig 1.2	A general structure of MPS with local memory	7
Fig 2.1	Single bus multiprocessors organization	13
Fig 2.2	Multiple-bus multiprocessor organization	13
Fig 2.3	Partial-bus multiprocessors organization	14
Fig 2.4	Crossbar switch organization for multiprocessors	16
Fig 2.5	Multiport memory organization	16
Fig 2.6	2X2 Switching element (a) Straight connection (b) Crossed connection	18
Fig 3.1	An $n \times k \times z$ multiple-bus	32
Fig 3.2	An $n \times k$ crossbar multiprocessor	32
Fig 3.3	A k n -ported memory system	33
Fig 3.4	Graph representation of multiple-bus MPS	35
Fig 3.5	Graph representation of crossbar switched MPS	35
Fig 3.6	Graph representation of multiport memory MPS	36
Fig 3.7	Connection matrix 3X3 crossbar MPS	37
Fig 3.8	Connection matrix of 3-port memory MPS	37
Fig 3.9	Connection matrix of 3X3X2 multiple-bus MPS	38
Fig 3.10	Connection matrix of 2X2 crossbar system	43
Fig 3.11	Tableau associated with $H(X;s,t)$ computation using Theorem 3.1	50
Fig 3.12	Tableau associated with $H(X;s,t)$ computation using Theorem 3.3	52
Fig 3.13	Algorithm 3.3	53
Fig 3.14	Algorithm 3.4	54

Fig 3.15	Algorithm 3.5	57
Fig 3.16	Tableau associated with $H(X;s,t-s)$ computation using Theorem 3.1	61
Fig 3.17	Algorithm 3.6	63
Fig 3.18	Variation of MPS reliability on reliability of IN	72
Fig 3.19	Dependence of System reliability criteria on processor reliability	72
Fig 3.20	Variation of MPS reliability ($\alpha=2, \beta=1$) with time	73
Fig 3.21	Variation of MPS reliability ($\alpha=\beta=4$) with time	73
Fig 4.1	An $n \times k \times z$ partial bus multiprocessor with G groups	80
Fig 4.2	An $a^N \times b^M$ Delta network	81
Fig 4.3	BW of a $n \times k \times z$ multiple-bus Vs no. of buses with, $r=1.0$ & 0.5 (URM)	93
Fig 4.4	BW of crossbar Vs no. of memory modules with $r=1.0$ (URM)	98
Fig 4.5	BW of a crossbar Vs memory request probability (NURM, URM)	98
Fig 4.6	BW of a crossbar Vs memory access probability with $r=1.0$ (UBRM)	102
Fig 4.7	BW of a crossbar Vs memory access probability (NURM)	106
Fig 4.8	BW of $n \times k$ crossbar Vs no. of memory modules $\alpha=m=0.8$ (URM, UBRM, NURM)	108
Fig 4.9	Variation of BW of $16 \times 16 \times z$ MPS with number of buses (URM, UBRM, NURM)	108
Fig 4.10	Graph representation of partial bus MPS	109
Fig 4.11	Memory utilization of crossbar Vs no. of memory modules, $r=1.0$ (URM)	116
Fig 4.12	Memory utilization of $n \times k$ crossbar with $r=1.0$ (NURM) Vs m.	116
Fig 4.13	A $3^2 \times 2^2$ Delta network	121

NOTATIONS & ACRONYMS

ANSI	American National Standard Institution
BA	bandwidth availability
B	bus, $\{B_f\}$
b_f	the reliability of B_f and $\bar{b}_f = 1 - b_f$ its unreliability
BW	bandwidth
CP	conservative policy
DFP	data flow path
END	exclusive and mutually disjoint
EP	exhaustive policy
$E(\alpha, \beta)$	the event of an MPS with α -out-of- n processors updating α -out-of- k memory modules while execution of a task
$H(X; s, t)$	pr {atleast, t units out of s are good}, where $X = (x_i)$
$I_j^{t, l}$	refers to the j th input index of the l th module in the t th stage of a Delta network
I_j^t	refers to the j th input index at the t th stage of the Delta network
k	number of memory modules
m	the prob. with which processor access its favourite memory (NURM)
M	memory module, $\{M_j\}$, $1 \leq j \leq k$
MIN	multistage interconnection network
m_j	the reliability of M_j and $\bar{m}_j = 1 - m_j$ its unreliability
$m_j(\tau)$	$e^{-\lambda_{M_j} \tau}$, λ_{M_j} is the failure rate of the memory module j
MPS	multiprocessor system
MRP	memory request probability
n	number of processors

$n \times k$	a crossbar switch with n processors and k memory modules
$n \times k \times z$	a multiple-bus with n processors, k memory modules and z buses
NURM	non uniform reference model
O_i^t	refers to the i th output index at the t th stage of a Delta network
$O_i^t(l)$	refers to the i th output index of the l th switch at the t th stage of a Delta network
P_i	processor, $\{P_i\}$, $1 \leq i \leq n$
p_i	the reliability of P_i and $\bar{p}_i = 1 - p_i$ its unreliability
P_{ij}	prob. with which the i th processor access j th memory module, if already a request exists from it
$P_{ijf}(\tau)$	the prob. that i, j, f units of processor, memory modules and buses respectively are available at time τ
r_i	prob. with which the processor P_i makes a request for a memory access in a cycle; $r_i = r$ for statistically identical processor
$R(n, k)$	$H(X; n, k)$
$R_m(\tau)$	pr $E(2, 1)$, the multiprocessing reliability
$R[MMT](\tau)$	multisource - to - multi-terminal reliability
$R[MST](\tau)$	multisource - to - single-terminal reliability
$R[SMT](\tau)$	single source - to - multi-terminal reliability
$R[SST](\tau)$	single source - to - single-terminal reliability
$R_S(\tau)$	pr $E(1, 1)$, the system reliability
$R_{\alpha\beta}(\tau)$	pr $E(\alpha, \beta)$, threshold reliability
$R_u(\tau)$	$R_S(\tau) - R_m(\tau)$, uniprocessor reliability
S_{ij}	crosspoint switch connecting P_i and M_j
s_{ij}	reliability of S_{ij} and $\bar{s}_{ij} = 1 - s_{ij}$, its unreliability
$S_{ij}(\tau)$	$e^{-\lambda_{S_{ij}} \tau}$, $\lambda_{S_{ij}}$ is the failure rate of S_{ij}

UBRM unbalanced reference model
 URM uniform reference model
 VLSI Very Large Scale Integration
 X $\{X_i\}$, and its reliability set $\{x_i\}$
 x_i reliability of unit X and $\bar{x}_i = 1 - x_i$, its
 unreliability
 Z_j the muliport connected to M_j
 z_j reliability of Z_j , and $\bar{z}_j = 1 - z_j$, its unreliability
 $Z_j(\tau)$ $e^{-\lambda_j \tau}$, λ_j is the failure rate of Z_j

CHAPTER I

INTRODUCTION

The continuous development of computer technology supported by the VLSI revolution stimulated the research in the field of multiprocessors. The main motivation for the change over from conventional architectures towards multiprocessor ones is the possibility to obtain a significant processing power together with the improvement of price/performance, reliability and flexibility figures. Currently such systems are moving from research laboratories to real field applications. That multiprocessing is not just an intellectual curiosity but a technique of value in real life is clearly demonstrated by writing some of the available exploratory multiprocessor systems, such as, Cm, C. mmp, μ^* , PLURIBUS, TOMP, Intel 4321, etc., in addition to so many commercial multiprocessors. Further technological advances and new generation components are likely to further enhance the trend. Whatever one's favourite reason for multiprocessor systems, it is undeniable that MPS's will play an important role in computer systems of tomorrow.

1.1 PARALLEL PROCESSING

Whenever a computer designer has reached for a level of performance beyond that provided by his contemporary technology, parallel processing has been his only alternative. This point will be demonstrated in three ways: by the outline of historic achievements in device technology, with a projection on the

development of the architectural features, and by a review of taxonomic types into which the spectrum of parallel processors can be divided.

1.1.1. Device Technology

Computation speed has increased by order of magnitude over the past three decades of computing with a major share of increase in speed attributable to inherently faster electronic parts. In addition to the speed of the devices, the factors reliability, reductions in hardware cost and physical size have greater effect on enhanced performance. However, better devices are not the sole factor contributing to high performance. Today we can not obtain speed increases, as we have done in the past, by simply increasing the basic speed of the logic components ; we must necessarily take other approaches to increase computation speed.

1.1.2 Processing Techniques

The term parallel processing is used in a very general sense to cover methods that involve a deliberate attempt to increase speed by exploitation of concurrent events in the computing process. According to Hwang et al [1] concurency implies parallelism, simultaneity, and pipelining. Parallel events may occur at the same time instant; and pipelined events occur in overlapped time spans. Parallelism can be achieved in systems with one or more than one processing unit.

1.1.2.1 Parallelism in Uniprocessors

There are some hardware and software means to promote parallelism in uniprocessors. Hardware approaches emphasize resource multiplicity and time overlapping. The operating system software approaches to achieve parallel processing with better utilization of system resources [1]. However, there is a limit to the speed obtainable from a computer based on a single processor. The closer we approach this limit, the more rapidly does the cost of such a computer rise [2]. An alternative and radically different solution is to have a newer architecture, enjoying the time tested device of parallelism i.e. using a number of operating processors.

1.1.2.2 Parallel Computer Systems

State-of-the-art parallel computer systems can be characterized [1,3,4] into three structural classes : pipeline processors, array processors, and multiprocessors. The three parallel approaches to computer system design are not mutually exclusive.

Pipelined processor attains speed by dividing each processor unit into several stages. An analogy for a pipelined processor is the assembly line. Each of many sections is finely tuned to effectively perform one function on the object being assembled. Array processors consists of identical processing units under the control of a common broadcast unit. All processors perform the same operation simultaneously on different

data stored in their private memories.

Multiprocessor systems consists of several autonomous processors which can each execute separate programs. These systems can be categorized either as loosely coupled or tightly coupled. Multiprocessors which employ the shared memory interconnect approach, have been termed tightly coupled. In contrast, a loosely coupled system has disjoint primary memory address spaces and processing events do not share a common memory. A tightly coupled system, from here afterwards referred to as multiprocessor system or simply, MPS, generally requires synchronization between cooperating processors, whereas in loosely coupled system concurrent processes may be performed asynchronously.

1.1.3. Taxonomics

In literature, a wide variety of different architectures have been proposed and it is natural for us to seek a taxonomy which would permit us to grasp this diversity in simple terms. The taxonomy of Flynn [5,6], Feng [7], and Shore [3] have been discussed quite widely and some of the associated technology has become a part of the language of computer science.

Flynn's classification is based on multiplicity of instruction streams and data streams. He proposed four types of computer architectures such as: single instruction stream, single data stream (SISD), single instruction multiple data stream (SIMD), multiple instruction single data stream (MISD) and

multiple instruction multiple data stream (MIMD). Feng classifies according to word length, i.e., the number of bits which are processed in parallel in a word, and the number of words which are processed in parallel. For example, a system which contains 'n' processors each with a word length 'b' bits is represented by (n,b). Shore based his classification on how the computer is organised from its constituent parts. Handler [8] compares several alternative ways of describing the architecture and introduces a more comprehensive scheme based on degree of parallelism and pipeline. His classification makes distinction at three processing levels: program control unit, arithmetic and logic unit and elementary logic circuitary.

1.1.4 Multiprocessor Systems

An MPS may be viewed as a logical result of the efforts to increase computer performance by exploiting parallelism in hardware [1,3,4,9-13]. It is relatively easy to enumerate the capabilities that must be provided by the hardware based on the fundamental definition of an MPS.

Definition 1.1

ANSI [14] defines a multiprocessor as : " A computer employing two or more processing units under integrated control". The definition does not exclude future developments in computer architecture, but does not seem to have had any impact on contemporary architecture. Subsequently, Enslow [4] suggested a more detail definition which included :

1. two or more processors having access to common memory, where private memory is not excluded,
2. shared I/O,
3. a single integrated operating system,
4. hardware and software interactions at all levels,
5. the execution of a job must be possible on different processors, and
6. hardware interrupts.

1.1.4.2 System Structure

A general model of the hardware system is shown in Fig.1.1. A set of processors and a set of memory modules are connected by means of an interconnection network (IN). More generally the processors are allowed to issue an access request to the IN in order to perform data transfer; the memory modules receive the request for an access from the processors and can accept and honour them. There are two sources of conflicts due to memory requests. First, more than one request can be made to the same memory module. Second, there may be more than one request through the same communication link of the IN to different memory units. To resolve these conflicts each IN is associated with an arbitration mechanism [1]. One way to reduce the number of processor-memory requests, and hence the IN traffic, is to have a local memory (cache) associated with each processor, as shown in Fig. 1.2 [10].

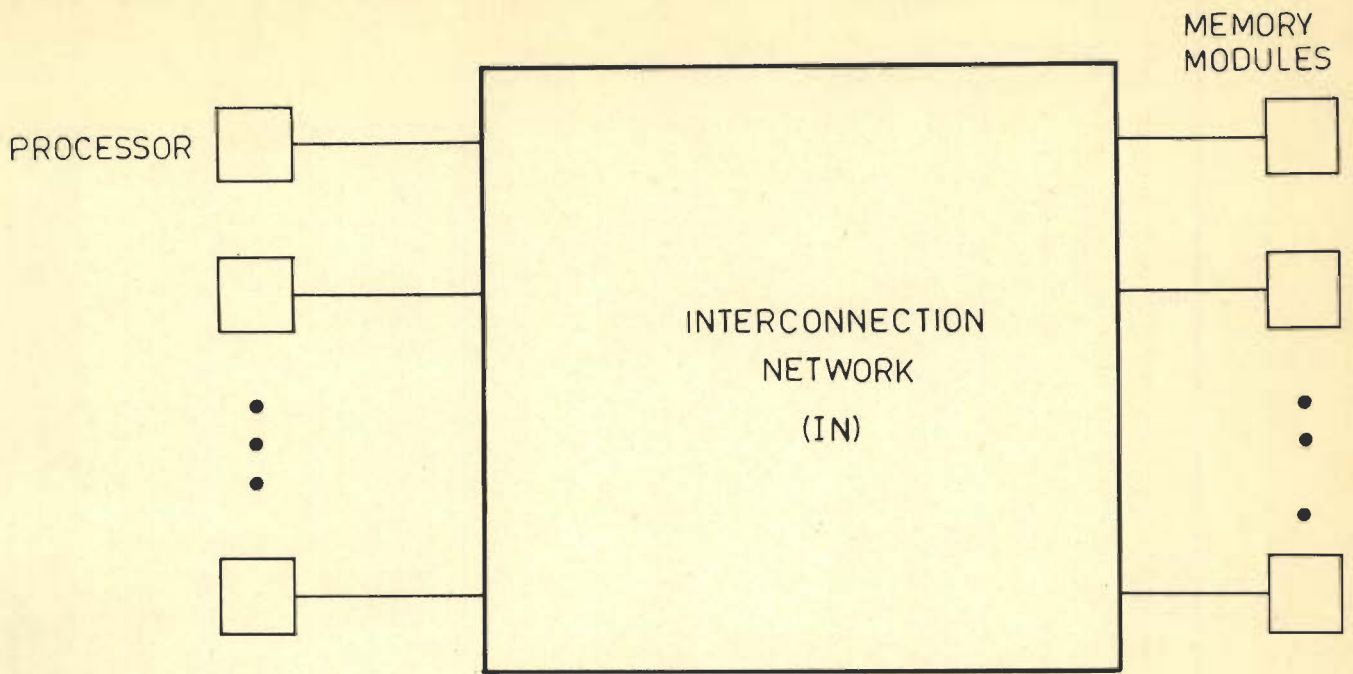


FIG.1.1 A GENERAL STRUCTURE OF MPS

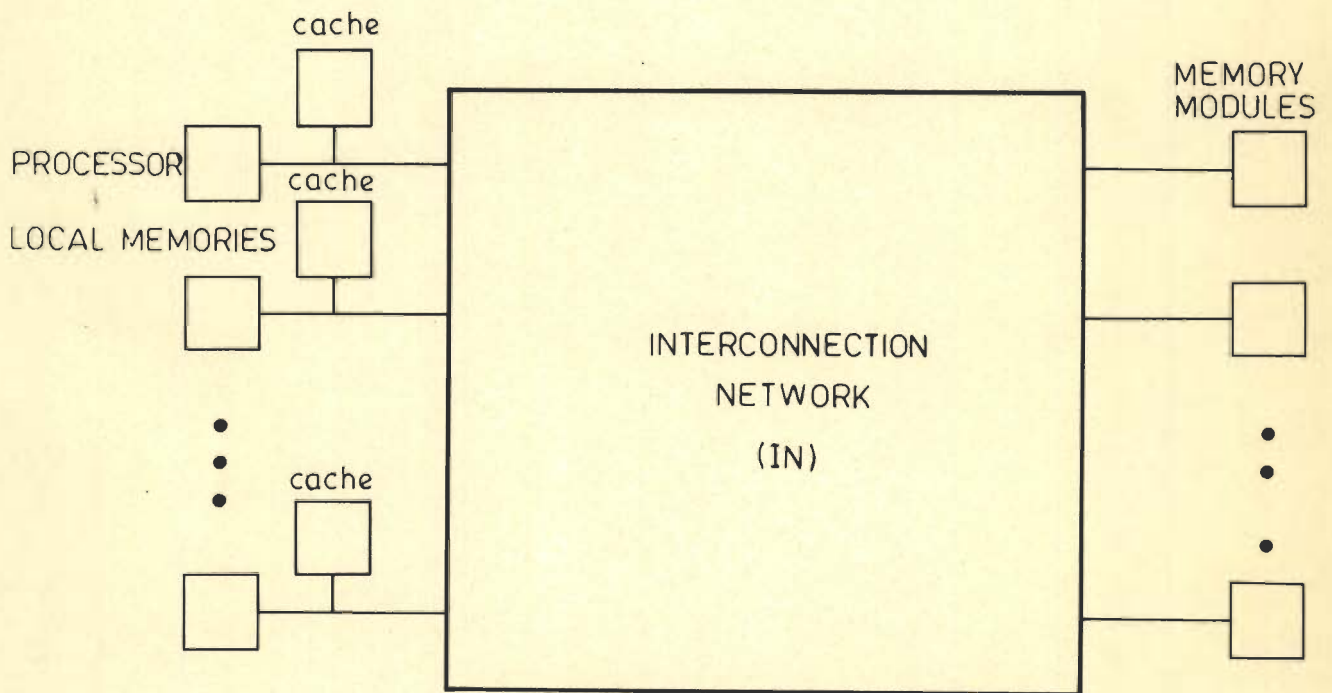


FIG.1.2 A GENERAL STRUCTURE OF MPS WITH LOCAL MEMORY

1.2. PROBLEM FORMULATION

The major design problems in an MPS are :

- a) Network topology
- b) Control strategy
- c) Fault tolerance
- d) Performance evaluation
- e) Reconfiguration techniques

Topology design relates to interconnection structure between processors and memory modules. Control strategy concerns how to route data from a source (processor) to various destinations (memory modules). Reconfiguration techniques are used to allocate hardware resources, such as processors and connection switches. When a task requests resource through a system controller or to provide fault tolerance when the current configuration contains faulty components. Performance analysis must be performed to observe the characteristics of the hardware resources, which in turn will be used to decide whether the resources are adequate in regard to system requirements.

A fault tolerant MPS can tolerate faults to some degree and still provide reliable and gracefully degradable communication between processors and memory modules. The reliability and bandwidth availability are the two important measures of fault tolerance. Out of these, the problems of performance evaluation and fault tolerant aspects of design are considered for different IN topologies, in this dissertation.

1.3 STATEMENT OF THE PROBLEM

This thesis attributes itself to the problem of designing an interconnection structure considering both performance and fault tolerant issues. The object is to find efficient mathematical models that represent the system completely and accurately so that the performance of the multi-processors could be compared under different operating conditions. To define the specific tasks which the present work is intended to cover, a list of major subproblems for the present study is given below.

- i) To examine the different interconnection networks, such as multiple-bus, crossbar, multiport memory system and MINs, considered for design.
- ii) to develop time and memory efficient reliability algorithm for a 't-out-of-s' redundant system, which is used as a modeling tool for the analysis of MPS's.
- iii) to obtain closed form solutions for the computation of memory interference, and hardware resources utilizations in an MPS so as to study the parameter dependence on performance. The parameters of interest are processors, memory modules, memory request probability and memory access probabilities.
- iv) to find out the responsiveness of the system for a given input request.
- v) to study the fault tolerant behaviour of the system such as reliability and bandwidth availability as these

specifications are crucial factors in the design of current and future computer systems, especially fault tolerant MPSs.

- vi) to study the performance degradability due to system component failures during the execution of a task.

1.4 OUTLINE OF THE THESIS

Chapter II contains a general description of some existing multiprocessor architectures and characteristics of the interconnection networks. Chapter III analyzes the multiprocessor systems for reliability computation and the reliability modeling using t-out-of-s systems. Chapter IV provides tools for evaluation of performance of MPSs assuming variety of performance indices. Chapter V considers the fault tolerance and performance degradation of non failure-critical multiprocessor models. Finally, chapter VI concludes the work carried out in the thesis along with some proposals for further research in this area.

CHAPTER II

DESIGN OF MPS - AN OVERVIEW

Since processors and memory modules are available as standard integrated circuits, the key design problem is how to put them together in an efficient and reliable way. Therefore, the design of interconnection structures requires special attention. The major design issues, in an MPS, considered are performance evaluation and fault-tolerance. Since the design of any system in its current state of development is as much an art or skill as it is a science, this chapter has two purposes: first to present those aspects of system evaluation criteria that are under consideration; and second to review the associated evaluation techniques. Section 2.1 gives an overview of topological aspects of interconnection networks. Section 2.2 and Section 2.3 discuss the modeling techniques and evaluation criteria for determination of performance. The fault-tolerance considerations for analysis are discussed in section 2.4. Section 2.5 concludes the chapter.

2.1 INTERCONNECTION NETWORKS

Focussing on the flow of data and parallelism that can be obtained with a multiprocessor, it is indeed the topology and mode of operation of the network that interconnects the functional units that becomes of paramount importance. Four main types of system organizations are possible for the processor-memory switching apparatus [1,4,15]. All the four topologies are

regular and can be classified as either static or dynamic.

The system organization to be covered and the discussion on each assume that the entire system is at one physical location within distances so that unit to unit transfer can be made at full machine speed.

2.1.1 Time Shared Multiple-Bus

There are several degrees of complexity in the system organization depending on the number of buses. The simplest one is to have all processors and memory modules connected to a single bus which can be totally passive (ref. Fig. 2.1). It is obvious that there is no possibility for concurrent transactions in the organization. To attain more parallelism, at the price of more complexity, one can have several buses as shown in Fig. 2.2. Priorities can be given to specific units if one adds a bus and a memory arbiter to resolve the conflicts due to bus and memory interferences respectively [16]. In this case, the interconnection subsystem becomes an active device.

A modification of the multiple-bus multiprocessor that has been proposed by Lang et al [17] to provide better cost-effectiveness, is known as the partial-bus architecture. Fig. 2.3 depicts a partial-bus system. The memories and buses are divided into a number of groups. All processors are connected to all buses, whereas each group of memory modules is connected to one set of buses only [18].

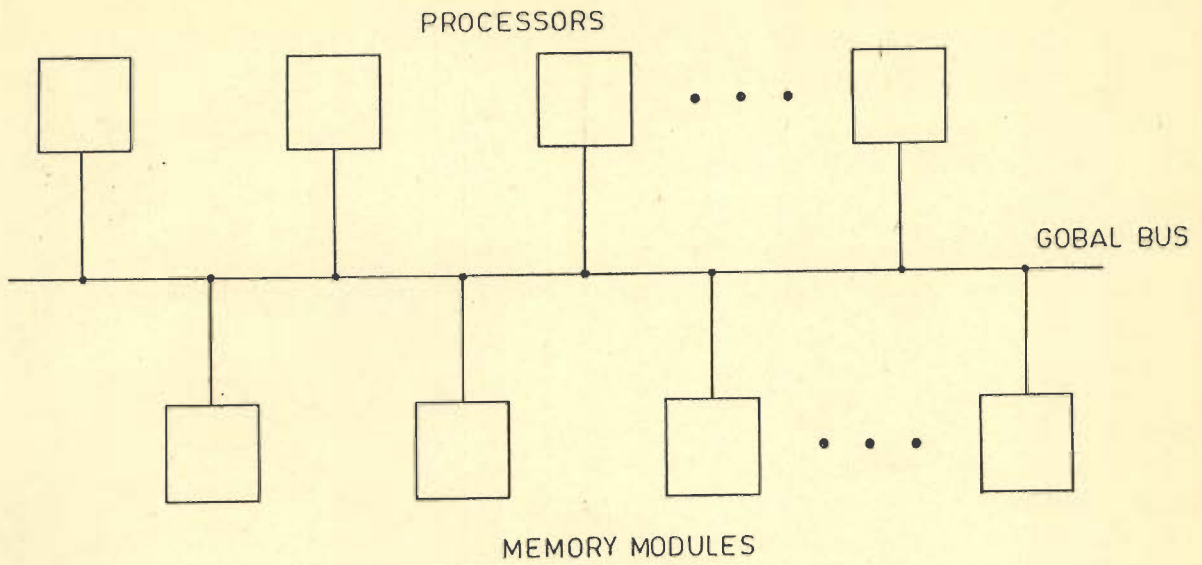


FIG.2.1 SINGLE BUS MULTIPROCESSORS ORGANIZATION .

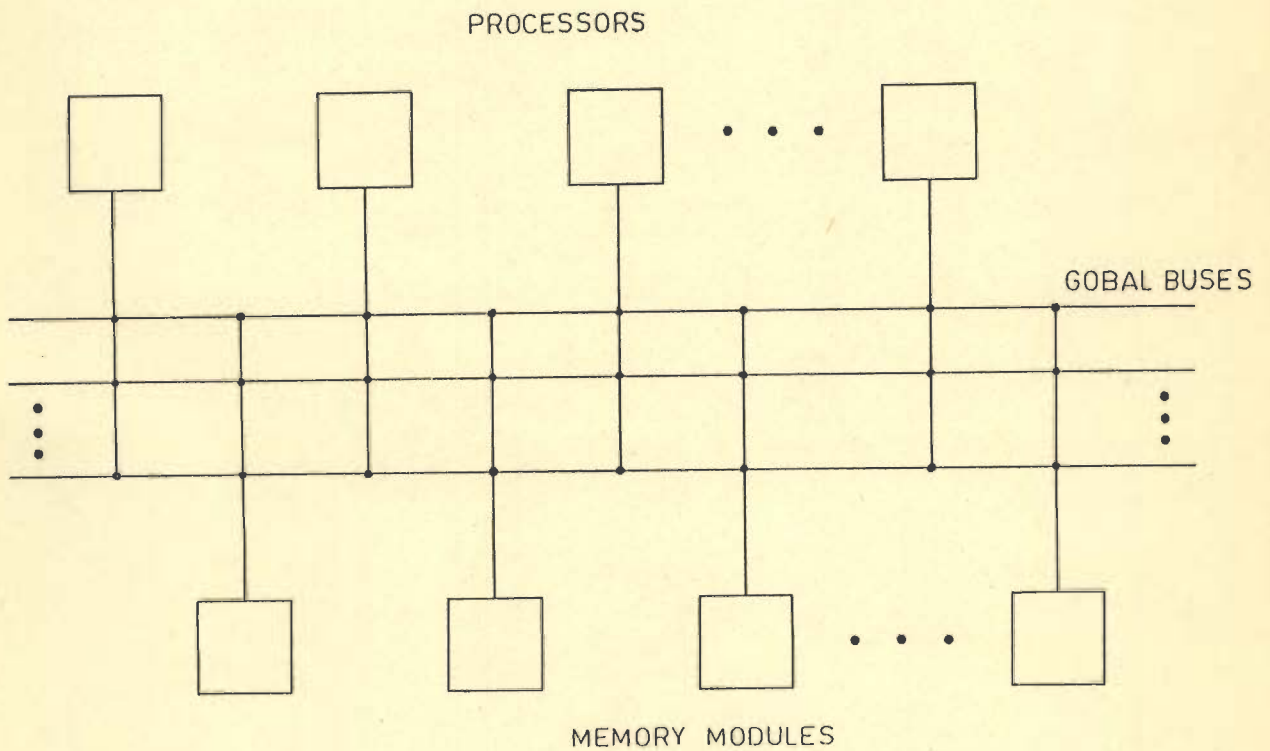


FIG.2.2 MULTIPLE - BUS MULTIPROCESSOR ORGANIZATION .

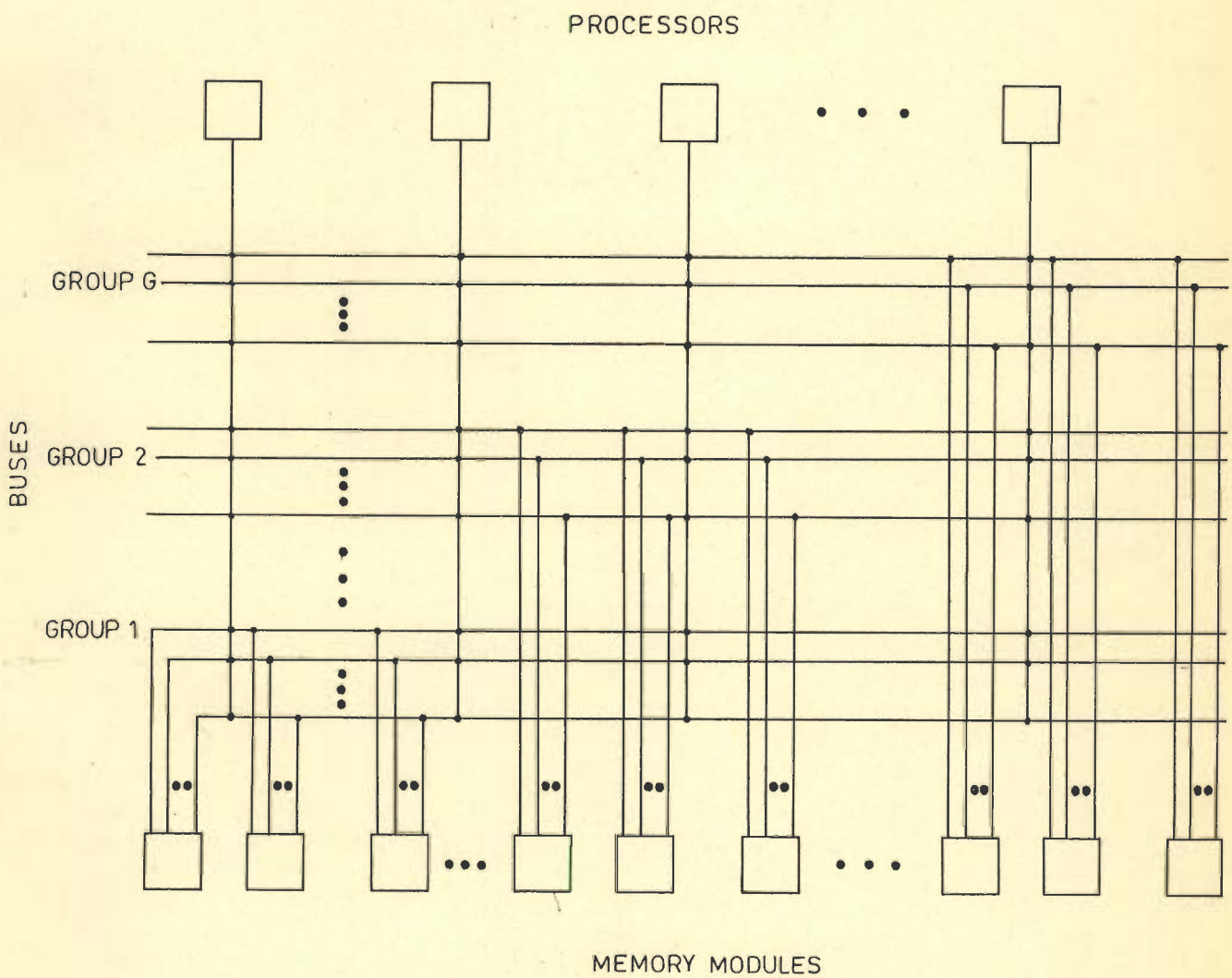


FIG. 2.3 PARTIAL - BUS MULTIPROCESSORS ORGANIZATION.

2.1.2 Crossbar Switch

If the number of buses in a time shared bus system is increased, a point is reached at which there is a separate path available for each memory module, as shown in Fig. 2.4. This interconnection network is called a nonblocking crossbar. This is the most extensive and expensive scheme since each crosspoint must have hardware capable of switching parallel transmissions and of resolving conflicting requests for a given memory module. The switching device becomes rapidly the dominant factor with the increase in complexity in the cost of overall system.

2.1.3 Multiport Memory System

In this organisation (ref., Fig. 2.5) the switching is concentrated in the memory module. Each processor has access through its own bus to all memory modules and the conflicts that occur if two or more processors request access to the same memory module are in general resolved through hardware fixed priorities [19].

2.1.4 Multistage Interconnection Networks (MINs)

They are well suited for communication among tightly coupled system components, and offer a good balance between cost and performance. Design, analysis, and development of MINs during the last decade have made them the most current technology [20-23].

A MIN consists of more than one stage of switching elements and is usually capable of connecting an arbitrary

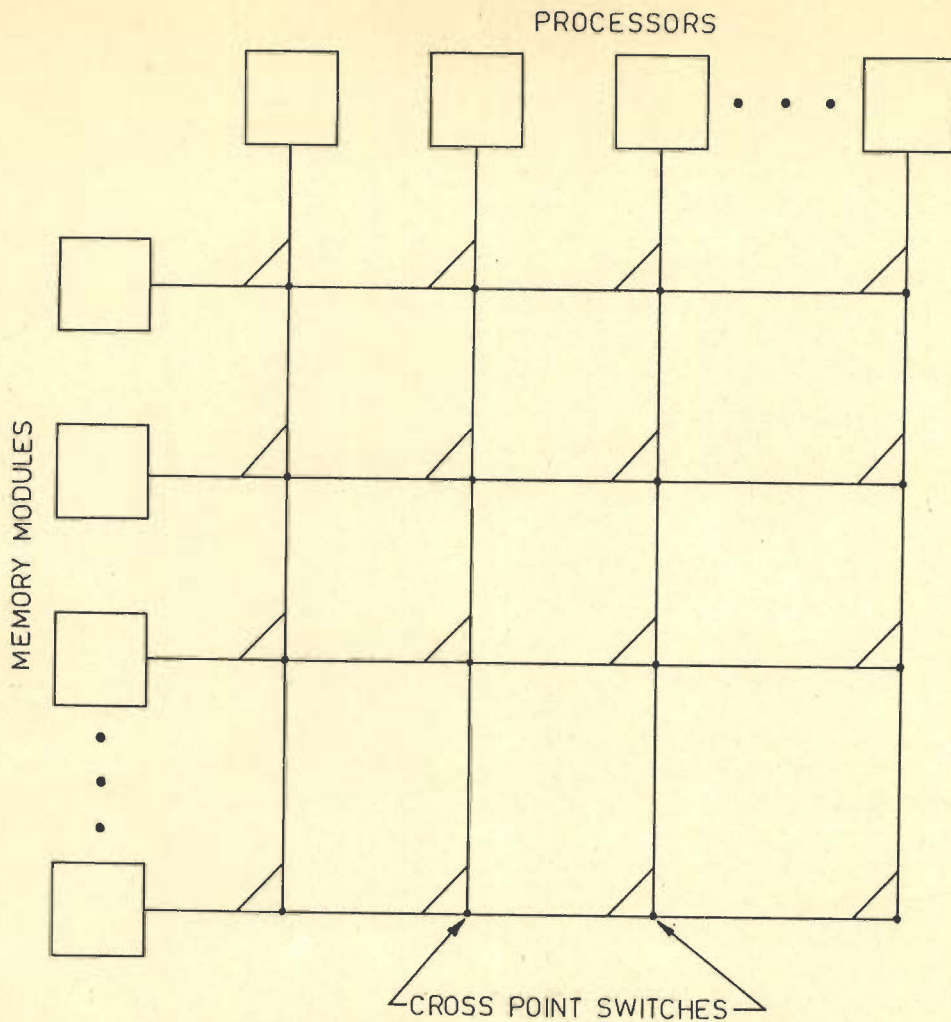


FIG. 2.4 CROSSBAR SWITCH SYSTEM ORGANISATION FOR MULTIPROCESSORS

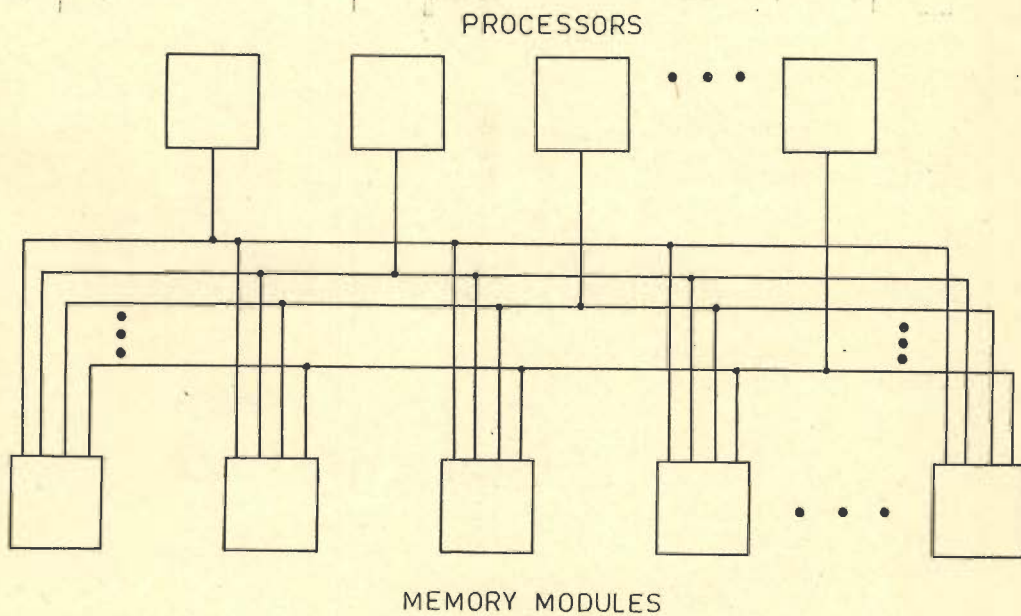


FIG. 2.5 MULTIPORT MEMORY ORGANIZATION .

processor to an arbitrary memory module. Each switching element can perform a very simple circuit switching function. Consider, for example, the basic 2 x 2 switching element shown in Fig. 2.6. The switching element can be set in two configurations performing a straight connection, as shown in Fig. 2.6(a) and crossed connection, as shown in Fig. 2.6(b). Generally, a multistage consists of N stages where $n = 2^N$ is the number of inputs and outputs. A matrix of $n \log_2 n$ basic switching elements can interconnect the set of input terminals to the set of output terminals. The interconnection patterns from stage to stage determine the network topology. In all these cases a convenient setting of basic switching elements can connect any input terminal to any output terminal.

2.1.5 Operational Characteristics

In selecting the architecture of an IN, three design decisions can be identified [20, 24]. These are based on its timing, switching and control characteristics.

The timing control of an IN can be either synchronous or asynchronous. An IN transfers data using either circuit switching or packet switching. Based on control strategy, an IN may be classified as centralized or decentralized. In centralized control, a global controller receives all requests and transmits the messages in the IN. In the decentralized system, requests are handled independently by different devices in the IN. These three operational characteristics with the

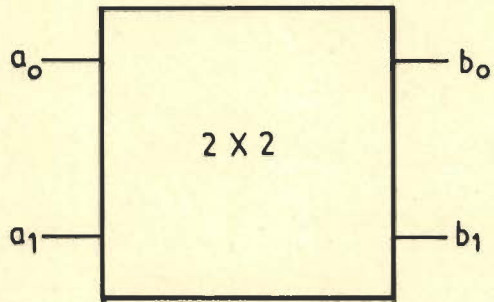
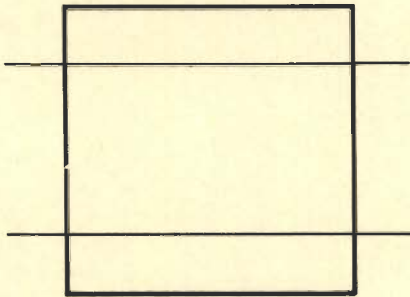
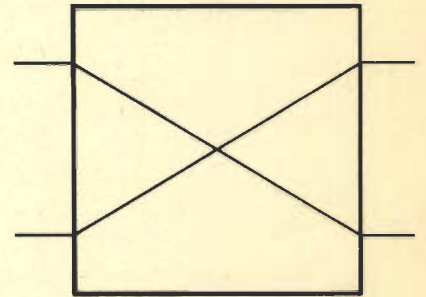


FIG.2.6 2x2 SWITCHING ELEMENT



(a)

(a) STRAIGHT CONNECTION



(b)

(b) CROSSED CONNECTION

topology define an IN.

2.2 SYSTEM EVALUATION

System evaluation is an essential activity in all branches of engineering. Any system which is being designed must satisfy certain specifications. Design methodologies and evaluation procedures are issued by designers to obtain system to meet these specifications [25, 26].

2.2.1 Evaluation Techniques

The most popular evaluation technique can be classified into two categories : measurement techniques, and modeling techniques. There are two major problems with measurements on the system [27]. First, measurement is not feasible in the design and development stages of the system ; the system is not measurable if it is not operational. Second, measurement of most systems is a complex activity which involves considerable human and machine cost. Modeling is the alternative when measurement is intractable.

There are two major approaches of modeling systems: simulation and analytical modeling. Simulation models are more prevalent in practice because they represent aspects of the modeled system more faithfully than analytic models. However, simulation models are very expensive to use, and the results of the simulation are harder to interpret [27].

Recent advances in modeling techniques are making

analytic models increasingly capable of representing more and more aspects of the modeled system. This has promoted an increased interest in the use of analytic modeling in the design of computer systems. Modeling of a system is a two phase effort. The analysis phase is to evaluate the performance criteria of the system by considering the system architecture, work load etc. The second phase of modeling effort is the design phase.

There are at least two types of performance measures ; Deterministic and Probabilistic.

In a deterministic model all variables are deterministic. If atleast one of the variable is random, the model is said to be stochastic or probabilistic. Deterministic approach is applied to worst-case or best-case studies [28,29]. Because of their potential for representing complex, highly variable phenomena in a relatively compact way, probabilistic models are much more widely applied to the study of computer systems than deterministic models.

Probabilistic Models

These models are useful when the behaviour of the system is not predictable in deterministic fashion. The role of the probability theory is to analyse the behaviour of the system assuming the given probability assignments and distributions. The results of this analysis is as good as underlying assumptions. Probability models tend to fall into one of the three classes: i) Markov Chain , ii) Queuing theory and iii)

Combinatorial theory.

2.2.1.1 Markov Models

State and state transition are the two central concepts of Markov models. The state of the system represents all that must be known to describe the system at any time i.e., it concentrates on the rate at which transition takes place between different states and then use this information to determine the probabilities that the system is in each of these states at any given time [29-31]. Markov theory is the basis of elementary queue theory [32-33].

Drawbacks of Markov Models

- i) Unmanageable large state space to represent an open queue network. So this approach applies primarily to closed networks [25, 34].
- ii) In Markov modeling, using continuous parameter and discrete-parameter analysis, the processor time distribution in each state is exponential and geometrical respectively. This is in many situations an unrealistic assumption [25,30]. Continuous time models are usually less accurate than discrete time models when discrete time events are considered.
- iii) Semi-Markov process, a generalization of a Markov process, allows state durations to have arbitrary distributions. This allows certain states that can not occur in the real system and hence the results are prone to errors [34].

2.2.1.2 Queuing Models

A queuing model is defined by its sources, its service centres and their interconnections. Jobs are generated by the sources. Servers are generally used to model the resources demanded by the jobs. The interconnections specify the paths which the jobs are allowed to flow in their journey through the model from centre to centre. Each server has at least one queue. Queuing theory encompasses the set of analytic models that most adequately describe computer systems. On the other hand, a discouragingly large fraction of practical queuing systems continue to elude exact analysis. Some researchers find these two observations a pessimistic commentary on work in queuing theory [11].

2.2.1.3 Combinatorial Models

Combinatorial models attempt to categorize the set of operational states of a system in terms of the functional states of its components, in such a way that the probabilities of each of these states can be determined by combinatorial means [31]. The commonly used combinatorial models are fault tree, reliability block diagrams, Sterling numbers, and t-out-of-s redundant systems. If the components are s-dependent reliability block diagrams with series-parallel (well nested) structure and fault trees, without repeated nodes can be analyzed using linear time algorithms [35]. However, if the component states are not s-dependent or the structure is not series-parallel the analysis can not in general be done using linear time. Methods for

analysing such structures use sterling numbers and t-out-of-s system concept with conditioning (using theorem of total probability). Methods based on Sterling numbers [36] are computationally complex.

2.2.2 t-out-of-s system : Modeling

The t-out-of-s systems are a generalization of the series-parallel model. However, instead of requiring one of the s units for the system to function t units are required. If the units are assumed to be identical, all the 't' states need not be enumerated. Any combination of the t of the s units is enumerated by s take t combinatorial coefficients, denoted by $\binom{s}{t}$ [29,30]. It makes no difference whether the problem is defined in terms of s-dependent or s-independent component probabilities. Each term is the product of a number of component probabilities. If they are s-dependent between some of the events presented in a term, they can be accounted for by standard representation of conditional events [37]. In Chapter III, the reliability computation of t-out-of-s systems is considered.

2.3 PERFORMANCE EVALUATION CONSIDERATION

Performance evaluation measures the capacity of IN of a MPS in terms of parameters that represent major characteristics of an IN application model. As no single parameter can give a truly accurate measure of systems performance, different parameters are needed to characterize a system for a given application. A wide variety of performance criteria is available

in literature. In this present work an attempt has been made at evolving methodologies for the design of an MPS by considering the following criteria.

- i) processor-memory interference [17,34,39-74]
- ii) response time [42,47,68,73]
- iii) hardware utilizations [34,42,60,68,73]

Sharing of memory modules between multiple tasks results in memory interference. This interference may be quite severe in MPS where memory modules are shared by a number of independent processors through INs. The combined effect of interference due to IN contentions and that due to memory conflicts is investigated in Chapter IV.

Response time measures, sometimes also called waiting time, describe the length of time from a request for service until the request is completed.

A measure closely related to throughput is utilization, that is, the fraction of a time a specified component (processor, memory module or bus) is busy. Utilization is an even more direct indicator of capacity of the system being used. The definitions and analysis of these performance criteria are also discussed in Chapter IV.

2.4 FAULT-TOLERANCE SYSTEM CONSIDERATION

Fault-tolerant systems may be either non-degradable or degradable. In case, a component fails non-degradable systems are

able to recover by bringing up another standby component and continue operation, almost transparently, with the same computing power, i.e., system operation is not degraded by failures. Degradable systems on the otherhand operate in a degraded mode as a consequence of component failures [75]. For the performance evaluation of a degradable fault tolerance system, the important parameters of interest are reliability and bandwidth availability.

2.4.1 Reliability

Real time systems fall into two classes. Failure-critical and non failure-critical. In the former class of systems no down-time can be tolerated, that is, any failure during a specified interval (mission time) is construed as a mission failure. Reliability defined as the probability that no failures (at all) occur during the mission time, is an appropriate measure for this class of systems. This reliability measure and its associated measures such as multiprocessing reliability and multiterminal reliability are of great interest to the system designers and are discussed in Chapter III.

In the non failure-critical models, some down-time can be tolerated. Failures are allowed to occur as long as some constraints are met [38]. An interesting study of these systems is the computation of bandwidth availability of the system.

2.4.1.1 Multiprocessing Reliability

The multiprocessor approach for executing a job

introduces new requirements: first each job must be partitioned into tasks; second, each task must be scheduled for execution on one or more processors; third, each task may require to access one or more modules through atleast one of the interconnection links. In a real situation the components of multiprocesor fail at random. If a task needs at least I processors and J memory modules, different reliability criteria can be defined depending upon the availability of I and J [76].

When $I = 2$, and $J = 1$, the reliability criterion is referred to as multiprocessing reliability and when $I = J = 1$, it is the system reliability.

Sections 3.1 , 3.3 and 5 discuss the analysis of multiprocessing reliability.

2.4.1.2 Terminal Reliability

These criteria find their use in packet switching communication. A meaningful measure of the reliability and availability of a communication network is the terminal reliability between the source and destination. One of the communication systems of recent interest is the MPS, where a several processors (sources) connected to a set of memory modules (destinations) through an IN (channel) [77].

Various measures for probabilistic networks have been analyzed in literature [78]. They are :

- a) a source communicates with a destination (SSI reliability)

- b) a source communicates with a number of destinations (SMT reliability)
- c) a set of sources communicate with a destination (MST reliability)
- d) a set of sources communicate with a set of destinations (MMT reliability)

The analysis of these reliability criteria is considered in Section 3.3.2.

2.4.2 Bandwidth Availability (BA)

The bandwidth availability of a gracefully degrading multiprocessor is the expected value of available BW in the system at any time ' τ '. Usually in the memory interference measurement, the BW is computed by assuming non failure-critical models. The failures of the components in failure-critical models degrades the performance of the system. The simultaneous consideration of both the performance and fault-tolerance issues leads the way to the computation of BA [59].

2.4.3 Evaluation Techniques-Review

Many algorithms have been proposed for communication networks and a summary of these techniques can be found in [79]. These known methods can be classified as follows:

- 1) Decomposition techniques [80-82]
- 2) Graphical approach [76,83,84]
- 3) Enumeration methods [76,84-88]

2.4.3.1 Decomposition Techniques

The most common approach to modeling complex systems consists of structurally dividing the system into smaller subsystems (eg., processor, memory module and IN) analyzing the dependability of subsystems separately and then combining the subsystem solutions. If the subsystems fault tolerant behaviours are mutually s-independent, then a decomposition into subsystem, separate analysis of subsystems and aggregation to obtain final solution can be used [89].

2.4.3.2 Graphical Approach

The reliability analysis of MPS using decomposition technique or series parallel approach can be made very easily using graph models. A probabilistic graph contains a finite set of weighted vertices connected by undirected edges. Each vertex, corresponding to a physical component in the MPS, is weighted by its reliability of functioning correctly. All component reliabilities are represented in vertices. The edges are used only to specify incidence relations between vertices [76].

2.4.3.3 Enumeration Methods

The simplest technique for reliability evaluation is to enumerate the favourable states by examining all the elementary events. The events can be expressed using exhaustive policy (EP) or conservative policy (CP) (discussed in Chapter III). Using CP only $(m+1)$ exclusive and mutually disjoint (EMD) events are generated as against 2^m EMD terms by

EP to represent an m-variable system completely. Algorithms based on CP produces more compact results than EP and hence preferred for large systems.

Algorithms based on pathset and cutset enumeration with reduction to mutually exclusive events are generally efficient and produce compact expressions. In path (cutset) enumeration methods, the terminal reliability expression is obtained by enumeration of all simple paths (all prime cutsets) between a pair of terminal nodes which represent a complete favourable (unfavorable) non-disjoint events. To obtain reliability either the inclusion-exclusion [90], probability theory or more efficient boolean algebra [80] can be utilized for mutually exclusive events.

In all the algorithms, the most time consuming step is the disjoint process to obtain the mutually exclusive events which is required in each step of the algorithm. Chapter V considers the techniques cited above for computing the reliability criteria of an MPS.

2.5 Conclusion

This chapter reviews some design aspects of an MPS. We have typically considered topological performance evaluation and fault-tolerance to highlight the issue. A review of the evaluation techniques is also presented. The analysis of these criteria is carried out in detail in the chapters to follow.

CHAPTER III

MULTIPROCESSOR SYSTEM RELIABILITY

As discussed earlier, two criteria namely multiprocessing and terminal reliability, of failure-critical models are considered for performance analysis of MPSs. Because of various reasons mentioned in [88] exact/symbolic methods found their use more important than approximate techniques. The algorithms for exact reliability analysis can be divided broadly into two categories depending upon their approach in attacking the problem [84]. The algorithms of the first category are based on primary method of starting from the enumeration of multiprocessing events and compute the reliability through one of the techniques discussed in Chapter II. The other category is modeling.

We discuss in Section 3.1 the computation of multiprocessing reliability through path enumeration. The analysis of t-out-of-s systems which have been used later for modeling of MPSs, is presented in Section 3.2.

The reliability evaluation of MPS is done based on the following assumptions.

- a) Each unit and the system is either good or failed.
- b) There is no repair.
- c) The states of all units need not be mutually statistically independent.
- d) Sensing and switching of failed units out of the system is

perfect.

- e) The system is good (bad) if and only if atleast t of its s units are good (bad).
- f) Reliability of each unit is known and the units of the system are contiguously numbered.

The processors are connected to the memory modules through any of the INs shown in Fig. 3.1 through 3.3 . Fig. 3.1 shows an $n \times k \times z$ multiple-bus architecture having n processors P_1, P_2, \dots, P_n , k memory modules, M_1, M_2, \dots, M_k , and z buses B_1, B_2, \dots, B_z . When $z = k$, the system behaves as an $n \times k$ crossbar, as depicted in Fig. 3.2. The processor P_i is connected to memory modules M_j through a separate crosspoint switch S_{ij} , for all i , and j , $1 \leq i \leq n$ and $1 \leq j \leq k$. In multiport system, shown in Fig.3.3, the memory module M_j is connected to multiport Z_j .

The topology of a network is the pattern of connections in its structure. This can be represented by a graph, where processors form the source nodes and memory modules the destinations. The topology is determined by the pattern of links connecting sources and destinations. Different INs are often compared graphically because comparison by topology is independent of hardware. Nodes in the graph of a IN can be numbered along with the sources and destinations and then a IN can be described in terms of the algebraic relations among the nodes. The algebraic model is useful in discussing control and communication routing strategy. The probabilistic-graphical

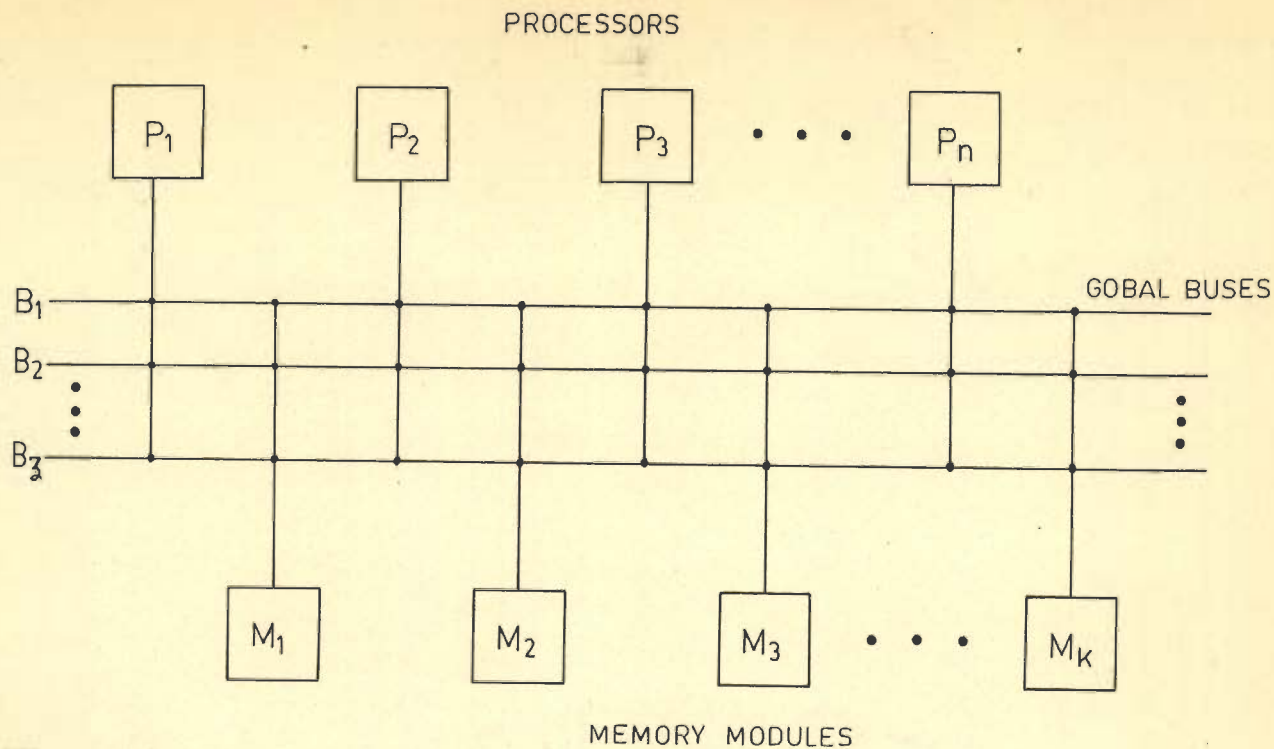


FIG. 3.1 AN $n \times k \times z$ MULTIPLE-BUS

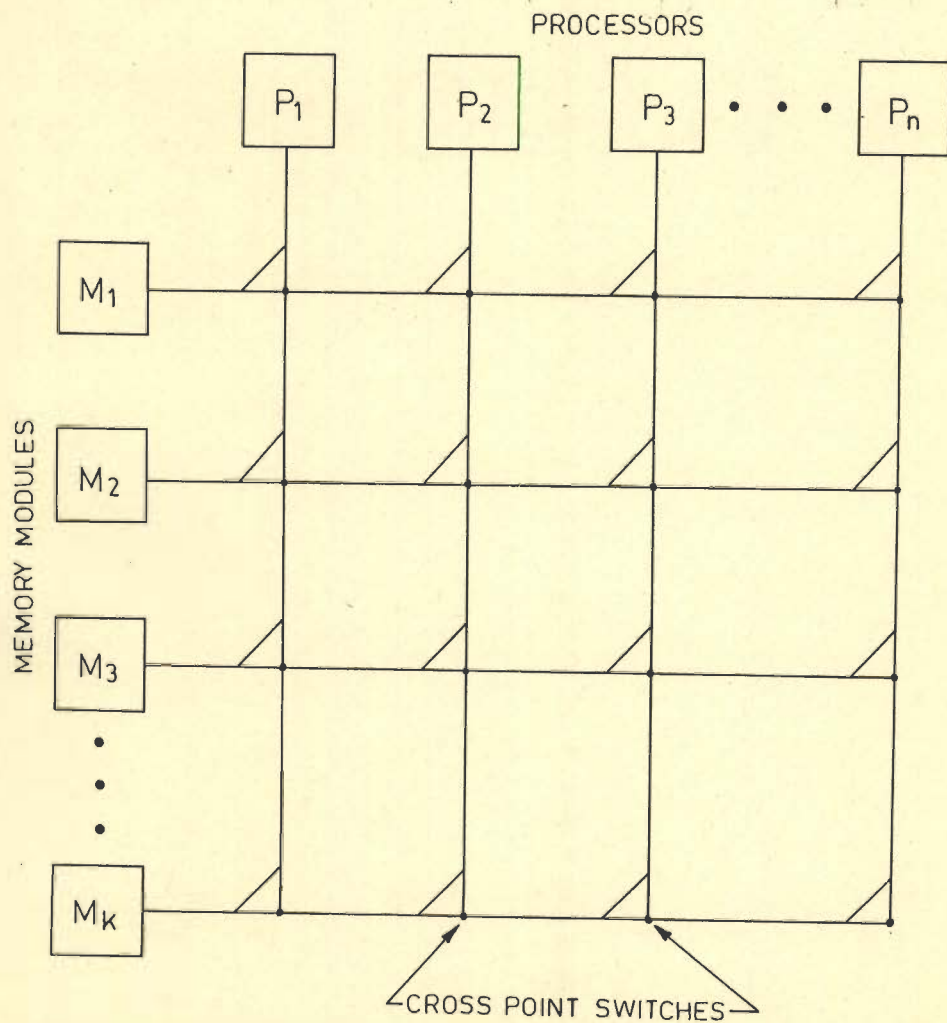


FIG. 3.2 AN $n \times k$ CROSSBAR MULTIPROCESSOR

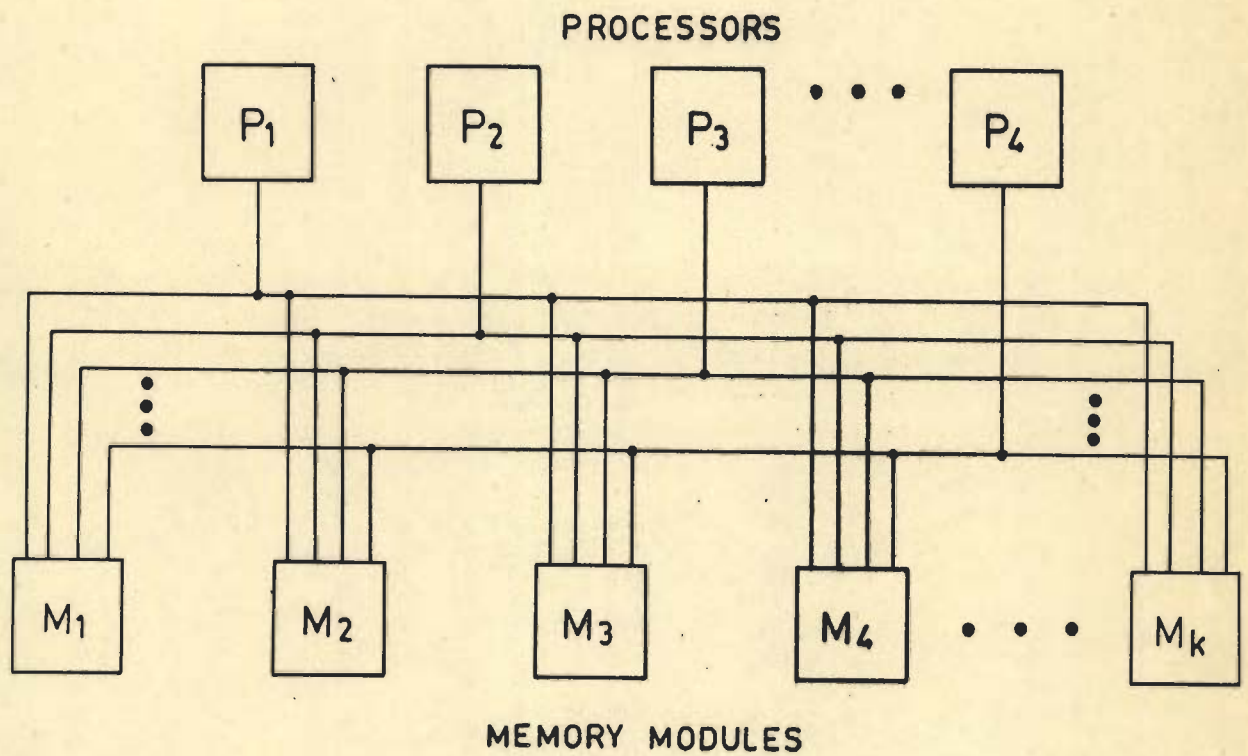


FIG. 3.3 A k n -PORTED MEMORY SYSTEM.

representation of three architectures, multiple-bus, crossbar and multiport are shown in Figures 3.4 through 3.6 respectively [76].

3.1 MULTIPROCESSING RELIABILITY

Assuming the multiprocessor systems under study use several processors in parallel to solve one problem, i.e., it uses a task oriented approach. The analysis of multiprocessing reliability criterion is considered below through a primary method based on path enumeration technique.

3.1.1 Primary Method

For determination of multiprocessing reliability it is necessary to compute all events that give rise to multiprocessing. These events can be easily obtained from the connection matrix of the system [91-92].

3.1.2 Construction of Connection Matrix

The connection matrix, C_n , for an MPS with n processors, is an n -row matrix. Each row of C_n corresponds to a processor; the i th row is thus labelled P_i , $1 \leq i \leq n$, for the i th processor. The number of columns in C_n depends upon the IN used. The connection matrix lists possible data flow paths (DFP's) between processors and memory modules. The construction of the connection matrix for each of the three INs is described below.

a) Crossbar MPS

In the crossbar switched MPS, there is one unique path

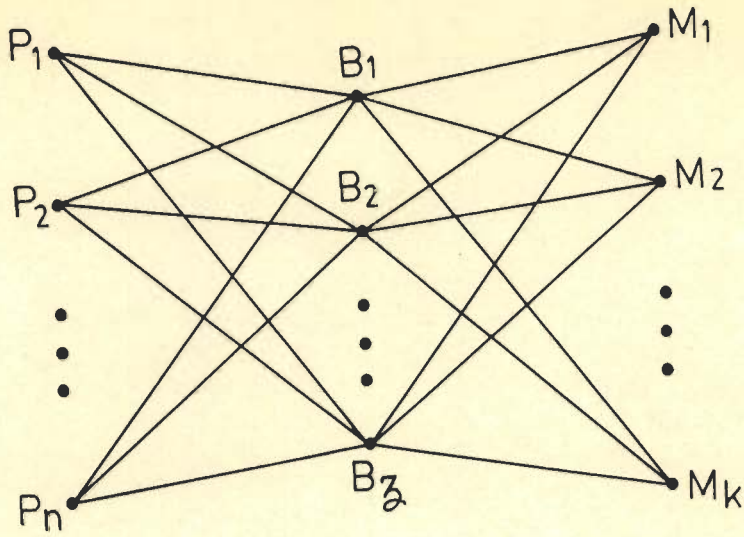


FIG. 3.4 GRAPH REPRESENTATION OF MULTIPLE-BUS MPS.

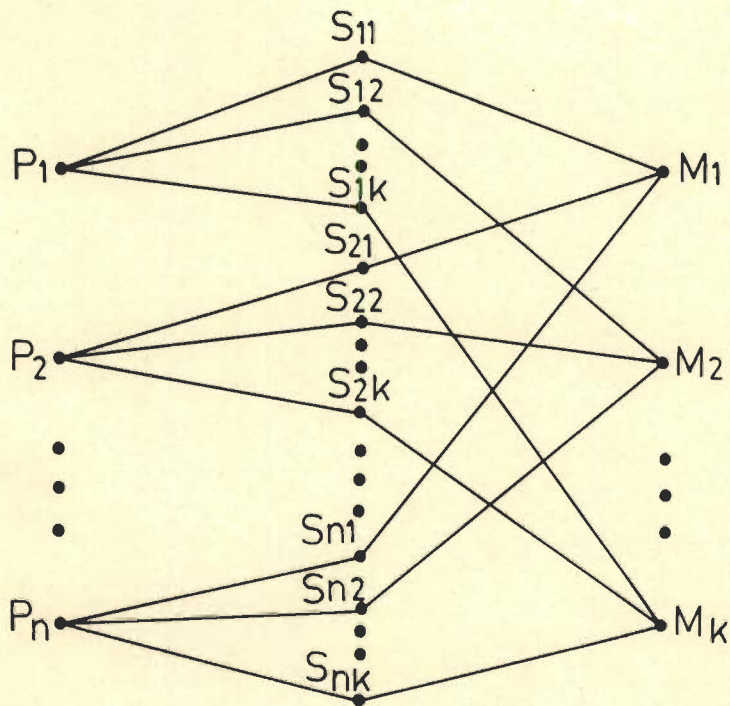


FIG. 3.5 GRAPH REPRESENTATION OF CROSSBAR SWITCHED MPS.

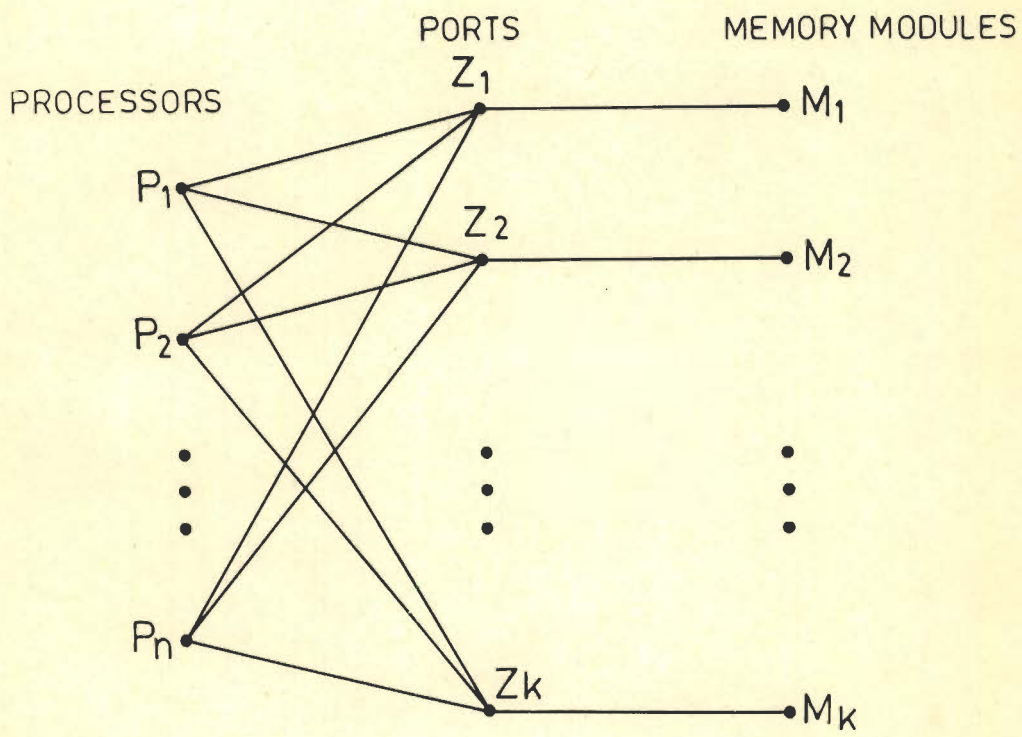


FIG. 3.6 GRAPH REPRESENTATION OF MULTIPOINT MEMORY MPS.

from each processor to every memory module. The connection matrix, C_n , consists of k columns, a column corresponding to a memory module. The element $C_{i,j}$ of C_n represents the unique path between P_i and M_j , viz., $P_i S_{ij} M_j$. These paths may be numbered as shown in the connection matrix below (Fig. 3.7), for $n = k = 3$. The path number is given by $(i-1)k+j$.

	M_1	M_2	M_3
P_1	1	2	3
P_2	4	5	6
P_3	7	8	9

FIG. 3.7 CONNECTION MATRIX 3 x 3 CROSSBAR MPS

b) Multiport Memory MPS

In this architecture also, there is unique path from P_i to M_j through Z_j , the n -ported controller for M_j . The connection matrix, C_n , therefore, has k columns as shown in Fig. 3.8 for the case $n = k = 3$. In this case again the path number is $(i-1)k+j$.

	$Z_1 M_1$	$Z_2 M_2$	$Z_3 M_3$
P_1	1	2	3
P_2	4	5	6
P_3	7	8	9

FIG. 3.8 CONNECTION MATRIX OF 3-PORT MEMORY MPS

c) Multiple-bus MPS

In this system, a processor P_i can be connected to a

memory module M_j through any one of the z buses. Thus there are z paths between P_i and M_j , viz., $P_i B_f M_j$, $f = 1, 2, \dots, z$. These paths are all numbered sequentially as shown in Fig. 3.9, for $n = k = 3$ and $z = 2$. The path number is given by $(i-1)kz + (j-1)z+f$.

	$M_1 B_1$	$M_1 B_2$	$M_2 B_1$	$M_2 B_2$	$M_3 B_1$	$M_3 B_2$
P_1	1	2	3	4	5	6
P_2	7	8	9	10	11	12
P_3	13	14	15	16	17	18

FIG. 3.9 CONNECTION MATRIX OF 3 x 3 x 2 MULTIPLE-BUS MPS.

3.1.3 Reduction of the connection matrix

The matrix, C_n , gives multiprocessing events for the case when all the n processors are active in executing a task. In order to get the multiprocessing events where less than n processors (minimum 2) are active, lower order matrices are to be obtained from C_n , which contains the same number of columns as C_n , but a reduced number of rows. Thus C_{n-1} contains $(n-1)$ rows, C_{n-2} contains $(n-2)$ rows and so on. C_{n-1} is obtained by removing one row at a time from C_n . There will be $\binom{n}{n-1}$ such matrices. $C_{n-1}(i)$, $i = 1, 2, \dots, n$ is obtained by removing i th row from C_n . $C_{n-2}(i, j)$, $i, j = 1, 2, \dots, n$, and $i \neq j$ are obtained by removing rows i and j from C_n . In general, matrix C_d contains d rows ($2 \leq d \leq n$) obtained by removing $(n-d)$ rows from C_n . C_d will generate the multiprocessing events with d processors active. Row matrices, denoted by C_i 's, will generate SISD

uniprocessing events and are not useful in determining multiprocessing reliability. Here we consider the matrices with atleast two rows. Example 3.1 illustrates the technique of reduction of a matrix.

Example 3.1: Consider the connection matrix, C_3 , given in Fig. 3.7. By reduction, we get only the lowest order matrices, C_2 's, and are given by

$$C_2(1) = \begin{pmatrix} 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}, \quad C_2(2) = \begin{pmatrix} 1 & 2 & 3 \\ 7 & 8 & 9 \end{pmatrix}, \quad C_2(3) = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$$

3.1.4 Enumeration of Events

The multiprocessing events are generated by recursively expanding each of the matrices C_i , $2 \leq i \leq n$. Algorithm 3.1 below gives the method of expansion. Let x and y denote the number of rows and columns of the matrix, A_x , to be expanded. Multiplication and addition used in the algorithm are boolean operations.

Example 3.2 illustrates this technique.

Example 3.2: Consider the connection matrix of Fig. 3.7

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} = 1 \begin{pmatrix} 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} + 2 \begin{pmatrix} 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} + 3 \begin{pmatrix} 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

Algorithm 3.1

Expansion of a connection matrix, A_x , for crossbar switched MPS.

Input: x, y, A_x

Output: Multiprocessing event expression with x processors active.

Step 1: Consider row 1. For the j th element, $a_{1,j}$ generate a reduced matrix $A_{x-1} (1)$ of size $(x-1)$ by deleting row 1. Multiply the element $a_{1,j}$ with $A_{x-1} (1)$. Repeat this for each element of row 1, $a_{1,j}$, $j = 1, 2, \dots, y$, and sum up all the terms to obtain an expression of the form

$$\sum_{j=1}^y a_{1,j} A_{x-1} (1)$$

Step 2: Repeat the above procedure with each matrix in the expression till it contains only row matrices.

Step 3: Multiply the coefficient of a matrix with each element in the row matrix and add.

$$\begin{aligned}
&= 1.4 (7 \ 8 \ 9) + 1.5 (7 \ 8 \ 9) + 1.6 (7 \ 8 \ 9) + \\
&\quad 2.4 (7 \ 8 \ 9) + 2.5 (7 \ 8 \ 9) + 2.6 (7 \ 8 \ 9) + \\
&\quad 3.4 (7 \ 8 \ 9) + 3.5 (7 \ 8 \ 9) + 3.6 (7 \ 8 \ 9) \\
&= 1.4.7 + 1.4.8 + 1.4.9 + 1.5.7 + 1.5.8 + 1.5.9 + 1.6.7 + 1.6.8 + \\
&\quad 1.6.9 + 2.4.7 + 2.4.8 + 2.4.9 + 2.5.7 + 2.5.8 + 2.5.9 + 2.6.7 + \\
&\quad 2.6.8 + 2.6.9 + 3.4.7 + 3.4.8 + 3.4.9 + 3.5.7 + 3.5.8 + 3.5.9 + \\
&\quad 3.6.7 + 3.6.8 + 3.6.9
\end{aligned}$$

This algorithm will directly give all the multiprocessor events for a crossbar system. Each product term in the expansion denotes an event for multiprocessor. For example 1.4.7 means the event of all the three processors and the memory module M_1 are being active in the execution of a task.

The connection matrices for the multiple-bus system and the multiprocessor memory system can also be expanded by following Algorithm 3.1. However, this will produce many duplicate terms because of some elements being common to many DFP's. This duplication can be easily avoided by slight modification of Algorithm 3.1, as shown below. This Algorithm 3.2 will produce only non-cancelling terms for multiple-bus and multiprocessor memory system. Here we denote by $A_{x-1}^j(i)$ the matrix obtained from A_x by deleting row i and all the columns to the left of column j . Thus $A_{x-1}^1(i)$ is same as $A_{x-1}(i)$.

Example 3.3: Consider the connection matrix of Fig. 3.8.

For the first expansion matrices required are:

Algorithm 3.2

Expansion of a connection matrix A_x for multiple-bus and multipoint systems.

Input: x, y, A_x

Output: multiprocessing event expression with x processors active.

This algorithm is same as Algorithm 3.1 except that Step 1 is modified as follows.

Step 1: Consider row 1. For the j th element in row 1, i.e., $a_{1,j}$, obtain a reduced matrix $A_{x-1}^j(1)$. Multiply $a_{1,j}$ with $A_{x-1}^j(1)$. Repeat this for each element of row 1, $a_{1,j}$, $j = 1, 2, \dots, y$ and sum up all the terms to obtain an expression of the form $\sum_{j=1}^y a_{1,j} A_{x-1}^j(1)$.

$$A_2^1(1) = \begin{pmatrix} 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}, \quad A_2^2(1) = \begin{pmatrix} 5 & 6 \\ 8 & 9 \end{pmatrix}, \quad A_2^3(1) = \begin{pmatrix} 6 \\ 9 \end{pmatrix}$$

Thus after first expansion the expression is

$$1. \begin{pmatrix} 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} + 2. \begin{pmatrix} 5 & 6 \\ 8 & 9 \end{pmatrix} + 3. \begin{pmatrix} 6 \\ 9 \end{pmatrix}$$

and so on.

3.1.5 Reliability Computation

The following steps are required for computing the multiprocessing reliability, of any MPS, from the event expression.

- 1) Replace the event expression by path expression.
- 2) Disjoin the events in the path expression exhaustively to get a boolean expression, in the form of sum-of-disjoint products of the interconnecting elements involved in the paths [92].
- 3) Replace the boolean variables with their probabilistic values to get the multiprocessing reliability.

Example 3.4: Consider 2 x 2 crossbar system. The connection matrix is shown in Fig. 3.10.

	M ₁	M ₂
P ₁	1	2
P ₂	3	4

FIG. 3.10. CONNECTION MATRIX OF 2 X 2 CROSSBAR SYSTEM.

The event expression is $13 + 14 + 23 + 24$

Path expression:

$$P_1 P_2 S_{11} S_{21} M_1 + P_1 P_2 S_{11} S_{22} M_1 M_2 + P_1 P_2 S_{12} S_{21} M_1 M_2 + P_1 P_2 S_{12} S_{22} M_2$$

Boolean expression after disjoining:

$$P_1 P_2 S_{11} S_{21} M_1 + P_1 P_2 \bar{S}_{11} S_{12} S_{21} M_1 M_2 + P_1 P_2 S_{11} \bar{S}_{21} S_{22} M_1 M_2 + \\ P_1 P_2 \bar{S}_{11} S_{12} \bar{S}_{21} S_{22} M_2 + P_1 P_2 \bar{S}_{11} S_{12} S_{21} S_{22} \bar{M}_1 M_2 + P_1 P_2 S_{11} S_{12} S_{21} S_{22} \bar{M}_1 M_2 + \\ P_1 P_2 S_{11} S_{12} \bar{S}_{21} S_{22} \bar{M}_1 M_2$$

Reliability of the MPS:

$$2p^2 s^2 m + 2p^2 s^2 m^2 - 4p^2 s^3 m^2 + p^2 s^4 m^2$$

assuming $p_i = p$, $m_j = m$ and $s_{ij} = s$ for all i and j .

3.2 t-out-of-s SYSTEMS

In this section we discuss t-out-of-s systems. The results derived here shall be used later for analysis of MPSs.

If a system with s components requires t ($\leq s$) or more components to function for the system to be good, then such a system is called a t-out-of-s: G system. If we let $t=s$, we have a series system; if we let $t=1$, then we have a system with parallel redundancy. A case of particular importance is a system with $t=s-1$. In such a system the failure of a single unit is not sufficient to cause system failure, but the failure of two units does cause system failure. This is sometimes referred to as fail-safe design.

The dual of t-out-of-s: G system is the t-out-of-s: F system which is defined as a system which requires at least t units to fail for the system to fail. Because of duality, there

is no real distinction between a t-out-of-s:G system and a t-out-of-s:F system. The probability of success of a t-out-of-s:G system is the complement of the probability of failure for an (s-t+1)-out-of-s : F. To save effort, it is only necessary to ascertain which number, $\binom{s}{t}$ or $\binom{s}{s-t+1}$ is larger. If the system is t-out-of-s:G and $\binom{s}{t}$ is larger, the probability of system failure produces lesser number of terms for computation and hence is advantageous; otherwise the probability of success is advantageous. If the two numbers are equal, both the systems yield the same computation effect. Instead of discussing both the systems separately, we can consider one, say t-out-of-s:G system, and substitute t by (s-t+1) when advantageous. There has been a great deal of interest in the study of t-out-of-s systems (37,93-101) because of the following reasons. t-out-of-s systems are more general than pure series or parallel system, and some interconnections can be modeled using this technique [99,100].

3.2.1 Previous work : A Review

Many researchers have presented algorithms for calculating the reliability of a t-out-of-s:G system where each of the s components of the system has a given reliability. The method of inclusion-exclusion principle [102] produces a larger number of pairs of identical terms with opposite signs, and requires $2\binom{s}{t} - 1$ terms and each term involves at least t multiplications. The first success in completely suppressing pairs of cancelling terms was scored by Satyanarayana et al [103] in

analysing the reliability of networks. But these results can not be applied to t-out-of-s systems, because for $1 < t < s$ these structures are not networks.

Cooksey [93] and Heidtmann [94] have considerably reduced the number of terms, to $\sum_{i=t}^s \binom{s}{i}$, by avoiding generation of cancelling terms. However, each term has to be multiplied by a positive or negative constant which represents the number of repetitions of the term. McGrady [95] also generates the same number of terms but each term involving exactly (s-1) multiplications.

Recent algorithms [37,96-99] have further improved on the computational effort. Locks [37] used the method of disjoint products [104] and generates $\binom{s}{t}$ terms. It however, involves many terms in the intermediate steps which disappear in final expression. Barlow and Heidtmann [96] used a recursive approach and is good for numerical computation but their method generates more than $\binom{s}{t}$ terms. The algorithm presented by Jain and Gopal [97] generates $\binom{s}{t}$ terms, but recursiveness of the evaluation process have not fully been exploited and it takes more multiplications than necessary [98].

In the following subsections we derived some efficient algorithms to obtain the exact reliability of a t-out-of-s:G system. Section 3.2.2 considers the techniques used in developing the model. Section 3.2.3 presents two recursive algorithms. An improved recursive algorithm is presented in

Section 3.2.4 and Section 3.2.5 presents an efficient non-recursive algorithm. A comparison of these algorithms [93-95,97] is done in Section 3.2.6.

3.2.2 Theory

In order to express the various events the two methods generally considered are conservative and exhaustive. Fratta and Montanari compared their relative merits [84]. In the present work conservative method is used for algorithm development, because it minimizes the number of disjoint events.

Conservative Policy

Let us consider a set of boolean stochastic variables $\{x_1, \dots, x_m\}$. The set of stochastic variables is first transformed into a sequence (x_1, x_2, \dots, x_m) , i.e., an arbitrary order is chosen. The set E contains the following $m+1$ events.

$$\begin{aligned}
 E_1 &= \{x_1 = 1\} \\
 E_2 &= \{x_1 = 0 ; x_2 = 1\} \\
 &\vdots \\
 E_m &= \{x_1 = \dots = x_{m-1} = 0 ; x_m = 1\} \\
 E_{m+1} &= \{x_1 = \dots = x_m = 0\}
 \end{aligned}$$

Note that every event except E_{m+1} contains exactly one positive assignment, while the number of negative assignment varies from 0 to m . This policy will be called as conservative policy while the dual policy will be called conservative negative. Both the variants of conservative policy are illustrated in Table 3.1.

TABLE 3.1
 VARIANTS OF CONSERVATIVE POLICY
 (Variables (x_1, x_2, \dots, x_m))

Event name	Negative	Assignment	Positive
E ₁	0		1
E ₂	10		01
E ₃	110		001
.			
.			
.			
E _m	111...10		000...01
E _{m+1}	111...11		000...00

3.2.3 Algorithm Development

Let us consider a system with 'n' units. The vectors $X = (p_1, p_2, \dots, p_n)$ and $\bar{X} = (q_1, q_2, \dots, q_n)$ represent the reliability and the unreliability values of these individual units. Then the following theorems hold.

Theorem 3.1 : Let s and t be positive integers such that $s \geq t$, then in a system with n units, the reliability is given by

$$R(s, t) = p_{n-s+1} H(X; s-1, t-1) + q_{n-s+1} p_{n-s+2} H(X; s-2, t-1) \\ + q_{n-s+1} q_{n-s+2} H(X; s-2, t)$$

Proof: Consider two units with conservative positive expansion of logical 1, 01, and 00. Logical 1(0) implies that a unit is good (failed) and is replaced by p(q) value of the unit. Each factor modifies $H(X; s, t)$ in the equation and hence the theorem is proved. Figure 3.11 illustrates the tableau for computing $H(X; s, t)$.

Corollary 3.1 - For $s=t$ Theorem 3.1 becomes:

$$H(X; s, s) = p_{n-s+1} H(X; s-1, s-1).$$

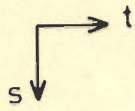
which recursively reduces to:

$$H(X; s, s) = \prod_{i=n-s+1}^n p_i, \quad s \neq 0.$$

Theorem 3.2 - For the integer values of s and t, we have

$$H(X; s, t) = \begin{cases} 0; & s < t \\ 1; & t = 0 \end{cases}$$

Proof: $H(X; s, t)$ is assumed to be $\binom{s}{t}$ hence the theorem follows



	t+1	t	
(s-2)	(s-2,t)	(s-2,t)	
(s-1)	(s-1,t-1)		
s		(s,t)	

FIG. 3.11 TABLEAU ASSOCIATED WITH $H(x;s,t)$
COMPUTATION USING THEOREM 3.1

245423

Central Library University of Roorkie
ROORKEE



from the definition of s units taken t at a time [102]

Theorem 3.3 : Let s and t be positive integers such that $s \geq t$ then-

$$\begin{aligned} H(X;s,t) &= p_{n-s+1} H(X;s-1, t-1) + \dots \\ &+ q_{n-s+1} \dots q_{n-t-1} p_{n-t} H(X;t, t-1) \\ &+ q_{n-s+1} \dots q_{n-t} H(X;t,t) \end{aligned}$$

Proof: Same as Theorem 3.1.

Theorem 3.3 gives an expression for $H(X;s,t)$ considering $(s-t)$ units and applying conservative positive assignment to obtain elementary events. Thus, it has $(s-t+1)$ terms as against only 3 terms of theorems 3.1. Fig. 3.12 illustrates the tableau for computing $H(X;s,t)$.

Example 3.5- consider a 4-out-of-7 : G system, then

$$H(X;7,4) = p_1 H(X;6,3) + q_1 p_2 H(X;5,3) + q_1 q_2 H(X;5,4)$$

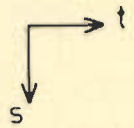
using theorem 3.1, and

$$\begin{aligned} H(X;7,4) &= p_1 H(X;6,3) + q_1 p_2 H(X;5,3) + q_1 q_2 p_3 H(X;4,3) \\ &+ q_1 q_2 q_3 H(X;4,4) \end{aligned}$$

using Theorem 3.3. In both the cases $s = n = 7$ and $t = k = 4$.

Recursive Approach

Algorithms 3.3 and 3.4 are directly based on Theorems 3.1 and 3.3. These are presented in Fig.3.13 and Fig.3.14 respectively. The simplification, if any, is done using Theorems 3.2 and Corollary 3.1. The algorithms are presented in Algol-like notation.



	$t-1$	t	
$s=t$	$(t, t-1)$	(t, t)	
$s-2$	$(s-2, t-1)$		
$s-1$	$(s-1, t-1)$		
s		(s, t)	

FIG. 3.12 TABLEAU ASSOCIATED WITH $H(x; s, t)$ COMPUTATION USING THEOREM 3.3.

```

procedure H(X;s,t)
  begin
    R ← 0;
    case
      : s < t: return (R)
      : t = 0 : R ← 1; return (R)
      : s = t : R ← p(n-s+1) * H(X;s-1,t-1) return (R)
    else
      R ← p(n-s+1) * H(X;s-1,t-1) + q(n-s+1) * p(n-s+2) *
        H(X;s-2,t-1) + q(n-s+1) * q(n-s+2) * H(X;s-1,t-1)
    end;
  end.

```

FIG.3.13 ALGORITHM 3.3


```

procedure H(X;s-1,t-1)
  begin
    R ← 0;
    case
      : s < t : return (R)
      : t ← 0 : R ← 1;return (R)
    else
      for i ← (n-s+1) to (n-t+1)do
        begin
          G ← p(i);
          for j ← (n-s+1) to (i-1) do
            G ← G * q(j);
          R ← R + G * H(X;s-1,t-1);
        end;
      return(R)
    end;
  end;
end.

```

FIG.3.14 ALGORITHM 3.4

3.2.4 An Efficient Recursive Algorithm

Let us consider the following example.

Examples 3.6- Consider a 3-out-of-5:G system. The symbolic expression for reliability is obtained using both conservative negative and positive assignments.

$$\begin{aligned} R(X;5,3) = & q_1 [p_2 \{ p_3 (p_4 + q_4 p_5) + q_3 p_4 p_5 \} + q_2 p_3 p_4 p_5] \\ & + p_1 q_2 [p_3 (p_4 + q_4 p_5) + q_3 p_4 p_5] \\ & + p_1 p_2 q_3 (p_4 + q_4 p_5) + p_1 p_2 p_3 \end{aligned}$$

Which can be written as

$$\begin{aligned} R(X;5,3) = & q_1 H(X;4,3) + p_1 q_2 H(X;3,2) + p_1 p_2 q_3 H(X;2,1) \\ & + p_1 p_2 p_3 H(X;2,0) \end{aligned}$$

Definition 3.1 generalizes examples 3.5 and 3.6. Instead of deriving the coefficients of H function from the conservative positive policy as in Theorem 3.1, we derive the coefficients from conservative negative policy (Example 3.6) and suitably modify the Theorem 3.1.

Consider t units with conservative negative expansion as shown in Table 3.1. Each of their factors modifies H(X;s,t) in Theorem 3.1 and hence -

$$\begin{aligned} H(X;s,t) = & q_1 H(X;s-1,t) + p_1 q_2 H(X;s-2,t) + \dots \\ & + \dots + p_1 p_2 \dots p_{t-1} q_t H(X;s-t,1) \\ & + p_1 p_2 \dots p_t H(X;s-t,0) \end{aligned}$$

Where $H(X;s-t,0) = 1$, based on this the Definition 3.1 follows.

Definition 3.1 The symbolic expression for reliability of t-out-of-s: G system is__

$$R(s,t) = \sum_{i=1}^t G_i H(X;s-i, t-i+1) + G_{t+1} \quad (1)$$

where

$$G_j = \begin{cases} \prod_{i=1}^t p_i & ; j = t+1 \\ q_j \cdot \prod_{i=1}^{j-1} p_i & ; 1 \leq j \leq t \end{cases} \quad (2)$$

Algorithm 3.5 is developed based on Definition 3.1 and Theorems 3.1 and 3.2 and is shown in Fig. 3.15. The algorithms 3.3 and 3.4 do not require the calculation of G_j 's. Therefore, the program size is smaller than Algorithm 3.5, but there are more iterations. Because of this, these algorithms take slightly more computational time than Algorithm 3.5.

Example 3.7 Consider the system with 7 units. The units reliabilities are $p_1 = 0.9$, $p_2 = 0.8$, $p_3 = 0.7$, $p_4 = 0.6$, $p_5 = 0.5$, $p_6 = 0.4$, $p_7 = 0.3$.

The system is 4-out-of-7: G. The reliability is computed using Algorithm 3.1, which ultimately evaluates

$$R(7,4) = G_5 + G_4 H(X;3,1) + G_3 H(X;4,2) + G_2 H(X;5,3) + G_1 H(X;6,4)$$


```

program R(n,k):
  begin
    for i ← 1 to n do
      begin
        read p(i);
        q(i) ← [1-p(i)];
      end;
    begin
      for i ← 1 to (n-1) do
        for j ← 1 to (k-1) do
          M(i,j) ← 2;
        end;
      G ← p(k);
      for i ← 1 lto (k-1) do G ← G * p(i);
      R ← G;
      for j ← k downto 1 do
        begin
          G ← q(j);
          for i ← 1 to (j-1) do G ← G * p(i);
          R ← R + G * H(X;n-j,k-j+1);
        end;
      end.

```

FIG.3.15 ALGORITHM 3.5

```

procedure H(X;s,t)
begin
R ← M(s,t);
if R ≤ 1 then return (R)
R ← 0;
case
: s < t : M(s,t) ← R; return (R)
: t = 0 : R ← 1; M(s,t) ← R; return (R)
: s = t : R ← p(n-s+1) * H(X;s-1,t-1); M(s,t) ← R; return (R)
else
R ← p(n-s+1) * H(X;s-1,t-1) + q(s-t+1) * p(n-s+2) * H(X;s-2,t-1)
    * q(n-s+1) * q(n-s+2) * H(X;s-2,t);
M(s,t) ← R; return (R)
end;
end.

```

FIG.3.15 ALGORITHM 3.5 CONT'D

The G_5 is first obtained without using $H(X;s,t)$. The remaining factors call $H(X;s,t)$ recursively for their computation. The final expression along with the values for various G_j 's and $H(X;s,t)$ are in Table 3.2.

3.2.5. Non Recursive Approach

Theorem 3.1 requires $[(s-t+1)(t-\frac{1}{2})-1]$ functions to be computed out of which approximately $[(s-t)(t-1) - 1]$ terms are to be computed more than once for all values of s and t .

Here we present a non recursive algorithm which computes the system reliability much more efficiently. This algorithm drastically reduces not only the amount of computation needed but also the memory requirement.

Theorem 3.1 has been used recursively for computation of reliability $H(X;s,t)$. For given values of $s=n$ and $t=k$, the various $H(X;s,t)$ values required to be computed are shown in Fig. 3.16 for $n = 7$ and $k = 4$. In the recursive use of Theorem 3.1, however, it can be seen, that many of the H functions must be calculated more than once. For example, in the computation of $H(X;7,4)$, $H(X;4,2)$ is required for $H(X;6,3)$ as well as for $H(X;5,3)$. This could be avoided by storing the complete matrix, so that no H function is required to be computed more than once. But if we observe Theorem 3.1, it is seen that storing the complete matrix is quite unnecessary. For computing values in any row (i.e., for a given s), only two previous rows are required

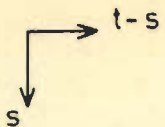
TABLE 3.2

COMPUTATION OF G_i AND $H(X;s,t)$ OF EXAMPLE 3.7

FACTOR	VALUE	EXPRESSION	#	REMARKS
G_5	0.3024	$p_1 p_2 p_3 p_4$	3	
G_4	0.2016	$p_1 p_2 p_3 q_4$	3	
G_3	0.216	$p_1 p_2 q_3$	2	
G_2	0.18	$p_1 q_2$	1	
G_1	0.1	q_1		
$H(3,1)$	0.79	$p_5 H(2,0) + q_5 p_6 H(1,0) + p_5 q_6 H(1,1)$	3	$H(1,0)=1$
$H(1,1)$	0.3	p_7		$H(2,0)=1$
$H(4,2)$	0.614	$p_4 H(3,1) + q_4 p_5 H(2,1) + q_4 q_5 H(2,2)$	5	$H(0,0)=1$
$H(2,1)$	0.58	$p_6 H(1,0) + q_6 p_7 H(0,0) + q_6 q_7 H(0,1)$	1	$H(1,0)=1$
$H(2,2)$	0.12	$p_6 H(1,1)$	1	$H(0,1)=0$
$H(5,3)$	0.5	$p_3 H(4,2) + q_3 p_4 H(3,2) + q_3 q_4 H(3,3)$	5	
$H(3,2)$	0.35	$p_5 H(2,1) + q_5 p_6 H(1,1) + q_5 p_6 H(1,2)$	3	$H(1,2)=0$
$H(3,3)$	0.06	$p_5 H(2,2)$	1	
$H(6,4)$	0.43492	$p_2 H(5,3) + q_2 p_3 H(4,3) + q_2 q_3 H(4,4)$	5	
$H(4,3)$	0.234	$p_4 H(3,2) + q_4 p_5 H(2,2) + q_4 q_5 H(2,3)$	3	$H(2,3)=0$
$H(4,4)$	0.036	$p_4 H(3,3)$	1	

No. of multiplications

Note:- For simplicity, the X terms are dropped from the H functions.



	$t-s$	$t-s+1$	$t-s+2$
$s=t$	$(t, t-s)$	$(t, t-s+1)$	$(t, t-s+2)$
$s-2$		$(s-2, t-s+1)$	$(s-2, t-s+2)$
$s-1$	$(s-1, t-1)$		
s			
s	$(s, t-s)$		

FIG. 3.16 TABLEAU ASSOCIATED WITH $H_{s,t-s}$ COMPUTATION USING THEOREM 3.1

(i.e., $s-1$ and $s-2$). Thus we can avoid storing the complete matrix by starting with row $s=0$ and keeping only three rows of the matrix at any time. For example, from the rows labelled $s=0$ and $s=1$ we can obtain the row $s=2$. The row $s=3$ can be obtained from row $s=1$ and $s=2$ discarding $s=0$, and so on, till the row $s=n$ is finally computed.

Further, it is clear that all entries above the diagonal through $(0,0)$ are zeros and the entries below the diagonal through (s,t) are not needed in the computation. Hence by a transformation which makes these diagonals vertical the length of each row can be reduced to $(s-t+1)$.

These considerations lead us to an algorithm that avoids completely duplication of computation and at the same time requires minimal memory. Algorithm 3.6 is also presented in Algol-like notation in Fig. 3.17.

3.2.6 Comparisons

Considering a 4-out-of-G system, the different methods available in the literature [93-95,97] are compared based on their computational complexity.

- i) The event-space approach generates 64 terms, each term involves 6 multiplications [95].
- ii) The methods in [93,94] generate 64 terms but the total number of multiplications is only 233 which is 61% of that event space approach [97].


```

procedure H(X;s,t)
for i ← 1 to n do
    begin
        read p(i); q(i) ← 1-p(i);
    end;
begin
    r ← 0;
    if s<t then return (r)
        if t = 0
    then
        r ← 1;return (r)
    um = n; vm = k-n;
    J(0,1) ← J(1,1) ← J(2,1) ← 0;
    J(0,0) ← 1; J(1,-1) ← 1; J(1,0) ← p(n);
    for u ← 2 to um do
        begin
            u0 ← u mod 3;
            u1 ← (u-1) mod 3;
            u2 ← (u-2) mod 3;
            vx ← vm;
            if u≥k then
                J(u0,0) ← p(n-u+1) * J(u1,0);
            if -u>=vm then
                begin
                    J(u0,-u) ← 1;
                    vx ← 1-u;
                end
        end
end

```

```

    end;
    v1 ← vm + [(um - u) div 2] * 2;
    if v1 < vx then v1 = k-u;
    if v1 > -1 then v1 ← -1;
    for v ← v1 downto vx do
        J(u0,v) ← p(n-s+1) * H(X;s-1,t-1) + q(n-s+1) * p(n-s+2)
            * H(X;s-2,t-1) + q(n-s+1) * q(n-s+2) * H(X;s-2,t);
    end;
    r ← J(um mod 3,vm);
    return (r)
end;
end.

```

FIG.3.17 ALGORITHM 3.6

- iii) The algorithm presented by Jain and Gopal [97] generates 35 terms, ie, only 55 percent of [93,93] and requires only 76 multiplication, which is only 32% of those of [93,94]
- iv) The methods presented in this section-
 - a. All the algorithms generate only 35 terms.
 - b. The algorithm 3.5 requires only 37 multiplications which is nearly 49 percent of that presented in [97].
 - c. The use of algorithms 3.3 and 3.4 require 42 and 64 multiplications which are only 55 and 82 percent of that of Jain and Gopal respectively.
 - d. Memory and time complexity.

Recursive Approach

- i) For all t , with $1 \leq t \leq s$, computing time increases as $t \rightarrow s/2$ and reaches maximum for $t = s/2$. Hence computational complexity is bounded by $s^2/4$ for a variable t .
- ii) For a given n , space complexity of this method is bounded by $t^2/2$.

Non-recursive approach

This method (Algorithm 3.6) requires minimal amount of memory and space complexity is $O(s-t)$. For a given value of $s-t$, the computational time is proportional to s .

3.3 Reliability Modeling

Algorithms which are efficient and which can easily be implemented on a computer are needed to analyze the reliability of large systems. In this section some reliability algorithms are presented, based on t-out-of-s system modeling which drastically reduce the amount of computation. Unlike the algorithms of section 3.1, these algorithms directly evaluate exclusive and mutually disjoint (EMD) events. Closed form solutions for multiprocessing and terminal reliability criteria are obtained. The reliability expressions for the three architectures are derived.

3.3.1 Multiprocessing Reliability

The threshold reliability is the probability when at least α processors and β memory modules are in operation to execute a task. As explained in Chapter II. The multiprocessing reliability and system reliability are special cases of threshold reliability.

(a) Multiple-bus system

The hardware resources of any $n \times k \times z$ multiple-bus MPS (Fig. 3.4) are divided into three independent groups: processors, memory modules and buses. Since the availability of processors, memory modules and buses are independent, the reliability of the MPS is the series reliability of these three groups of components according the principle of functional decomposition [89]. Based on the definitions in section 2.4.1.1,

consider the case where a task needs at least α processors and β memory modules. For the system operation at least one of the buses must be available. The threshold reliability can therefore be simply written as a product.

$$R_{\alpha\beta}[\text{Bus}] = H(P;n,\alpha) H(M;k,\beta) H(B;z,1) \quad (3.1)$$

where $P = \{P_i\}$, $M = \{M_j\}$, $B = \{B_f\}$ for all $i, j, \text{ and } f$.

Since multiprocessing reliability is the probability of atleast two processors and one memory module being active in the execution of a task, we have from equation (3.1)

$$R_m(\tau) [\text{Bus}] = H(P;n,2) H(M;k,1) H(B;z,1) \quad (3.2)$$

The case when atleast one processor and one memory are surviving for execution of a task, then the system reliability is given by

$$R_s(\tau) [\text{Bus}] = H(P;n,1) H(M;k,1) H(B;z,1) \quad (3.3)$$

Using eq.(3.3) system availability can also be computed with the knowledge of the individual component availabilities.

The uniprocessor reliability is the probability that one processor is executing a task in the system. This can easily be expressed by using an expression that specifies exactly i units availability in a task execution. This can be determined by taking the availability of at least i units and subtracting the availability of atleast $(i+1)$ units [93]. For computing $R_u(\tau)$, we need to compute the probability that exactly one

processor and one memory module are active in execution of a task. Then-

$$R_U(\tau) [\text{Bus}] = [H(P;n,1) - H(P;n,2)] H(M;k,1) H(B;z,1) \quad (3.4)$$

Where $[H(P;n,1) - H(P;n,2)]$ is the reliability that exactly one processor out of n is active in the execution of the task.

Example 3.8: Consider a MPS system with $n=k=z=4$. The problem is to compute $R_{\alpha\beta}$ when $\alpha = 2$ and $\beta = 3$.

$$\begin{aligned} R_{23}(\tau) &= H(P;4,2) H(M;4,3) H(B;4,1) \\ &= [p_1 (\bar{p}_2 + p_2 \bar{p}_3 + p_3 \bar{p}_2 \bar{p}_3 + p_4) + p_1 p_2 (p_3 + \bar{p}_3 p_4)] * \\ & [m_1 (m_2 (\bar{m}_3 + m_3 m_4) + \bar{m}_2 m_3 m_4) + \bar{m}_1 m_2 m_3 m_4] * (b_1 + \bar{b}_1 \bar{b}_2 + \bar{b}_1 \bar{b}_2 \bar{b}_3 + \bar{b}_1 \bar{b}_2 \bar{b}_3 b_4) \end{aligned}$$

$$\begin{aligned} R_S(\tau) &= H(P;4,1) H(M;4,1) H(B;4,1) \\ &= (p_1 + \bar{p}_1 p_2 + \bar{p}_1 \bar{p}_2 p_3 + \bar{p}_1 \bar{p}_2 \bar{p}_3 p_4) * (m_1 + \bar{m}_1 m_2 + \bar{m}_1 \bar{m}_2 m_3 + \bar{m}_1 \bar{m}_2 \bar{m}_3 m_4) * \\ & (b_1 + \bar{b}_1 \bar{b}_2 + \bar{b}_1 \bar{b}_2 \bar{b}_3 + \bar{b}_1 \bar{b}_2 \bar{b}_3 b_4) \end{aligned}$$

$$R_{21}(\tau) = H(P;4,2) H(M;4,1) H(B;4,1)$$

If the reliabilities of all the system components are same and equal to 0.9 then

$$R_{23} = 0.9441, R_S = 0.9997, R_U = 3.5993 \times 10^{-3}, R_{21} = 0.9961.$$

(b) Crossbar System

In case of an $n \times k$ crossbar the failure of either a memory module or a crosspoint switch reduces the size of the crossbar to $n \times (k-1)$ [59] and hence the availability of memory modules and crosspoint switches are not disjoint [see Fig.3.5].

Hence the system reliability is obtained by considering the series reliability of processors and the combination of memory and crossbar. It can be seen that in a crossbar each memory module is connected to 'n' crosspoint switches.

The probability, θ_j , that the jth memory module is available to one of the processors is given by

$$\theta_j = H(S_j; n, 1) M_j \quad (3.5)$$

where $S_j = \{S_{1j} \mid j \text{ fixed}\}$ represents the probabilities of the crosspoint switches connected to the jth memory module, and expanding H function.

$$\theta_j = (S_{1j} + \bar{S}_{1j}S_{2j} + \dots + \bar{S}_{1j}\bar{S}_{2j}\dots\bar{S}_{(n-1)j}S_{nj})M_j$$

Hence for a crossbar switched MPS, we have-

$$R_{PB}(\tau)[\text{crossbar}] = H(P; n, \alpha) H(\theta; k, \beta) \quad (3.6)$$

$$R_m(\tau)[\text{crossbar}] = H(P; n, 2) H(\theta; k, 1) \quad (3.7)$$

$$R_S(\tau)[\text{crossbar}] = H(P; n, 1) H(\theta; k, 1) \quad (3.8)$$

$$R_U(\tau)[\text{crossbar}] = [H(P; n, 1) - H(P; n, 2)] H(\theta; k, 1) \quad (3.9)$$

Example 3.9 Consider a 4 X 4 crossbar

$$\begin{aligned} R_{PB}(\tau) &= H(P; 4, 2) H(\theta; 4, 3) \\ &= [p_1 (p_2 + \bar{p}_2 p_3 + \bar{p}_2 \bar{p}_3 p_4) + \bar{p}_1 p_2 (p_3 + \bar{p}_3 p_4)] * \\ &\quad [\theta_1 (\theta_2 (\theta_3 + \bar{\theta}_3 \theta_4) + \bar{\theta}_2 \theta_3 \theta_4) + \bar{\theta}_1 \theta_2 \theta_3 \theta_4] \end{aligned}$$

and $\theta_j = H(S_j; 4, 1) M_j$

$$\theta_j = (S_{1j} + \bar{S}_{1j}S_{2j} + \bar{S}_{1j}\bar{S}_{2j}S_{3j} + \bar{S}_{1j}\bar{S}_{2j}\bar{S}_{3j}S_{4j})M_j$$

$$R_s(\tau) = H(P; 4, 1) H(\phi; 4, 1)$$

$$= (P_1 + \bar{P}_1 P_2 + \bar{P}_1 \bar{P}_2 P_3 + \bar{P}_1 \bar{P}_2 \bar{P}_3 P_4) * (\phi_1 + \bar{\phi}_1 \phi_2 + \bar{\phi}_1 \bar{\phi}_2 \phi_3 + \bar{\phi}_1 \bar{\phi}_2 \bar{\phi}_3 \phi_4) *$$

$$R_{21}(\tau) = H(P; 4, 2) H(\phi; 4, 1)$$

If the reliabilities of all processors, memory modules and cross-point switches are same and are equal to 0.9, we have

$$R_{23} = 0.9441; R_s = 0.9998; R_u = 3.5996 \times 10^{-3}, R_{21} = 0.9962.$$

(c) Multiport System

In this organization, shown in Fig. 3.3, the switching is concentrated in the memory module. Each processor has access through its own bus to all memory modules. This is a special case of crossbar, where all crosspoint switches in the same crossbar column (Fig. 3.6) are combined and hence -

$$S_{ij} = Z_j, 1 \leq i \leq n, 1 \leq j \leq k \quad (3.10)$$

$$\text{and } \phi_j = z_j M_j, \text{ for all } j$$

With this value of ϕ , the equations (3.6)-(3.9) give the corresponding reliability measures for an MPS with k n -ported memories. Using these equations its multiprocessing reliability can be easily computed.

The variation of multiprocessing reliability, in case of the three architectures ($n=k=4$) with variation of the success probability of interconnection links is shown in Fig. 3.18. The dependence of system reliability and uniprocessors reliability on

the success probability of IN is same. It can also be concluded that for low bus failure rate, the reliability of the three systems will be very close. Fig. 3.19 gives the variation of the three reliability measures with processor reliability in case of a 4 X 4 X 4 multiple-bus system. The uniprocessor reliability records its maximum value when the reliability of each processor is equal to 1/n. When p_i ($= p$, $1 \leq i \leq n$) reaches unity $R_u = R_m$.

Assuming the failures are exponentially distributed, we define λ_{P_i} , λ_{M_j} , $\lambda_{S_{ij}}$, λ_{Z_j} and λ_{B_f} are the failure rates of the processors P_i , the memory module M_j , the crossbars which S_{ij} , the multipore Z_j and the multiple-bus B_f respectively. Then

$$p_i(\tau) = e^{-\lambda_{P_i}\tau} \quad ; \quad m_j(\tau) = e^{-\lambda_{M_j}\tau} \quad \text{and} \quad b_f(\tau) = e^{-\lambda_{B_f}\tau}$$

$$s_{ij}(\tau) = e^{-\lambda_{S_{ij}}\tau} \quad ; \quad z_j(\tau) = e^{-\lambda_{Z_j}\tau} \quad \text{give the corresponding}$$

reliabilities. For simplicity, we assume the elements of a group have the same failure rate, then-

$$\lambda_{P_i} = \lambda_P = \lambda_{M_j} = \lambda_M = 0.0001 \quad \text{for all } 1 \leq i, j \leq k$$

$$\text{and } \lambda_{z_j} = \lambda_z = \lambda_{S_{ij}} = \lambda_S = \lambda_{B_f} = \lambda_B = 0.00005 \quad \text{for all } i \text{ and } j.$$

Analytic results showing the variation of multiprocessing reliability R_{21} and R_{44} with time, for three systems having $n = k = 4$ and /or $z = 1$ are shown as a function of time in Figures 3.20 and 3.21. If we need high reliability for a short time crossbar is considered to be the best architecture. It can also be noticed that the common bus has the highest reliability, multiport is next and the crosspoints have the

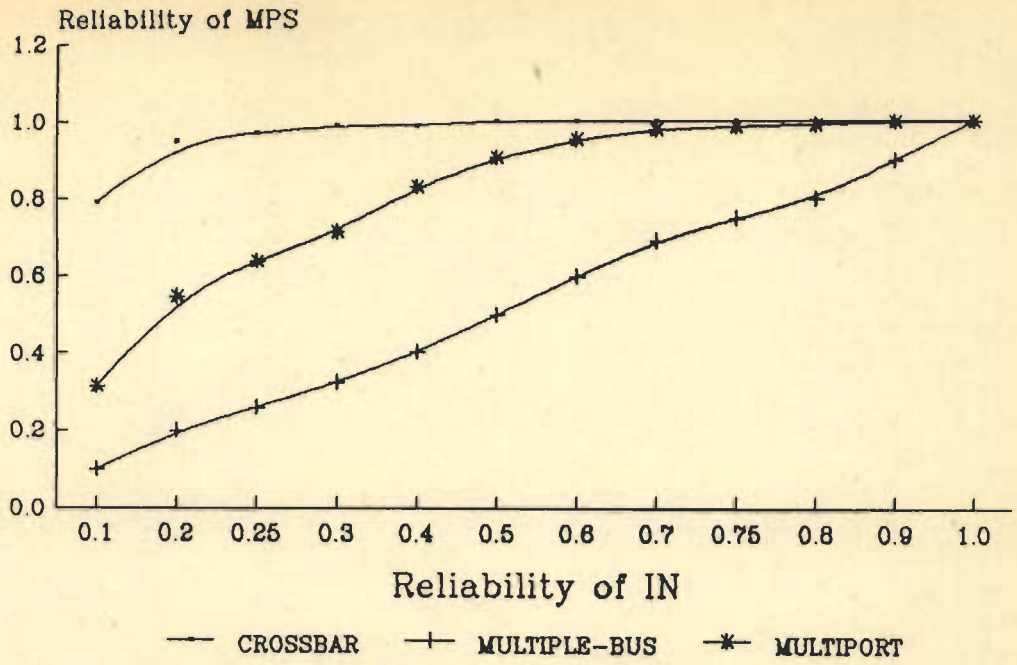


FIG.3.18 VARIATION OF REALIABILITY WITH RELIABILITY IN

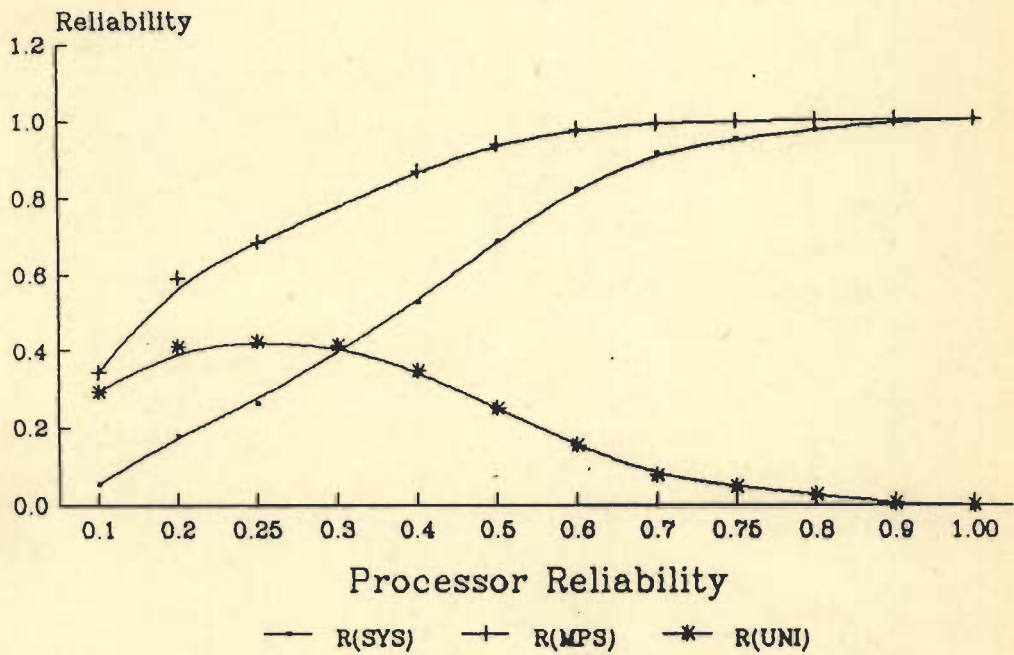


Fig.3.19 DEPENDENCE OF SYSTEM RELIABILITY CRITERIA ON PROCESSOR RELIABILITY

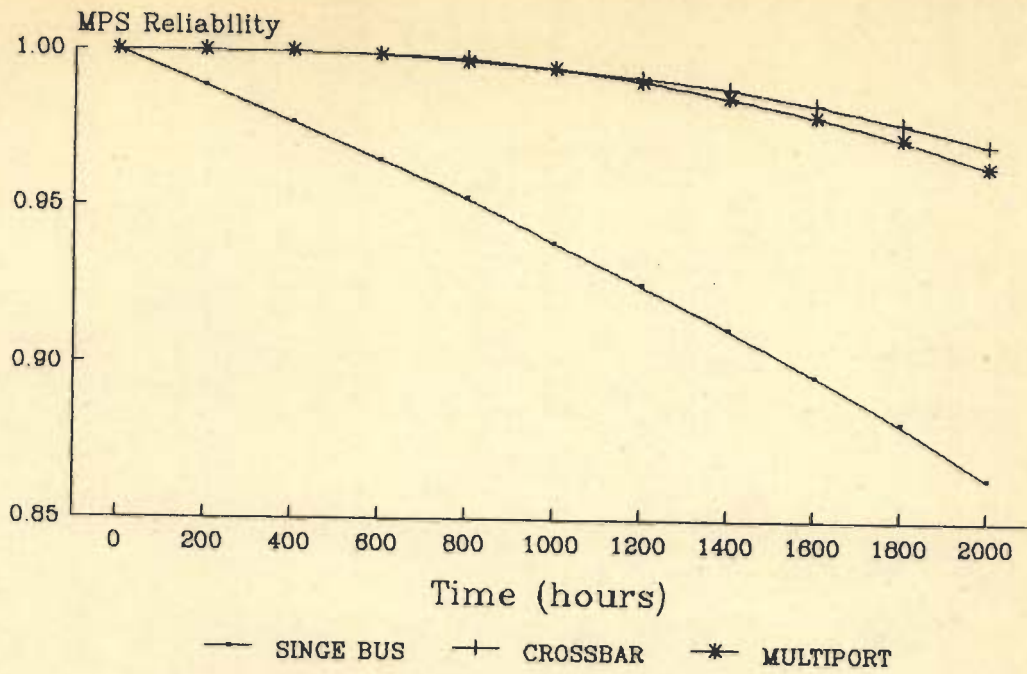


FIG.3.20 VARIATION OF MPS RELIABILITY
(ALPHA = 2, BETA = 1) WITH TIME

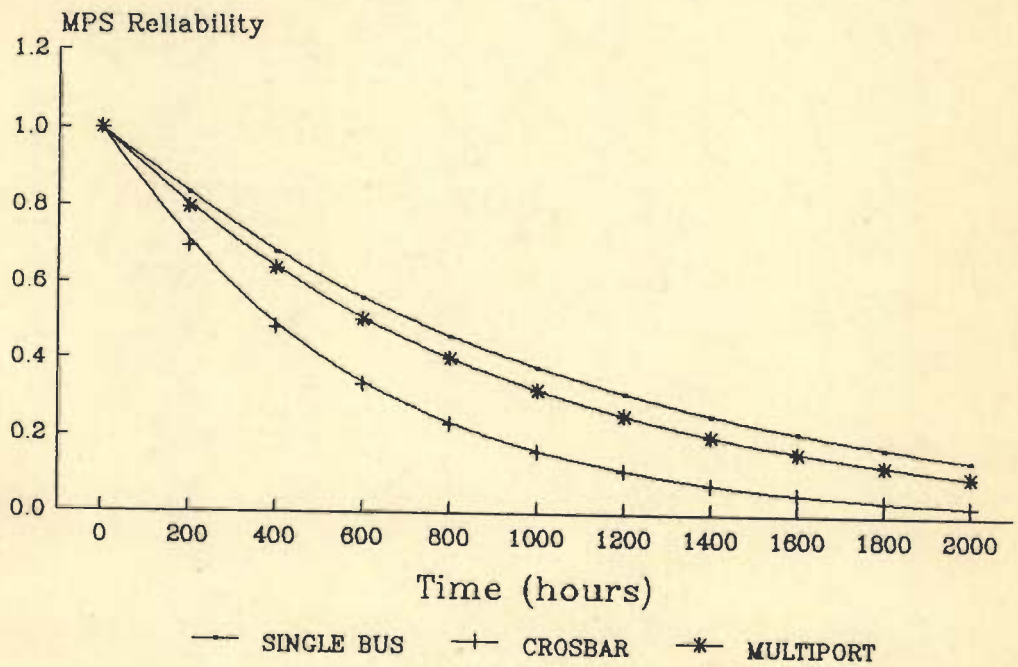


FIG.3.21 VARIATION OF MPS RELIABILITY
(ALPHA = BETA = 4) WITH TIME

lowest reliability when large time spans are considered. These results are in agreement with the results presented by Hwang et al [76].

3.3.2 Terminal Reliability

There are mainly three basic forms of connections through a network. A one-to-one connection passes information from one network port, the source, to another network port, the destination. Multiple one-to-one connections may be active simultaneously. Information flow from one source simultaneously to two or more destinations is supported by broadcast connection [119]. The reliability criteria discussed in Chapter II, SST, SMT, MST, and MMT can represent all the cases cited.

i) Multiple-bus system

a) SST reliability: The SST reliability, $R_B(\tau)[SST]$ of a multiple-bus system, is obtained by considering that exactly one of the n processors communicates exactly with one of the k memory modules through one of the buses, and is given by

$$R_B(\tau)[SST] = [H(P; n,1) - H(P; n,2)] * H(B; z,1) * [H(M; k,1) - H(M; k,2)] \quad (3.11)$$

where $[H(P; n,1) - H(P; n,2)]$ is the reliability that exactly one processor out of n is active in communication. Similarly $[H(M; k,1) - H(M; k,2)]$ is the reliability that exactly one memory module out of k is active.

b) SMT reliability: The SMT reliability, $R_B(\tau)$ [SMT], of a multiple-bus system, is obtained by considering that exactly one of the n processors communicates more than one (say y) memory modules, through one of the buses, and is given by

$$R_B(\tau)[SMT] = [H(P; n,1) - H(P; n,2)] H(B; z,1) * [H(M; k,y) - H(M; k,y+1)] \quad (3.12)$$

where $[H(M; k,y) - H(M; k,y+1)]$ is the reliability that exactly y memory modules out of k are active in communication.

c) MST reliability: The MST reliability, $R_B(\tau)$ [MST], of a multiple bus system is obtained by considering that more than one (say x) processors communicate with exactly one memory module through one of the buses, and is given by,

$$R_B(\tau)[MST] = [H(P; n,x) - H(P; n,x+1)] H(B; z,1) * [H(M; k,1) - H(M; k,2)] \quad (3.13)$$

where $[H(P;n,x) - H(P;n,x+1)]$ is the reliability that exactly x processors out of the n are active in communication.

d) MMT reliability: The MMT reliability, $R_B(\tau)$ [MMT] of a multiple-bus system is obtained by considering that more than one (say x) processor communicate with more than one (say y) memory module through one of the buses, and is given by

$$R_B(\tau)[MMT] = [H(P; n,x) - H(P; n,x+1)] H(B; z,1) * [H(M; k,y) - H(M; k,y+1)] \quad (3.14)$$

e) System reliability: The system reliability, $R_B(\tau)[MPS]$ of a multiple-system is obtained by considering that atleast one processor communicates with atleast one memory module, through one of the buses, and is given by,

$$R_B(\tau)[MPS] = H(P;n,1) H(B;Z,1) H(M;k,1) \quad (3.15)$$

ii) Crossbar system

The corresponding set of reliability expressions (3.11) - (3.15) for the crossbar are obtained by replacing the bus and memory terms by Eq. (3.5) with suitable values of j , and, accordingly, expressed in terms of H function as follows.

$$R_c(\tau)[SST] = [H(P;n,1) - H(P;n,2)][H(\phi;k,1) - H(\phi;k,2)] \quad (3.16)$$

$$R_c(\tau)[SMT] = [H(P;n,1) - H(P;n,2)][H(\phi;k,y) - H(\phi;k,y+1)] \quad (3.17)$$

$$R_c(\tau)[MST] = [H(P;n,x) - H(P;n,x+1)][H(\phi;k,1) - H(\phi;k,2)] \quad (3.18)$$

$$R_c(\tau)[MMT] = [H(P;n,x) - H(P;n,x+1)][H(\phi;k,y) - H(\phi;k,y+1)] \quad (3.19)$$

$$R_c(\tau)[MPS] = H(P;n,1) H(\phi;k,1) \quad (3.20)$$

iii) Multiport memory system

The corresponding set of reliability expressions of the multiport system are obtained directly from Eqs. 3.16 through 3.20 by substituting $\phi_j = z_j M_j$.

3.4 CONCLUSION

In this chapter the reliability evaluation of the multiple-bus, crossbar and multiport memory architectures are discussed. Expressions for multiprocessing reliability are presented using enumeration path technique. Algorithms for

reliability are presented considering the MPS as a t-out-of-s redundant system. The models are extended to compute the terminal reliability and multiprocessing reliability for the three architectures. The variation of multiprocessing reliability with time, also, has been studied.

CHAPTER IV

PERFORMANCE EVALUATION

Performance evaluation of an MPS measures the capability of the system in terms of parameters that represent major characteristics of a system application model. The model allows the user to quantify the effects of changing various system design parameters on such performance measures as: bandwidth (BW) and some fundamental measures such as hardware utilizations, waiting time, probability of acceptance etc.

Section 4.1 describes the behaviour of the multiprocessors under study and Section 4.2 reviews the relevant previous work in stochastic modeling of these systems. The Section 4.3 gives the set of assumptions on which the models are developed. In Section 4.4 memory interference models are analysed. Section 4.5 extends these models to compute other fundamental measures. Section 4.6 considers performance of Delta network and Section 4.7 summarizes the important contributions of the chapter.

4.1 CHARACTERISTICS OF MULTIPROCESSORS

A process associated with each processor can be in one of the three states: **running** on its processor, **waiting** to access a memory module, or **accessing** a memory module. The memory module can be in one of the two states: **busy**, when it is being accessed by a processor; and **idle**, when it is not being accessed.

The processing time between memory requests is the processor's interrupt time, which is assumed to be a geometric random variable. The parameter of that random variable is the memory request probability (MRP). The MRP is the probability that an actively executing processor will generate a request in the next memory cycle. If the MRP is same for all processors, then the processors are said to be statistically identical. The amount of time spent actively accessing memory per request is the memory access time. When the memory access time is constant, time is said to be divided into cycles and memory access time is sometimes called memory cycle time. The MRP of the particular processor is said to be 1, if it sends memory requests in every memory cycle. The distribution of access across the memory modules by a processor is that processor's memory access probabilities. There are two extreme cases of memory access. The first one is the uniform memory reference where the request of each processor is directed to a particular memory module chosen at random, but all memory modules equally likely to be chosen. The second is a general memory referencing, where the demand pattern of each processor is equivalent to a sequence of Bernoulli trials, i.e., each processor requests a different memory module with a different probability.

The processors are connected to the memory modules through any of the INs shown in Figs. 3.1, 3.2, 4.1 and 4.2. In case of $n \times k \times z$ multiple-bus system (Fig. 3.1), if $z \geq k$, the system is called a bus-sufficient system, otherwise it will be

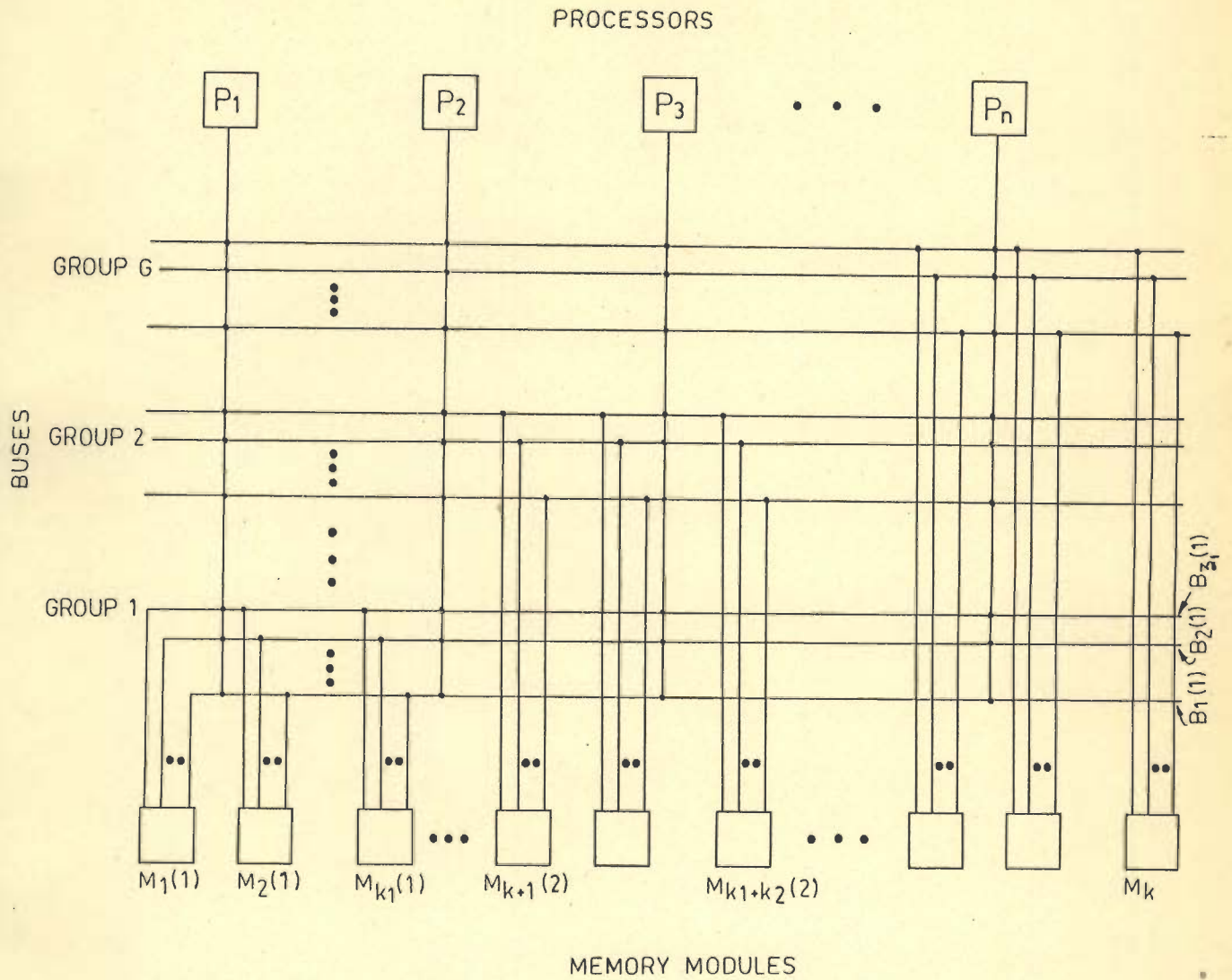


FIG. 4.1 AN $n \times k \times z$ PARTIAL-BUS MULTIPROCESSOR WITH G GROUPS.

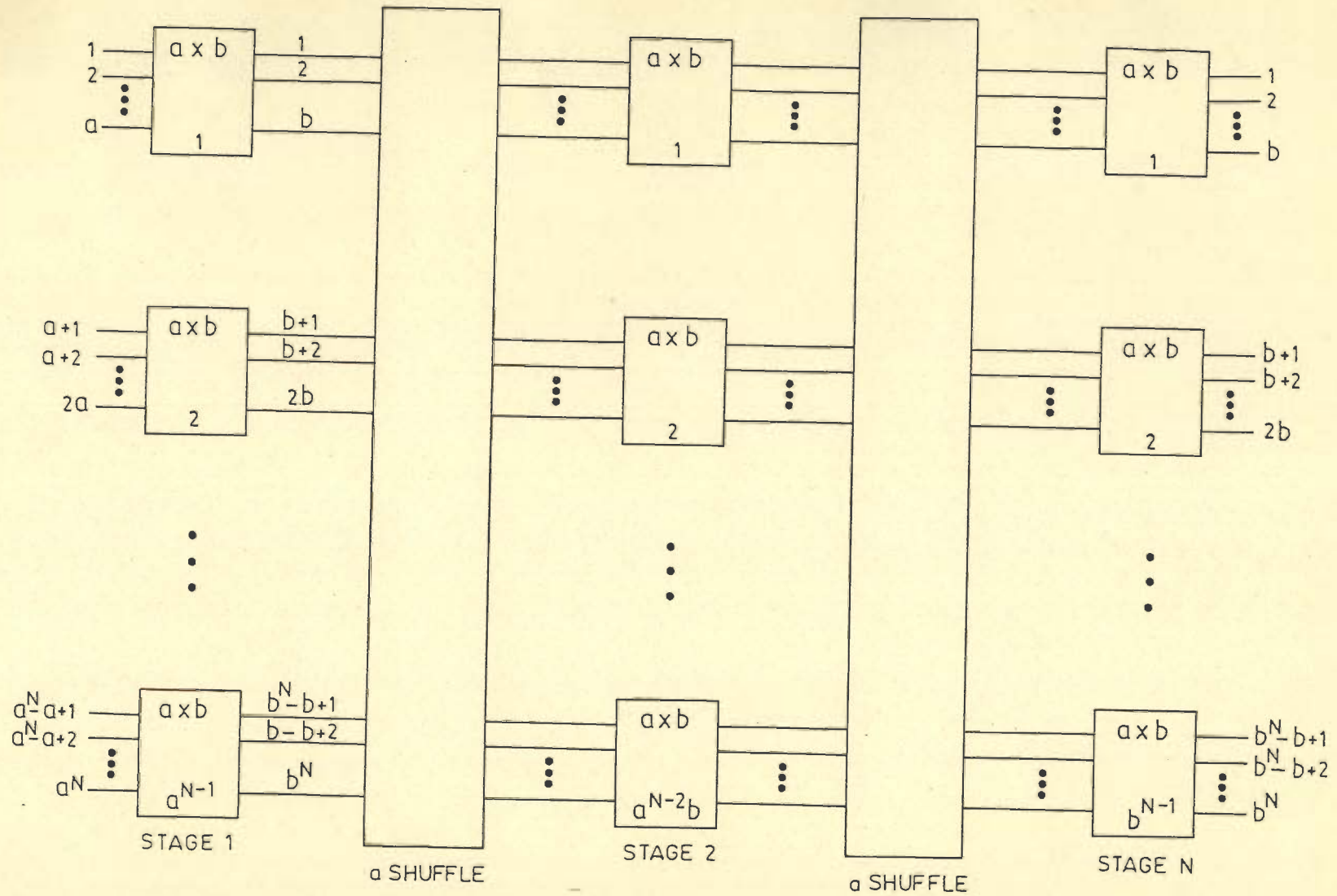


FIG. 4.2 AN $a^N \times b^N$ DELTA NETWORK

called a **bus-deficient system**. The cost of such an interconnection is $O(z(n + k))$. When $z = k$, the system behaves as a $n \times k$ crossbar (Fig. 3.2). The cost of such a crossbar is $O(n \times k)$. The cost of multiple-bus with $n/2$ buses is the same as that of a $n \times n$ crossbar. Fig. 4.1 depicts a partial bus architecture, having n processors, k memory modules and z buses. The memory modules are divided into G groups. The cost of a partial bus connection is only $O(z(n + (k/G)))$. A class of MINs, called Delta network, is a $n \times n$ network, whose cost is proportional to $O(n \log_2 n)$. Fig. 4.2 depicts a $a^N \times b^N$ Delta networks. The BW of a MPSs is defined as the expected number of memory modules busy in a cycle. The most important factors affecting the BW are MRP, the pattern of memory access, memory cycle time and the degree of conflicts that the processor requests experience due to network contention and memory contention.

4.2 REVIEW OF EXISTING MEMORY INTERFERENCE MODELS

There is extensive literature on the performance evaluation of MPSs. Bandwidth is a widely accepted performance index for synchronous multiprocessors. Several performance studies using different evaluation techniques have been reported for different interconnection networks under a variety of assumptions [17,34, 39-74]. The behaviour of memory interference in MPSs are modeled using crossbar [39-52] multiple-bus [17,34,58-74], partial-bus [17,59,60,71] and MINs [47,55,79,107-119].

The analysis of crossbar and multiple-bus MPSs has been attempted using simulation techniques [17,41,58,62,63], mathematical models [39,47,52-55,57,59,61,73], queuing models [40,42,50-53,58,63,70,72] and Markov-chains [34,42,43,49,54,55,65-69]. Out of these, some results of the probabilistic models coincide with simulation models [43,53,58] and some are in closed form [47,53,55,57,59,60] and unfortunately some are applicable only to systems with small number of processors and memory modules [43,49-51,53,56,58]. Yen et al [39] proposed a classification for models according to the approach used in their formulation.

Some researchers presented models for systems with interleaved memory modules [40,42,50-52]. The results for bandwidth of a crossbar obtained by Rau [50] are within a maximum error of 0.25 percent. Baskett and Smith [51] have given asymptotic results and also presented results of simulations with real programs.

The models are further classified according to the assumptions they make with respect to MRP, memory access pattern and memory cycle time. Holliday et al [58] have presented a good summary of the most of the existing models and presented exact results, using petrinet model. However, their approach, as claimed by them, requires to build complete state space and hence many interesting models can not be studied because of their requirement for a large state space.

Many authors deal with basic case of constant memory access time and uniform access probabilities assuming statistically identical processors, using MRP either equal to unity [43,46,48,49,51-54] or less than unity [17,39,57,59,60-63]. Bhandarkar [53] attempted an exact analysis with MRP of 1 and as this analysis leads to larger state space, he presented only approximate analysis for $MRP < 1$. Ravi's [52] some-what complicated combinatorial result, developed using sterling numbers, has been shown that the MRP is exactly equal to 1 by Chang et al [41]. The continuous time Markov-chain models of Marsan [65,66] and Irani and Onyuksel [69] assumed an exponentially distributed access time.

Some researchers with identical processors considered non-uniform memory access probabilities [45,46,49,55-57,63,72,73]. The exact solution method of Du and Baer [46] is a modification of Bhandarkar's exact method. Towsley [72] developed a model based on approximate queue analysis, that gives BW predicted generally within 1% of simulation results at the cost of considerable computation. Most of these analyses except the queuing models are based on the assumption where an unsuccessful request is considered lost and are not resubmitted.

The model of Bhuyan [61] employed Stirling numbers and the BW formulae are computationally complex compared to those given by Mudge et al [60] and Goyal et al [62]. Bhuyan [55] and Das and Bhuyan [59] presented non-uniform models with identical

processors and these are not as generalized as Bernoulli trials. Modeling the memory access process as a Bernoulli process has been attempted in [51,53,56,57] and is widely used as a basis for memory reference models. Hoogendoorn [56] could not present closed form solutions. Mudge et al [57] obtained BW equations for a crossbar with non-identical processors.

4.3 ASSUMPTIONS

Analytic models are presented in this chapter with the following assumptions:

1. Processors requests for memory modules are independent.
2. At the beginning of a memory cycle a processor i makes a request to access a memory module with probability r_i , $1 \leq i \leq n$. For systems with statistically identical processors $r_i = r$ for all i .
3. The demand pattern of each processor is equivalent to a sequence of Bernoulli trials, i.e., an access request from processor i , if made, is directed to memory module j , with a probability p_{ij} . Thus, the access rate from processor i to memory module j is defined as $q_{ij} = r_i \times p_{ij}$.
4. If more than one processor issues a request to a particular idle memory module, that memory will choose a processor at random to get the connection. The other requests are rejected.
5. If more than one processor seeks the same interconnection

link for memory access, arbitrarily one of the requests is allowed to pass through and the remaining are rejected.

6. The requests generated by each processor in successive cycles are independent of requests issued in the previous cycles.

The sixth assumption is unrealistic because a rejected request will indeed be resubmitted in the next cycle. However, this assumption leads to simpler analysis, and it does not result in a substantial difference in actual results [53].

When there are no conflicts, either due to bus interference or memory interference, the expected number of requests for shared memory is given by $\sum r_i$. This is referred to as requested bandwidth, BW_{CF} . Therefore

$$BW_{CF} = \sum_{i=1}^n r_i \quad (4.1)$$

where subscript CF refers to the conflict free condition.

In the presence of conflicts not all these requests make successful memory accesses and hence the actual BW is always less than BW_{CF} . The bandwidth analysis will be done in two parts corresponding to two types of interferences, bus interference and memory interference [60,64].

4.4 MEMORY INTERFERENCE-ANALYSIS

Let E_j be the event that there is atleast one request for the memory module M_j . The probability of acceptance, x is

the probability that atleast one request gains access to a given memory module. If the module M_j is under consideration, then-

$$\text{Pr } [E_j] = x_j = q_{1j} + \bar{q}_{1j} q_{2j} + \dots + \bar{q}_{1j} \bar{q}_{2j} \dots \bar{q}_{(n-1)j} q_{nj}$$

or, using H function of Chapter III, we have

$$x_j = H(Q_j ; n, 1) \quad (4.2)$$

where $Q_j = \{q_{ij} \mid j = \text{constant}\}$, represents probabilities with which different processors access memory module j .

If the events E_j , $1 \leq j \leq k$, are assumed to be independent [64], the expected number of memory modules, in a bus sufficient system (for all values of $z \geq k$)-

$$BWS = \sum_{j=1}^k x_j = \sum_{j=1}^k H(Q_j ; n, 1) \quad (4.3)$$

where suffix 'S' refers bus-sufficient condition.

Bus Interference Analysis

The assumption that E_j 's are independent allows us to express $E(i)$, the probability that exactly i memory modules are accessed in a memory cycle. This can be determined by taking probability of accessing atleast i memory modules and subtracting from it the probability of accessing atleast $(i+1)$ memory modules [93], and is given by

$$E(i) = H(X; k, i) - H(X; k, i+1)$$

Where $X = \{x_j\}$, $1 \leq j \leq k$

In the case where $i \leq z$, there are sufficient buses to

handle memory requests without any conflict for memory access. In case where $i > z$, only z of these requests are accepted and rest will be rejected. Hence the bandwidth of the system, can be expressed as

$$BW = \sum_{i=1}^z i E(i) + z \sum_{i=z+1}^n E(i) \quad (4.4)$$

where the two terms on the righthand side correspond to the two cases, $i \leq z$ and $i > z$.

On simplification, equation (4.4) reduces to

$$BW_{nkz} = \sum_{i=1}^z H(X; k, i) \quad (4.5)$$

where

BW_{nkz} is the BW of a $n \times k \times z$ multiple-bus system.

For a crossbar system, the equation for bandwidth, BW_{nk} , is obtained by substituting $z = k$ in (4.5).

$$BW_{nk} = \sum_{i=1}^k H(X; k, i) = \sum_{i=1}^k x_j = BW_s \quad (4.6)$$

The expressions (4.5) and (4.6) for BW computation are very general. These are valid for all system configurations, i.e., for all values of n, k and z .

4.4.1 Uniform Reference Model(URM)

If all memory modules are accessed with equal probability, we get a uniform reference model, i.e., an equally likely case.

$p_{ij} = p = 1/k$ for all i and j . Hence in an equally likely case $x_j = x$ for all j , $1 \leq j \leq k$ and equations (4.5) and (4.6) are suitably modified as

$$BW_{nkz} = \sum_{i=1}^z H(X;n,1), \text{ where } X = \{x_j\} = \{x\} \quad (4.7)$$

$$BW_{nk} = k x \quad (4.8)$$

$$\text{Where } x = p(1 + \bar{p} + \bar{p}^2 + \dots + \bar{p}^{(n-1)}) = p \sum_{f=1}^n (\bar{p}^{f-1})$$

Table 4.1 shows the available simulation results presented in [17] for a bus system. The bandwidth is calculated for various values of z , n and k , with r ($r_i = r$ for all i) the independent processor request rate, assigned the values 1.0 and 0.5. The data here clearly indicates, especially when $r=0.5$, that BW changes very little after z reaches $k/2$ [Fig. 4.3]. Table 4.2 shows the BW predicted by equation (4.7). The difference between the simulation results and analytic values (Table 4.2) is presented in Table 4.3. It can be seen that except for small values of $n(=k)$, this difference is less than about 10 percent indicating reasonable good agreement between our results and those of [17]. Tables 4.4 and 4.5 show bandwidth values of crossbar calculated from equation (4.8) with $r = 1.0$ and $r = 0.5$ respectively. The simulation results presented in [53,55] and the corresponding percentage deviations of our results appear in Table 4.6. The analytic results are within 9 percent of simulation results. In view of results presented in Tables 4.2, 4.3 and 4.5 it is seen that the multiple-bus system with a number

TABLE 4.1

BW OF $n \times k \times z$ MULTIPLE-BUS MPS OBTAINED BY SIMULATION (URM)

NO. OF BUSES (z)	4X4		8X8		12X12		16X16	
	r=1	r=0.5	r=1	r=0.5	r=1	r=0.5	r=1	r=0.5
1	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
2	1.97	1.65	2.00	2.00	2.00	2.00	2.00	2.00
3	2.55	1.77	3.00	2.87	3.00	3.00	3.00	3.00
4	2.62	1.77	3.93	3.33	4.00	3.95	4.00	4.00
5			4.62	3.45	4.99	4.67	5.00	4.98
6			4.90	3.47	5.93	5.03	6.00	5.85
7			4.94	3.47	6.68	5.13	6.98	6.43
8			4.95	3.47	7.12	5.16	7.92	6.70
9					7.27	5.16	8.72	6.82
10					7.28	5.16	9.27	6.83
11					7.30	5.16	9.27	6.83
12					7.30	5.16	9.61	6.83
13							9.63	6.84
14							9.63	6.84
15							9.63	6.84
16							9.63	6.84

TABLE 4.2
 BW OF $n \times k \times z$ MULTIPLE-BUS CALCULATED FROM EQ.4.7 (URM)

NO. OF BUSES	4 X 2		8 X 2		8 X 4		2 X 4		2 X 8		4 X 8	
	r = 1	r=0.5	r = 1	r=0.5	r = 1	r=0.5	r = 1	r=0.5	r = 1	r=0.5	r = 1	r=0.5
1	0.999	0.91	1.00	0.99	1.00	0.99	0.97	0.70	0.97	0.70	0.99	0.913
2	1.589	1.21	1.83	1.55	1.80	1.54	1.70	0.82	1.29	0.82	1.55	1.209
3					2.19	1.68	1.87	0.83	1.34	0.83	1.72	1.257
4					2.27	1.69	1.98	0.83	1.35	0.83	1.76	1.262
5									1.35	0.83	1.76	1.262
6									1.35	0.83	1.76	1.262
7									1.35	0.83	1.76	1.262
8									1.35	0.83	1.76	1.262
9												
10												
11												
12												
13												
14												
15												
16												

TABLE 4.2 (Contd.)

NO. OF BUSES	2 X 2		4 X 4		8 X 8		12 X 12		16 X 16	
	r = 1.0	r = 0.5	r = 1.0	r = 0.5	r = 1.0	r = 0.5	r = 1.0	r = 0.5	r = 1.0	r = 0
1	0.938	0.684	0.990	0.882	1.000	0.984	1.000	0.998	1.000	1.000
2	1.500	0.875	1.893	1.431	1.997	1.881	2.000	1.978	2.000	1.996
3					2.974	2.572	3.000	2.895	3.000	2.997
4					3.875	2.986	3.993	3.669	4.000	3.910
5					4.595	3.165	4.967	4.231	4.998	4.740
6					5.038	3.217	5.880	4.566	5.991	5.406
7					5.217	3.226	6.663	4.724	6.965	5.874
8					5.251	3.226	7.240	4.781	7.891	6.153
9							7.582	4.796	8.718	6.292
10							7.730	4.799	9.388	6.348
11							7.771	4.799	9.857	6.367
12							7.777	4.799	10.129	6.372
13									10.253	6.373
14									10.293	6.373
15									10.302	6.373
16									10.303	6.373

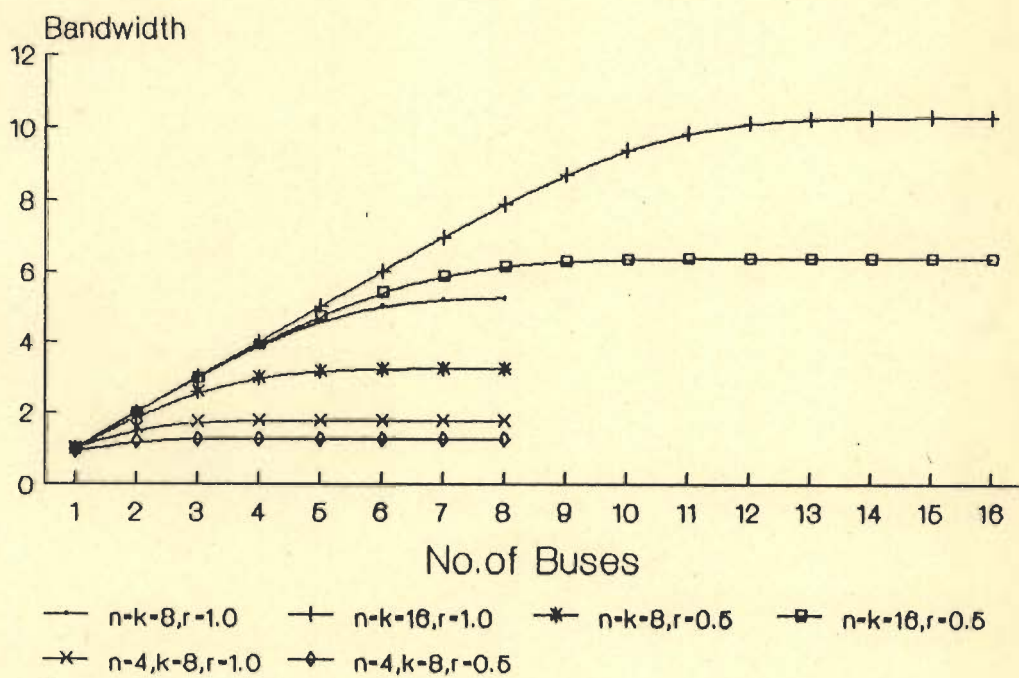


FIG. 4.3 BW OF $n \times k \times z$ MULTIPLE-BUS Vs. NO. OF BUSES WITH $r=1$ & 0.5 (URM)

TABLE 4.3

PERCENTAGE DIFFERENCE BETWEEN CALCULATED (TABLE 4.2) AND SIMULATED VALUES (TABLE 4.1) OF
BW OF A MULTIPLE-BUS (URM)

NO. OF BUSES	4 X 4		8 X 8		12 X 12		16 X 16	
	r = 1.0	r = 0.5	r = 1.0	r = 0.5	r = 1.0	r = 0.5	r = 1.0	r = 0.5
1	1.000	11.800	0.000	1.600	0.000	0.200	0.000	0.000
2	3.908	1.327	0.150	5.950	0.000	1.100	0.000	0.200
3	1.333	8.135	0.866	10.383	0.000	3.500	0.000	0.766
4	-4.351	6.497	1.399	10.330	0.175	7.113	0.000	2.250
5			0.541	8.260	0.460	9.400	0.040	4.819
6			-2.816	7.291	0.843	9.224	0.015	7.589
7			-5.600	7.031	0.254	7.914	0.214	8.646
8			-6.060	7.031	-1.685	7.344	0.366	8.164
9					-4.291	7.054	0.022	7.741
10					-6.181	6.996	-1.272	7.057
11					-6.452	6.996	-3.431	6.778
12					-6.534	6.996	-5.401	6.705
13							-6.469	6.828
14							-6.884	6.828
15							-6.978	6.828
16							-6.938	6.828

TABLE 4.4

BW OF $n \times k$ CROSSBAR MPS OBTAINED FROM EQ. 4.8 WITH $r=1.0$ (URM)

n	k	2	4	8	16	32	64	128	256	512	1024
2		1.500	1.750	1.875	1.938	1.969	1.984	1.992	1.996	1.998	1.999
4		1.875	2.734	3.311	3.640	3.816	3.907	3.953	3.997	3.988	3.994
8		1.992	3.600	5.251	6.453	7.178	7.576	7.785	7.892	7.946	7.973
16		2.000	3.960	7.056	10.303	12.745	14.255	15.056	15.540	15.768	15.883
32		2.000	4.000	7.889	13.971	20.414	25.335	28.411	30.136	31.050	31.520
64		2.000	4.000	7.999	15.743	27.805	40.641	50.516	56.725	60.217	62.070
128		2.000	4.000	8.000	15.996	31.450	55.474	81.096	100.880	113.352	120.378
256		2.000	4.000	8.000	16.000	31.991	62.864	110.813	162.007	201.608	226.605
512		2.000	4.000	8.000	16.000	32.000	63.980	125.692	221.490	323.830	403.064
1024		2.000	4.000	8.000	16.000	32.000	64.000	127.985	251.348	442.844	647.475

TABLE 4.5

BW OF $n \times k$ CROSSBAR MPS OBTAINED FROM EQ. 4.8 WITH $r=0.5$ (URM)

n	k	2	4	8	16	32	64
2		0.875	0.938	0.969	0.984	0.992	0.996
4		1.367	1.655	1.820	1.908	1.954	1.977
8		1.800	2.626	3.226	3.589	3.783	3.892
16		1.980	3.528	5.151	6.373	7.128	7.548
32		2.000	3.944	6.986	10.207	12.668	14.206
64		2.000	3.999	7.871	13.903	20.320	25.258

TABLE 4.6
 BW OF $n \times k$ CROSSBAR
 $r=1.0$, URM
 (THEORETICAL, SIMULATION & %ERROR)

NO. OF PROCESSORS ($n = k$)	SIMULATION RESULTS	THEORETICAL RESULTS	%ERROR
2	1.50	1.500	0.00
4	2.61	2.734	-4.75
8	4.91	5.251	-6.95
16	9.72	10.303	-6.00
35	19.06	20.414	-6.91
64	37.33	40.641	-8.87
128	75.13	81.096	-7.94
256	149.93	162.007	-8.06
512	300.41	323.830	-7.80

of buses $z=(k/2)+1$ produces a BW very nearly equal to that obtained with the crossbar. It can also be observed that the results presented are close to the simulation results of [53] when $k > n$ than when $k < n$, like in the other models [54,55,57,59,60]. From the exact Markov-chain we know that the performances of $i \times j$ and $j \times i$ systems are almost equal [53]. Hence when $k < n$, we write

$$BW = \sum_{i=1}^n H(Y;n,i) \quad (4.9)$$

Where $Y = \{y_i\}$, $1 \leq i \leq n$, and y_i is the probability that the request from the processor i is accepted by one of the memory modules, and is given by

$$Y_i = H(Q_i ; k,1), \text{ where } Q_i = \{q_{ij} \mid i = \text{constant}\}$$

Fig. 4.4 shows the effect of adding a memory module on the bandwidth of a crossbar with $n = 2,4,8$ and 16. Fig. 4.5 illustrates how the bandwidth of crossbar varies with r .

4.4.2 Local Reference Model

In real system very often each processor will have a local or preference memory which will be accessed with a higher probability than other memory modules. We consider two types of local reference models [74].

Unbalanced Memory Reference Model (UBRM)

In this type of local reference, all processors have a preference for a certain memory module, say, m_f , $1 \leq f \leq k$,

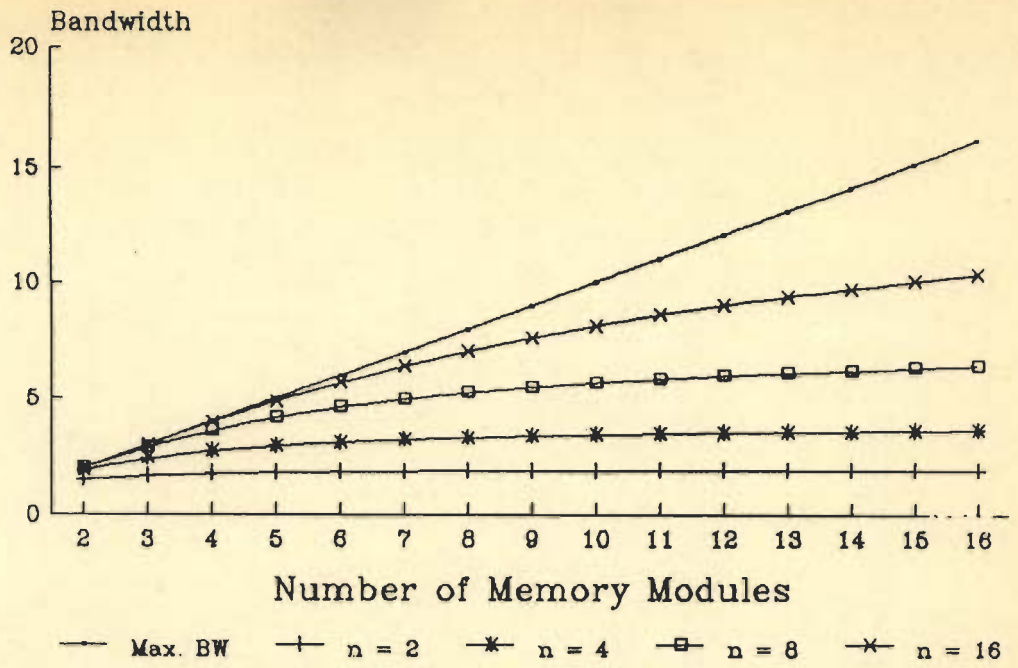


FIG. 4.4 BW OF CROSSBAR VS NO.OF MEMORY MODULES WITH $r = 1.0$ (URM)

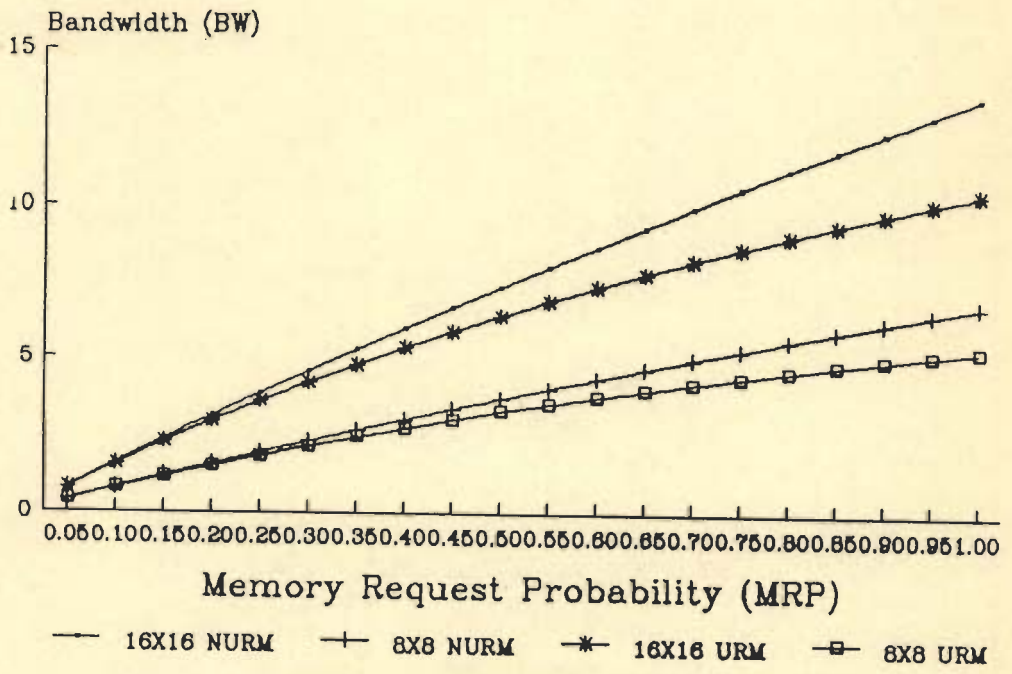


FIG. 4.5 BW OF A CROSSBAR Vs MEMORY REQUEST PROBABILITY (NURM, URM)

selecting it with probability α and request other memory modules with equal probabilities. Then

$$\left. \begin{aligned} p_{ij} &= \alpha ; j = f \\ p_{ij} &= (1-\alpha) / (k-1), ; j \neq f \end{aligned} \right\} \quad (4.10)$$

Table 4.7 shows some results obtained with $r = 0.5$ and 1.0 for a multiple-bus system using equations (4.5) and (4.10).

It can be observed that the BW predicted by UBRM is almost equal to that of URM with $z=1$. The data here clearly indicates that BW changes very little after z reaches $k/2$. It can also be concluded that the performance of UBRM is significantly poorer than that of a URM.

Table 4.8 gives the BW of crossbar predicted by equations (4.6) and (4.10) with r assigned the values 1.0 and 0.5 . Fig. 4.6 shows the BW of a crossbar plotted as a function of α for various values of n and k . This figure clearly demonstrates that for a given MPS the BW_{nk} falls as α increases from $1/k$ to 1 . The BW_{nk} also falls as α decreases from $1/k$ to 0 .

It is evident from the Fig. 4.6 that the performance of MPS would be improved by making α close to 0 and over the range $\alpha > 0.75$, the variation of BW is small. The results suggested by the approximate model described by Sethi and Deo [49] records the maximum bandwidth when $\alpha = 0$, which, they considered as the equally likely case.

TABLE 4.7

BANDWIDTH OF $n \times k \times X \times z$ MULTIPLE-BUS MPS WITH ALPHA=0.8 (UBRM)

NO. OF BUSES	2 X 2		4 X 2		8 X 2		8 X 4		2 X 4		4 X 4		2 X 8		4 X 8		8 X 8		16 X 16	
	$r = 1.0$	$r = 0.5$	$r = 1.0$	$r = 0.5$	$r = 1.0$	$r = 0.5$	$r = 1.0$	$r = 0.5$	$r = 1.0$	$r = 0.5$	$r = 1.0$	$r = 0.5$	$r = 1.0$	$r = 0.5$	$r = 1.0$	$r = 0.5$	$r = 1.0$	$r = 0.5$	$r = 1.0$	$r = 0.5$
1	0.974	0.708	0.999	0.915	1.000	0.993	1.000	0.993	0.974	0.706	0.999	0.914	0.973	0.706	0.999	0.913	1.000	0.993	1.000	1.000
2	1.680	0.830	1.589	1.214	1.832	1.553	1.809	1.542	1.301	0.828	1.562	1.210	1.296	0.828	1.555	1.209	1.803	1.539	1.960	1.799
3							2.196	1.683	1.345	0.837	1.708	1.249	1.349	0.838	1.728	1.257	2.245	1.709	2.777	2.258
4							2.273	1.696	1.347	0.837	1.722	1.251	1.354	0.839	1.761	1.262	2.405	1.740	3.353	2.448
5													1.354	0.839	1.765	1.262	2.443	1.744	3.680	2.506
6													1.354	0.839	1.765	1.262	2.448	1.744	3.827	2.519
7													1.354	0.839	1.765	1.262	2.449	1.744	3.880	2.522
8													1.354	0.839	1.765	1.262	2.449	1.744	3.895	2.522
9																			3.898	2.522
10																			3.899	2.522
11																			3.899	2.522
12																			3.899	2.522
13																			3.899	2.522
14																			3.899	2.522
15																			3.899	2.522
16																			3.899	2.522

TABLE 4.8

BW OF $n \times k$ CROSSBAR MPS WITH ALPHA=0.8 (UBRM)

k	2		4		8		16		32		64	
	r = 1.0	r = 0.5	r = 1.0	r = 0.5	r = 1.0	r = 0.5	r = 1.0	r = 0.5	r = 1.0	r = 0.5	r = 1.0	r = 0.5
2	1.320	0.830	1.347	0.837	1.354	0.839	1.357	0.839	1.359	0.840	1.360	0.840
4	1.589	1.214	1.722	1.251	1.765	1.262	1.783	1.266	1.791	1.269	1.795	1.270
8	1.832	1.553	2.273	1.696	2.449	1.744	2.527	1.765	2.564	1.774	2.582	1.779
16	1.972	1.814	3.005	2.256	3.598	2.439	3.899	2.522	4.050	2.562	4.125	2.581
32	2.000	1.966	3.670	2.986	5.232	3.583	6.238	3.890	6.799	4.045	7.095	4.123
64	2.000	1.999	3.964	3.657	6.905	5.213	9.647	6.224	11.514	6.791	12.600	7.090

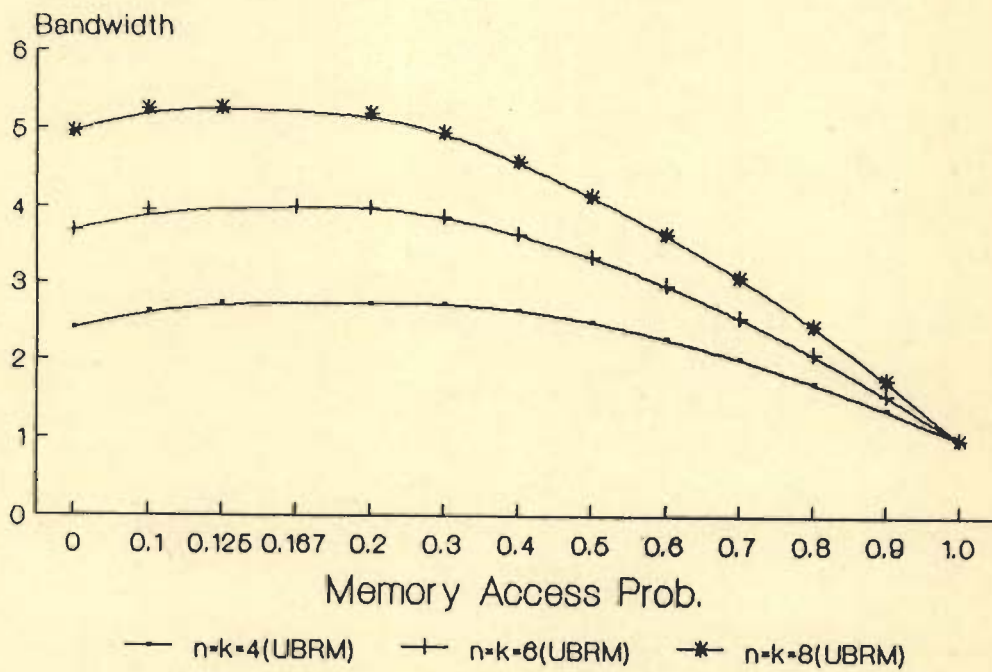


FIG.4.6 BW OF CROSSBAR Vs. MEMORY ACCESS PROB. WITH $r=1$ (NURM, UBRM)

Non-Uniform Reference Model (NURM)

In another example of favourite memory each processor may be biased towards a different memory module, say P_i is biased to M_i . This is referred to as nonuniform memory reference model (NURM). In NURM

$$\left. \begin{aligned} p_{ij} &= m, 0 \leq m \leq 1.0 && ; i = j \\ p_{ij} &= 1/k && ; i > k \\ p_{ij} &= (1-m)/(k-1) && ; i \neq j \end{aligned} \right\} \quad (4.11)$$

Tables 4.9 and 4.10 give the BW of $n \times k \times z$ multiple-bus and $n \times k$ crossbar respectively, with $r = 1.0$ and $r = 0.5$ for a fixed $m = 0.8$, computed using equations (4.5), (4.6) and (4.11).

For $r = 1.0$ a processor requests a particular memory most of the time, thereby reducing memory access conflicts. The BW of the system is then very much bus dependent. The results indicate that the number of buses for a multiple-bus should be determined by taking both r and m into consideration. When r is less than 1, there is lot of flexibility for selecting only the required number of buses in multiple-bus system. The crossbar in this case is clearly underutilized and therefore for the same degradation of performance, fewer buses than for $r=1$ are required. The Fig. 4.7 shows the variation of BW of a $n \times k$ crossbar with m . The minimum BW is recorded when $m = 1/k$, and BW improves as m increases from $1/k$ to 1, the variation is opposite to that of a UBRM. It is evident from the figure that

TABLE 4.9

BN OF $n \times n \times n$ MULTIPLE-BUS MPS WITH $\mu=0.8$ (NURM)

NO. OF BUSES	2 X 2		4 X 2		8 X 2		8 X 4		2 X 4		4 X 4		2 X 8		4 X 8		8 X 8		16 X 16	
	r=1.0	r=0.5	r=1.0	r=0.5	r=1.0	r=0.5	r=1.0	r=0.5	r=1.0	r=0.5	r=1.0	r=0.5	r=1.0	r=0.5	r=1.0	r=0.5	r=1.0	r=0.5	r=1.0	r=0.5
1	0.974	0.708	0.998	0.908	1.000	0.991	1.000	0.990	0.974	0.706	0.999	0.914	0.973	0.706	0.999	0.981	1.000	0.993	1.000	1.000
2	1.680	0.920	1.920	1.393	1.995	1.808	1.999	1.892	1.709	0.946	1.984	1.536	1.716	0.953	1.986	1.550	2.000	1.934	2.000	2.000
3							2.985	2.512	1.874	0.970	2.858	1.788	1.920	0.988	2.881	1.839	3.000	2.727	3.000	2.993
4							3.506	2.729	1.894	0.971	3.350	1.832	1.947	0.987	3.482	1.914	3.995	3.267	4.000	3.969
5													1.949	0.987	3.673	1.924	4.967	3.542	5.000	4.892
6													1.949	0.987	3.702	1.925	5.838	3.538	6.000	5.711
7													1.949	0.987	3.705	1.925	6.454	3.558	7.000	6.367
8													1.949	0.987	3.705	1.925	6.694	3.660	8.000	6.827
9																			8.998	7.103
10																			9.989	7.239
11																			10.954	7.294
12																			11.848	7.312
13																			12.569	7.316
14																			13.089	7.317
15																			13.334	7.317
16																			13.384	7.317

TABLE 4.10

BW OF $n \times k$ CROSSBAR MPS WITH $m=0.8$ (NURM)

k	2		4		8		16		32		64	
	r = 1.0	r = 0.5	r = 1.0	r = 0.5	r = 1.0	r = 0.5	r = 1.0	r = 0.5	r = 1.0	r = 0.5	r = 1.0	r = 0.5
2	1.680	0.920	1.884	0.971	1.949	0.987	1.976	0.994	1.988	0.997	1.994	0.999
4	1.920	1.393	3.350	1.832	3.705	1.925	3.859	1.965	3.931	1.983	3.996	1.992
8	1.995	1.808	3.794	2.723	6.694	3.660	7.358	3.836	7.682	3.920	7.842	3.960
16	2.000	1.981	3.979	3.563	7.551	5.410	13.384	7.317	14.670	7.660	15.330	7.831
32	2.000	2.000	4.000	3.949	7.947	7.078	15.068	10.775	26.764	14.630	29.297	15.309
64	2.000	2.000	4.000	4.000	7.999	7.883	15.881	14.108	30.104	21.506	53.524	29.257

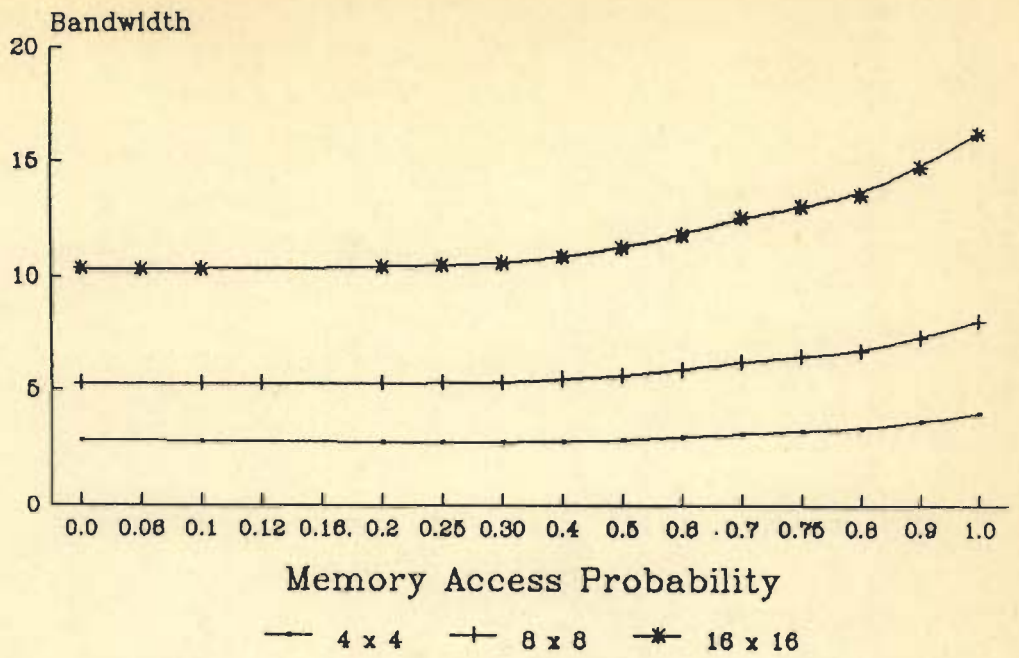


FIG. 4.7 BW OF CROSSBAR VS MEMORY ACCESS PROB.(m)

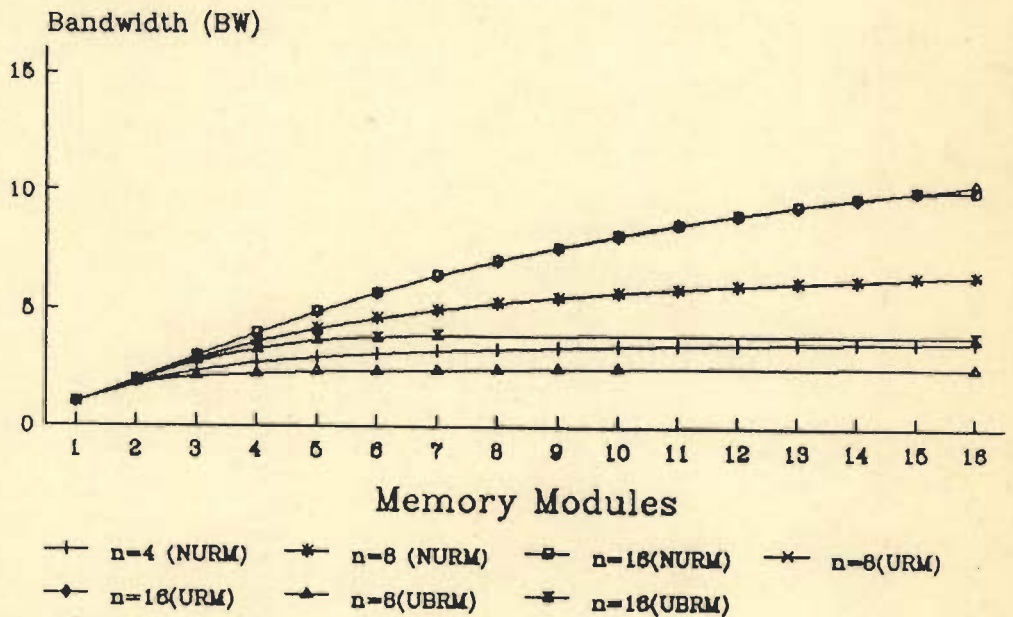


FIG. 4.8 BW OF $n \times K$ CROSSBAR VS NUMBER OF MEMORY MODULES, ALPHA = $m = 0.8$ (URM, UBRM, NURM)

the performance of MPS would be improved by selecting m close to 1. However, in the range $0.8 \leq m \leq 1.0$, the variation of BW is small. Fig. 4.8 compares the variations of bandwidth of the these three cases of $n \times k$ crossbar with number of memory modules. The variation of BW of a $16 \times 16 \times z$ multiple buses system with number of buses in the three models. URM, UBRM and NURM are shown in Fig. 4.9 for comparison. The BW variation of URM and NURM modules, of a crossbar system, with MRP are compared in Fig. 4.5.

4.4.3 Partial Bus System

In this section we generalize the BW expression, derived for multiple-bus, to cover the partial bus architecture, which is shown in Fig. 2.3 and its graphical representation is in Fig. 4.10. The k memory modules are divided into G groups and in the g th group, z_g buses are used to connect k_g memory modules. In each group the memory modules and buses are sequentially numbered. $B_i(g)$ and $M_f(g)$ respectively denote the bus i and memory module f of the g th group, for $1 \leq i \leq z_g$ and $f \leq j \leq k_g$ and

$$\sum_{g=1}^G z_g = z \quad \text{and} \quad \sum_{g=1}^G k_g = k$$

The probability, $x_f(g)$, that atleast one of the processors successfully accesses memory module f of the g th group.

$$x_f(g) = H(Q_f(g); n, 1) \tag{4.12}$$

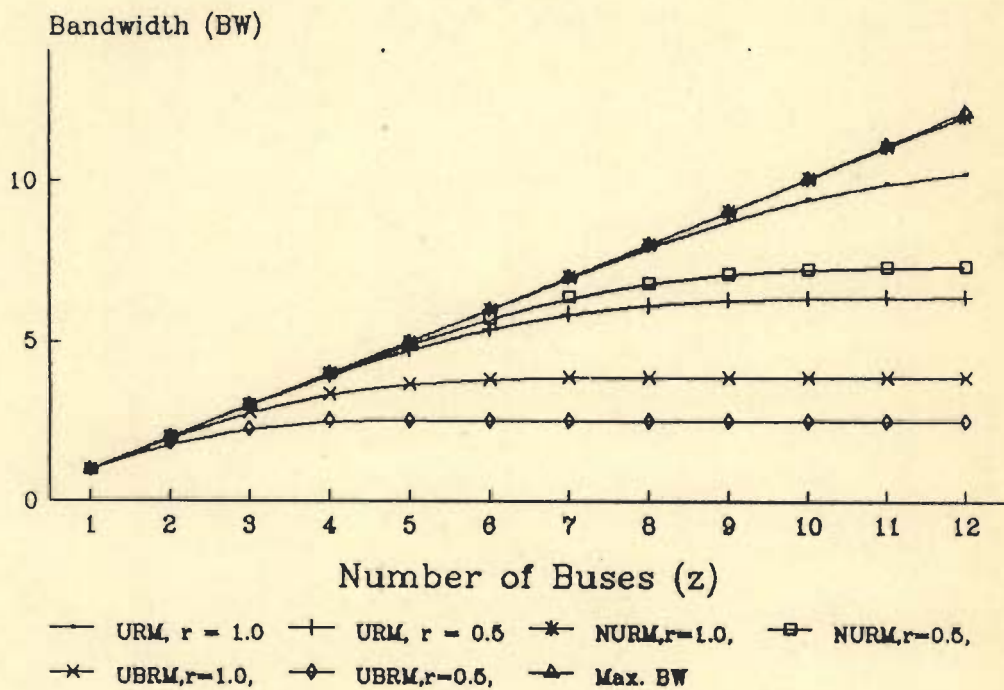


Fig. 4.9 VARIATION OF BW OF 16X16Xz MPS WITH NUMBER OF BUSES (URM, UBRM, NURM)

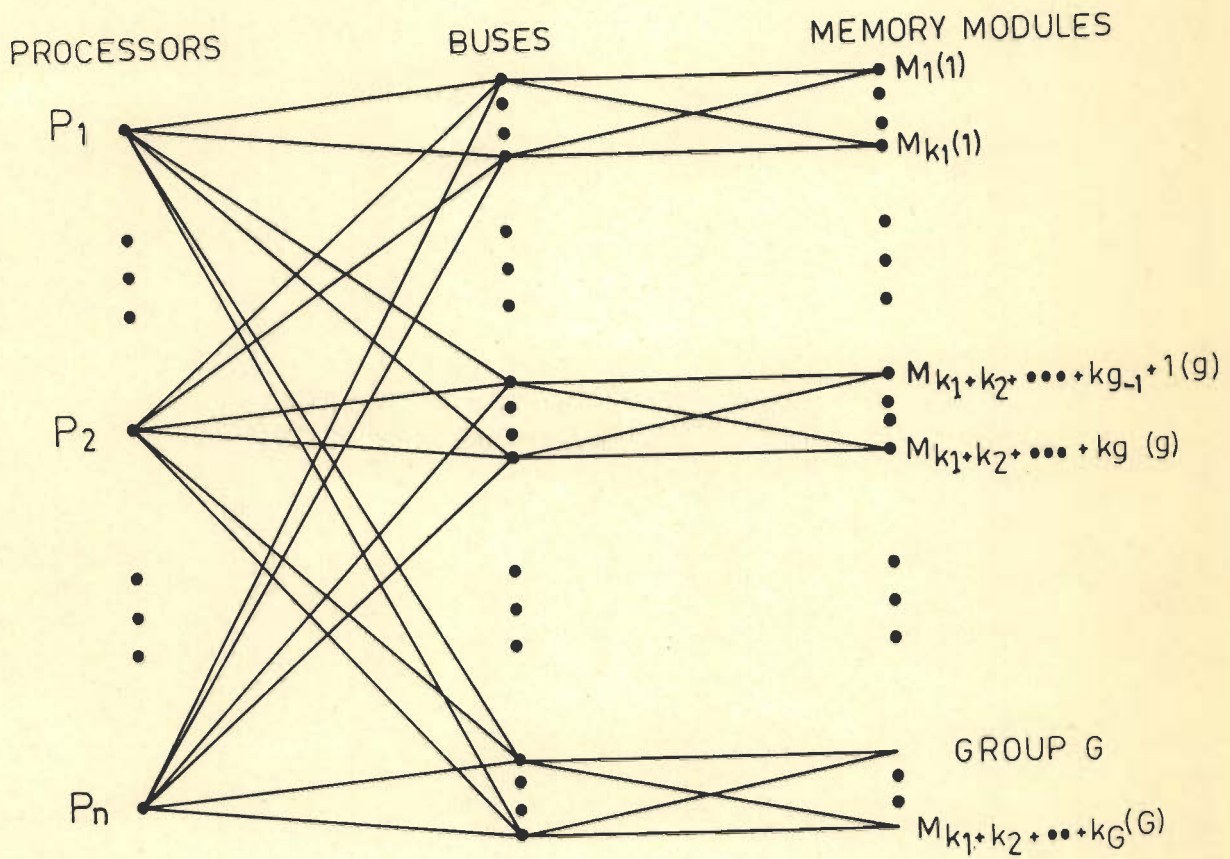


FIG.4.10 GRAPH REPRESENTATION OF PARTIAL-BUS MPS .

where $Q_f(g) = \{q_{ij} \mid j = f, \text{ a constant } \}$

$$j = f + \sum_{g^1=1}^{g-1} k_g^1 \quad (4.13)$$

Then probability, $E^1(i)$, that exactly i memory modules are accessed and i memory modules are distributed over G groups with i_g units in the g th group.

$$E^1(i) = \sum_{g=1}^G [H(X(g); k_g, i_g) - H(X(g); k_g, i_g + 1)] \quad (4.14)$$

where $X(g) = \{x_f(g)\}$ and $\sum_{g=1}^G i_g = i$

Then from (4.5), we get

$$BW_{Gnkz} = \sum_{i=1}^n \sum_{g=1}^G H(X(g); k_g, i_g) \quad (4.15)$$

where BW_{Gnkz} is the BW of a partial connected multiple-bus. Lang et al [17] also simulated the partial bus organization with $G=2$. The analytic data, obtained using Eq. (4.15) is given in Table 4.11 along with NURM results, with $m = 0.8$. Comparison of these two sets of results shows good agreement (within 7 percent) between the analytic and simulation results, when $r = 1.0$ (Table 4.12). Our results coincide with those of Mudge et al [60]. Das and Bhuyan [59], Mudge [60] and Lin and Jou [71] presented only URM's. The model of Das et al requires tedious calculations, even for small values of 'G' in NURM especially when $n \neq k$. Lin and Jou's results show errors less than 8 percent, for $r = 1.0$.

TABLE 4.11

BANDWIDTH OF $n \times k \times 2$ PARTIAL-BUS MPS OBTAINED FROM EQ.4.15 WITH $r=1.0$ (URM, NURM)

NO. OF BUSES	Number of processors $n = k$							
	4		8		12		16	
	$m = 1/k$	$m = 0.8$	$m = 1/k$	$m = 0.8$	$m = 1/k$	$m = 0.8$	$m = 1/k$	$m = 0.8$
1 + 1	1.800	1.947	1.972	2.000	1.996	2.000	2.000	2.000
2 + 2	2.734	3.350	3.731	3.968	3.950	3.999	3.992	4.000
3 + 3			4.880	5.714	5.711	5.983	5.936	6.000
4 + 4			5.251	6.694	6.997	7.864	7.710	7.991
5 + 5					7.628	9.353	9.096	9.933
6 + 6					7.776	10.039	9.923	11.675
7 + 7							10.244	12.904
8 + 8							10.303	13.384

TABLE 4.12

BW OF $n \times k \times z$ PARTIAL-BUS MPSCOMPARISON OF ANALYTICAL RESULTS WITH SIMULATION RESULTS, WITH $r=1.0$ (URM)

NO. OF BUSES	NUMBER OF PROCESSOR (=k)											
	4			8			12			16		
	SIMULATION	ANALYTIC	%ERROR	SIMULATION	ANALYTIC	%ERROR	SIMULATION	ANALYTIC	%ERROR	SIMULATION	ANALYTIC	%ERROR
1 + 1	1.74	1.80	3.45	1.87	1.97	5.35	1.92	2.00	4.17	1.93	2.00	3.60
2 + 2	2.62	2.73	4.20	3.61	3.73	3.32	3.81	3.95	3.70	3.86	4.00	3.60
3 + 3				4.72	4.88	3.39	5.52	5.71	3.44	5.73	5.94	3.67
4 + 4				4.93	5.25	6.50	6.77	7.00	3.40	7.48	7.71	3.07
5 + 5							7.24	7.63	5.40	8.79	9.10	3.50
6 + 6							7.28	7.78	6.90	9.43	9.92	5.20
7 + 7										9.59	10.24	6.78
8 + 8										9.63	10.30	6.96

4.4.4 Errors in the BW Analysis

The major source of error is the assumption that blocked requests are discarded. In reality, blocked requests are resubmitted or queued until the memory they request allows them access, thereby increasing the request rate [1,39,56,60]. The errors are significant when $r < 1.0$. Hwang et al [1], Yen et al [39], Hoogendoorn [56] and Mudge et al [60] considered the effects of this assumption and presented iterative techniques to refine the BW equations to take this effect into account.

An approximate analysis of the behaviours of the system with rejected requests is made based on the assumption that the resubmitted request addresses the memory models uniformly [1]. The processor may be in active cycle, when it is connected to a memory module and in blocked cycle, when its request is rejected by the memory module.

The Probability, PA, that an arbitrary request is accepted is

$$PA = BW / \sum_{i=1}^n r_i \quad \text{and} \quad r = \sum_{i=1}^n r_i / n \quad (4.16)$$

the probability that the system is in active cycle, qa, is given by

$$qa = PA / (PA + r(1-PA)) \quad (4.17)$$

The request rate, r, defined in conflict free access is referred to as static request rate of the processor; while the

request rate, r^1 , defined in memory interference cycle time (i.e., blocked cycles) are referred to as the dynamic request rate of the processor.

$$r^1 = r q_A + (1 - q_A) = r / (r + PA(1 - r)) \quad (4.18)$$

$$\text{and } PA = BW / r^1 n \quad (4.19)$$

Equation 4.18 and 4.19 define an iterative process by which one can compute PA for a given n , k , and r^1 can be initialized to r for the iteration process [1]. Note that when $r = 1$, r^1 must be 1 [39].

4.5 Fundamental Measures

So far, the focus was on one particular performance measure, namely effective memory bandwidth. In the remainder of this chapter, given a complex and hopefully more accurate, representations of performance than that seen in previous section.

It is useful to note that most of the meaningful measures for computer systems fall into the two fundamental classes, throughput and response time measures.

Throughput measures attempt to gauge how well the system is being used rather than how responsive the system is to the demands of the user. Utilization is a measure which is closely related to throughput [11]. In literature, the studies related to utilization are found in [34,42,47,57,58,60,68,71,73], wait time in [42,47,68,73] and probability of acceptance in

[1,42,47].

Memory Utilization

A meaningful analysis of the effective utilization of the MPS must consider all its components such as processors, memory modules and INs. Memory utilization (MU) is the fraction of time that a memory module is being accessed by some processor (busy).

Utilization of M_j is therefore given by

$$\begin{aligned} MU_j &= H(Q_j ; n, 1), \quad Q_j = \{q_{ij} \mid j = \text{constant}\} & (4.20) \\ &= q_{1j} + \bar{q}_{1j} q_{2j} + \dots + \bar{q}_{1j} \bar{q}_{2j} \dots \bar{q}_{(n-1)j} q_{nj} \end{aligned}$$

For a known BW the average number of busy memory modules, MU, can be computed as

$$MU = BW / k \quad (4.21)$$

Memory utilization summed over all memories is the expected memory bandwidth, for either a bus sufficient system or crossbar

$$BW = \sum_{j=1}^k MU_j$$

Fig.4.11 shows the variation of average memory utilization in a $n \times k$ crossbar with variation of k , by considering $n = 4, 8$, and 16 in a URM. For any number of memory modules, as $n > k$ the curves shift being concave upward to convex upward indicating better utilization of memory. For $n \times k$ crossbar the memory utilization decreases with n . Fig. 4.12 shows how the memory utilization of a $n \times k$ crossbar vary with m in a NURM.

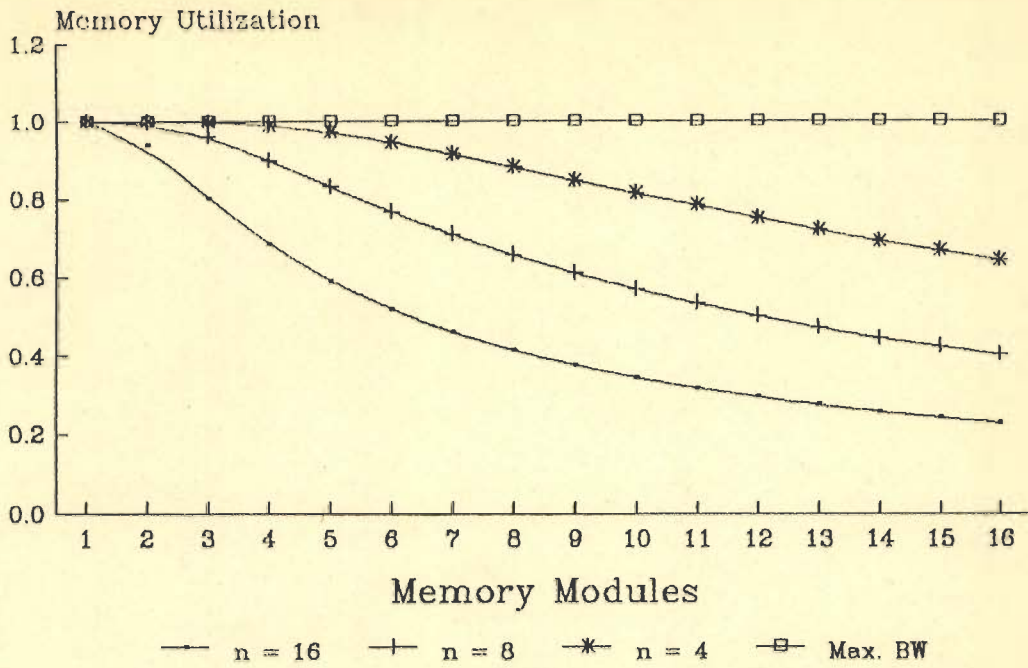


FIG. 4.11 MEMORY UTILIZATION OF CROSSBAR Vs. NUMBER OF MEMORY MODULES, $r=1$ (URM)

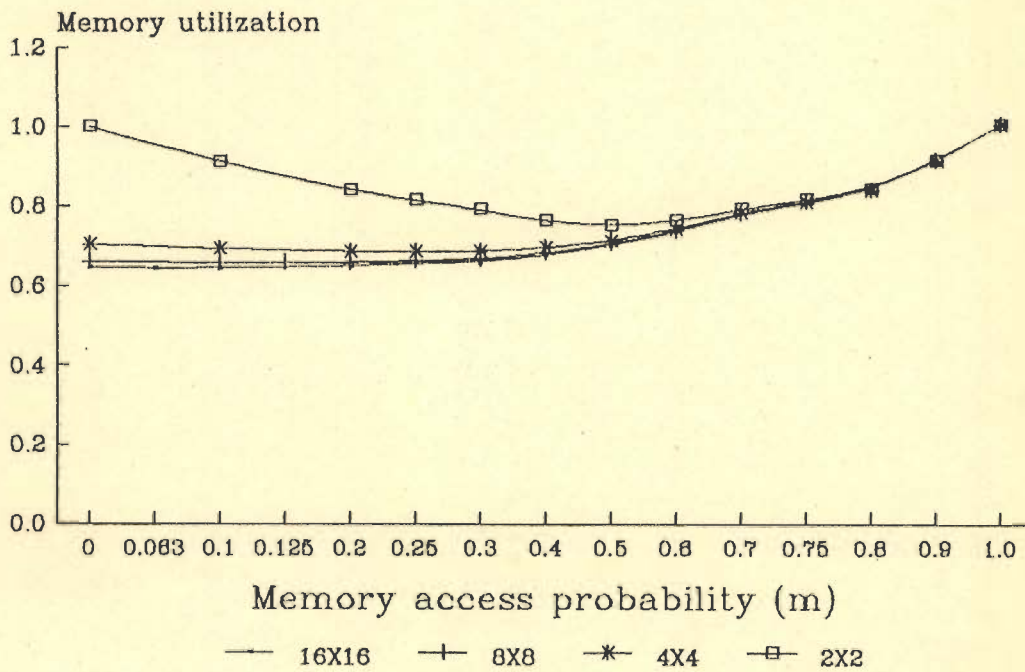


FIG. 4.12 MEMORY UTILIZATION OF $n \times k$ CROSSBAR WITH $r=1.0$ (NURM) Vs. m

Probability of acceptance (PA) is defined as the probability that one of the processor request is accepted by one of the memory modules and is simply given by the ratio of expected BW to the expected number of requests generated per cycle [47].

The Probability that the j th memory module accepts the request is utilization of M_j

$$MU_j = H(Q_j ; n, 1), Q_j = \{q_{ij} \mid j = \text{const.}\}$$

If the processor makes requests in every cycle ($r_i = 1$) then

$$PA = \frac{\sum_{j=1}^k MU_j}{n}, \text{ otherwise}$$

$$PA = \frac{\sum_{j=1}^k MU_j}{\sum_{i=1}^n r_i} = \frac{BW}{\sum_{i=1}^n r_i} \quad (4.22)$$

We observe that

- i) When $n \leq k$ utilization of all memory modules is same, and $MU_j = MU$
- ii) When $n = k$ and $r_i = r = 1.0$, then $MU = PA$

Channel Utilization

Utilization of the crossbar, U_c , is the ratio of the expected bandwidth (BW) to the maximum bandwidth BW_{max} , where the BW_{max} is the minimum value of (n, k) and utilization of a crossbar is given by-

$$U_c = BW / BW_{max}$$

where suffix C refers to crossbar. Notice that in a $n \times k$ crossbar 80-90% of the maximum number of channels are available when $r = 1.0$ as against 45-50% when $r = 0.5$ in NURM. The channel utilization is poorer in URM. In $n \times k \times z$ multiple-bus system, the utilization is $BW / \min(n, k, z)$. The bus utilization in case of a $8 \times 8 \times z$ MPS is shown as a function of MRP in Table 4.13.

Wait Time

It is the length of time from a request for service until the request is completed. The PA is measure of wait time (WT). A higher PA indicates a lower WT and a lower PA indicates higher WT [47]. The expected wait time of a request is

$$WT = (1/PA - 1) \quad (4.23)$$

The wait time of a processor waiting for its favourite memory decreases to zero as $r \rightarrow 1$. Nevertheless, the overall average wait time experienced by a processor is lower in the NURM than in URM [34].

Processor Utilization (PU)

PU is the fraction of time that a processor has its associated process running on it. It gives the average number of processors busy. In fact, since a processor may or may not generate a nonlocal request, processor utilization is more significant for indicating the whole system performance than memory bandwidth. The average processor utilization is given by [73].

TABLE 4.13
BUS UTILIZATION IN 8 X 8 X z CONFIGURATION (URM)

MRP (r)	NUMBER OF BUSES (z)							
	1	2	3	4	5	6	7	8
0.1	55.3	36.4	25.5	19.2	15.3	12.8	10.9	9.6
0.2	80.2	70.8	47.2	36.5	29.3	24.4	20.9	18.3
0.4	91.3	79.0	64.5	51.7	42.0	35.1	30.1	26.4
0.4	96.3	88.7	77.1	64.5	53.4	44.8	38.5	33.7
0.5	98.4	94.1	85.7	74.7	63.3	53.6	40.1	40.3
0.6	99.3	97.0	91.4	82.5	65.7	61.5	53.0	46.4
0.7	99.7	98.5	95.0	88.2	78.7	68.4	59.3	51.9
0.8	99.9	99.3	97.1	99.2	84.2	74.5	64.9	57.0
0.9	100.0	99.7	98.4	95.0	88.6	79.6	70.0	61.5
1.0	100.0	99.9	99.1	96.9	91.9	84.0	74.5	65.6

$$PU = 1 - \left(\sum_{i=1}^n r_i \right) (1 - PA) / n = 1 - \left(\sum_{i=1}^n r_i \right) / n + BW/n \quad (4.24)$$

Processing power is the sum of the utilization of all processors and it is given by $n \times PU$.

4.6 Delta Network

A crossbar allows all possible one-to-one connections between processors and memory modules, but the cost grows rapidly with the increase in network size. Multistage Interconnection Networks (MINs), an alternative to crossbars, have assumed paramount importance in recent times [20,23]. One class of IN mostly useful in MPS is the Delta network [47]. Delta, Baseline [107], Indirect Binary n-cube [109] and Omega [110] are all topologically equivalent [111]. The analysis of one can easily be extended to others.

The processor-memory interference of MINs using circuit switching [107,111-114] and packet switching [115-117] have been studied recently. The performance analysis of Delta networks is reported in [47,113]. They examined the performance with the assumption of uniformly directed processor requests to all memory modules (URM). Bhuyan [55] analyzed NURM model for Omega network. Conternon and Melen [113] modeled the Delta network with random memory access, whereas Patel [47] models are applicable only to uniform memory reference. Here we show how the analysis of crossbar (Section 4.1) can be used to determine the performance of the Delta network.

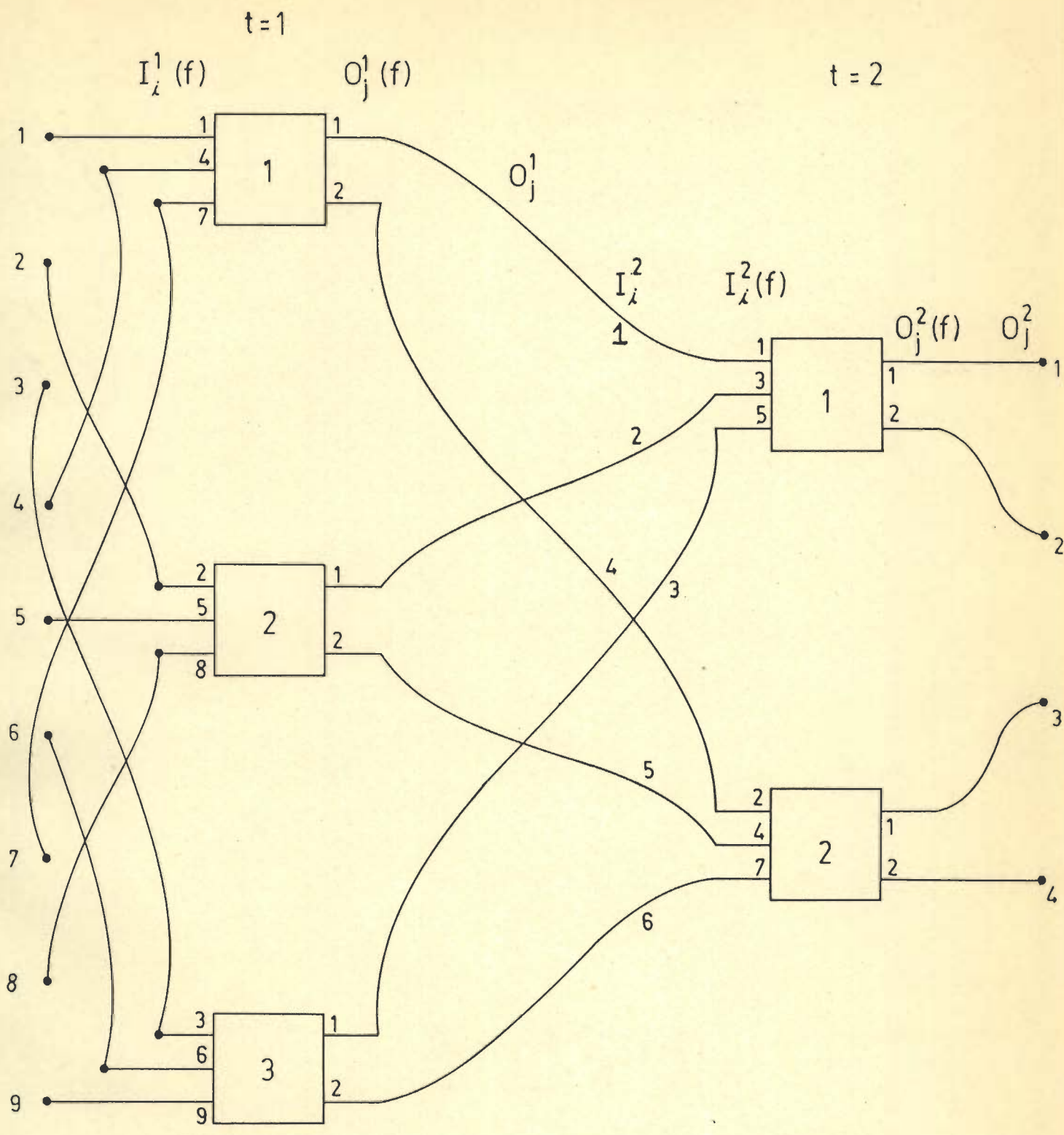


FIG. 4.13 A $3^2 \times 2^2$ DELTA NETWORK

Model

The Delta network is basically an $a^N \times b^N$ switching network connecting a^N processor to b^N memory modules as shown in fig. 4.2. It consists of N stages and each stage consists a number of $a \times b$ crossbar switches. The switch has a capacity of connecting any one of its 'a' inputs to any one of its 'b' outputs. Each switch has a control unit which selects an output. There are $a^{N-t} \times b^{t-1}$ switches in the t th stage and are labelled as $1, 2, \dots, a^{N-t} b^{t-1}$. The control switch of the f th switch in the t th stage is denoted by C_f , $1 \leq f \leq a^{N-t} b^{t-1}$. The number of inputs to the t th stage is $a^{N-t+1} b^{t-1}$ and outputs $a^{N-t} b^t$. The interconnection pattern between two stages of the network are based on generalized shuffle $S_{a \times b}(i)$.

$S_{a \times b}(i)$ an a -shuffle of ab objects, where a and b are positive integers, is defined as a permutation of ab indices, $\langle 1, 2, \dots, ab \rangle$, given by [47]

$$S_{a \times b}(i) = \begin{cases} a(i-1) + \left\lfloor \frac{(i+b-1)}{b} \right\rfloor \bmod ab & \text{for } 0 \leq i < ab \\ i & i = ab \end{cases}$$

We extend this definition to give a shuffle mechanism between the stages of a Delta networks as follows.

Definition 4.1. $S_{b \times a}(i)$, b shuffle of $a^{N-t} b^t$ objects where 'b' and 'a' are positive integers is a permutation of $a^{N-t} b^t$ indices, $\langle 1, 2, \dots, a^{N-t} b^t \rangle$, given by

$$S_{b \times a}(i) = \begin{cases} b(i-1) + \lfloor (i+a-1)/a \rfloor \bmod a^{N-t} & \text{for } 0 \leq i < a^{N-t} \\ i & \text{for } i = a^{N-t} \end{cases} \quad (4.25)$$

The inputs of the first stage of MIN are obtained after $S_{a \times a}$ shuffle of the processor outputs given by

$$S_{a \times a}(i) = \begin{cases} a(i-1) + \lfloor (i+a-1)/a \rfloor \bmod a^N & \text{for } 0 \leq i < a^N \\ i & \text{for } i = a^N \end{cases} \quad (4.26)$$

Let $O_i^t(f)$ and $I_j^t(f)$ indicate the output and input indices at the t th stage of an f th switch of the Delta network, for all $1 \leq i \leq b$, and $1 \leq j \leq a$. $O_i(f)$ after shuffle become I_j , the input indices of the next stages. The input/output indices of the switches and stages are related as:

$$i^t = i^t(f) + (f-1)b : t \text{ th stage outputs} \quad (4.27)$$

$$j^t = j^t(f) + (f-1)a : t \text{ th stage inputs} \quad (4.28)$$

Assume that we wish to connect the sources O_i^0 to the destinations O_i^N . The O_i^0 are the inputs I_j^0 of the first stage. $S_{a \times a}$, the first stage of generalized shuffle, modifies the sources to $I_j^1(f)$ and then are connected as inputs, to switches of the first stage. These switches at the first stage will connect the sources to the output O_i^1 depending upon the control bits, C 's of the first stage. These are the inputs, I_j^2 , to the second stage. After the second stage generalized shuffle, $S_{b \times a}$

these are moved to $I_j^2(f)$ at the input of the second stage of switches and O_i^2 at the output. At the output, of the t th stage of switches the sources reach O_i^t and so on. The inputs to the switches of the t th stage, $I_j^t(f)$ are obtained after shuffling the inputs of the t th stage, I_j^t using definition 2.

$$J^t(f) = \begin{cases} b(J^t - 1) + \left[(J^t + a - 1) / a \right] \bmod a^{N-t} b^t & \text{for } 0 \leq i_0 < a^{N-t} b^t \\ I_j & I_j = a^{N-t} b^t \end{cases} \quad (4.29)$$

$$\text{and } f = \left[(J + a - 1) / a \right] \quad (4.30)$$

Example 4.1: Consider $3^2 \times 2^2$ Delta network. It consists of three (3 X 2) crossbar switches in the first stage and two in the second as shown in Fig. 4.13. The inputs of these 5 switches are computed and tabulated in Table 4.14.

BW of a Delta Network

x_j , the probability of acceptance of at least one of the inputs of $a \times b$ crossbar can be obtained at j th output from eq. (4.2).

$$x_j = H(Q_j ; a, 1), \quad Q_j = \{q_{ij} \mid j = \text{constant}\}, 1 \leq j \leq b \quad (4.31)$$

and

$$BW_{a \times b} = \sum_{i=1}^b x_j \quad (4.32)$$

Extending this analysis to the t th stage of a Delta network.

$$q_{i|d}^t = H(q_{i|e}^t ; a, 1) \quad (4.33)$$

where $d = O_i(f)$ and $e = I_j(f)$

TABLE 4.14

Inputs to Switches of Two Stage DELTA NETWORK

O _j	t	1			2	
	I _j	I _j (f)			I _j (f)	
	f	1	2	3	1	2
1		1			1	
2		4			3	
3		7			5	
4			2			2
5			5			4
6			8			6
7				3		
8				6		
9				9		

q_{1d} is the probability that the request from the l th input is accepted by d th output if a request is made (similar to q_{1j} defined in Section 4.1)

Compute q_{1d}^1 using H function (ref. Chapter III).
 Recursively compute q_{1d}^N using equation (4.25) through (4.33) for $t = 1$ to N

$$BW[\Delta] = \sum_{i=1}^b q_{1d}^N$$

Example 4.2- Consider $3^2 \times 2^2$ Delta network

- i) q_{1d}^0 are the outputs of the processors and become the inputs to the first stage q_{1e}^1
- ii) q_{1e}^t (f) and q_{1d} are computed using S_{ax} shuffle (eq.(4.26) and (4.27) (Ref. Table 4.14)

$$q_{1d} = H(q_{1e}; 3, 1), \quad q_{1e} = (q_{1e} \mid e = \text{constant}), \quad 1 \leq e \leq 2$$

$$\text{For } l = 1, \quad q_{1e}^1 + \bar{q}_{1e}^1 \quad q_{4e}^1 + \bar{q}_{1e}^1 \bar{q}_{4e}^1 \quad q_{7e}^1$$

$$\text{For } l = 2, \quad q_{2e}^1 + \bar{q}_{2e}^1 \quad q_{3e}^1 + \bar{q}_{2e}^1 \bar{q}_{3e}^1 \quad q_{6e}^1$$

$$\text{For } l = 3, \quad q_{3e}^1 + \bar{q}_{3e}^1 \quad q_{5e}^1 + \bar{q}_{3e}^1 \bar{q}_{5e}^1 \quad q_{9e}^1$$

- iv) These output are shuffled and are fed to the second stage as q_{1e}^t (f) (Ref. Table 4.14)
- v) Compute q_{1d}^2 using (4.33)

$$vi) \quad BW = \sum_{l=1}^3 \sum_{d=1}^2 q_l^2 d$$

4.7 CONCLUSION

In this chapter approximate performance estimates for models of multiple-bus MPSs have been presented. These models include the important properties of constant memory access time, memory request probabilities less than unity, and bus contention. Two forms of non-uniformity in the memory access probabilities were also treated along with uniform memory reference. The crossbar is shown to be a special case of multiple-bus. The performance estimates provided several important insights. One is that assuming a Bernoulli distribution for memory access probabilities. Two, is that low request rates (MRP) only a few buses are needed to have the performance of a crossbar. However when only a few buses are used, a minimum request rate exists. Exceeding that request rate causes a dramatic collapse in performance. The analysis was extended to partial-bus and Delta networks.

CHAPTER V

Fault Tolerance

A fault tolerant interconnection network can tolerate faults to some degree and still provide reliable and gracefully degradable communication. There are several ways to achieve fault tolerance: multiple paths between an input/output pair, multiport connection and fault tolerant switching elements. Fault tolerance is an inherent characteristic of the interconnection network. Once the IN has been constructed, there is little one can add to enhance its deserved fault tolerance level [23]. This implies that fault tolerance must be considered as one of the prime factors in choosing a proper interconnection network for system connection.

For the evaluation of non-failure critical multi processor models, the most appropriate performance measure is the bandwidth availability (BA). It is a measure of variation of available BW with time. The reliability measure discussed in Chapters III and IV are not sufficient to evaluate these systems because they do not reveal the performance degradation due to failures in the mission time.

Several simulation and analytic models have been developed to predict the reliability of computer systems assuming failures in mission time [35,59,89,122-132]. Recent papers presented in literature, illustrate the use of three highly developed reliability software modeling tools, viz. The Hybrid

Automatic Reliability Predictor (HARP) [89,22,123], Symbolic Hierarchical Automated Reliability and Performance Evaluator [SHARPE] [35] and Computer-Aided Reliability Estimation (CARE) [124-126] in evaluating the reliability and other performance measures of fault-tolerant systems and architectures.

Das and Bhuyan [59] using Markov chain techniques developed availability models by considering statistically identical processors, memory modules and interconnection links. Ingle and Siewiorek [127] in their analysis considered failures of processors and memory modules assuming undegradable INs. Chou and Abraham [128] analyzed the models using resource guardian. The models [127-128] are not applicable to tightly coupled multiprocessors [59]. Das et al [129] presented analytic models for dependability and performance evaluation of multiprocessor systems with both on line and off line maintenance.

The system availability is modeled based on task (job) requirement [59,129,130] as discussed in Section 3.1.5. The task based availability assumes that the system remains operational as long as the minimum number of resources (processors, memory modules and interconnection links) for the concurrent execution of a task are available on the system. The bandwidth availability is directly proportional to the number of memory modules available for the execution of the task and reliability of the system at any time.

The reconfiguration process for graceful degradation of the system can be taken into account by assuming coverage factors for each of the three groups. Coverage is the conditional probability of successful error recovery given that error has occurred [131, 132].

In this section the equations for threshold reliability are suitably modified to take into account the failures in mission time. A fault tolerant IN is one that provides service, in at least some cases, even when it contains a faulty component or components. A fault can be either permanent or transient; unless stated otherwise, it is assumed in this section that faults are permanent.

5.1 MULTIPLE-BUS SYSTEM

The probability that i processors are executing a task is given by $[H(P; n, i) - H(P; n, i+1)]$. Similarly the problem of j memory modules and f buses being available for the execution of the tasks are $[H(M; k, j) - H(M; k, j+1)]$ and $[H(B; z, f) - H(B; z, f+1)]$ respectively. The availability of the MPS, $P_{ijf}(\tau)$, when i processors, j memory modules and f buses are available for execution of a task, is the product of the availability of these mutually independent groups of components and hence-

$$P_{ijf}(\tau) = [H(P; n, i) - H(P; n, i+1)] * [H(M; k, j) - H(M; k, j+1)] * [H(B; z, f) - H(B; z, f+1)] \quad (5.1)$$

The reliability of the multiple-bus system, $R_{\alpha\beta}$

[Bus], when a task needs at least α processors, β memory modules and a bus for communication, is obtained by considering probabilities of all non-failed states of the components of all three groups separately and then summing up the probabilities.

$$\begin{aligned}
 R_{\alpha\beta}[\text{Bus}] = & \sum_{i=\alpha}^n C_P^{n-1} [H(P; n, i) - H(P; n, i+1)] * \\
 & \sum_{j=\beta}^k C_M^{k-j} [H(M; k, j) - H(M; k, j+1)] * \\
 & \sum_{f=1}^z C_B^{z-f} [H(B; z, f) - H(B; z, f+1)]
 \end{aligned} \tag{5.2}$$

where C_P , C_M and C_B are the coverage factors for processor, memory module and bus. By including C_P , C_M and C_B in equation 5.1, the fault tolerant effect can be taken into account and the modified availability equation can be referred to as $A_{i,j,f}(\tau)$. Then the reliability expression can be written as

$$R_{\alpha\beta}(\tau) [\text{Bus}] = \sum_{i=\alpha}^n \sum_{j=\beta}^k \sum_{f=1}^z A_{i,j,f}(\tau) \tag{5.3}$$

The BA is the sum of the expected BW of all the non failed states of the system [59].

$$\text{BA}(\tau) [\text{Bus}] = \sum_{i=\alpha}^n \sum_{j=\beta}^k \sum_{f=1}^z A_{i,j,f}(\tau) \text{BW}_{i,j,f} \tag{5.4}$$

where $\text{BW}_{i,j,f}$ is the bandwidth of a $i \times j \times f$ multiple-bus MPS [ref., Eq.(4.5)].

5.2 CROSSBAR SYSTEM

If at any time, 'i' processors and 'j' memory modules are executing a task, the availability of the crossbar switched MPS, $A_{ij}(\tau)$, is obtained as

$$A_{ij}(\tau) = C_P^{n-1} [H(P; n, i) - H(P; n, i+1)] * C_{BM}^{k-j} [H(\emptyset; k, j) - H(\emptyset; k, j+1)] \quad (5.5)$$

where C_{BM} is the coverage factor of a memory-bus combination.

The expressions for reliability, $R_{\alpha\beta}(\tau)$ [Bus] and bandwidth availability $BA(\tau)$ [crossbar] are

$$R_{\alpha\beta}(\tau) [\text{Crossbar}] = \sum_{i=\alpha}^n \sum_{j=\beta}^k A_{ij}(\tau) \quad (5.6)$$

$$BA(\tau) [\text{Crossbar}] = \sum_{i=\alpha}^n \sum_{j=\beta}^k A_{ij}(\tau) BW_{ij} \quad (5.7)$$

where BW_{ij} is the bandwidth of an $i \times j$ crossbar [ref., Eq. (4.6)].

5.3 PARTIAL-BUS SYSTEM

The probability, $E(i)$, that exactly i processors are active in a system is given by

$$E(i) = H(P; n, i) - H(P; n, i+1)$$

If the task needs exactly j memory modules over G groups and let the group g contains ' j_g ' memory modules and ' f_g '

buses, such that $\sum_{g=1}^G j_g = j$ and $\sum_{g=1}^G f_g = f$. The probability $q(j, f)$ that exactly j memory modules and f buses are active in a system is given by

$$q(j, f) = \sum_{g=1}^G [H(\varnothing(g); k_g, j_g) - H(\varnothing(g); k_g, j_g + 1)] \quad (5.8)$$

where $\varnothing(g) = \{\varnothing_{t(g)}\}$

$\varnothing_{t(g)}$, the probability that atleast a single bus connected to the memory modules t in group g , is expressed as

$$\varnothing_{t(g)} = H(B(g); f_g, 1) M_{t(g)}, \quad 1 \leq t \leq f \quad (5.9)$$

where $B(g) = \{B_{t(g)}\}$, represents the successive probabilities of the buses in the g th group. Since the availability of the events of selecting of processors and memory modules are independent of the system, reliability is obtained by computing the series reliability of $z(i)$ and $q(j, t)$.

The probability, $A_{i,j,f}$ that exactly i processors, j memory modules and f buses are busy in an MPS is given by

$$A_{i,j,f}(\tau) = x(i)q(j, f) \quad (5.30)$$

and the threshold reliability $R_{\alpha\beta}(\tau)$ [partial-bus] is given by

$$R_{\alpha\beta}(\tau) \text{ [partial-bus]} = \sum_{i=\alpha}^n \sum_{j=\beta}^k \sum_{f=1}^z A_{i,j,f}(\tau) \quad (5.31)$$

and the equation for bandwidth availability is

$$BA(\tau) \text{ (partial-bus)} = \sum_{i=\alpha}^n \sum_{j=\beta}^k \sum_{f=1}^z A_{ijf}(\tau) BWG_{ijf} \quad (5.32)$$

where BWG_{ijf} is the bandwidth of the partially connected multiple bus MPS. The results of this model coincides with those of Das et al [59]. If the statistical properties of the elements of each group differ, then Das model cannot be directly be used.

5.4 CONCLUSION

The reliability models for the real time systems assuming non failure-critical models have been presented. Performance degradation of the MPSs are studied. The equations for BA are computed for the three types of configurations viz., multiple-bus, crossbar, and partial-bus.

CHAPTER VI

CONCLUSIONS AND SCOPE OF FUTURE WORK

This chapter summarizes the dissertation and discusses several conclusions drawn in it. Section 6.2 presents related problems that may be studied in future as an extension to the present work.

6.1 SUMMARY AND CONCLUSIONS

This thesis documents some design aspects such, as modeling and analysis, of MPS's.

Modeling

A framework for analyzing the MPSs. the t-out-of-s system modeling technique has been evolved. The interconnection networks that are modeled are multiple-bus, crossbar, partial-bus and multistage interconnection networks.

Analysis

The criteria that have been considered for analysis are bandwidth, reliability, bandwidth availability, graceful degradation and some fundamental performance criteria such as memory utilization, processor utilization, channel utilization and processor waiting time.

i) Memory Bandwidth

Closed form solutions for the bandwidth (BW) of the multiple-bus have been derived using more general models. The dependability of BW on the number of processors, number of memory

modules, number of buses, memory request probability and memory access pattern has been analyzed. The analysis holds good for both uniform memory reference and local memory reference. The comparison of crossbar with multiple-bus has been made. It has also been shown how these models can be extended to compute the BW of partial-bus and Delta networks.

ii) **Fault Tolerance**

First, three recursive algorithms are presented for determination of exact reliability of t-out-of-s redundant systems. These give reliability expressions with minimal number of terms and involve fewer multiplications in comparison to other methods. The recursive nature of the algorithms enables one to design easily the number of units in the system to meet the reliability target. Second, an alternative representation, in the form of non-recursive algorithm, which is more memory and time efficient was presented.

These models are extended to compute the terminal reliability and multiprocessing reliability of the MPS, assuming failure-critical nature of operation. Multiple bus, crossbar and multiport memory INs are considered. Expressions for reliability and bandwidth availability are presented considering graceful degradation also. The models are also extended to partial-bus configuration.

iii) **Fundamental Criteria**

The BW models are extended to compute some of the

related performance measures which may be useful in some situations, in particular: the probability of request being accepted, the processor utilization, memory utilization; channel utilization and the expected waiting time of a processor before a request is allowed memory access.

The results obtained here give an overall view of the design aspects of four possible multiprocessor configurations. However, the selection of a proper architecture depends on the specific application. For example, if the requirements are such that we need high communication BW for a short duration, a crossbar seems to be a viable solution. On the other hand, if the BA as well as the duration are important, then multiple-bus provides better characteristics than crossbar. It is also interesting to note that when $r < 1.0$, the multiple-bus looks attractive in view of the flexibility to choose the number of buses depending on MRP and memory access probabilities. The cost and performance of a MIN is a reasonable balance between a shared bus and a crossbar. If the BA requirements are not stringent, the partial-bus connection can be a cost effective alternative to the multiple-bus.

Uniform memory reference is shown to be a special case of local memory reference analysis. With favourite memories (NURM) the bandwidth is much higher than uniform reference model because of fewer conflicts.

The validity of the models can be seen from closeness in the results of the analysis and those obtained from simulations.

6.2 SUGGESTIONS FOR FUTURE INVESTIGATION

As always happens at the end of a research project when we look back what has been accomplished we realize that every problem which has been solved has also give rise to a set of open research issues and that for every solution which has been choosen we can always think of a better one. This is true also for this analytic design study. Some of these important issues are listed below:

- i) It can be observed that the processor-memory analysis relies on two assumptions: temporal independence and spatial independence. The temporal independence requires that successive memory requests by a processor be independent, which is clearly not valid in reality for resubmitted blocked requests. Spatial independence corresponds to independent requests to different memories, an assumption that also has limited validity. The effects of these assumptions are to be investigated.
- ii) The fixed access times incorporated into the models are the norm in real memories. But a distinguishing feature is to investigate using memory access time as a exponentially distributed random variable.
- iii) In the performance evaluation of interconnection networks,

it is assumed that in the event of conflicts a request is accepted at random with an equal probability. In many cases, the BW does not depend on this selection policy because it just counts the number of requests accepted in a cycle. The BW does not indicate which requests are accepted and which are rejected. Hence in case of non-uniform memory reference models equal acceptance rule discriminates against remote or less frequent requests because it rejects them most of the time rather it should be given priority over other processors which request that particular memory more often. Future research has to be done to derive mathematical expressions for arbitrary memory references.

- iv) Performance of MPSs are based on the assumption that the reconfiguration process, provided by a maintenance processor, is taken into account by the coverage factor. Further investigations are required in the area to find the effect of the maintenance processor reliability on the system dependability.
- v) In the present work, reliability models for only multiple-bus and crossbar are considered. Reliability analysis of MINs requires further investigation.
- vi) The model proposed for studying the performance of Delta networks in circuit switching environment may give a significant contribution to a better understanding of the

problems involved in the analysis by investigating the behaviour of these models in packet switching environment as well.

BIBLIOGRAPHY

- 001 K.Hwang, F.A.Briggs, **Computer Architecture And Parallel Processing**, McGraw-Hill, New York, 1984.
- 002 D.P.Agrawal, V.K.Janakiram, "Evaluating the performance of Multicomputer Configurations", **IEEE Computer**, vol 19, May 1986, pp 23-37.
- 003 R.W.Hockney, C.R.Jesshope, **Parallel Computer:Architecture, Programming And Algorithms**, Adam Hilger Ltd., Bristol, England,1981.
- 004 P.H.Enslow (Ed), **Multiprocessors And Parallel Processing**, **Wiley-Interscience**, New York, 1974.
- 005 M.J.Flynn, "Very high-speed computing systems", in **Proc. IEEE** Vol 54, 1966, pp 1901-1909.
- 006 M.J.Flynn, "Some computer organizations and their effectiveness", **IEEE Trans.Computers**, vol C-21, Sep 1972,pp 948-960.
- 007 T.Feng, "Some characteristics of associative / parallel processing" in **Proc.Sagamore Comp.Conf.,Syracuse Univ.,1972**, pp 5-16.
- 008 W.Handler, "The impact of classification schemes on computer architecture", in **Proc.Int.Conf.on Parallel Processing,1977**, pp 7-15.

- 009 H.S.Stone (Ed), **Introduction To Computer Arcitecture**, SRA, Chicago, 1980.
- 010 G.Conte, D.D.Corso (Eds), **Multi-Microprocessor Systems For Real Time Applications**, D.Reidel, Holland, 1985.
- 011 S.H.Fuller, "Performance evalation" In Ref.[8], pp 527-584.
- 012 M.Satyanarayana,**Multiprocessors-A Comparative Study**, Prentice-Hall, New York, 1980.
- 013 R.H.Kuhn, D.A.Padua(Eds), **Tutorial On Parallel Processing**, **IEEE Computer** Press, Los Angeles, 1981.
- 014 "Vacabulary for information processing" **American National Sandard**, X.3.12-1970.
- 015 J.L.Baer, "Multiprocessing systems", **IEEE Trans.Computers**, vol C-25, Dec 1976, pp 1271-1276.
- 016 Y.Parker, **Multi-Microprocessor Systems**, Academic Press, London, 1983.
- 017 T.Lang, M.Valero, I.Alegre, "Bandwidth of crossbar and multiple-bus connections for multiprocessors", **IEEE Trans. Computers**, vol C-31, Dec. 1982, pp 1227-1234.
- 018 T.Lang, M.Valero, M.A.Fiol, "Reduction of connections for multibus organization", **IEEE Trans.Computers**, vol C-32, Aug 1983, pp 707-715.

- 019 P.H.Enslow, "Multiprocessor organization", **ACM Computing Surveys**, vol 9, Mar 1977, pp 103-129.
- 020 T.Y.Feng, "A survey of interconnection networks", **IEEE Computer**, vol 14, Dec 1981, pp 12-27.
- 021 **IEEE Computer**, vol 20, Jun 1987.
- 022 **IEEE Computer**, vol 14, Nov 1981.
- 023 C.L.Wu, T.Y.Feng (Eds), **Tutorial On Interconnection Networks For Parallel And Distributed Processing**, **IEEE Computer Press**, Los Angeles, 1984.
- 024 L.N.Bhuyan (Guest Ed), "Interconnection networks for parallel and distributed processing", **IEEE Computer**, vol 20, Jun 1987, pp 9-12.
- 025 D.Ferrari, **Computer System Performance Evaluation**, Prentice-Hall, New Jersey, 1978.
- 026 C.H.Saucer, K.M.Chandy, **Computer Systems Performance Modeling**, Prentice-Hall, New Jersey, 1981.
- 027 K.S.Trivedi, "Analysis modelling of computer systems", in **Tutorial:Distributed Design**, Edited by M.P.Mariani and D.F.Palmer, **IEEE Computer Press**, Los Angeles, 1979, pp 391-409.
- 028 D.P.Siewiorek, C.G.Bell, A.Newell, **Computer Structures: Principles And Examples**, McGraw-Hill, Tokyo, 1982.

- 029 D.P.Siewiorek, R.S.Swarz, **The Theory And Practice Of Reliability System Design**, Digital Press, Bedford, 1982.
- 030 K.S.Trivedi, **Probability & Statistics With Reliability, Queuing, And Computer Science Applications**, Prentice-Hall, New Jersey, 1982.
- 031 D.K.Pradhan (Ed), **Fault-Tolerant Computing Theory & Tecniques**, vol 2, Prentice-Hall, New Jersey, 1986.
- 032 L.Kleinrock, **Queuing Systems, vol 1, Theory**, John Wiley & Sons, New York, 1975.
- 033 L.Kleinrock, **Queuing Systems, vol 2, Computer Applications**, John Wiley, New York, 1976.
- 034 T.N.Mudge, H.B.Sadoun, B.A.Makrucki, "Memory - intererence model for multiprocessors based on semi-markov processes", **IEE Proc.** vol 134, Pt.E, Jul 1987, pp 203-214.
- 035 R.A.Sahner, K.S.Trivedi, "Reliability modeling using SHARPE", **IEEE Trans. Reliability**, vol R-36, Jun 1987, pp 186-193.
- 036 C.L.Lin, **Introduction To Combinatorial Mathematics**, McGraw-Hill, New York, 1968, pp 38-40.
- 037 M.O.Locks, "Comments on : Improved method of inclusion-exclusion applied to k-out-of-n systems", and "Author reply", **IEEE Trans. Reliability**, vol R-33, Oct 1984, pp 321-322.

- 038 A.Goyal, V.F.Nicola, A.N.Tantawi, K.S.Trivedi, "Reliability systems with limited repairs", **IEEE Trans. Reliability**, vol R-36, Jun 1987, pp 202-207.
- 039 D.W.L.Yen, J.H.Patel, E.S.Davidson, "Memory interference in synchronous multiprocessor systems", **IEEE Trans.Computers**, vol C-31, Nov 1982, pp 1116-1121.
- 040 B.R.Rau, "Program behaviour and the performance of interleaved memories", **IEEE Trans.Computers**, vol C-28, March 1979, pp 191-199.
- 041 D.Y.Chang, D.J.Kuck, D.H.Lawrie, "On the effective bandwidth of parallel memories", **IEEE Trans.Computers**, vol C-26, May 1977, pp 480-490.
- 042 H-C.Du, "On the performance of synchronous multiprocessors", **IEEE Trans.Computers**, vol C-34, May 1985, pp 462-466.
- 043 C.E.Skinner, J.R.Asher, "Effects of storage contention on system performance", **IBM Syst.J.**, vol 8, Apr 1969, pp 319-333.
- 044 A.J.Smith, " Multiprocessor memory organization and memory interference", **ACM Commu.** vol 20, Oct 1977, pp 754-761.
- 045 K.O.Siomalas, B.A.Bowen, "Performance of crossbar multiprocessor systems", **IEEE Trans. Computers**, vol C-32, July 1983, pp 689-695.
- 046 H.C-Du, J.L.Baer, "On the performance of interleaved memories

- with non-uniform access probabilities", in **Proc.Int.Conf.on Parallel Processing**, Aug 1983, pp 429-436.
- 047 J.H.Patel, "Performance of processor-memory interconnections for multiprocessors", **IEEE Trans.Computers**, vol C-30, Oct 1981, pp 771-780.
- 048 W.A.Wulf, C.G.Bell, "C.MMP-A MULTI-MINI-PROCESSOR", in **Fall Joint Comput. Conf.Proc.**, vol 41, Pt 2, 1972, pp 765-777.
- 049 A.S.Sethi, N.Deo, "Interference in multiprocessor systems with localized memory access probabilities", **IEEE Trans.Computers**, vol C-28, Feb 1979, pp 157-163.
- 050 B.R.Rau, "Interleaved memory bandwidth in a model of multi-processor computer systems", **IEEE Trans.Computers**, vol C-28, Sep 1979, pp 678-681.
- 051 F.S.Baskett, A.J.Smith, "Interference in multiprocessor computer systems with interleaved memory", **ACM Commun.**, vol 19, Jun 1976, pp 327-334&
- 052 C.V.Ravi, "On the bandwidth and interference in interleaved memory systems", **IEEE Trans.Computers**, vol C-21, Aug 1972, pp 899-901.
- 053 D.P.Bandarkar, "Analysis of memory interference in multi-processor", **IEEE Trans.Computers**, vol C-24, Sep 1975, pp 897-908.

- 054 W.D.Strecker, "Analysis of the instruction execution rate in certain computer structures", **Ph.D Dissertation**, Carnegie-Mellon Univ., Pittsburgh, PA, 1970.
- 055 L.N.Bhuyan, "An analysis of processor-memory interconnection networks", **IEEE Trans.Computers**, vol C-34, Mar 1985, pp 279-283.
- 056 C.H.Hoogendoorn, "A general model for memory interference in multiprocessors", **IEEE Trans.Computers**, vol C-26, Oct 1977, pp 998-1005.
- 057 T.N.Mudge, B.A.MakrucKi, "Probabilistic analysis of a crossbar switch", in **Proc.9th Int.Symp.Comput.Architect.**, Apr 1982, pp 311-320.
- 058 M.A.Holliday, M.K.Vernon, "Exact performance estimates for multiprocessor memory and bus interference", **IEEE Trans.Computers**, vol C-36, Jan 1987, pp 76-85.
- 059 C.R.Das,L.N.Bhuyan, "Bandwidth availability of multiple-bus multiprocessors", **IEEE Trans.Computers**, vol C-34, Oct 1985, pp 918-926.
- 060 T.N.Mudge, J.P.Hayes, G.D.Buzzard, D.C.Winsor, "Analysis of multiple-bus interconnection networks", in **Proc.Int.Conf.Parallel Processing**, Aug 1984, pp 228-232.
- 061 L.N.Bhuyan, "A combinatorial analysis of multibus multiprocessors", in **Proc.Int.Conf.Parallel Processing**, Aug 1984, pp 225-227.

- 062 A.Goyal, T.Agerwala, "Perfomance analysis of future shared storage systems", **IBM.J.Res. & Develop.**, vol 28, Jan 1984, pp 95-108.
- 063 D.Towsley, "An approximate analysis of multiprocessor systems", in **Proc.ACM Sigmetrics Conf.Meas.and Mod.Comput. Syst.**, Aug 1983, pp 207-213.
- 064 T.N.Mudge, J.P.Hayes, D.C.Winsor, "Multiple bus architectures", **IEEE Computer**, vol 20, Jun 1987, pp 42-49.
- 065 M.A.Marsan, M.Gerla, "Markov models for multiple-bus multiprocessor systems", **IEEE Trans. Computers**, vol C-31, Mar 1982, pp 239-248.
- 066 M.A.Marsan, G.Balbo, G.Conte, "Comparative performance analysis of single bus multiprocessor architectures", **IEEE Trans. Computers**, vol C-31, Dec 1982, pp 1179-1191.
- 067 M.A.Marsan, G.Balbo, G.Conte, Gregoretti, "Modeling bus contention and memory interference in a multiprocessor system", **IEEE Trans.Computers**, vol C-32, Jan 1983, pp 60-72.
- 068 T.N.Mudge, H.B.A.Sadoun, "A semi-markov model for the performance of multiple-bus systems", **IEEE Trans.Computers**, vol C-34, Oct 1985, pp 934-942.
- 069 I.H.Onyuksel, K.B.Irani, "A markovian queueing network model for performance evaluation of bus-deficient multi-processor systems", in **Proc. Int.Conf.Parllel**

Processing, Aug. 1983, pp 437-439.

- 070 K.B.Irani, I.H.Onyuksel, " A closed form solution for the performance analysis of multiple-bus multiprocessor systems", **IEEE Trans. Computers**, vol C-33, Nov 1984, pp 1004-1012.
- 071 Y.C.Liu, C.J.Jou, "Effective memory bandwidth and processor blocking probability in multiple-bus systems", **IEEE Trans. computers**, vol C-36, Jun 1987, pp 761-764.
- 072 D.Towsley, "Approximate models of multiplebus multiprocessor systems", **IEEE Trans.Computers**, vol C-35, Mar 1986, pp 220-227.
- 073 T.N.Mudge, J.P.Hayes, G.D.Buzzard, D.C.Winsor, "Analysis of multiplebus interconnection networks", **Journal of Parallel And Dist. Computing**, vol 3, 1986, pp 328-333.
- 074 G.Chola, M.A.Marsan, B.Balbo, "Product form solution techniques of the performance analysis of multiplebus multiprocessor systems with non-uniform memory reference", **IEEE Trans. Computers**, vol C-37, May 1988, pp 532-540.
- 075 A.Goyal, A.N.Tantawi, "Evaluation of performability for degradable computer systems", **IEEE Trans.Computers**, vol C-36, Jun 1987, pp 738-744.
- 076 K.Hwang, T.P.Chang, "Combinatorial reliability analysis of multiprocessor computers", **IEEE Trans.Reliability**, vol R-31, Dec 1982, pp 469-473.

- 077 V.P.Kumar, S.M.Reddy, "Augmented shuffle exchange multistage interconnection network", **IEEE Computer**, vol 20, Jun 1987, pp 32-40.
- 078 A.Satyanarayana, J.N.Hagstrom, "A new algorithm for the reliability analysis of multi-terminal networks", **IEEE Trans. Reliability**, vol R-30, Oct 1981, pp 325-334.
- 079 C.L.Hwang, F.A.Tillman, M.H.Lee, "System reliability evaluation techniques for complex large systems-A Review", **IEEE Trans. Reliability**, vol R-30, Dec 1981, pp 416-423.
- 080 K.B.Misra, "An algorithm for the reliability evaluation of redundant networks", **IEEE Trans. Reliability** vol R-19, Nov 1970, pp 146-159.
- 081 K.K.Aggarwal, Y.C.Chopra, J.S.Bajwa, "Reliability evaluation by network decomposition", **IEEE Trans. Reliability**, vol R-31, Oct 1982, pp 355-358.
- 082 A.Satyanarayana, A.Prabhakar, "New topological formula and rapid algorithm for reliability analysis of complex networks", **IEEE Trans. Reliability**, vol R-27, Jun 1978, pp 82-100.
- 083 A.Satyanarayana, "A unified formula for analysis of some network reliability problems", **IEEE Trans. Reliability**, vol R-31, Apr 1982, pp 23-32.

- 084 L.Fratta, U.G.Montanari, " A recursive method based on case analysis for computing network terminal reliability", **IEEE Trans.Commun.**, vol COM-26, Aug 1978, pp 1166-1177.
- 085 S.Arnborg, "A reduced state enumeration-another algorithm for reliability evaluation", **IEEE Trans.Reliability**, vol R-27, Jun 1978, pp 101-105.
- 086 L.Faratta, U.G.Montanari, "A boolean algebra method for computing the terminal reliability in a communication network", **IEEE Trans.Circuit Theory**, vol CT-20, May 1973, pp 203-211.
- 087 S.Rai, K.K.Aggarwal, "An efficient method for reliability evaluation", **IEEE Trans.Reliability**, vol R-27, Jun 1978, pp 101-105.
- 088 S.Hariri, C.S.Raghavendra, "SYREL: A symbolic reliability algorithm based on path and cutset methods", **IEEE Trans. Reliability**, vol C-36, Oct 1987, pp 1224-1231.
- 089 S.J.Bavuso, J.B.Dugan, K.S.Trivedi, E.M.Rothamann, W.E.Smith, "Analysis of typical fault-tolerant architecture using HARP", **IEEE Trans.Reliability**, vol R-36, Jun 1987, pp 176-185.
- 090 M.O.Locke, "Recursive disjoint products, inclusion-exclusion, and min-cut approximations", **IEEE Trans. Reliability**, vol R-29, Dec 1980, pp 368-371.

- 091 S.Rai, Arun Kumar, E.V. Prasad, "Computing terminal reliability of a computer network", **Reliability Engineering**, vol 16, No 2, 1986, pp 109-119.
- 092 S.Rai, Arun Kumar, "On path enumeration", **Int. J. Electronics**, vol 60, No 3, 1986.
- 093 W. Cooksey JR., "An algorithm to determine parallel system reliability", **IEEE Trans. Reliability**, vol R-24, Oct 1975, pp 287.
- 094 K.D. Heidtmann, "Improved method of inclusion-exclusion applied to k-out-of-n systems", **IEEE Trans. Reliability**, vol R-31, Apr 1982, pp 36-40.
- 095 P.W. McGrady, "The availability of a k-out-of-n:G network", **IEEE Trans. Reliability**, vol R-34, Dec 1985, pp 451-452.
096. R.E. Balow, K.D. Heidtmann, "Computing k-out-of-n system reliability", **IEEE Trans. Reliability**, vol R-33, Oct 1984, pp 322-323.
- 097 S.P. Jain, K. Gopal, "Recursive algorithms for reliability evaluation of k-out-of-n:G system", **IEEE Trans. Reliability**, vol R-34, Jun 1985, pp 144-146.
- 098 T. Risse, "On the evaluation of the reliability of k-out-of-n systems", **IEEE Trans. Reliability**, vol R-36, Oct 1987, pp 433-435.

- 099 S.Rai, A.K.Sarje, E.V.Prasad, Arun Kumar, "Two recursive algorithms for computing the reliability of k-out-of-n systems", **IEEE Trans. Reliability**, vol R-36, Jun 1987, pp 261-265.
- 100 K.K.Aggarwal, S.Rai, "Reliability evaluation in computer communication networks", **IEEE Trans. Reliability**, vol R-30, Apr 1981, pp 32-35.
- 101 E.V.Prasad, A.K.Sarje, "Performance evaluation of multiple-bus multiprocessor systems", Communicated to **IEEE Trans. Computers**.
- 102 R.E.Barlow, F.Proshan, **Statistical Theory Of Reliability And Life Testing**, Holt, Rinehart And Winston, New York, 1975.
- 103 A.Satyanarayana, A.Prabhakar, "New topological formula and rapid algorithm for reliability analysis of complex networks" **IEEE Trans. Reliability**, vol R-27, Jun 1978, pp 82-100.
- 104 J.A.Abraham, "An improved algorithm for network reliability", **IEEE Trans. Reliability**, vol R-28, Apr 1979, pp 58-61.
- 105 A.K.Sarje, E.V.Prasad, "An efficient non-recursive algorithm for computing the reliability of k-out-of-n systems" , Under Review, **IEEE Trans. Reliability**, Ref. TR 87-160.
- 106 E.Horowitz, S.Sahni, **Fundamentals Of Data Structures**, Galgotia Book Source, New Delhi, 1983.

- 107 M.Lee, C.L.Wu, " Performance analysis of circuit switching baseline interconnection networks", in **Proc. 11th Int. Symp. Comput.Arch.**, 1984, pp 82-90.
- 108 C.L.Wu, T.Y.Feng, " On a class of multistage interconnection networks", **IEEE Trans.Computers**, vol C-29, Aug 1980, pp 694-702.
- 109 M.C.Pease, "The indirect binary n-cube microprocessor array", **IEEE Trans.Computers**, vol C-26, May 1977, pp 458-473.
- 110 H.D.Lawrie, "Access and alignment of data in an array processor", **IEEE Trans.Computers**, vol C-25, Dec 1976, pp 1145-1155.
- 111 L.N.Bhuyan, D.P.Agrawal, "Design and performance of generalized interconnection networks", **IEEE Trans.Computers**, vol C-32, Dec 1983, pp 1081-1090.
- 112 S.Thanawastien, V.P.Nelson, "Interference analysis of shuffle/exchange networks", **IEEE Trans.computers**, vol C-30, Aug 1981, pp 545-556.
- 113 R.Conterno, R.Melen, "An analytical model for a class of processor - memory interconnection networks", **IEEE Trans. Computers**, vol C-36, Nov 1987, pp 1374-1378.
- 114 C.P.Kruskal, M.Snir, "The performance of multistage interconnection networks for multiprocessors", **IEEE Trans. Computers**, vol C-32, Dec 1983, pp 1091-1098.

- 115 D.M.Dias, J.R.Jump, " Analysis and simulation of buffered delta networks", **IEEE Trans.Computers**, vol C-30, Apr 1981, pp 273-282.
- 116 C.Y.Chin, K.Hwang, "Packet switching networks for multi-processors and data flow computers", **IEEE Trans.Computers**, vol C-33, Nov 1984, pp 991-1003.
- 117 D.H.Lawrie, D.A.Padua, " Analysis of message switching with shuffle exchanges in multiprocessors", in **Proc. of the Workshop on Interconnection Networks for Parallel and Distributed Processing**, Apr 1980, pp 116-123.
- 118 R.G.Bennetts, "Analysis of reliability block diagrams by boolean techniques", **IEEE Trans. Reliability**, vol R-31, Jun 1982, pp 159-165.
- 119 G.B.Adams III, D.P.Agrawal, H.J.Siegel, " A survey and comparison of fault tolerant multistage interconnection networks", **IEEE Computer**, vol 20, Jun 1987, pp 14-27.
- 120 E.V.Prasad, A.K.Sarje, S.Rai, "Reliability modelling and analysis of multiprocessor systems", Communicaed to **J.of Microelecronics and Reliability**.
- 121 S.Rai, Arun Kumar, E.V.Prasad, "Computing performance index of a computer network", **Reliability Engineering**, vol 16, No 2, 1986, pp 153-161.

- 122 R.M.Geist, K.S.Trivedi, J.B.Dugan, M.Smotherman", Design of Hybrid Automated Realiability Predictor", in Proc. **IEEE/AIAA 5th Digital Avionics Systems Conf.**, Nov 1983.
- 123 J.B.Dugan, K.S.Trivedi, M.Smotherman, R.M.Geist, "The Hybrid Automated Reliability Predictor", **AIAA Journal On Guidance, Control and Dynamics**, May 1986, pp 319-331.
- 124 J.J.Stiffler, L.A.Bryant, "CARE III phase III report-Mathematical Description", **NASA Contract Report 3566**, Nov 1982.
- 125 K.S.Trivedi, R.M.Geist, "Decomposition in reliability analysis of fault-tolerant systems", **IEEE Trans.Reliability**, vol R-32, Dec. 1983, pp 463-468.
- 126 K.S.Trivedi, R.M.Geist, " A tutorial on the CARE III approach to reliability modeling", **NASA Contract Report 3488**, Dec. 1981.
- 127 A.D.Ingle, D.P.Sierwiorek, "Reliability models for multiprocessor systems with and without periodic maintenance", in **Proc. 7th Annual.Int.Conf.FTCS**, Los Angeles, CA, Jun 1977, pp 3-9.
- 128 T.C.K.Chou, J.A.Abraham, "Performance availability model of shared resource multiprocessors", **IEEE Trans.Reliability**, vol R-29, Apr 1980, pp 70-74.
- 129 C.R.Das, L.N.Bhuyan, V.V.S.Sarma, "Effect of maintenance on the dependability and performance of multiprocessor systems", **IEEE Trans.Reliability**, vol R-36, Jun 1987, pp 208-215.

- 130 C.R.Das,L.N.Bhuyan,"Reliability simulation of multiprocessor systems",in **Proc. Int. Conf. Parallel Processor**, Aug 1985, pp 591-598.
- 131 M.D.Beaudry,"Performance related reliability measures for computing systems",**IEEE Trans.Computers**, vol C-27, Jan 1978, pp 540-547.
- 132 T.F.Arnold, "The concept of coverge and its effect on the reliability model of a repairable system", **IEEE Trans. Computers**, vol C-22, Mar 1983, pp 251-254.

