On Fault Tolerant Multistage Interconnection Networks

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering
in
Computer Sc. & Engineering

Thapar University, Patiala

By:
Sheetal
80632025

Under the supervision of:
Rinkle Aggarwal
Lecturer (SS)

MAY 2008

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA – 147004

Certificate

I hereby certify that the work which is being presented in the thesis entitled, “On Fault Tolerant Multistage Interconnection Networks”, in partial fulfillment of the requirements for the award of degree of Master of Engineering in Computer Science & Engineering submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of Ms. Rinkle Aggarwal and refers other researcher’s works which are duly listed in the reference section.

The matter presented in this thesis has not been submitted for the award of any other degree of this or any other university.

(Sheetal)

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

(Rinkle Aggarwal)
Lecturer (SS),
Computer Science and Engineering Department,
Thapar University,
Patiala.

Countersigned by

(SEEMA BAWA)
Professor & Head,
Computer Science & Engineering Department,
Thapar University,
Patiala.

(R.K.SHARMA)
Dean (Academic Affairs),
Thapar University,
Patiala.

Acknowledgement

It is a great pleasure for me to acknowledge the guidance, assistance and help I have received from Ms. Rinkle Aggarwal, Lecturer (SS), Computer Science and Engineering Department. I am thankful for her continual support, encouragement, and invaluable suggestions. She not only provided me help whenever needed, but also the resources required to complete this thesis report on time.

I am also thankful to Dr. Seema Bawa, Head, Computer Science and Engineering Department for her kind help and cooperation.

I would also like to thank all the staff members of Computer Science and Engineering Department for providing me all the facilities required for the completion of my thesis work.

My special thanks to Dr. Himanshu Aggarwal, Reader, Computer Engineering Department, University College of Engineering, Punjabi University, Patiala for providing me invaluable support.

I would like to say thanks for support of my Classmates. I want to express my appreciation to every person who contributed with either inspirational or actual work to this thesis.

I am deeply grateful to my parents and brother for the inspiration and ever encouraging moral support, which enabled me to pursue my studies.

Sheetal

Abstract

The performance of a computer system depends directly on the time required to perform a basic operation and the number of these basic operations that can be performed concurrently. High performance computing systems can be designed using parallel processing. Parallel processing is achieved by using more than one processors or computers together they communicate with each other to solve a given problem. An interconnection network is a complex connection of switches and links permitting processors in a multiprocessor system to communicate among themselves or with memory modules.

Multistage interconnection networks play an important role in parallel systems. Multistage Interconnection Networks (MINs) consist of more than one stages of small interconnection elements called switching elements and links interconnecting them. MINs provide better way for the communication between different processors or memory modules with less complexity, fast communication, good fault tolerance, high reliability and low cost.

In this thesis two new networks MASEN and IASEN are proposed. These two networks are derived from an existing regular network ASEN-2. Both MASEN and IASEN are irregular networks, they have different number of switches at each stage. ASEN-2 network is a single switch fault-tolerant. If both switches in a loop are simultaneously faulty then some sources are disconnected from some destinations.

MASEN and IASEN provides multiple paths between each source and destination pair. It has been found that the bandwidth of both MASEN and IASEN is almost double than the existing ASEN-2 network. Reliability of MASEN and IASEN network is better than ASEN-2 and these networks also provides better permutation capabilities. For the small size networks cost of MASEN is less than ASEN-2 but it increases as the network size increases. whereas the cost of IASEN is little bit higher than ASEN-2.

Contents

CHAPTER 1
Introduction

1.1 Parallel Systems

There are many different ways to organize computational structures to achieve parallelism. Parallelism is a strategy multiple systems are working simultaneously for performing large, complex tasks faster. A large task can either be performed serially, one step following another, or can be decomposed into smaller tasks to be performed simultaneously, i.e., in parallel. Parallelism is achieved by:

Breaking up the task into smaller tasks
Assigning the smaller tasks to multiple processing elements
Processing elements solve the problem by coordinating with each other

A parallel system is a collection of processing elements that communicate and cooperate to solve large problems faster. Many research efforts around the world are being conducted with the purpose of determining that hardware and software organizations that are best suited for general purpose parallel processing [8]. The processors in a parallel system communicate with and work in cooperation with other processors. The primary reasons for using parallel systems:

Save time
Solve larger problems
Provide concurrency (do multiple things at the same time)

The structure of interconnection network is one of the key objectives to improve system performance. The communication subsystems linking processors, memory modules and I/O controllers in a parallel processing system is one of the most important architecture features and has a profound impact on system capabilities, performance, size and cost [3]. An Interconnection network of the processors provides the desired connectivity and performance at minimum cost is required for communication in parallel processing systems with a large number of components. A multistage interconnection network (MIN) is usually segmented into several stages and links to connect any sources to any destinations [1].

1.2 Interconnection Networks

An interconnection network is a complex connection of switches and links permitting processors in a multiprocessor system to communicate among themselves or with memory modules or I/O devices. It is the path, in which the data must travel in order to access memory in a shared memory computer or to communicate with other processes in a distributed memory environment or to use any I/O devices [8].

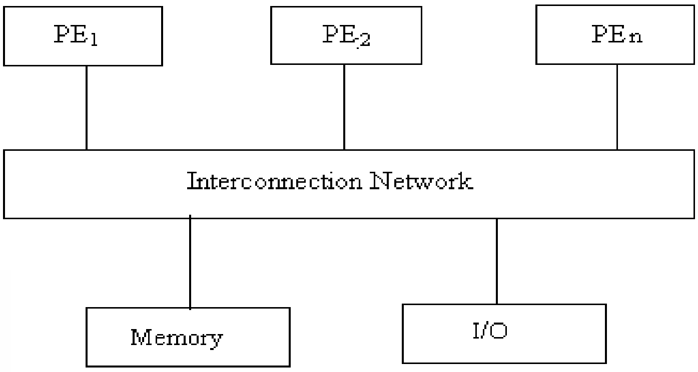


Figure 1.1: An Interconnection Network

1.3 Design Dimensions of An Interconnection Network

The key design dimensions for interconnection networks are:

Switching Methodology

Operational Mode

Control Strategy

Network Topology

1.3.1 Switching Methodology

Two major switching methodologies are:

Circuit switching: In circuit switching an end to end physical path is actually established between a source and a destination. This path exists as long as the data transmission is not complete. Circuit switching is suitable for bulk transmission of data.

Packet switching: In packet switching data is divided into packets and routed through the interconnection network without establishing a physical end to end connection path. Packet switching is more efficient for short messages.

1.3.2 Operational Modes

Operational modes can either be synchronous or asynchronous or a combination of the two.

Synchronous: This mode is useful for either a data manipulating function or for a data instruction broadcast. Synchronous control techniques are characterized by a global clock, which broadcasts clock signals to all devices in a system so that the entire system operates in lock-step fashion.

Asynchronous: Asynchronous communication is needed for multi processing in which connection requests are issued dynamically. Asynchronous techniques do not utilize a single global clock, but rather distribute the control function throughout the system, often utilizing many individual clocks for timing.

1.3.3 Control Strategy

A typical interconnection network consists of a number of switching elements and interconnection links. Interconnection functions are realized by properly setting control of switching elements. The control strategy can be of two types:

Centralized control strategy: In this all control settings are managed by a centralized controller. A main control system manages all of the switching elements.

Distributed control strategy: In this all control settings are managed by individual switching elements.

1.3.4 Network Topology

A network can be represented by a graph in which nodes indicate switches and edges represent communication links. Topology is the pattern in which the individual switches are interconnected to other elements such as processors, memories and other switching elements. The topologies can be categorized into two groups:

1 **Static:** In static topology, links between two processors are with passive and dedicated buses which cannot be reconfigured for direct connection with other processors. Static networks that are generally used in message- passing architectures [14]. The following network topologies are commonly used:

Ring Network

Star Connected Network
Completely Connected Network
Tree Network
Mesh Network
Hypercube Network

2 **Dynamic:** On the other hand, links in dynamic topology can be reconfigured by setting network's active switching elements [14]. Dynamic interconnection networks implement one of the following interconnection techniques:

Crossbar Networks
Bus based Network
Multistage Interconnection Networks

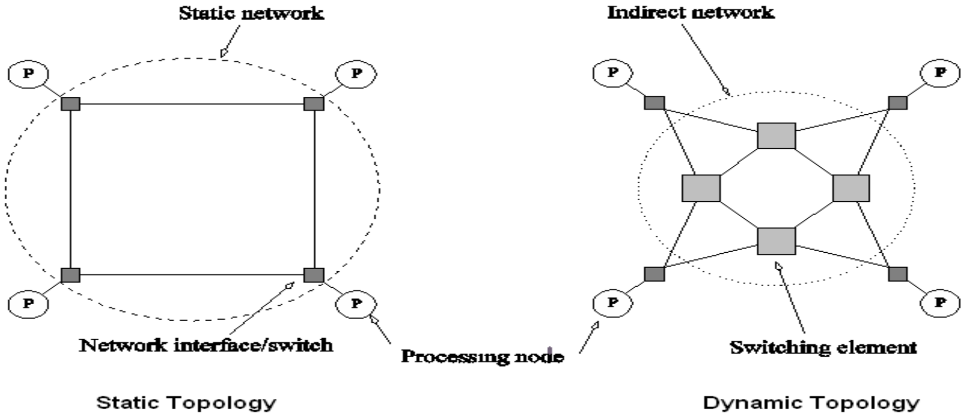


Figure 1.2: Types of Network Topology

1.3.4.1 Static Topology

Ring Network: In ring network, every device has two adjacent neighbors for communication. In a ring network, all the communication messages travel in the same direction whether clockwise or anti-clockwise. Damage of cable or device can result in the breakdown of the whole network [14].

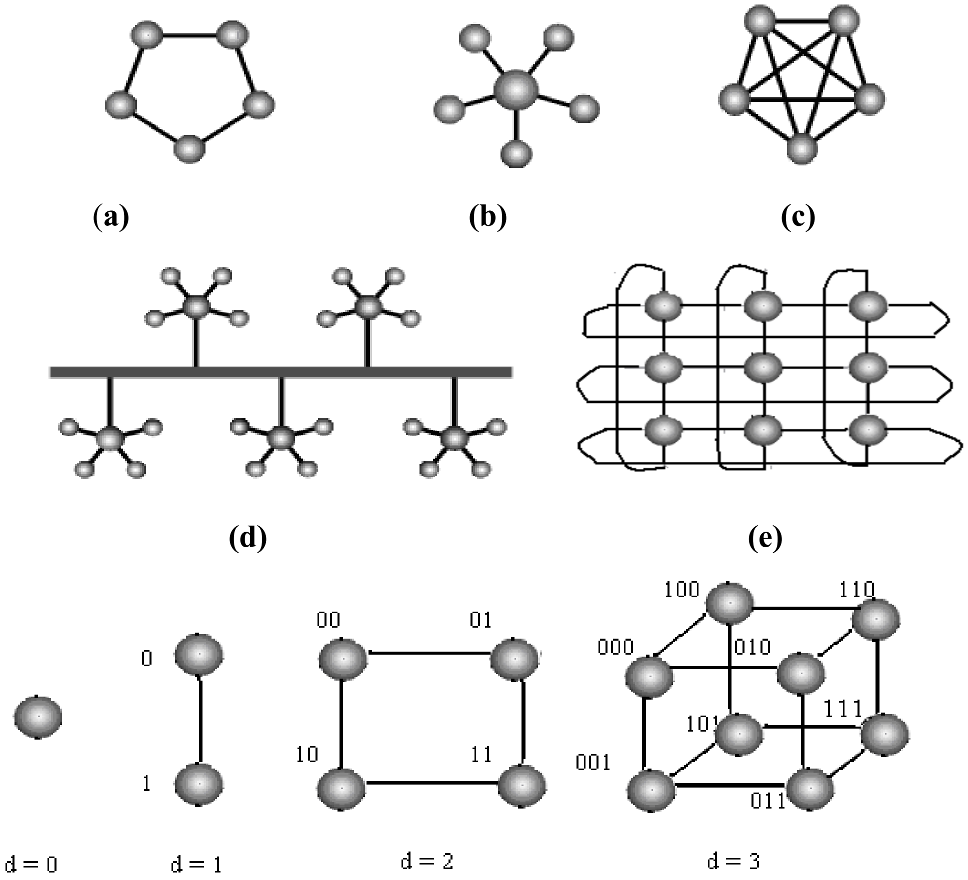
Star Connected Network: In the computer networking the most commonly used topology is the star topology. All the computers in the star topologies are connected to central device like hub, switch or router. The functionality of all these devices is different. The main disadvantage of this kind of topology is that if central device stops working then there will be no transmission between any nodes.

Completely Connected Network: In this topology, all nodes are directly connected to every other node with a point to point link. In this message sent to the destination can take any possible shortest, easiest route to reach its destination. In the previous topologies star and ring, messages are usually broadcasted to every computer [14]. In the Ring topology message can travel in only one direction i.e. clockwise or anticlockwise.

Tree Network: Tree topologies are comprised of the multiple star topologies on a bus. Tree topologies integrate multiple star topologies together onto a bus. Only the hub devices can connect directly with the tree bus and each hub functions as a root of a tree of the network devices, as shown in Figure 1.3 (d). This bus/star/hybrid combination supports future expandability of the computer networks, much better than a bus or star.

Mesh Network: A mesh simply connects one processor to four others, as shown in figure 1.3 (e). Processors along the top and bottom edges would be connected to the processor in the same column along the opposite edge. The processors on the last column would be connected to the first processor on the next row. The processor in the bottom right corner would have a connection to the processor in the top left corner. With this layout, there are 2N connections, but it takes at most N-1 shifts to get data from one processor to any other.

Hypercube network: A zero dimensional hypercube is a single processor and a one-dimensional hypercube connects two zero dimensional hypercube i.e. a line connecting two nodes defines a one-dimensional cube. A square with four nodes is a two-dimensional cube. Generally a hypercube of dimension d+1 is constructed by connecting corresponding processors in two hypercube of dimension d [14]. In hypercube two processors are connected if and only if the binary representation of their labels differs in a single position. It may be considered as a mesh with additional, long distance connections. A disadvantage of the hypercube interconnect is that it is more complex than the mesh. In the Figure 1.3 (f), processors in the cubes of dimension 1, 2, and 3 are labeled with integers, represented as binary numbers. Two processors are neighbors in dimension d if and only if their binary labels differ only in the dth place.



(a) (b) (c)

(d) (e)

(f)

Figure 1.3: Various types of Static Network Topologies are (a) Ring (b) Star (c) Completely connected (d) Tree (e) Mesh (f) Hypercube of dimension zero to three.

1.3.4.2 Dynamic Topology

Crossbar Networks: The crossbar makes a connection from a given vertical bus to the appropriate horizontal bus and allows traffic to flow along this path. In crossbar network, the other horizontal or vertical buses can be supporting a flow of traffic at the same time. For example, if each horizontal bus needs to talk to a separate vertical bus, then they all can be moving data at the same time. This completely eliminates the single-shared-resource limitation of the system bus. The crossbar is a preferable approach for high performance multiprocessors [24].

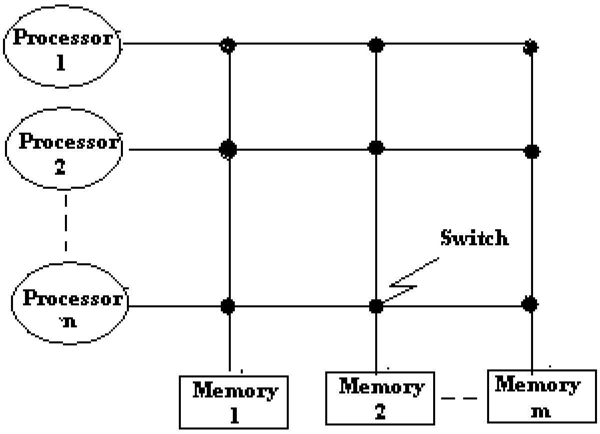


Figure 1.4: A Crossbar Network

Bus network: Bus topology uses a common backbone to connect all the network devices in a network in a linear shape shown in Figure 1.5. Some of the simplest and earliest parallel machines used bus. All processors access a common bus for exchanging data. The distance between any two nodes is O(1) in a bus. The bus also provides a convenient broadcast media. However, the bandwidth of the shared bus is a major bottleneck.

A bus is highly non scalable architecture, because only one processor can communicate on the bus at a time. A bus network design offers minimum bandwidth. It is highly inefficient and unreliable because of a single bus, the failure of which will make it unusable [8]. Buses are commonly used in shared memory parallel computers to communicate read and write requests to a shared global memory. A single bus organization is simple and inexpensive. By adding more number of processors or memory increases the bus contention, which decreases the bus throughput.

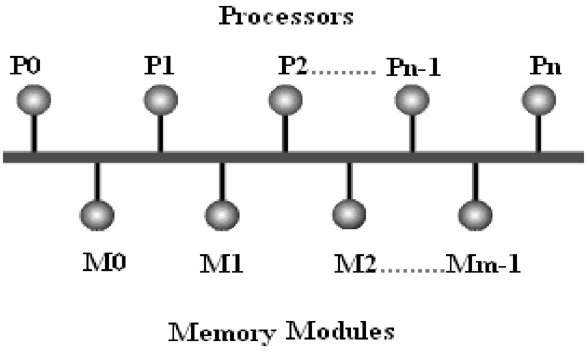


Figure 1.5: A Bus Network

1.4 Multistage interconnection networks

Multistage interconnection networks (MINs) consist of more than one stages of small interconnection elements called switching elements and links interconnecting them. Multistage interconnection networks (MINs) are used in multiprocessing systems to provide cost-effective, high-bandwidth communication between processors and/or memory modules. A MIN normally connects N inputs to N outputs and is referred as an N × N MIN. The parameter N is called the size of the network [25].

There are several different multistage interconnection networks proposed and studied in the literature. Figure 1.6 illustrates a structure of multistage interconnection network, which are representatives of a general class of networks. This figure shows the connection between p inputs and b outputs, and connection between these is via n number of stages.

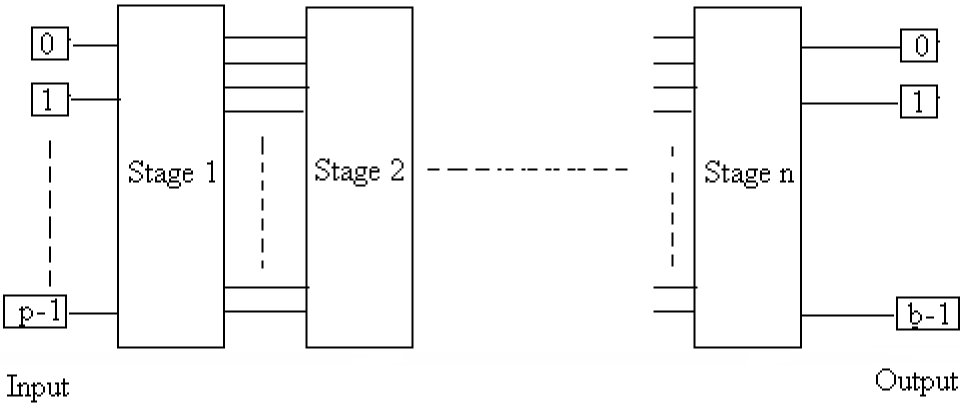


Figure 1.6: A Multistage Interconnection Network (MIN)

A multistage interconnection network is actually a compromise between crossbar and shared bus networks, indicated in the table 1.1, describing the properties of various types of multiprocessor interconnections networks [8]. Multistage interconnection networks are:

Attempt to reduce cost

Attempt to decrease the path length

In a multistage interconnection network, as in a crossbar, switching elements are distinct from processors. Instead messages pass through a series of switch stages.

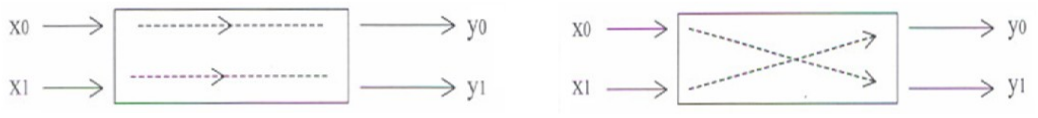
Table 1.1: Properties of different Networks

Property	**Bus**	**Crossbar**	**Multistage**
Speed	Low	High	High
Cost	Low	High	Moderate
Reliability	Low	High	High
Configurability	High	Low	Moderate
Complexity	Low	High	Moderate

The network can be constructed from unidirectional or bi-directional switches and links. In a unidirectional MIN, all messages must traverse the same number of wires, and so the cost of sending a message is independent of processor location. In effect, all processors are equidistant. In a bi-directional MIN, the number of wires traversed depends to some extent on processor location, although to a lesser extent than a mesh or hypercube [7].

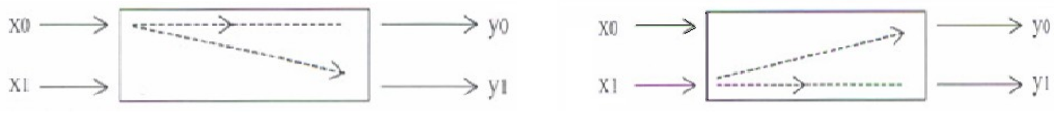
1.4.1 Switching Elements

The switching element is the basic element of a multistage interconnection network. It may be viewed as a very small network. These switches are the devices having multiple inputs and multiple outputs [1]. A switch box have any one of the following four states i.e. straight, exchange, upper broadcast and lower broadcast shown in Figure. 1.7.



Straight

Exchange



Lower Broadcast

Upper Broadcast

Figure 1.7: Types of Switching Elements

1.5 Classification of Multistage Interconnection Networks

Multistage interconnection networks can be classified according to different categories. The main classification categories are:

According to number of paths

According to switch

According to control

According to availability of path

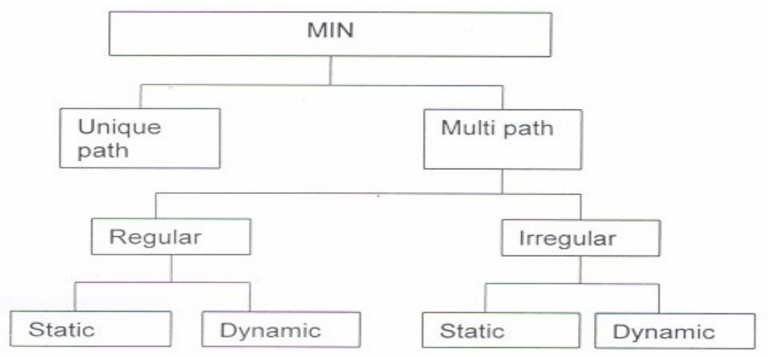


Figure 1.8: Types of Multistage Interconnection Networks (MINs)

1.5.1 Classification According to Number of Paths

Unique path networks*:* These networks provide unique path between every source and destination. The failure of any switching element along the path disconnects some source-destination pairs. These are not reliable for a large multiprocessor system, as they cannot tolerate even a single fault. These networks provide poor performance as a source destination connection may be blocked by a previously established connection [19].

Multi path networks*:* These provide more than one path between source and destination. In case, there is a failure of one switching element in the path, the request is routed through some alternative path. Multi path multistage interconnection networks can be static or dynamic. For static networks, backtracking is used if fault is encountered,. In dynamic networks, if a fault is encountered in a particular stage, a switching element in preceding stage will re-route data through an alternative available path [19].

1.5.2 Classification According to Switches

Regular networks: Regular multistage interconnection networks have an equal number of switching elements per stage. As a result they may impose equal time delay to all h requests passing through them [22].

Irregular networks: Irregular multistage interconnection networks have unequal number of switching elements at each stage and thus they are inherently multi path in nature. For a given source destination pair, multiple paths are available [22].

1.5.3 Classification According to Control

Flip controlled networks: Flip controlled multistage interconnection networks have a common control signal for switching in various switching elements at a given stage [8]. These networks are less complicated due to lesser number of control signals but have lesser bandwidth.

Distributed control networks: Distributed control multistage interconnection networks have a separate control signal for every switching element. These have higher bandwidth due to selection of source destination pair at a given time and are quite complex [8].

1.5.4 Classification According to Availability of Path

Blocking networks: Blocking is the conflict that arises between two paths established between two source and destination pairs. When sources generate connection requests to an identical destination, it is possible that some paths need to pass simultaneously through an identical output of the SE. A path can never share the output with another path at any point in time, and hence blocking arises in the SE. Omega network is a type of blocking network [24].

Non blocking networks: A network is called non blocking if it is possible to route from any source to any destination, in presence of other established source-destination routes, provided no two sources have same destination. In other words, a network that can handle all possible connections without blocking is called non-blocking network [24].

1.6 Routing of MINs

In networks the process of moving data packet from source to destination is called routing. Routing is a key feature of the networks because it enables messages to pass from one computer to another and eventually reach the destination. Each intermediary switch performs routing by passing along the message to the next switch. The routing in multistage interconnection networks take place through the generation of routing tags, which specify a fault free path. Each switch in a MIN performs routing according to the routing tag [6]. There are several objectives of routing strategy:

Minimize the transmit time

Minimize the costs

Maximize the network throughput capability

To minimize the transmit times under conditions of changing load, many control signals or overheads would be sent so that network throughput would be reduced. On the other hand, maximizing the throughput could be done at the expense of packet transmission time.

1.6.1 Routing Tag

Routing tag is a way of describing the path through the network. For multistage interconnection networks, these tags are generally expressed as a multi-digit binary number expressed by the destination. Each successive digit in this binary code is used to find out the switch in the next stage along a desired path. For each multistage interconnection network routing tag may be different [6].

1.6.2 Types of Routing in MINs

There are basically three types of routing that is commonly used in multistage interconnection networks.

- **Non adaptive routing:** In this method a source leans a fault when a path is attempting to establish reaches the faulty network component. A notice of fault is sent to the source, which than tries next alternative available path. This method has poor performance though it requires little hardware.
- **Adaptive routing :** The adaptive routing can be of following types :

  Notification on demand: with notification on demand, a source maintains a table of faults it encountered in attempting to establish paths and uses this information to guide the future routing.

Broadcast routing: With broadcast notification of a fault, all the sources are notified of the fault components as they are diagnosed.

- **Dynamic routing:** A fault free path need not to be specified by a source if the routing tags are modified in response to the faults as a path is followed or established. The dynamic routing can be accomplished in multistage interconnection networks constructed of switches, which are capable of performing the necessary tag revision.

1.7 Performance Parameters

The performance parameters applicable for MINs are:

1.7.1 Permutation Passability

A one to one correspondence between a source and a destination is called Permutation. Permutation passability means how many input requests occurring simultaneously at the input are able to pass through a given network, and how many of them will successfully mature i.e. will reach their destination [24]. The request always pass from the most suitable path available (generally, the minimum length path), if such path is busy or faulty then the request is pass through an alternate path. If no alternate path is available then the request has to be simply dropped or said to be having clash. So some of the requests will pass through the most favorable path, others have to be routed through an available alternative path. If no alternative paths are available then some requests can not be served at all [27].

1.7.2. Fault Tolerance

A fault tolerance multistage interconnection network provides service routing even under the faults. Fault can be permanent or transient in nature. Fault tolerance is a criteria that must be met for the network which has tolerated a given fault or faults. A network is called single fault tolerant if it can tolerate or function in case of a single fault. In general, if any set of i-faults can be tolerated by a network, then network is called i-fault tolerant [2].

1.7.3. Bandwidth (BW)

It is the most common performance parameter used in analyzing a synchronous interconnection network. It is defined as the number of processing elements requests honored per unit of time. In other words, it is the average number of requests honored per unit time. So bandwidth (BW) also takes into account the memory access conflicts caused by the random nature of the process requests. BW is also defined as the expected number of destination receiving requests in any given cycle. Thus it is the total number of requests matured. A high bandwidth is often desired at reasonably low network cost.

1.7.4 Throughput (TP)

Throughput is the maximum number of traffic accepted by the network per unit time. The average number of packets delivered from source to destination by network in unit time called Throughput. It is measured as packets per node per cycle. It can also be defined as the average number of cells delivered by the network per unit time per unit line.

1.7.5. Probability of Acceptance (P_a)

It is defined as the ratio of expected bandwidth to the expected number of requests generated per cycle. Expected means, the request generated by the source in a random access environment.

1.7.6. Processor Utilization (PU)

It is the expected percentage of time a processor is active. A processor is said to be active when it is doing internal computation without accessing the global memory.

1.7.7. Reliability

Reliability, of a system is the probability that it will successfully perform its intended operations for a given time under stated operating conditions. In the presence of multiple CPU’s if one goes down, the others can be able to take over its work. In case of networks if one switch is fail then the possibility that it will not fail for some time period. It is one of the major design issue for any network [20].

1.7.8. Cost Effectiveness

To estimate the cost of a network, one common method is to calculate the switch complexity with the assumption that the cost of a switch is proportional to the number of gates involved, which is roughly proportional to the number of 'crosspoints' within a switch. For example, a 4 x 4 switch has 16 units of hardware cost whereas a 2 x 2 switch has 4 units. For the multiplexers and demultiplexers, we roughly assume that each of *K* x 1 multiplexers or I x *K* demultiplexers has *K* units of cost [4].

Chapter 2

Literature Survey

2.1 Introduction

To solve the problem of providing fast, reliable and efficient communication at a reasonable cost in large parallel processing systems, many different networks between the extremes of single bus and the cross bar have been proposed. Such interconnection networks can be constructed from single or multiple stages of switches. In a single stage network, data may have to be passed through the switches several times before reaching the final destination. In multistage network, one pass of multistage stages of switches is usually sufficient. The way input units are connected with the output units, determine the functional characteristics of the network i.e. the allowable interconnections [8].

The single stage network is also called a recirculating network. Data items may have to recirculate through the single stage several times before reaching their final destination. Number of recirculations needed depends upon connectivity in a single stage network. In general, the higher is the hardware connectivity, the lesser is the number of recirculations.

2.2 Cube interconnection network

In a cube, vertical lines connect vertices whose addresses differ in most significant bit position. Vertices at both ends of diagonal lines differ in middle bit position. Horizontal lines differ in least significant bit positions [3]. The unit cube concept can be extended to an n-dimensional unit space, called n-cube, with n bits per vertex.

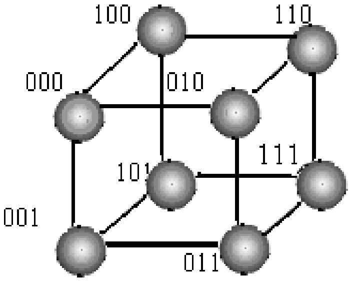


Figure 2.1: A three dimensional binary cube

The implementation of a single stage cube network is given in the figure 2.2 for 8 nodes. The interconnection of the switching elements, corresponding to three routing functions is given separately in this figure.

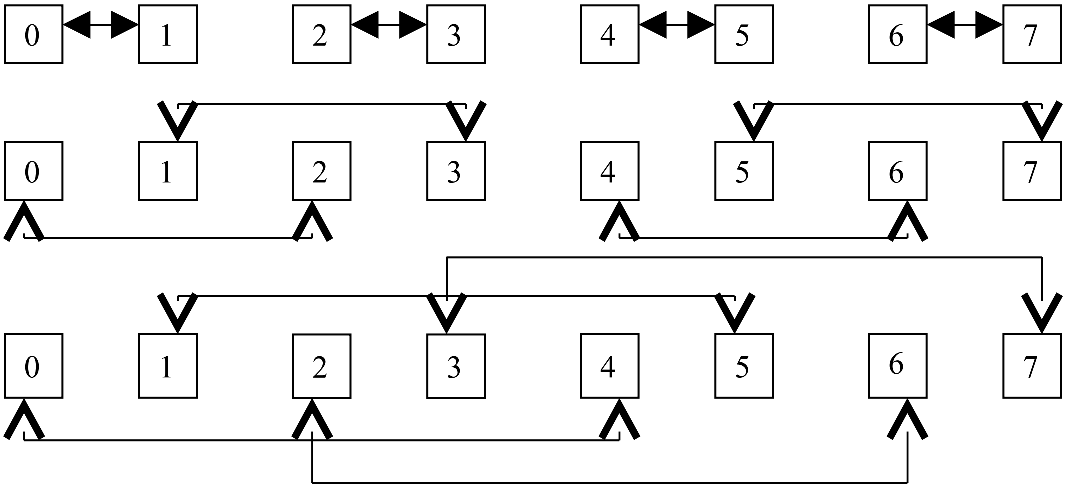


Figure 2.2: The Recirculating Cube Network for N = 8

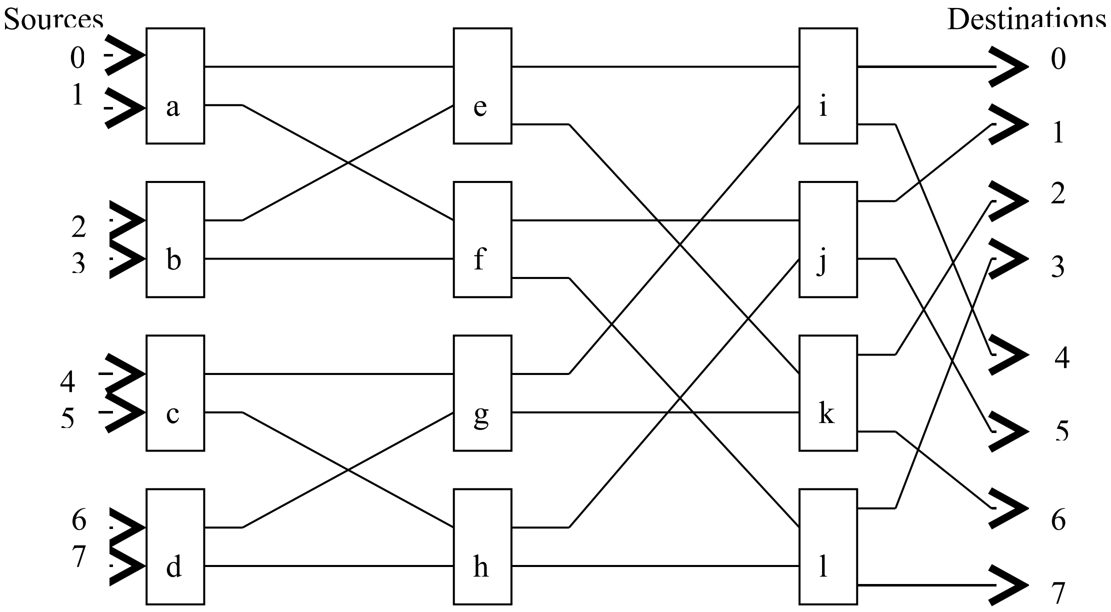


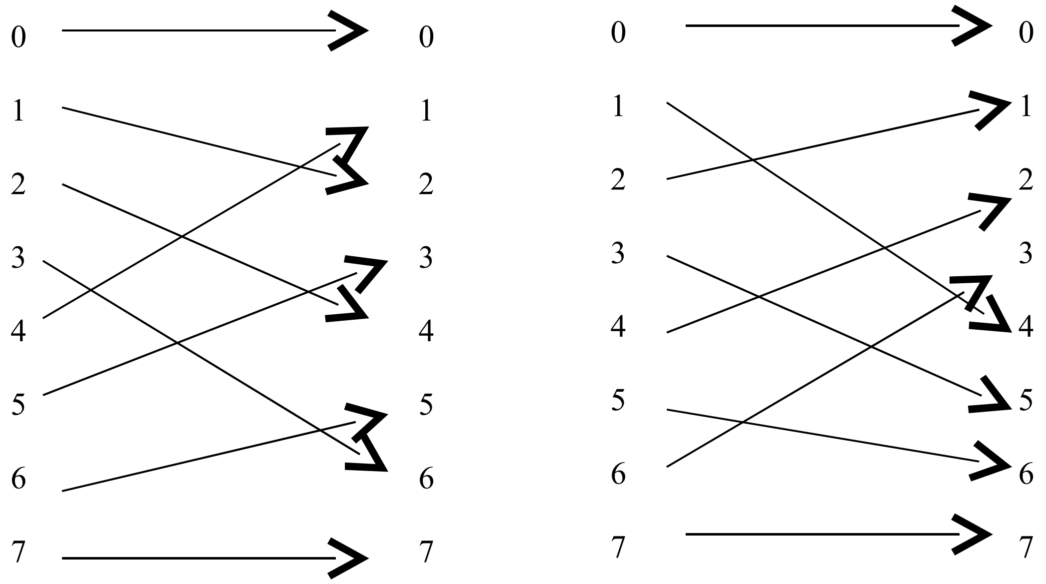
Figure 2.3: A Multistage Cube Network for N =8

The same set of cube routing functions, c_0, c_1, c_2 can also be implemented by using a three stage cube network. Two function switch boxes i.e. straight and exchange are used in construction of multistage cube network. The stages are numbered as 0 at input end and increased to n-1 at output. The stage i implements C_i

routing function for i = 0, 1, 2..(n-1). So, switch box at stage i connect an input line to output line that differs from it only at i^{th} bit position [14].

2.3 Shuffle Exchange Network

Shuffle exchange network is based on two routing functions, Shuffle and Exchange. A perfect shuffle of N = 8 is shown in figure 2.4(a). Perfect shuffle cuts the deck into two halves from the center and than intermixes them evenly. Inverse perfect shuffle does the opposite to restore the original ordering as shown in Figure 2.4(b). These shuffle exchange functions can be implemented as either recirculating network or a multistage network [14]. Figure 2.5 represent a single stage recirculating shuffle exchange network, where solid lines indicate exchange and dashed lines indicate shuffle.



(a) (b)

Figure 2.4: (a) A perfect shuffle and (b) The inverse perfect shuffle

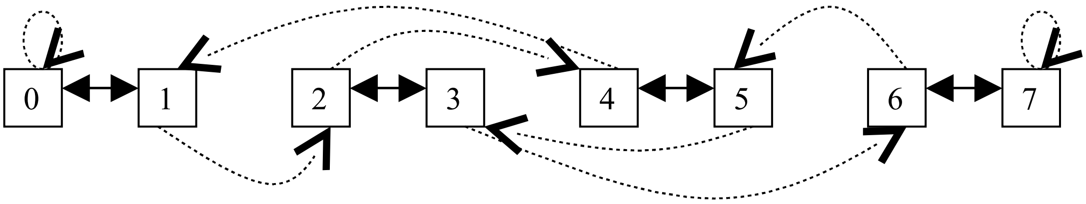


Figure 2.5: A shuffle exchange recirculating network for N = 8

The shuffle exchange has been implemented with multistage Omega network. Figure 2.6 represents Omega network for N= 8. An N × N Omega network consists of

$\log_2 N$ identical stages and between two stages there is a perfect shuffle interconnection. Each stage has N/2 switch boxes under independent box control [14].

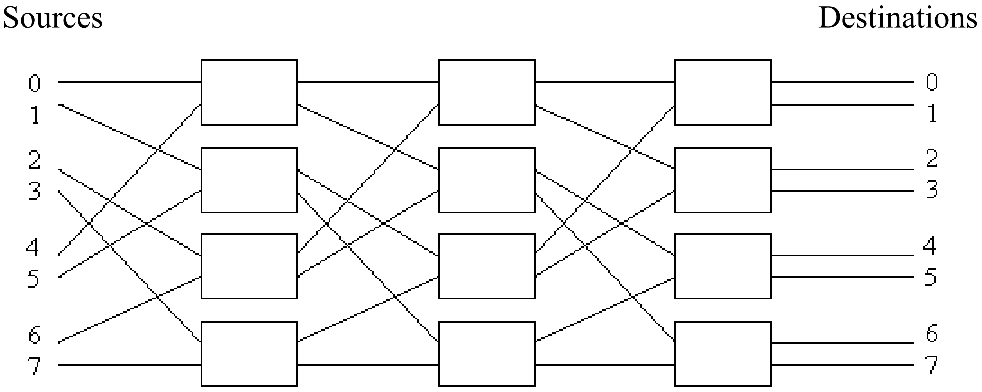


Figure 2.6: An Omega network for N = 8

2.4 Double Tree Network (DOT)

The double tree network was originally proposed as a fault-detecting and correcting network. It is an irregular network with the same number of inputs and outputs. This network has many redundant paths of different lengths between an input-output terminal pair. It consists of a right and a left half. Each half of the network resembles a binary tree, and is a mirror image of each other [19, 23]. An 8×8 DOT network is shown in Figure. 2.7.

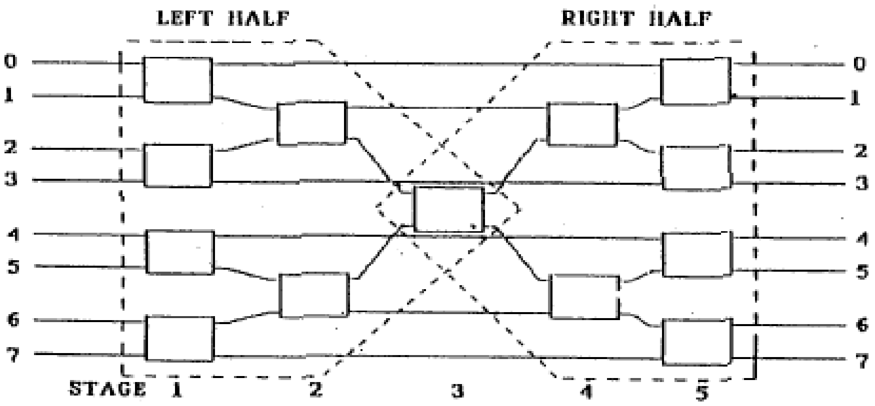


Figure 2.7: Double tree network for N=8

2.5 Modified Double Tree Network (MDOT)

MDOT is an example of non-fault tolerant irregular network, having different number of switches at each stage. Figure 2.8 indicates a modified double tree network for N = 8. Total number of stages in this type of network are 2^n-1, where $n = \log_2 N$

and the number of switching elements are $2^{n+1}-3$. The numbers of switching elements at the last and first stages are equal. Similarly, numbers of switching elements in the next stages are equal [4].

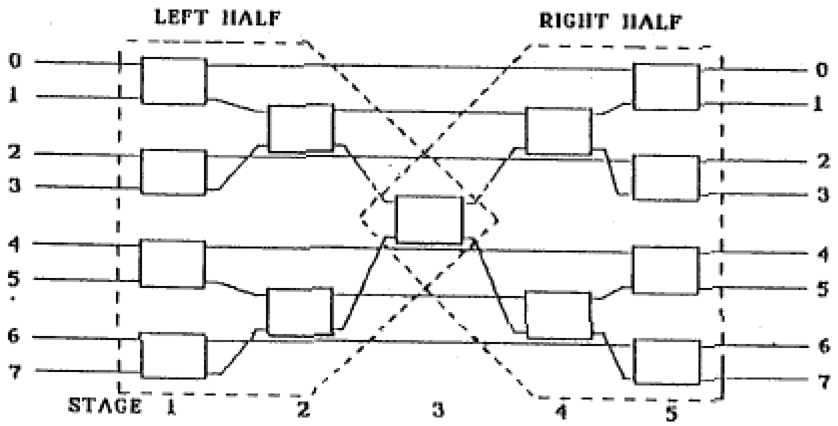


Figure 2.8: Modified double tree network for N=8

2.6 Four Tree network

Four Tree network is an irregular MIN and provides multiple paths of varying lengths between a Source-Destination pair. Four Tree network of size N×N is constructed with two identical groups G^i, each consisting of MDOT network of size N/2 × N/2, which are arranged one above the other. The two groups are formed based on the most significant bit of the source-destination terminals with a MSB 0 falling into group G^0 and with MSB 1 falling into G^1 [6].

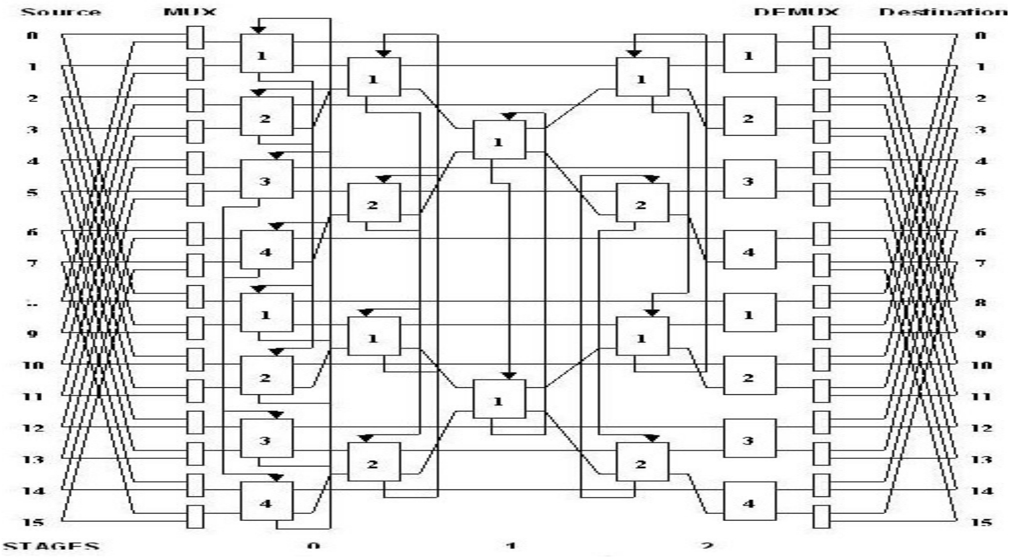


Figure 2.9: FT network for N=16

Chapter 3
Problem Statement

3.1 Problem Definition

Multistage interconnection networks play an important role in the parallel computing systems. A multistage interconnection network consists of more than one stages of interconnection elements and links. Reliability and efficiency in terms of the speed of operations and the cost are the major considerations in the design of multistage interconnection networks.

Permutation passability is an important parameter to study the behavior of any multistage interconnection network. Permutation passability means how many input requests occurring simultaneously at the input are able to pass through a given network and reach successfully at the intended destination. Network should have the capability of passing as many requests as it can. Permutation passability should be high.

A network bandwidth is the average number of requests honored per unit time. Bandwidth of the network should be high. Throughput (TP), Probability of Acceptance (P_a), Processor Utilization (PU) of a network is based on the bandwidth of the network. If bandwidth is high than the value of all these parameters is also increases.

Reliability is also the major consideration in the design of MIN. Reliability, of a system is the probability that it will successfully perform its intended operations for a given time under stated operating conditions. The reliability of the network should be high. Reliability is, in the presence of multiple CPU’s if one is faulty, the others can be able to take over its work.

3.2 Problem Statement

A lot of work has already been done in the design and analysis of regular multistage interconnection networks. Since irregular networks, in general, are less costly and inherently multi path in nature compared to regular multistage interconnection networks. So, analysis of irregular multistage interconnection network is important.

In this thesis, two new irregular networks are proposed. New networks are derived from an existing regular network ASEN-2. Routing of ASEN-2 is very simple. But there are some limitations of this network. As it is a regular network, so it provides less number of paths between each source and destination pair. It also provides limited permutation passability, less reliability and bandwidth as compared to other irregular networks.

The objectives of the thesis are:

To design fault-tolerant networks that can achieve the general goals, i.e. high reliability, good performance even in the presence of faults at reasonable cost.

To evaluate the permutation passability of these networks.

To evaluate the Bandwidth, Throughput (TP), Probability of acceptance (P_a), Processor Utilization (PU) of a network.

To calculate the reliability of these networks in terms of Mean Time to failure.

Comparison of the proposed networks with existing networks.

Chapter 4

Construction and Routing Scheme

4.1 ASEN-2 Network

Augmented Shuffle Exchange Network (ASEN-2) is a regular network, means it has same number of switches in each stage. ASEN-2 network is constructed from Shuffle Exchange network by adding a stage of 2×1 multiplexers at the initial stage and 1×2 demultiplexer at last stage. It provides multiple paths between a source and a destination.

ASEN-2 of size N × N with N number of sources and N number of destination consists of $\log_2 N - 1$ stages where the initial stage consists of N/2 switches of size 3×3 and the last stage consist of N/2 switches of size 2×2. ASEN-2 provides fault tolerance using links between the conjugate pairs of switches [15, 21]. A 16×16 ASEN-2 network is shown in Figure. 4.1.

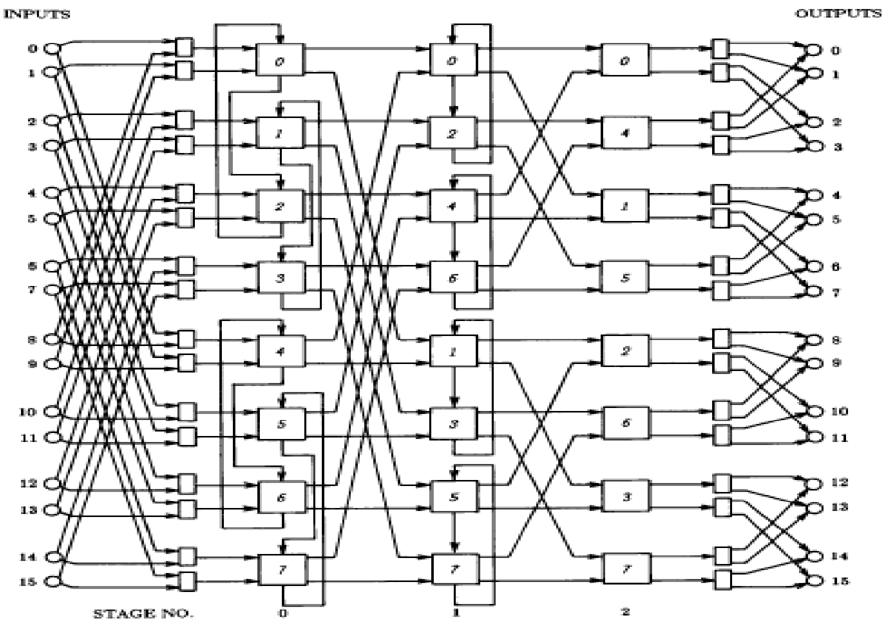


Figure 4.1: Augmented Shuffle Exchange network (ASEN-2)

4.1.1 Redundancy Graph

Redundancy graph is a method of showing all possible paths between a source and a destination. A node in the graph represents the switching elements and an edge represents the link between the switches [21]. Redundancy graph of ASEN is shown in figure 4.2.

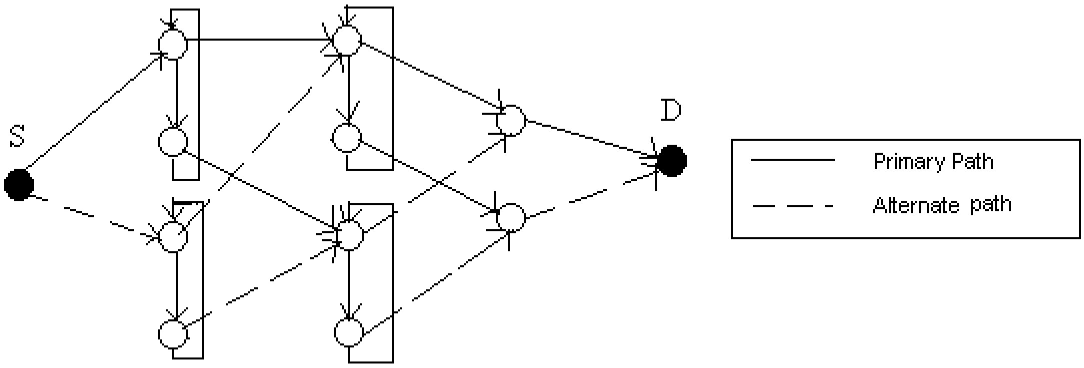


Figure 4.2: Redundancy Graph of ASEN-2

4.1.2. Routing of ASEN-2

The reliability and performance improvement obtained from a multiple path network depends upon how effectively the alternate paths available are used by the routing algorithm. One can use a backtracking routing algorithm that searches extensively for an available fault-free path.

Routing tag is used in the routing of a network. Routing tag is a way of describing the path through the network. For multistage interconnection networks, these tags are generally expressed as binary of the destination. Each successive digit of this binary code is used for finding the switch in the next stage. This control is called distributed if the devices, using the network switches can be set on their own based on the tag information [15]. The routing tag is the binary code of the destination. Let the source S and destination D be represented in binary code as:

$S=s_0, s_1 \dots s_{n-2}, s_{n-1}$

$D=d_0, d_1 \dots d_{n-2}, d_{n-1}$

Routing Procedure→

1. For each source submit the request for connection to the primary switch, along with the binary label (d_0) of the required destination. If the primary switch, or the multiplexer at the input of the primary switch, is faulty, then submit the request to the secondary switch. If the secondary switch is also faulty, drop the request.

2. For each switch in initial stage requests may arrive on any of the 3 input links. For each request if the required output link is busy or cannot be used because of a fault in the next stage, route the request via the auxiliary output link to the next switch in the

loop. If the auxiliary link is also unusable because it is busy or faulty, then drop the request.

3. For each switch in intermediate stage requests may arrive on any of three input links, as there are 3 × 3 switches are used. For each request, route it via corresponding output link if it is not available then submit the request to the alternate output link. If the alternate link is faulty or busy then drop the request.
4. For each switch in last stage requests may arrive on any of two input links, as 2×2 switches are there. For each request route it via the corresponding output link. If the required output link is busy or faulty then drop the request.
5. For each demultiplexer can receive maximum of one request. If a request arrives make a connection to the upper or the lower output link according to whether the routing bit is 0 or 1. For each destination up to two requests may arrive [15].

Chapter 5

Performance Analysis

5.1 Permutation Passability

Permutation passability means how many input requests occurring simultaneously at the input are able to pass through a given network, and how many of them will successfully mature i.e. will reach their destination. MINs can be of unique path or multi path. Unique path MINs provide a single path between a source and a destination and in multi path MINs more than one path are available. The requests always pass from the most favorable path available (generally, the path with minimum length). The alternate path is followed in following cases:

1. If such path is busy (currently using by another request)
2. If one or more switches are faulty

If no alternate path is available then the request is simply dropped or said to be clashed. So some of the requests will pass through the most favorable path, others have to be routed through an available alternative path. If no alternative paths are available then some requests can not be served at all [24].

In this chapter permutation passability of ASEN, MASEN and IASEN is evaluated. All three networks are multi path networks. Permutation passability is a very important performance parameter in the multi path MINs, i.e. if a number of requests simultaneously occur at source at a particular moment of time then how many of them successfully mature after being routed through the network. The network tries to send the request through the most favorable path if such path is not available then the request is routed through the alternative paths. If there are no alternative paths available, the request is dropped. The desirable character of any network is that it should be such that it allows maximum requests through it with minimum path length. These networks are analyzed with and without faults.

5.1.1 Permutation Without Faults

All the permutations are passed if every SE is without faults. In the event of no-fault operation, shortest available path is chosen. The shortest path-length is 2 irrespective of the size of the network in case of irregular MINs which secures the delay encountered to pass each permutation to be identical. In this we assume that no switch is faulty.

5.1.1.1 ASEN-2 Network

Case 1: Set of six source destination pairs are (4,2), (1,3), (9,6), (3,10), (8,2), (7,10). The various paths that will be followed for this set are:

4->MUX(4)->C->C1->A'->DEMUX(1)->2	Path Length=3
1->MUX(1)->A->A1->B1-> B'->DEMUX(3)->3	Path Length=4
9->MUX(9)->E->A1->C'->DEMUX(5)->6	Path Length=3
3->MUX(3)->B->F1->F'->DEMUX(11)->10	Path Length=3
8->MUX(8)->E->G->C1->A'->DEMUX(1)->2	Path Length=4
7->MUX(15)->H->H1->G1->E'->DEMUX(9)->10	Path Length=4

Total requests=6

Requests Matured= 6

Average Path Length (21/6) =2.666666

Case 2: Set of nine source destination pairs are (1,3), (2,4), (3,5), (6,5), (12,4), (13,2), (9,6), (4,12), (5,0). The various paths that will be followed for this set are :

1->MUX(1)->A->A1->A'->DEMUX(1)->3	Path Length=3
2->MUX(2)->B->B1->D'->DEMUX(6)->4	Path Length=3
3->MUX(11)->F->B1->A1->C'->DEMUX(4)->5	Path Length=4
6->MUX(6)->D->D1->D'->DEMUX(6)->5	Path Length=3
12->MUX(4)->C->C1->C'->DEMUX(4)->4	Path Length=3
13->MUX(13)->CLASH	Path Length=0
9->MUX(9)->E->A1->C'->DEMUX(5)->6	Path Length=3
4->MUX(12)->G->G1->G'->DEMUX(12)->12	Path Length=3
5->MUX(5)->CLASH	Path Length=0

Total requests=9

Requests Matured= 7

Average Path Length (22/9) =2.444444

Case 3: Set of 11 source destination pairs are (0,1), (4,5), (6,2), (7,8), (9,10), (12,14), (3,5), (1,11), (2,5), (15,7), (14,9). The various paths that will be followed are:

0->MUX(0)->A->A1->A'->DEMUX(0)->1	Path Length=3
4->MUX(4)->C->C1->C'->DEMUX(4)->5	Path Length=3
6->MUX(6)->D->D1->B'->DEMUX(3)->2	Path Length=3
7->MUX(15)->H->H1->F'->DEMUX(10)->8	Path Length=3
9->MUX(9)->E->E1->E'->DEMUX(8)->10	Path Length=3
12->MUX(12)->G->G1->G'->DEMUX(13)->14	Path Length=3
3->MUX(3)->B->B1->D'->DEMUX(6)->5	Path Length=3
1->MUX(1)->A->C->G1->E'->DEMUX(9)->11	Path Length=4
2->MUX(2)->B->D->D1->D'->DEMUX(6)->5	Path Length=4
15->CLASH	Path Length=0
14->MUX(14)->H->F->F1->F'->DEMUX(10)->9	Path Length=4

Total requests=11

Requests Matured= 10

Average Path Length (33/11) =3.000000

Case 4: Set of 8 source destination pairs are (1,3), (2,4), (3,5), (6,5), (9,6), (4,12), (5,14), (7,9). The various paths that will be followed for this set are:

1->MUX(1)->A->A1->A'->DEMUX(1)->3 Path Length=3
2->MUX(2)->B->B1->D'-> DEMUX(6)->4 Path Length=3
3->MUX(3)->B->D->D1->D'->DEMUX(6)->5 Path Length=4
6->MUX(14)->CLASH Path Length=0
9->MUX(9)->E->A1->C'->DEMUX(5)->6 Path Length=3
4->MUX(12)->G->G1->G'->DEMUX(12)->12 Path Length=3
5->MUX(5)->C->F1->H'-> DEMUX(15)->14 Path Length=3
7->MUX(15)->H->H1->F'->DEMUX(10)->9 Path Length=3

Total requests=8
Requests Matured=7
Average Path Length (22/8) =2.750000

5.1.2 Permutation Passability With Faults

In this we take the case if one or more switches are faulty. As multiple paths of varying lengths are available, faulty network will pass most of the request through the alternate paths. In this we calculate the ability of a network to work under faults. This will show that at a particular moment of time, if a number of requests simultaneously occur at source, how many of them successfully mature will i.e. will reach their destination. In this node A is considered as faulty.

5.1.2.1 ASEN-2 Network

Case 1: Set of six source destination pairs are (4,2), (1,3), (9,6), (3,10), (8,2), (7,10). The various paths that will be followed for this set are:

4->MUX(4)->C->C1->A'->DEMUX(1)->2 Path Length=3
1->MUX(9)->E->G->C1-> A'->DEMUX(1)->3 Path Length=4
9->MUX(9)->CLASH Path Length=0
3->MUX(3)->B->F1->F'->DEMUX(11)->10 Path Length=3
8->MUX(8)->E->A1->A'->DEMUX(1)->2 Path Length=3
7->MUX(15)->H->H1->G1->E'->DEMUX(9)->10 Path Length=4

Total requests=6
Requests Matured=5
Average Path Length (17/6) =2.833333

Case 2: Set of nine source destination pairs are (1,3), (2,4), (3,5), (6,5), (12,4), (13,2), (9,6), (4,12), (5,0). The various paths that will be followed for this set are :

1->MUX(9)->E->A1->A'->DEMUX(1)->3 Path Length=3
2->MUX(2)->B->B1->D'->DEMUX(6)->4 Path Length=3
3->MUX(11)->F->B1->A1->C'->DEMUX(4)->5 Path Length=4
6->MUX(6)->D->D1->D'->DEMUX(6)->5 Path Length=3
12->MUX(4)->C->C1->C'->DEMUX(4)->4 Path Length=3
13->MUX(13)->CLASH Path Length=0
9->MUX(9)->CLASH Path Length=0
4->MUX(12)->G->G1->G'->DEMUX(12)->12 Path Length=3
5->MUX(5)->CLASH Path Length=0

Total requests=9
Requests Matured= 6
Average Path Length (19/9) =2.111111

Case 3: Set of 11 source destination pairs are (0,1), (4,5), (6,2), (7,8), (9,10), (12,14), (3,5), (1,11), (2,5), (15,7), (14,9). The various paths followed for this set are:

0->MUX(8)->E->A1->A'->DEMUX(0)->1 Path Length=3
4->MUX(4)->C->C1->C'->DEMUX(4)->5 Path Length=3
6->MUX(6)->D->D1->B'->DEMUX(3)->2 Path Length=3
7->MUX(15)->H->H1->F'->DEMUX(10)->8 Path Length=3
9->MUX(9)->E->E1->E'->DEMUX(8)->10 Path Length=3
12->MUX(12)->G->G1->G'->DEMUX(13)->14 Path Length=3
3->MUX(3)->B->B1->D'->DEMUX(6)->5 Path Length=3
1->MUX(1)->CLASH Path Length=0
2->MUX(2)->B->D->D1->D'->DEMUX(6)->5 Path Length=4
15->CLASH Path Length=0
14->MUX(14)->H->F->F1->F'->DEMUX(10)->9 Path Length=4

Total requests=11
Requests Matured= 9
Average Path Length (29/11) =2.636363

Case 4: Set of 8 source destination pairs are (1,3), (2,4), (3,5), (6,5), (9,6), (4,12), (5,14), (7,9). The various paths that will be followed for this set are:

1->MUX(9)->E->A1->A'->DEMUX(1)->3	Path Length=3
2->MUX(2)->B->B1->D'-> DEMUX(6)->4	Path Length=3
3->MUX(3)->B->D->D1->D'->DEMUX(6)->5	Path Length=4
6->MUX(14)->CLASH	Path Length=0
9->MUX(9)->CLASH	Path Length=0
4->MUX(12)->G->G1->G'->DEMUX(12)->12	Path Length=3
5->MUX(5)->C->F1->H'-> DEMUX(15)->14	Path Length=3
7->MUX(15)->H->H1->F'->DEMUX(10)->9	Path Length=3

Total requests=8

Requests Matured= 6

Average Path Length (19/8) =2.37500

5.2 Bandwidth Analysis

Following assumptions have been made to compute bandwidth equations.

Assumptions:

Here are the assumptions on the basis of which the probabilistic relations have been carried out.

1) The IN operates in a synchronous mode, i.e., the requests issued by the processors begin and end simultaneously.
2) The requests are random and the request generated by a processor is independent of the request generated by another processor.
3) Requests which are not accepted are blocked or rejected.
4) The requests generated in a cycle are independent of the requests generated in the previous cycle.
5) "p_0" is the probability with which a processor generates a request. Thus, p_0 is the rate of request of a processor per cycle.
6) The probability with which processor P_i addresses memory M_i is zero i.e. there is no favorite memory.
7) Networks are of same size i.e. N × N.

Assume a multistage interconnection network (MIN) of size $a^n \times b^n$ constructed from a × b crossbar modules. Thus there are a^n sources connected by b^n destinations. Applying analysis of crossbar network to a x b crossbar switch and then extending the analysis for the complete MIN. Each stage of MIN is controlled by a distinct destination digit (in base b) for setting of individual a × b switches. Since the destinations are independent and uniformly distributed, so are the destination digits. We can apply the result of analysis of crossbar network to any a × b module in the multistage interconnection network.

Given the request rate p at each of the a inputs of a × b crossbar module, the expected number of request that it passes per unit time is given by:

$b\text{-}b\,(1\text{-}p/b)^{a}$

Dividing the above expression by the number of outputs lines of a × b modules gives us the rate of request on any one of b output lines:

$1\text{-}(1\text{-}p/b)^{a}$

Thus for any stage of a MIN, the output rate of requests p_{out} is a function of its input rate and is given by:

$p_{out} = 1\text{-}(1\text{-}p_{in}/b)^{a}$

Since the output rate of a stage is the input rate of the next stage, one can recursively evaluate the output rate of any stage starting at stage i. In particular, the output rate of the final stage n determines the bandwidth of a multistage interconnection network, that is, the number of requests accepted per cycle.

Let us define p_i to be the rate of requests on an output line of stage i, then the following equation determines the bandwidth. BW of $a^n \times b^n$ multistage interconnection network, the rate of requests generates by each source is given by:

$BW= b^n p_n$

Which is according to the definition is total number of requests matured.

Where $p_i =1(1\text{-}p_{i-1}/b)^{a}$ and $p_0=p$

5.2.1 Bandwidth of ASEN-2

ASEN-2 allows two ways of routing a connection request in every stage except the final stage. If the destinations are assumed to be fault free, it allows that all single switch faults can be tolerated. From the definitions of various performance parameters for the generalized MIN, probability equations for N×N ASEN-2 network is given by:

$P_1=1-(1-p_0/3)^3$

$P_2=1-(1-P_1/3)^3$

$P_3=1-(1-P_2/2)^2$

$BW = b^4 P_3$

where $p_0 = p$ and p is memory request rate, i.e. Probability of Request Generation

To have an idea that how these equations have been derived, Consider Figure 4.1 of ASEN-2 network. There are 3 stages. First and last stages have N/2 switches whereas intermediate stage has N/4 switches. As discussed above probability to reach i^{th} stage is given by $1 – (1- p_{i-1}/b)^a$. In the first stage there are 3 × 3 switches, therefore a=3 and b=3. In second stage, the numbers of switches are half of the first stage, probability to reach second stage would be half. In third stage the input lines are coming form first and second stage. The numbers of switches in second stage are half of the third stage and for requests coming from first stage the probability will be same.

5.3 Throughput (TP)

The average number of packets delivered from source to destination by network in unit time is called Throughput. Both IASEN and MASEN have same throughput but more than ASEN-2.

$$\text{Throughput (TP)} = BW / (a^n . T)$$

5.4 Probability of Acceptance (P_a)

It is probability that a request generated by a source should be successful in reaching the destination. Probability of acceptance is the probability that, in a random-access environment, a request submitted by a source is accepted by a destination without getting blocked by other requests or connections in the network. This probability is usually evaluated by assuming that all the sources simultaneously generate their requests for connection with a probability p and aim them at randomly chosen destinations. These requests propagate through the network one stage at a time. When two or more requests arrive at a switch requiring the same output link, the requests that are serviced are chosen at random and the others are blocked and dropped. In the case of ASEN, MASEN and IASEN, a maximum of three requests can arrive at a switch. Up to two requests with the same routing tag bit can be serviced. The probability of acceptance is defined as the ratio of the expected number of successful requests to the expected number of requests submitted by the sources.

Probability of acceptance ($P_{a)}$ = BW / (a^n . p)

5.5 Processor Utilization (PU)

It is the expected percentage of time a processor is active doing internal computation without accessing the global memory. Both MASEN and IASEN has same PU but more than ASEN-2.

PU = BW / (a^n .p. T)

5.6 Reliability

Reliability of ASEN-2, MASEN, IASEN networks are analyzed in terms of Mean time to Failure (MTTF). MTTF of a MIN is evaluated using simple series-parallel reliability models.

- **Series configuration model**: A series configuration model is constructed by connecting all the components in a series system. This type of configuration is very sensitive because the failure of a single component make whole of the system fail. The reliability of a series configuration model is always worse than the poorest component in it. Series configuration model shown in fig 5.1.

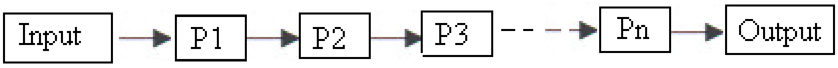


Figure 5.1: Series configuration model

- **Parallel configuration model**: In this all components are connected parallel to each other. In this if one component fails than the data can follow another path and system will be active. This type of model is shown in figure 5.2.

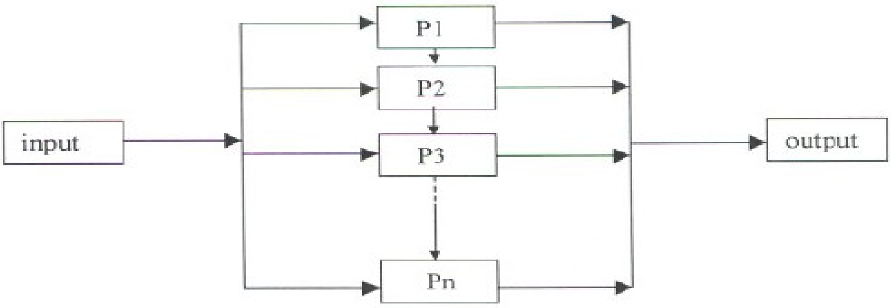


Figure 5.2: Parallel configuration model

- **Series-Parallel Configuration model:** The two loops in a conjugate pair are in parallel and all the conjugate pairs of loops are in series as shown in Figure 5.3.

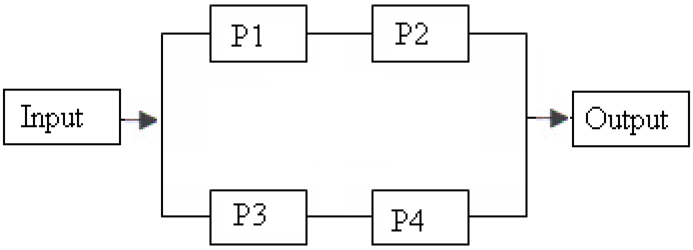


Figure 5.3: Series-Parallel configuration model

To make the Reliability analysis traceable, we need to have some assumptions [20]. The assumptions used in the analysis on the failure rates of the components are given below:

1. Switch failure occur independently in a network with a failure rate of for 2×2 crossbar switches (a reasonable estimate for is about 10^{-6}per hour).
2. Failure of the multiplexers and demultiplexers also occur independently with failure rates of $_{m}$ and $_{d}$respectively.
3. Assuming that the hardware complexity of a component is directly proportional to the gate counts of it, one can derive a failure rate of the component From the basic logic design of MUX and DEMUX, we can say that number of gates in a 2m × l MUX or a 1 × 2m DEMUX is roughly double of that in a m × 1 MUX or a 1 × m DEMUX. Based on the gate counts of crossbar switches, the number of gates in a 2 × 2 crossbar switch is approximately equal to that in a 2 × 1 MUX or a 1 × 2 DEMUX. Thus to simplify the analysis we can assume that $_{m} = m\ /2$ for a m × 1 MUX, where

$_m$ failure rate of MUX or $_d$ (= $_m$) for 1 × m DEMUX, where $_d$ failure rate of DEMUX.

The adaptive routing scheme considers a 2 × 2 switch in the last stage and its associated DEMUX as a series system, so we consider these three elements as single component (SE2d), and based on a gate count, a failure rate of $_{2d}$ = 2 can be assigned to this group of elements. Also let $_2$ and $_3$ be the failure rate for the 2 × 2 (SE2) and the 3 × 3 switch (SE3), then based on gate count, $_2$ = and $_3$ = 2.25 and $_{3m}$ = 4.25 .

4. Irregular MINs are inherently multi-path and the MTTF needs to be calculated at all existing path-lengths separately based upon the series and parallel models of reliability.

5.6.1 ASEN-2 Reliability

For the reliability both the optimistic and pessimistic analysis of the networks has been done. These were extended to incorporate the added complexity of the switches used in the ASEN-2, the multiplexers and demultiplexers used at the input and output interfaces of the network.

5.6.1.1 ASEN-2 Optimistic (Upper bound) Analysis

To obtain an upper bound for the ASEN-2 and observed that each source is connected to two multiplexers and each switch is connected is a conjugate pair. So if we assume that the ASEN-2 is operational as long as one of the two multiplexers attached to a source is operational and both components in a conjugate pair are not faulty. Block diagram of upper bound is shown in figure 5.4.

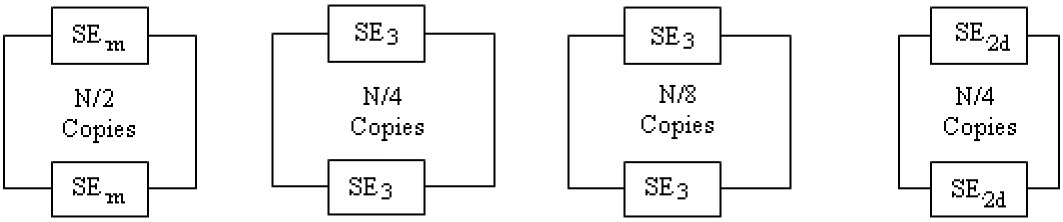


Figure 5.4: Upper Bound of ASEN-2

Reliability Equations are:

$$f1 = \left[1 - \left(1 - e^{-\lambda_m t}\right)^2\right]^{\left(N/2\right)}$$

$$f2 = \left[1 - \left(1 - e^{-\lambda_3 t}\right)^2\right]^{\left(N/4 + N/4(n-3)\right)}$$

$$f3 = \left[1 - \left(1 - e^{-\lambda_{2d} t}\right)^2\right]^{\left(N/4\right)}$$

$$R_{Optimistic} = f1 * f2 * f3$$

$$MTTF = \int_0^{\infty} R_{Optimistic}(\mathrm{t})\, \mathrm{dt}$$

5.6.1.2 ASEN-2 Pessimistic (Lower Bound) Analysis

At the input side of the ASEN-2, the routing algorithm does not consider the multiplexers to be an integral part of a given 3 × 3 SE. For example, as long as one of the two multiplexers attached to switch 0 is operational, switch 0 can still be used for routing. Hence, if we group two multiplexers with each switch on the input side and consider them as a series system (SE_{3m}), then we will have a conservative estimate of the reliability of these three components. Their aggregated failure rate will be $_{3m}$= 4.25 . Finally, these aggregated components and the switches in the intermediate stages can be arranged in pairs of conjugate loops. To obtain lower bound of the ASEN-2, assume the network is failed whenever more than one loop has a faulty element or more than one switch in the last stage fails. The block diagram is shown in Figure. 5.5.

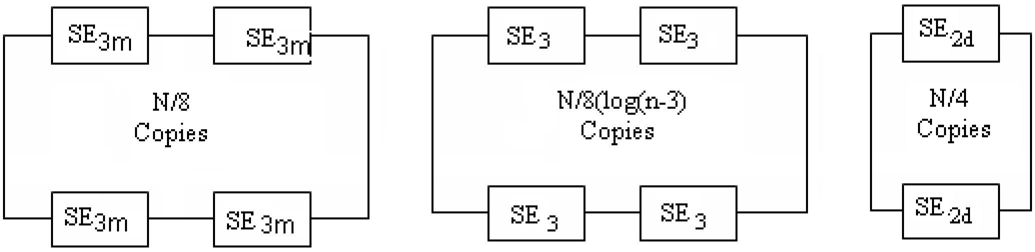


Figure 5.5: Lower Bound of ASEN-2

Reliability Equations are:

$$f1 = \left[1 - \left(1 - e^{-\lambda_{3m} t}\right)^2\right]^{\left(N/4\right)}$$

$$f2 = \left[1 - \left(1 - e^{-\lambda_3 t}\right)^2\right]^{\left(N/4(n-3)\right)}$$

$$f3 = \left[1 - \left(1 - e^{-\lambda_{2d} t}\right)^2\right]^{\left(N/4\right)}$$

$$R_{Pessimistic} = f1 * f2 * f3$$

$$MTTF = \int_{0}^{\infty} R_{Pessimistic}(t)\, dt$$

5.7 Cost Effectiveness

To estimate the cost of a network, one common method is used to calculate the switch complexity with an assumption that the cost of a switch is proportional to the number of gates involved, which is roughly proportional to the number of cross-points within a switch. For example a 2 × 2 switch has four units of hardware cost, whereas a 3 × 3 switch has nine units. For the multiplexers and demultiplexers, assume that each of m × 1 multiplexers or 1 × m demultiplexers has m units of cost [4].

Table 5.7: Cost Functions

MIN	Cost
ASEN	$3N(1.5 \log_2 N - 1)$
MASEN	$3N(1.5 \log_2 N) - 52$
IASEN	$3N(1.5 \log_2 N) - 20$

Chapter 6

Experimental Results

6.1 Permutation Passability Analysis

Permutation passability analysis is done on the basis explained below:

6.1.1 On the basis of Requests Matured

On the basis of number of request matured IASEN network is best among the two networks, as shown below in table 6.1 and in figure. 6.1. MASEN is good than the ASEN-2 network.

Table 6.1: Comparison on the basis of Requests Matured

Number of Requests	**ASEN-2**	**IASEN**	**MASEN**
Total Requests	34	34	34
Requests Matured Without Fault	30	34	33
Requests Matured With Fault	26	34	31

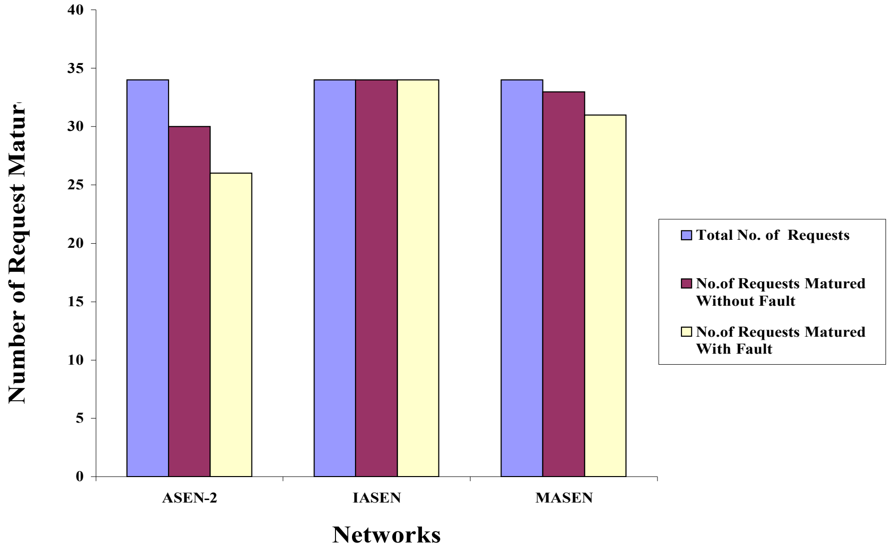


Figure 6.1: Comparison on the basis of Requests Matured

6.1.2 On the basis of Average Path Length

Path length of MASEN network is less than ASEN and IASEN.

Table 6.2: Comparison on the basis of Average Path Length

Criteria	**ASEN-2**	**IASEN**	**MASEN**
Without Fault	10.85	11.01	9.80
With Fault	9.94	10.86	9.73

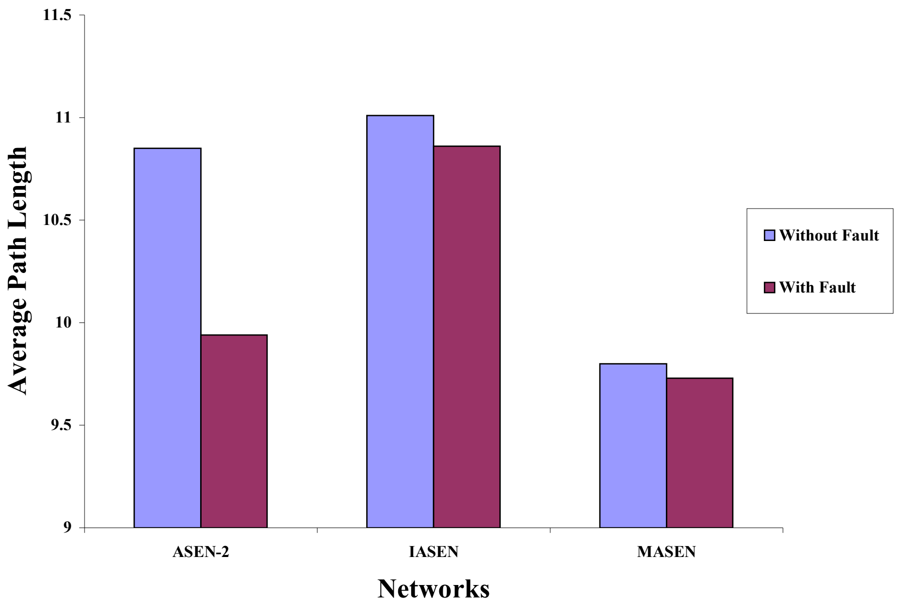


Figure 6.2: Comparison on the basis of Average Path Length

6.2 Bandwidth

In this section bandwidth comparison of ASEN-2, MASEN and IASEN is shown. The bandwidth of MASEN and IASEN is almost double than the ASEN-2 network as shown below in Table 6.3 and in figure 6.3. But both MASEN and IASEN networks has same bandwidth.

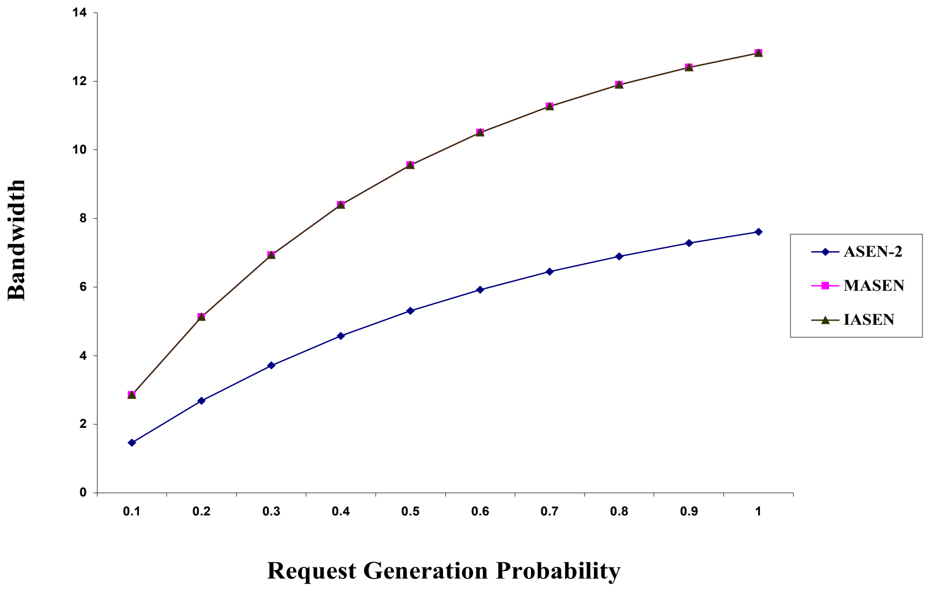


Figure 6.3: Comparison on the basis of Bandwidth

Values of the parameters throughput, probability of acceptance, processor utilization are depended on the bandwidth. So the value of these parameters increases as the bandwidth increases.

6.3 Throughput (TP)

The TP of both the networks is almost double than the ASEN-2. Throughput=BW/ (a^n . T). As shown below in table 6.4 and fig 6.4.

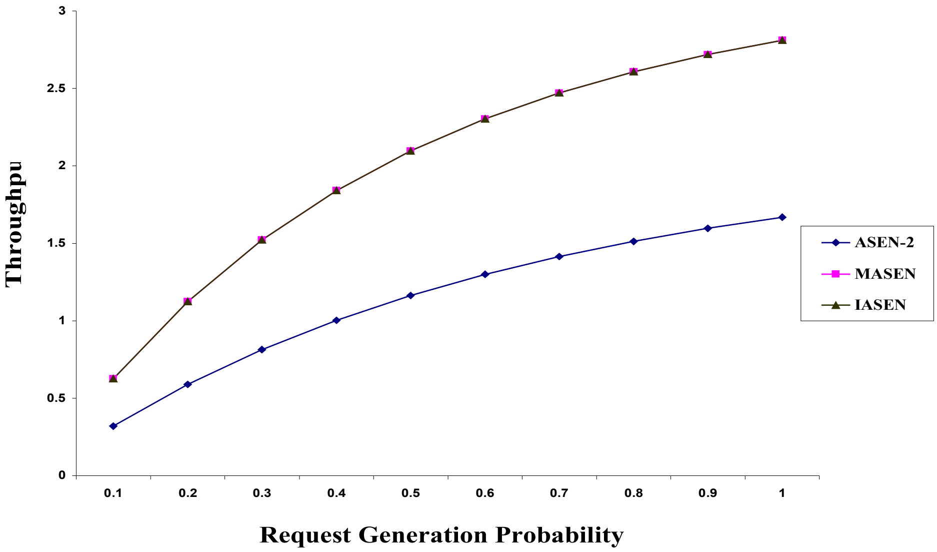


Figure 6.4: Comparison on the basis of Throughput

6.4 Probability of Acceptance (P_a)

Probability of acceptance= BW / (a^n . p), so the MASEN and IASEN networks has same probability of acceptance but almost double than the ASEN-2 network. As shown below in table 6.5 and Figure. 6.5.

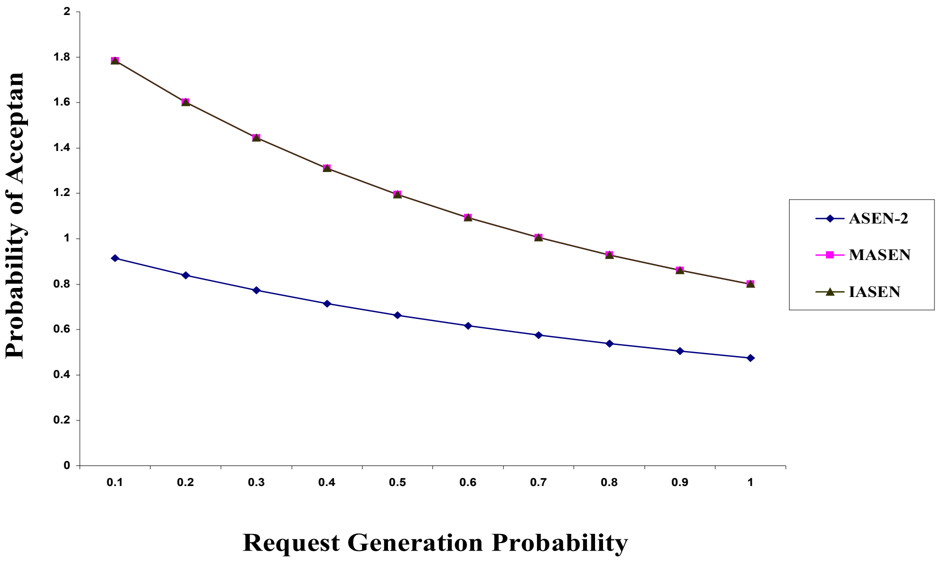


Figure 6.5: Comparison on the basis of Probability of Acceptance

6.5 Processor Utilization

Processor utilization of MASEN and IASEN is also almost double than ASEN-2. Processor utilization (PU) = BW / (a^n .p. T). Both MASEN and IASEN has same PU. Comparison of these networks on the basis of PU is shown below in table 6.6 and in Figure 6.6.

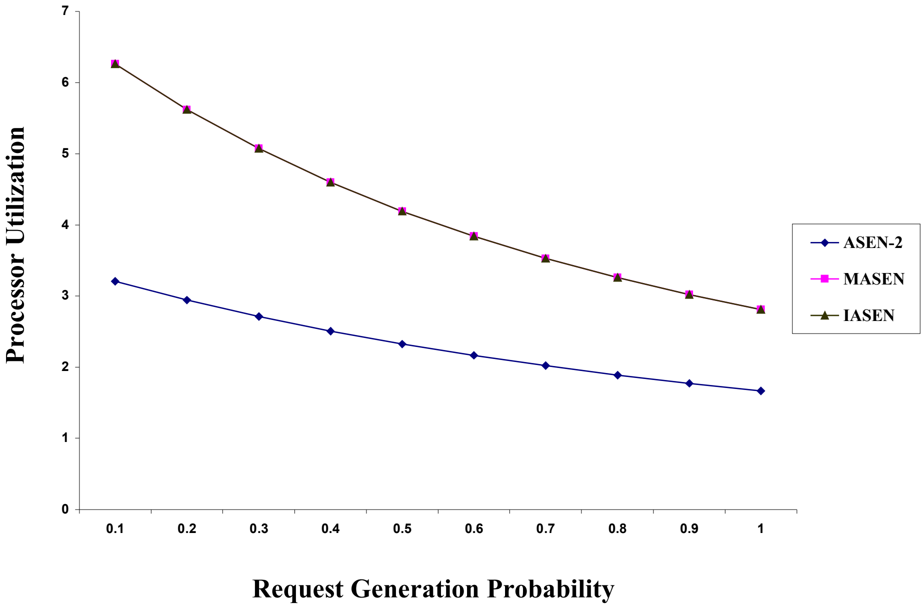


Figure 6.6: Comparison on the basis of Processor Utilization

6.6 Reliability

Reliability of ASEN-2, MASEN, IASEN networks are analyzed in terms of Mean time to Failure (MTTF). The upper bound MTTF and Lower Bound MTTF of these networks is shown below in the following two sections.

6.6.1 Comparative Analysis of Upper Bound MTTF

Optimistic approach is used for calculating the Upper bound MTTF (explained in chapter 5). The result is shown in Table 6.7 and in Figure 6.7.

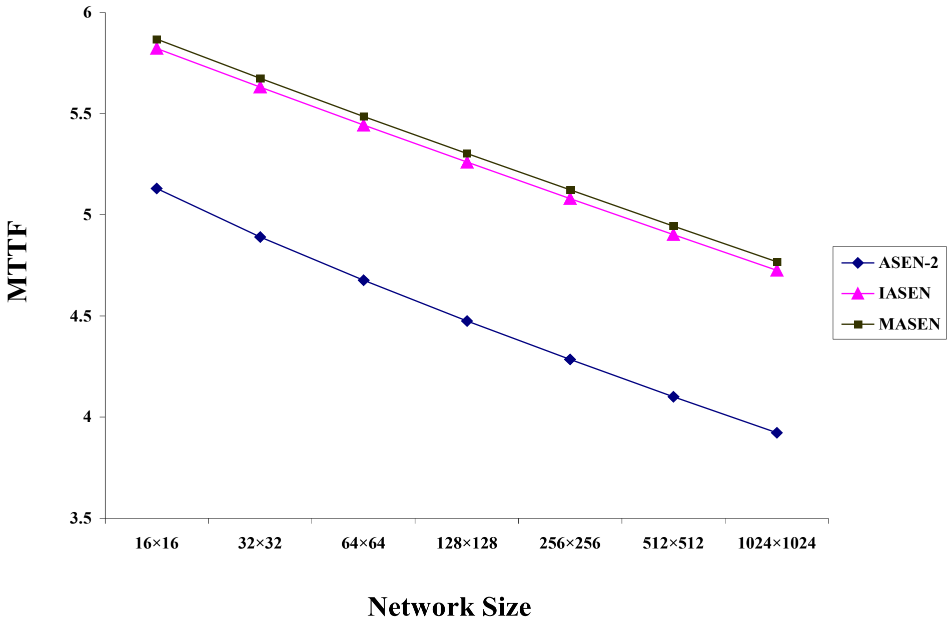


Figure 6.7: Comparative Analysis of Upper Bound MTTF

6.6.2 Comparative Analysis of Lower Bound MTTF

For the calculation of Lower bound MTTF a pessimistic approach is used (explained in chapter 5). Results are shown below in table 6.8 and in Figure. 6.8.

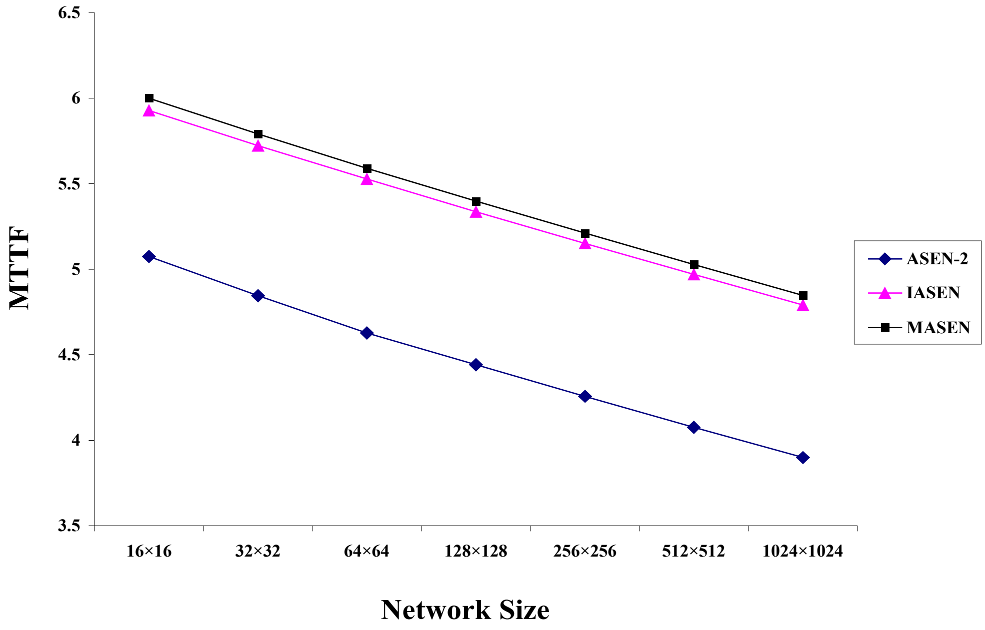


Figure 6.8: Comparative Analysis of Lower Bound MTTF

6.7 Cost

One common method is used to calculate the cost of ASEN, MASEN and IASEN networks (method is explained in section 5.7). Comparison of these networks is shown below in table 6.9 and in Figure. 6.9. Cost of MASEN and IASEN increases as the network size increases. Cost of these networks is comparable.

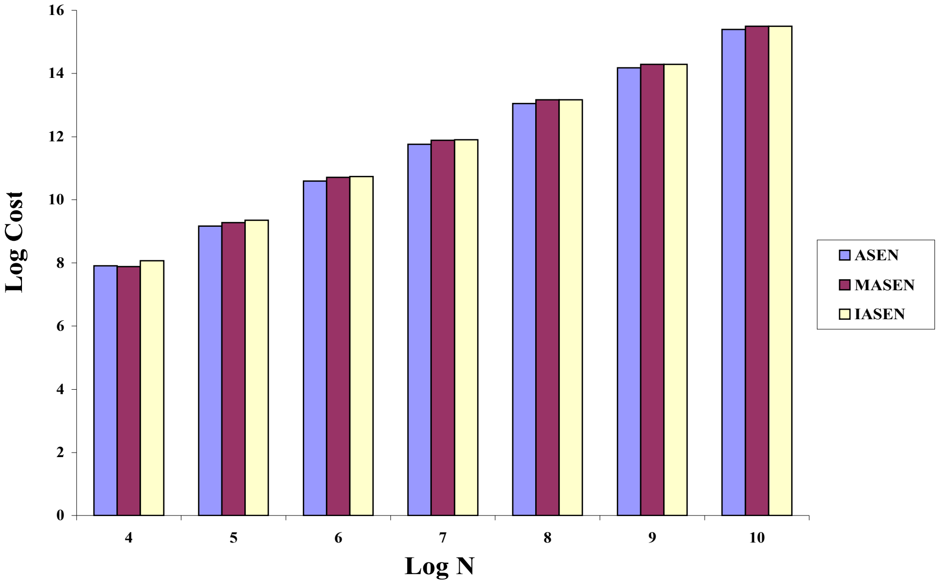


Figure 6.9: Comparative Cost of ASEN-2, MASEN, IASEN

Chapter 7

Conclusions and Future Scope

7.1 Conclusions

1. IASEN network provide more permutation passibility than MASEN and ASEN-2 network. MASEN provide permutation passibility less than IASEN but more than ASEN-2.
2. It has been found that the bandwidth of both MASEN and IASEN is same but almost double than the existing ASEN-2 network. As the performance parameters Throughput, Probability of Acceptance, Processor Utilization of a network is based on the bandwidth of the network. So accordingly the performance parameters are also improved.
3. Reliability is calculated in terms of mean time to failure (MTTF). MTTF is analyzed on the basis of the upper bound and the lower bound. The analysis shows that the reliability of MASEN network is better than IASEN and both of these network are more reliable and fault tolerant than ASEN-2.
4. There is always a compromise between cost and performance of a network. Both MASEN and IASEN have good overall performance but they are not more cost effective than ASEN-2.

7.2 Summary of Thesis

The work done in this thesis is summarized as:

Survey of existing MINs.

Two new irregular networks MASEN and IASEN are proposed, which are derived from an existing regular network ASEN-2.

Existing network ASEN-2 and two proposed networks MASEN, IASEN are analyzed for the following performance parameters:

- Permutation Passibility.
- Bandwidth.
- Throughput, Probability of Acceptance, Processor Utilization.
- Reliability.
- Cost.

7.3 Future Scope

No work is perfect done for the first time. There is always a scope for the improvement. The field of Irregular Networks can be further explored in the light of the following suggestions:

The designing of more irregular networks having better Bandwidth, Reliability and Permutation can be explored.

A comparative analysis of the multistage interconnection networks can be carried out with respect to the performance parameters.

An efficient algorithm can be defined.

Concept of optical MINs can also be used.

References

---

[1] Agrawal Dharma P., "Testing and Fault Tolerance of Multistage Interconnection Networks", IEEE Transactions on Computers, April 1982, pp. 41-53.

[2] Adams George B., Agrawal Dharma P., Siegel Howard Jay, “A Survey and Comparison of Fault-Tolerant Interconnection Networks”, IEEE Transactions on Computers, June 1987, pp. 14-27.

[3] Adams George B., Siegel Howard Jay, “The Extra Stage Cube: A Fault-Tolerant Interconnection Network for Supersystems”, IEEE Transactions on Computers, vol. C-31, no. 5, May 1982, pp. 443-454.

[4] Bansal P.K, Singh Kuldeep, Joshi R.C., “Quad Tree: A Cost-Effective Fault-Tolerant Multistage Interconnection network**”,** Proceeding of International Conference IEEE INFOCOM, 1992, pp.6D.1.1-6D.1.7.

[5] Bansal P.K, Singh Kuldeep, Joshi R.C, “On Fault tolerant Multistage Interconnection Network”, Conference on Computer Electrical Engineering, vol. 20, no.4, 1994, pp. 335-345.

[6] Bansal P.K, Singh Kuldeep, Joshi R.C, “Routing and path length algorithm for a cost-effective four-tree multistage interconnection network”, International Journal of Electronics, vol. 73, no.1, 1992, pp. 107-115.

[7] Bhogavilli Suresh K., Abu-Amara Hosame, “Design and Analysis of High Performance Multistage Interconnection Networks”, IEEE Transactions on Computers, vol. 46, no. 1, January 1997, pp. 110 -117.

[8] Bhuyan Laxmi N., Yang Qing, Aggarwal P. Dharma, “Performance of Multiprocessor Interconnection Networks”, Proceeding of IEEE, February 1989, pp. 25-37.

[9] Blaket James T., Trivedi Kishor S., “Reliabilities of Two Fault-Tolerant Interconnection Networks”, Proceeding of IEEE, 1988, pp. 300-305.

[10]Cam Hasan, “Rearrangeability of (2n−1)-Stage Shuffle-Exchange Networks”, SIAM Journal Computer Society for Industrial and Applied Mathematics vol. 32, no. 3, 2003, pp. 557–585.

[11]Chi Hsin-Chou and Wu Wen-Jen, “Routing Tree Construction for Interconnection Networks with Irregular Topologies”, Proceedings of the Eleventh Euromicro Conference on Parallel, Distributed and Network-Based Processing (Euro-PDP’03).

[12]Chuan-Lin Wu, Tse-Yun Feng, “The Reverse-Exchange Interconnection Network”, IEEE Transactions on Computers, vol. c-29, no. 9, September 1980, pp. 801-811.

[13]Fan Chenggong Charles, Bruck Jehoshua, “Tolerating Multiple Faults in Multistage Interconnection Networks with Minimal Extra Stages”, IEEE Transactions on Computers, vol. 49, no. 9, September 2000, pp 998-1004.

[14]Hwang K., Bridges F.A., “Computer architecture and parallel processing”, McGraw Hill book company, New York,1985.

[15]Kumar V.P., Reddy S.M., “Augmented Shuffle Exchange Multistage Interconnection Network”, Proceeding of International Conference IEEE, June 1987, pp. 30-40.

[16]Kuskal Clyde P., Snir Marc, “The Performance of Multistage Interconnection Networks for Multiprocessors”, IEEE Transactions on Computers, vol. c-32, no. 12, December 1983, pp. 1091-1098.

[17]Li Shuo-Yen Robert , Tan Xuesong Jonathan, “Preservation of Conditionally Nonblocking Switches Under Two-Stage Interconnection”, IEEE Transactions on Communications, vol. 55, no. 5, May 2007, pp 973-980.

[18]Malhotra Deepti, Aggarwal Rinkle, “Performance Analysis of Fault Tolerant Irregular MINs”, Proceedings of National Conference on Challenges and opportunities in Information Technologies (COIT-2007), RIMT-IET, Mandi Gobindgarh, March 23,2007, pp. 81-87.

[19]Mittal R., Cherian D., Mohan P.J., “Routing and Performance of the double tree (DOT) network” Proceeding of International Conference on Computer Digital Technology, vol. 142, no. 2, March 1995, pp. 93-97.

[20]Nitin, “On Analytic Bounds of Regular and Irregular Fault-tolerant Multistage Interconnection Networks”, Proceedings of International Conference, 2006.

[21]Sadawarti Harsh, Bansal P.K., “Fault Tolerant Irregular Augmented Shuffle Network”, Proceeding of the 2007 WSEAS International Conference on Computer Engineering and Applications, Australia, January 17-19, 2007, pp. 7-12.

[22]Sengupta J., Bansal P.K., “Performance of Regular and Irregular Dynamic MINs”, Proceeding of International Conference IEEE TENCON, 1999, pp. 427-430.

[23]Sengupta J., Bansal P.K., “Fault-Tolerant Routing in Irregular MINs”, Proceeding of International Conference IEEE TENCON, 1998, pp.638-641.

[24]Sengupta J., Bansal P.K., Gupta Ajay, " Permutation and Reliability measures of Regular and Irregular MINs", International Conference IEEE, 2000, pp. I-531-I-536.

[25]Sharma Sandeep and Bansal P.K., "A New Fault Tolerant Multistage Interconnection Network", Proceeding of International Conference IEEE TENCON, 2002, pp. 347-350.

[26]Sharma Sonia , Aggarwal Himanshu, "Performance Analysis of Modified Four tree Network", Proceedings of National Conference on Challenges & Opportunities in Information Technology (COIT-2007) RIMT-IET, Mandi Gobindgarh, March 23, 2007, pp 98-101.

[27]Ted H. Szymanski, V. Carl Hamacher, "On the Permutation Capability of Multistage Interconnection Networks", IEEE Transactions on Computers, vol. C-36, no. 7, July 1987, pp. 810-822.

[28]Wu Chuan-Lin, Feng Tse-Yun, "The Universality of the Shuffle-Exchange Network" IEEE Transactions on Computers, vol. C-30, no. 5, May 1981, pp. 324-332.

[29]Wu Chuan-Lin, Feng Tse-Yun, "On a Class of Multistage Interconnection Networks", IEEE Transactions on Computers, vol. C-29, no. 8, August 1980, pp. 694-702.

Appendix

Pseudo Code for the permutation Passibility

Global Parameters:

(i) An array garph[][] to store the network connections.

(ii) Two variable n, n_f for storing number of source destination pairs and number of faults.
(iii) An array s_d_pair[] for storing source and destination values.
(iv) An array faulty_nodes[] for storing the faulty nodes.
(v) An array dest_a[] to stores the binary of the destination.
(vi) An array path[][] for storing the nodes traversed in a path.
(vii) An array visit[][] that keeps track of all visited nodes.

Function Name: main()

Called by: operating system

Calling Functions: (i) enter_faults()
(ii) find_path()
(iii) display()

Parameters Passed: No

Purpose: This function includes all the initializations

Function Body:

(i) Initialize the graph matrix representing the value 1 if there is a path from a given source to the given destination otherwise, Initialize it to the infinite value (depicting no direct path exist between corresponding source and destination pair).
(ii) Get the number of source and destination pairs.
(iii) Input the source and destination pair values.
(iv) Input the number of faulty nodes
(v) The enter_faults() function is called for getting the faulty nodes
(vi) Next find_path() function is called to find out the path between each source and destination pairs.
(vii) Finally, display function is called to display the output.

Function Name: enter_faults ()

Called by: main function

Calling Functions: No

Parameters Passed: Number of faulty nodes

Purpose: To get the number of faulty nodes

Function Body:

- (i) Get the faulty nodes
- (ii) Return to main () function.

Function Name: find_path()

Called by: main function

Calling Functions:

- (i) dec_bin()
- (ii) find_mux ()
- (iii) find_mux1()
- (iv) find_mux2()
- (v) mux_to_stage1()
- (vi) stage_1()
- (vii) stage_2()
- (viii) stage_3()

Parameters Passed: No

Purpose: Finding the path between all source and destination pairs.

Function Body:

- (i) Firstly convert the destination value in binary by calling a function dec_bin().
- (ii) Then call a function find_mux(). That will find out the MUX used in the path.
- (iii) If the MUX found by the find_mux() function is busy then call find_mux1() to find out the next alternative.
- (iv) If the MUX found by the find_mux() function is also busy then call find_mux2() to find out the next alternative.
- (v) Then call mux_to_stage1() function that will find out the switch after the MUX.
- (vi) Then stage_1 (),stage_2() and stage_3() are called for finding the next switches in the path.
- (vii) Return to the main().

Function Name: dec_bin()

Called by: find_path()

Calling Functions: No

Parameters Passed: Value of destination

Purpose: For decimal to binary conversion

Function Body:

(i) Apply logic for the decimal to binary conversion.
(ii) Return the binary value to the function find_path().

Function Name: find_mux()

Called by: find_path()

Calling Functions: chek_clash()

Parameters Passed: Value of source and destinations

Purpose: To find out the MUX in the path

Function Body:

(i) Find out the appropriate MUX
(ii) Then call the function chek_clash() for finding the MUX is already in use or it is free.
(iii) Return the MUX value to the calling function i.e. find_path().

Function Name: find_mux1()

Called by: find_path()

Calling Functions: chek_clash()

Parameters Passed: source value.

Purpose: To find out the next MUX if the previous MUX find out by find_mux1() was busy.

Function Body:

(i) Find out the alternate MUX
(ii) Then call the function chek_clash() for finding the MUX is already in use or it is free.
(iii) Return the MUX value to the calling function i.e. find_path().

Function Name: find_mux2()

Called by: find_path()

Calling Functions: chek_clash()

Parameters Passed: source value.

Purpose: To find out the next alternate MUX if the previous MUX find out by find_mux1() was busy.

Function Body:

(i) Find out the next alternate MUX

(ii) Then call the function chek_clash() for finding the MUX is already in use or it is free.

(iii) Return the MUX value to the calling function i.e. find_path().

Function Name: mux_to_stage1()

Called By: find_path()

Calling Functions: chek_clash()

Parameters Passed: The MUX value that is used as new source.

Purpose: Find out the next switch in the path

Function Body:

(i) Find out next switch connected with MUX.

(ii) Then call the function chek_clash() for finding the switch is already in use or it is free.

(iii) Return the switch value to the function find_path().

Function Name: stage_1()

Called By: find_path()

Calling Functions: chek_clash()

Parameters Passed: The next switch value.

Purpose: Find out the next switch in the path

Function Body:

(i) Find out next switch

(ii) Then call the function chek_clash() for finding the switch is already in use or it is free.

(iii) Return the switch value to the function find_path().

Function Name: stage_2()

Called By: find_path()

Calling Functions: chek_clash()

Parameters Passed: The next switch value.

Purpose: Find out the next switch in the path

Function Body:

(i) Find out next switch
(ii) Then call the function chek_clash() for finding the switch is already in use or it is free.
(iii) Return the switch value to the function find_path().

Function Name: stage_3()

Called By: find_path()

Calling Functions: chek_clash()

Parameters Passed: The next switch value.

Purpose: Find out the next switch in the path

Function Body:

(i) Find out next switch
(ii) Then call the function chek_clash() for finding the switch is already in use or it is free.
(iii) Return the switch value to the function find_path().

Function Name: chek_clash()

Called By:

(i) find_mux()
(ii) find_mux1()
(iii) find_mux2()
(iv) mux_to_stage1()
(v) stage_1()
(vi) stage_2()
(vii) stage_3()

Calling Functions: No

Parameters Passed: The switch or MUX value sent by the calling functions

Purpose: Find out the next switch in the path is free or used by another request or faulty.

Function Body:

(i) Find out the switch or MUX is used by another request or free to use.

(ii) If free than it set visit is equal to 1

(iii) If not free or faulty then update the value and find out the alternate path if available.

(iv) Return to the calling function.

Function Name: display()

Called By: main function ()

Calling Functions: fprint ()

Parameters Passed: No

Purpose: To display the output in the desired format and get the path length of each source, destination pair.

Function Body:

(i) Check the clash value if it is –1 then that request is not successfully matured and after displaying the path to that switch , print clash.

(ii) Else call the function fprint() and print the evaluated path.

Function Name: fprint()

Called by: display ().

Calling Functions: No

Parameters Passed: value of the next node in the path.

Purpose: To display the switch number according to manipulated value.

Function Body:

(i) Display the values.

(ii) Return to the function display ().

Pseudo Code for Calculating Bandwidth, Probability of Acceptance, Throughput and Processor Utilization

Function Name: **main()**

Called by: operating system

Calling Functions: No

Parameters Passed: No

Purpose: This function calculates the bandwidth, probability of acceptance, throughput and processor utilization.

Function Body:

(i) Evaluate the bandwidth of ASEN-2 network by applying the bandwidth equations.

(ii) Then calculate probability of acceptance, throughput and processor utilization using the calculated value of bandwidth.

(iii) Display the values of all these parameters.

Pseudo Code for cost Evaluation

Function Name: main()

Called by: operating system

Calling Functions: (i) asen()
(ii) masen()
(iii) iasen()

Parameters Passed: No

Purpose: This function gets the choice and calls the appropriate function.

Function Body:

(i) Get the choice of network whose cost has to calculate.

(ii) According to the choice call the appropriate function either asen() or masen() or iasen() or all three.

Function Name: **asen()**

Called by: main()

Calling Functions: No

Parameters Passed: No

Purpose: This function calculate the cost of ASEN-2 network

Function Body:

(i) Evaluate the cost of the network according to the formula.

(ii) Finally, display the evaluated cost.

Pseudo Code for Reliability

Function Name: main()

Called by: operating system

Calling Functions: (i) upper()
(ii) lower()

Parameters Passed: No

Purpose: This function gets the choice and calls the appropriate function.

Function Body: (i) Get the choice for upper bound or lower bound or both.
(ii) Call the appropriate function.

Function Name: **upper()**

Called by: main()

Calling Functions: No

Parameters Passed: No

Purpose: This function calculate the upper bound of IASEN network

Function Body:

(i) Evaluate the upper bound according to the reliability equations.
(ii) Apply trapezoidal rule for the calculation of upper bound MTTF.
(iii) Finally, display the evaluated MTTF.

Function Name: **lower()**

Called by: main()

Calling Functions: No

Parameters Passed: No

Purpose: This function calculate the lower bound of IASEN network

Function Body:

(i) Evaluate the lower bound according to the reliability equations.
(ii) Apply trapezoidal rule for the calculation of lower bound MTTF.
(iii) Finally, display the evaluated MTTF.

List of Publications

---

[1] Sheetal and Rinkle Aggarwal, **“Comparative study of Fault Tolerant Multistage Interconnection Networks”**, National Conference on Advancements in Computer

Engineering (ACE-08), Fatehgarh Sahib (Punjab), India, April3-4 2008, pp.93-98 **[Presented and Published].**