On Fault Tolerant Irregular 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:

Manpreet Kaur
80632012

Under the supervision of:
Rinkle Aggarwal
Lecturer (SS)

MAY 2008

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA – 147004

CHAPTER 1 Introduction

1.1 Parallel Networks

With device characteristics approaching physical limits, parallel or distributed processing has been widely advocated as a promising approach for building high performance computing systems. The continued research in these areas arises from two factors: (a) the technological development in the area of VLSI chips and (b) the observation that significant exploitable software parallelism is inherent in many scientific and engineering applications [14]. To exploit this parallelism efficiently, a parallel system must be designed to considerably reduce the communication overhead between the processors. The communication architecture of the system might support one application well but might prove inefficient for others.

1.1.1 A classification of parallel systems

General-purpose parallel computer systems are divided into two categories:

1.1.1.1 Multicomputers

In a multicomputer, each processor has its own memory space, and communication between the processors occurs at a higher level as with a complete file or data set. A processor cannot directly access another processor's local memory as shown in Figure 1.1

1.1.1.2 Multiprocessors

A multiprocessor must permit all processors to share the main memory. All the processors address a common main memory space as shown in Figure 1.2

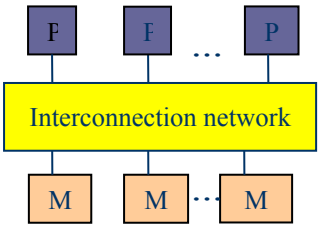


Figure 1.1: Multicomputer system

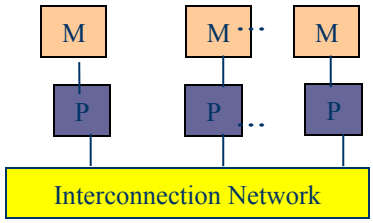
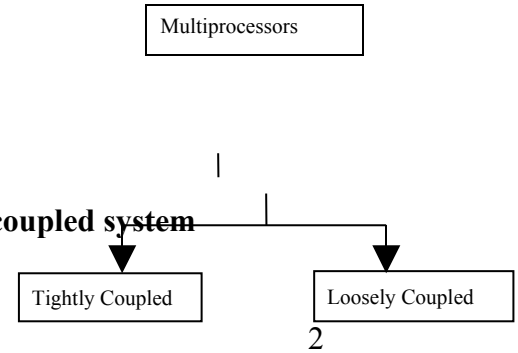


Figure 1.2: Multiprocessor system

Multiprocessors can be further divided as tightly coupled and loosely coupled.



1.1.1.2.1 Tightly coupled system

In a tightly coupled system, the main memory is situated at a central location so that the access time from any processor to the memory is the same. In addition to this central memory (also called main memory, shared memory, global memory, etc.), each processor might consist of some local memory or cache as shown in Figure 1.3.

1.1.1.2.2 Loosely coupled system

Systems, with small number of large, independent CPUs that have low speed connections between the CPUs are called loosely coupled. In these systems, the processors do not share memory but each processor has its own memory. In these systems, all the communication between processors is done by passing messages across the network as shown in Figure 1.4. If a processor wants to access data from another processor's memory, it must send a message through the communication subsystem, requesting the other processor to send these data. Partitioning and allocation of program and data into segments play crucial role in overall performance of an application in case of loosely coupled systems.

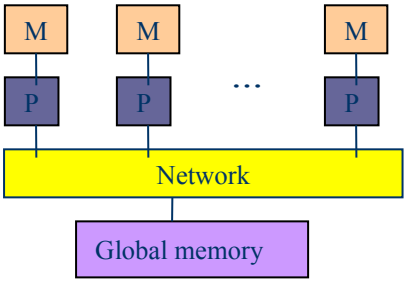


Figure 1.3: Tightly Coupled System

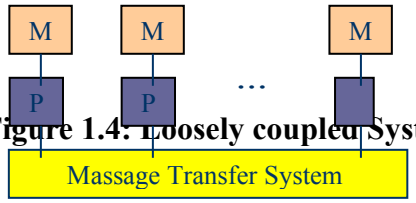


Figure 1.4: Loosely coupled System

As mentioned previously, the memory in a multicomputer is not shared. The interaction between the processors relies on message passing between the source and destination processors (nodes). The message passes over a link that directly connects two nodes and might have to pass through several such nodes in a store-and forward manner before it reaches its destination. Therefore, each interaction involves a lot of communication overhead, and only those applications that need less interprocessor communication are well suited to multicomputers. The multicomputers are usually based on topologies such as ring, tree, star, hypercube, etc.

1.2 Interconnection Networks

Interconnection Networks play a major role in the performance of modem parallel computers. Many aspects of Interconnection Networks, such as implementation complexity, routing algorithms, performance evaluation, and fault tolerance, have been the subject of research over the years [30]. There are many factors that may affect the choice of appropriate interconnection network for the underlying parallel computing environment. 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 [9]. It is the path, the data must travel in order to access memory in a shared memory computer or to communicate with another processor in a distributed memory environment.

1.2.1 A classification of Interconnection networks

1.2.1.1 Switching methodology: There are basically two major switching methodologies:

1.2.1.1.1 Packet switching

In packet switching, a message is broken into small packets transmitted through the network in a "store-and-forward" mode. Thus, a packet experiences a random delay at each switching point, depending on the traffic in the network along its path to the destination.

1.2.1.1.2 Circuit switching

Circuit switching actually establishes a physical path between a source and a destination. A time delay is needed when the path is being established. Once the path is completed, it is held for the entire data transmission. In general, circuit switching is much more suitable for long messages, and packet switching is more efficient for short messages.

1.2.1.2 Control strategy

Control strategy mainly concerns the way control signals direct the dataflow generated in a network. In a centralized control scheme, all the control signals come from a single source [11]. Obviously, the central controller creates a system bottleneck and directly affects the performance and reliability of the entire system. The design of this central controller must be very complex to retain good system performance. These drawbacks can be avoided through the use of distributed control

strategies in which a small controller is associated with each component of the system.

1.2.1.3 Operational modes

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

1.2.1.4 Network topology

A network can be represented by a graph in which nodes indicate switches and edges represent communication links. Interconnection networks are made up of switching elements and topology is the pattern in which the individual switches are interconnected to other elements such as processors, memories and other switching elements. Direct topologies connect each switch directly to a node, while in indirect topologies at least some of the switches connect to other switches [1]. The topologies can be categorized into two groups, static and dynamic as shown in Figure 1.5.

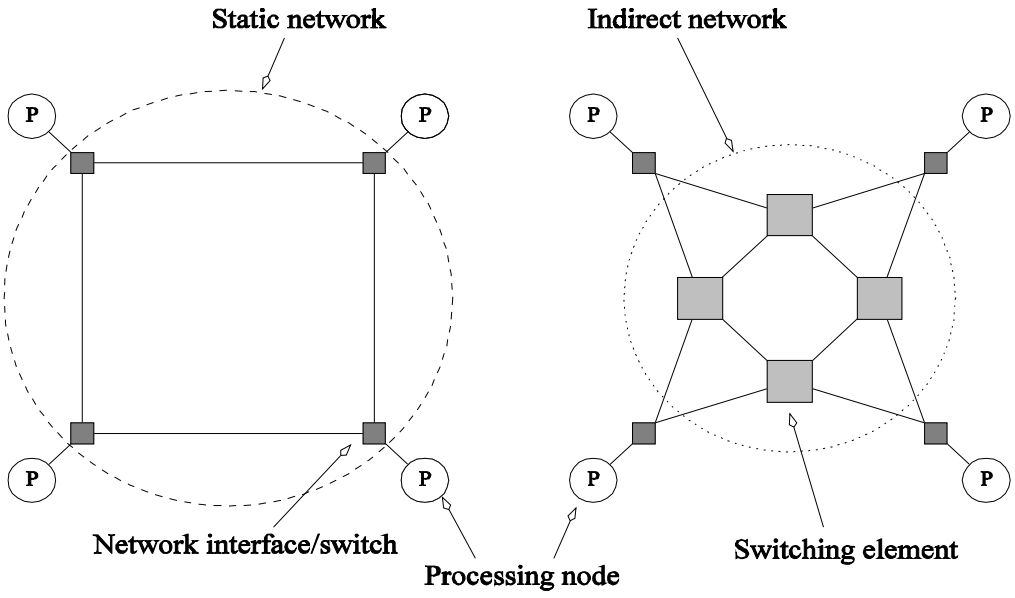


Figure 1.5: Static and Dynamic Network

1.2.1.4.1 Static network

In static topology, links between two processors are with passive and dedicated buses cannot be reconfigured for direct connection with other processors. Static networks that are generally used in message-passing architectures [12]. The following network topologies are commonly used.

1.2.1.4.1.1 Ring Network

The type of network topology in which each of the nodes of the network is connected to two other nodes in the network and with the first and last nodes being connected to each other, forming a ring as shown in Figure 1.6. All data that is transmitted between nodes in the network travels from one node to the next node in a circular manner and the data generally flows in a single direction only.

1.2.1.4.1.2 Star Network

A star topology connects all nodes to a central point of concentration as shown in Figure 1.7. This point is usually a hub or switch. Nodes communicate across the network by passing data through the hub. The main disadvantage of this kind of topology is that if central hub stops working then there will be no transmission at any node.

1.2.1.4.1.3 Fully connected Network

The type of network topology in which each node is connected to every other nodes of the network with a point-to-point link as shown in Figure 1.8. This makes it possible for data to be simultaneously transmitted from any single node to all of the other nodes.

1.2.1.4.1.4 Tree Network

The type of network topology in which a central 'root' node (the top level of the hierarchy) is connected to one or more other nodes that are one level lower in the hierarchy (i.e., the second level) with a point-to-point link between each of the second level nodes and the top level central 'root' node, while each of the second level nodes that are connected to the top level central 'root' node will also have one or more other nodes that are one level lower in the hierarchy (i.e., the third level) connected to it as shown in Figure 1.9.

1.2.1.4.1.5 Mesh Network

A mesh simply connects one processor to four others, as shown in Figure 1.10. Consider a chessboard with a processor in each square. A connection would be made from each square to four its neighbors.

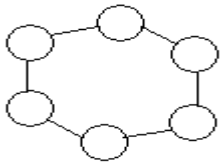


Figure 1.6: Ring

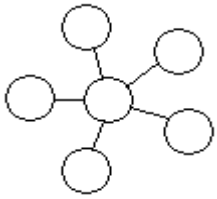


Figure 1.7: Star

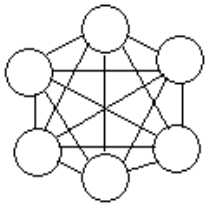


Figure 1.8: Fully connected

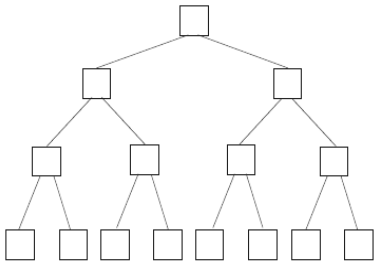


Figure 1.9: Tree Network

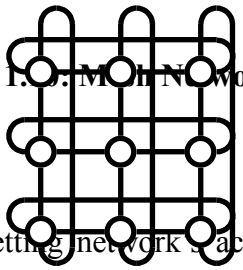


Figure 1.10: Mesh Network

1.2.1.4.2 Dynamic Network

Links in dynamic topology can be reconfigured by setting network active switching elements. Dynamic interconnection networks implement one of the following main alternatives:

1.2.1.4.2.1 Crossbar Network

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 as shown in Figure 1.11.

1.2.1.4.2.2 Bus Network

In a bus network, processors share a single communication channel, called bus shown in Figure 1.12. 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.

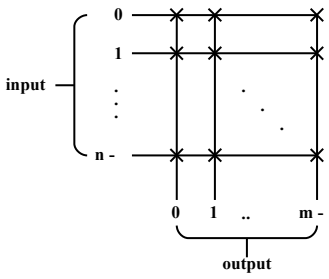


Figure 1.11: Crossbar Network

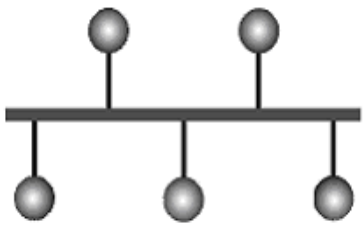


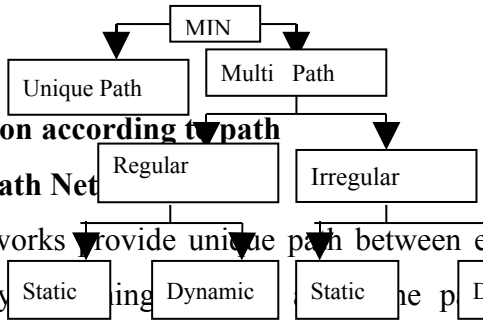
Figure 1.12: Bus Network

1.3 Multistage interconnection networks

As stated previously, the cost of a crossbar network is too high to be practical for building large multiprocessor systems. As an alternative to the crossbar network, multistage interconnection networks (MINs) have assumed importance in recent times. The main advantage of these networks is their cost-effectiveness. They allow a rich subset of one to one and simultaneous mappings of processors to memory modules, while reducing the hardware cost to O(N1ogN) in contrast to O(N2) for crossbar networks. Multistage interconnection networks (MINs) consist of more than one stages of small interconnection elements called switching elements and links interconnecting them. A multistage interconnection network is actually a compromise between crossbar and shared bus networks [29].

1.3.1 Classification of Multistage interconnection networks

Multistage interconnection networks can be classified according to different categories. The main classification categories are according to path, switches and control.



1.3.1.1 Classification according to path

1.3.1.1.1 Unique path Net

These networks provide unique path between every source and destination. The failure of any switching element in the path disconnects some source-destination pairs, so adversely affecting the capabilities of existing network [8]. These are not reliable for a large multiprocessor system, as they cannot tolerate even a single fault.

1.3.1.1.2 Multi path Network

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.

1.3.1.2 Classification according to switches

1.3.1.2.1 Regular Network

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

1.3.1.2.2 Irregular Network

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.

1.3.1.3 Other Classification

1.3.1.3.1 Blocking Network

In blocking network, simultaneous connections of more than one terminal may result in conflict in use of network communication links. For example: Omega network as shown in Figure 1.13.

1.3.1.3.2 Non blocking Network

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 [18]. In other words, networks that can handle all possible connections without blocking is called non-blocking network shown in Figure 1.14.

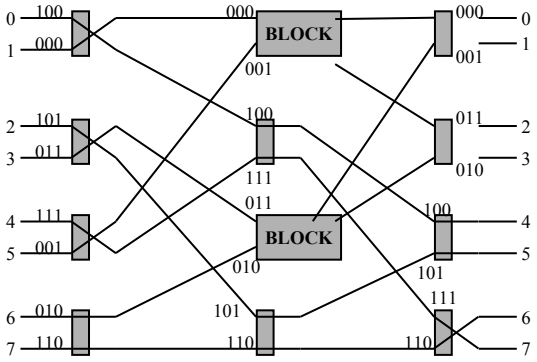


Figure 1.13: Blocking Network

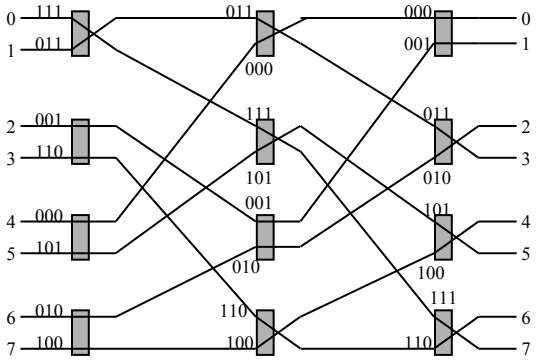


Figure 1.14: Non-Blocking Network

1.3.2 Types of connections in MINs

There are four types of connections, which are commonly used, in multistage interconnection networks. These are:

One to one connection*:* A one to one connection passes information from a source to destination. The exact route taken by the information is determined by path itself.

Multi path connection*:* Multi path means many one to one connections are active simultaneously.

Permutation connection*:* A set of one to one connections such that no two connections have the same source or destination. Such connections are meaningful only in cases of equal number of sources and destinations.

Broadcast connection*:* Information flows from source to various destinations either some or all. Thus a number of destinations simultaneously receive the information.

1. 3.3 Routing in MINs

No decision regarding the routing in a network is perfect one. To acquire nearly complete knowledge for routing would require so much overhead that traffic throughout would be simultaneously reduced. For example, if there is minimal traffic, the network path with minimum number of links will normally be the best. If a node or switching element fails, then the path with minimum number of links that bypasses the failures will be the best [24]. As traffic builds up, however this simple routing strategy can give poor result at times because the shortest path may happen to be congested. So the network as a whole should employ a routing strategy that would bypass areas of congestion.

1.3.4 Routing tags

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 integer expressed by the destination. Each successive digit in this integer encodes the settings for the switch in the next stage along a desired path [24]. This control is called distributed if the devices, using the network switches can be set on their own based on the tag information.

1.4 Organization of thesis

A survey of existing regular and irregular multistage interconnection networks is covered in chapter 2. Chapter 3 deals with problem definition and Chapter 4 covers the construction and routing of one existing MIN (FT) and two proposed MINS (FT-1 and FT-2). Chapter 5 explains the performance analysis of three networks (FT, FT-1 and FT-2). Chapter 6 shows the experimental results. The last Chapter is regarding the Conclusions and future scope of the work.

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 [2]. 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.

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. The unit cube concept can be extended to an n-dimensional unit space, called n-cube, with n bits per vertex.

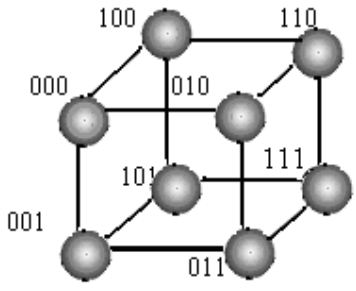


Figure 2.1: A three dimensional binary cube

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 Figure 2.2.

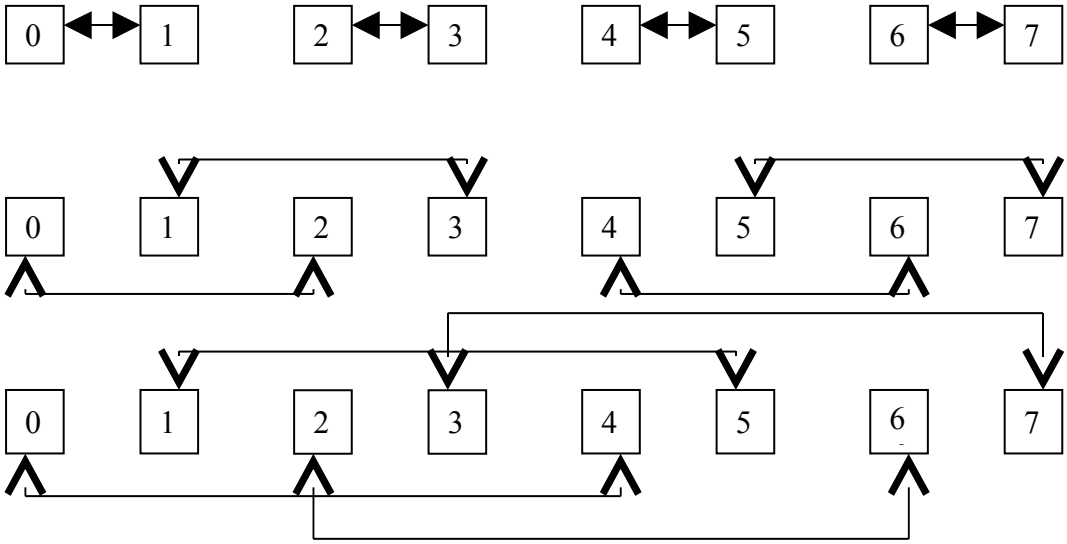


Figure 2.2: The Recirculating Cube Network for N = 8

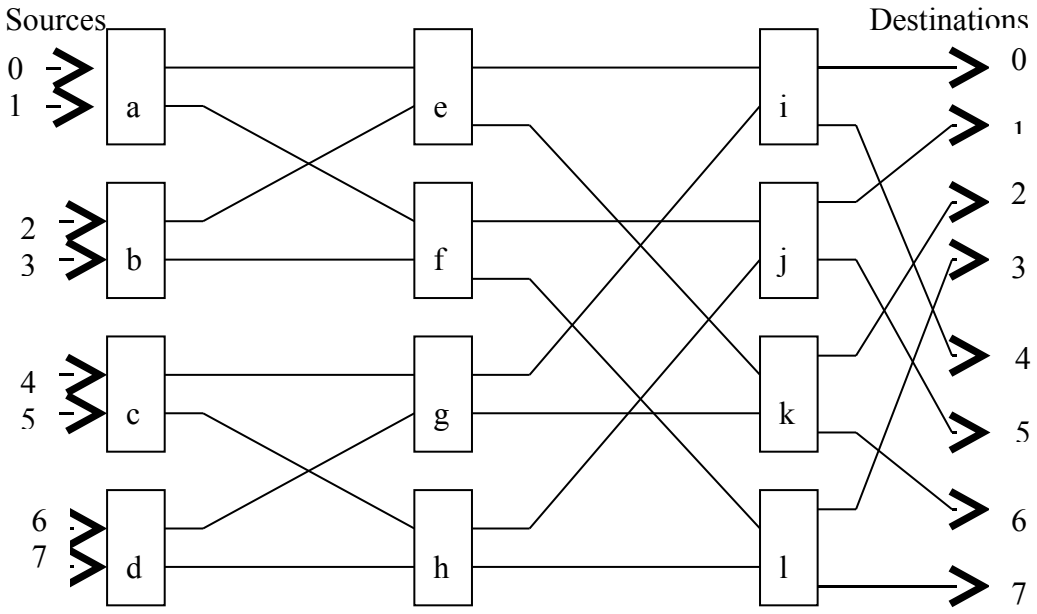


Figure 2.3: A Multistage Cube Network for N =8

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 functions switch boxes i.e. straight and exchange is used in constructing 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.

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 [10]. 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. Figure 2.5 represent a single stage recirculating shuffle exchange network, where solid lines indicate exchange and dashed lines indicate shuffle [22, 28].

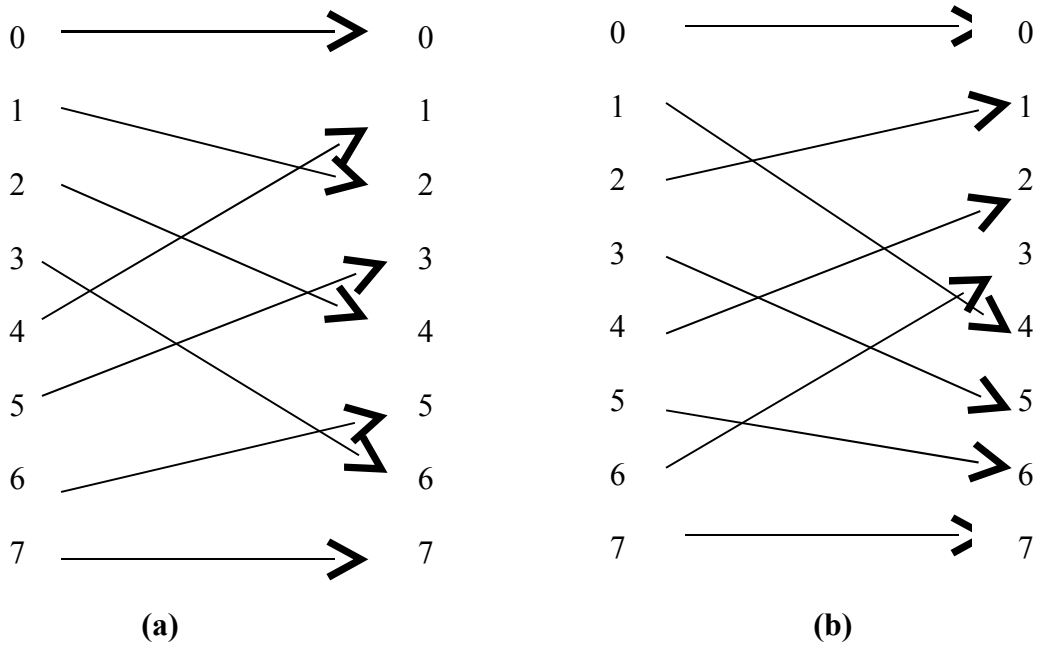


Figure 2.4 : (a) A perfect shuffle and (b) The inverse perfect shuffle

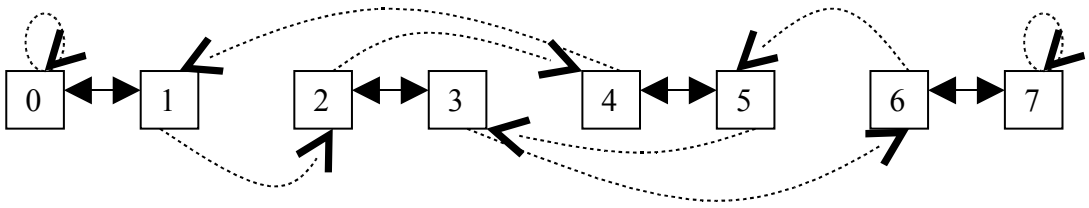


Figure 2.5: A shuffle exchange recirculating network for N = 8

The shuffle exchange has been implemented with multistage Omega network [17]. Figure 2.6 represents Omega network for N=8. An N x 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. Each box has four functions, straight, exchange, upper broadcast and lower broadcast.

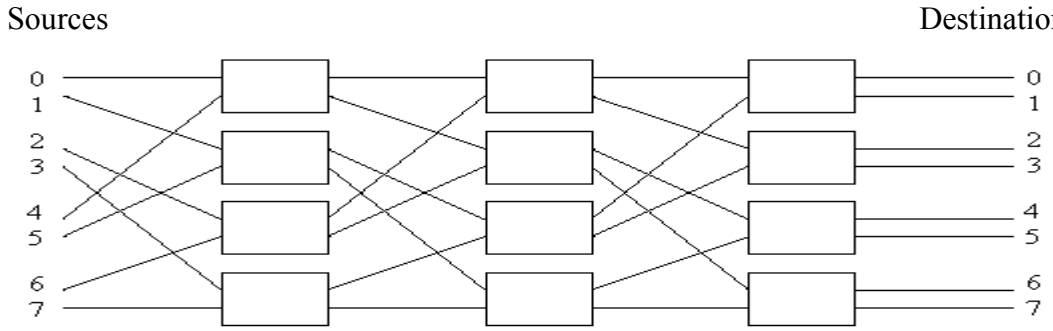


Figure 2.6: An Omega network for N = 8

2.4 Double Tree Network (DOT)

The double tree (DOT) network was originally proposed as a fault-detecting and correcting network. This network has many redundant paths of different lengths between an input-output terminal pair. It compromises of a right and a left half. Each half of the network resembles a binary tree, and is a mirror image of each other [20]. The structure of $2^m \times 2^m$ DOT network can be defined recursively as the $2^1 \times 2^1$ DOT network. An 8 × 8 DOT network is shown in Figure 2.7.

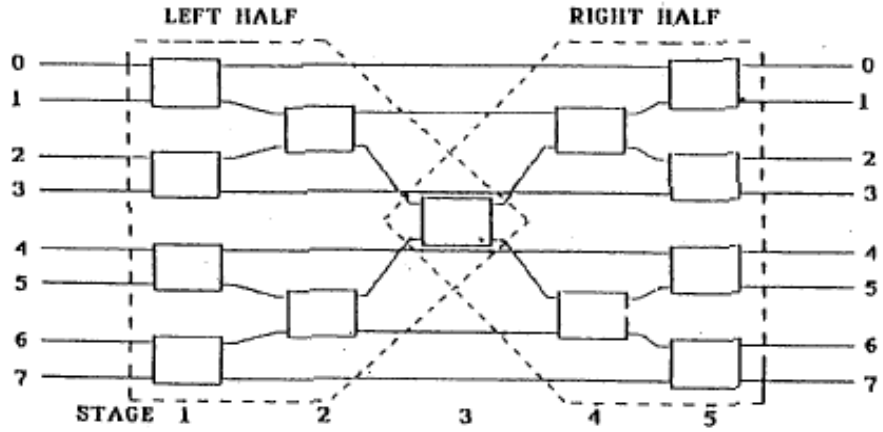


Figure 2.7: Double tree network for N=8

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 is 2n-1, where n = $\log_2 N$ and the number of switching elements are $2^{n+1}-3$ [19, 20]. The numbers of switching elements at the last and first stages are equal. Similarly, numbers of switching elements in the next stages are equal.

2.6 FDOT Network

An FDOT-k network of size NXN is designed by dividing N inputs and N outputs into k disjoint partition of N/k sources and N/k destinations where k (>=2) and N (>k) are the powers of 2.There are k independent subnetworks, and an extra one, such that a connection path between each source-destination pair can be established via any one of the subnetwork. All the (k+1) subnetworks are of identical type [19]. The extra subnetwork helps to enhance fault-tolerant capability and to keep a desired level of performance even in the presence of faults. Each source and destination are linked to all the (k+1) subnetworks via kx1 multiplexers and 1x k demultiplexers. An FDOT-k consists of (2n-1) number of stages and (k+1) (2n+1-3) number of switching elements, where n = log2 N/k. FDOT network of size 8x8 is shown in Figure 2.9.

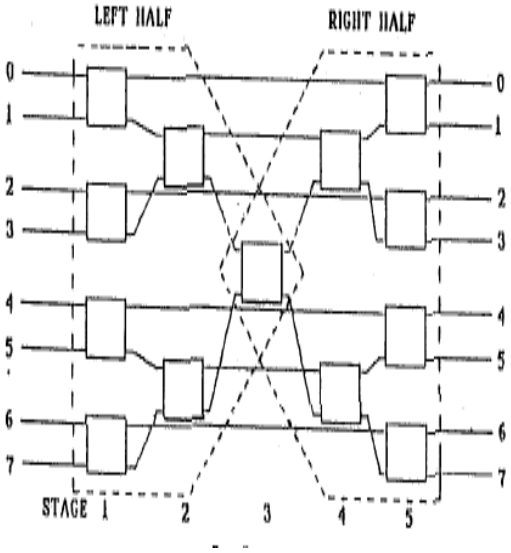


Figure 2.8: Modified double tree network for N=8

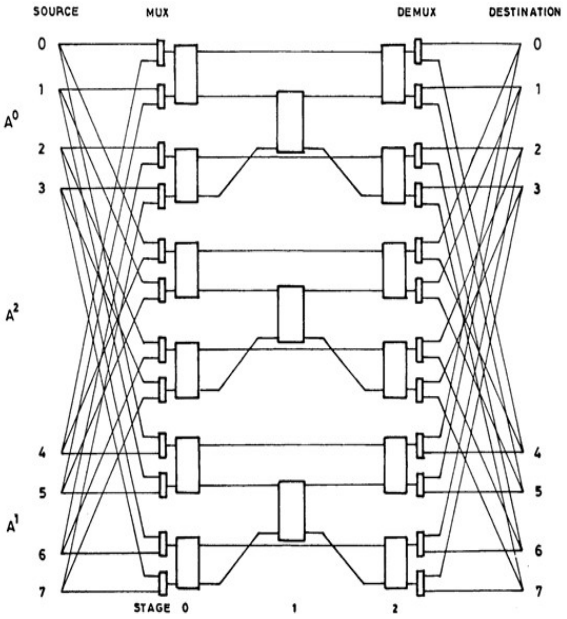


Fig 2.9: Fault Tolerant double tree network for N=8

CHAPTER 3 Problem Statement

3.1 Gaps between Existing and Targeted Work

Multistage interconnection networks play an important role in the parallel computing systems. A Multistage interconnection network consists of more than one stage of interconnection elements and links. Reliability and permutation are two major design parameters of a network, which needs to be maximized at a reasonable cost. A lot of work has already been done in the design and analysis of regular and irregular networks. Since irregular networks, in general, are less costly and multipath in nature as compared to regular multistage interconnection networks. So, analysis of irregular multistage interconnection network is important. A study of different irregular multistage interconnection networks is done and some modifications in the design of existing irregular MINs has been proposed. Irregular FT network is a single switch fault-tolerant if both switches in a loop are simultaneously faulty then the whole network breaks down so modification is required in FT network.

3.2 Problem Statement

Objective of the research was to design new MINs which perform better than the existing MINs. So, two new networks are proposed as FT-1 and FT-2 which are derived from FT network with improvement in fault tolerance capability.

Objectives of the Research work can be summarized as follows:

Designing

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

Construction and Routing

To explain the construction and routing of the proposed networks.

Reliability

The system used for the interconnection of various components should be reliable in nature. Thus, the reliability analysis for FT-1 and FT-2 irregular MINs in terms of mean time to failure (MTTF) is determined and analyzed.

Permutation Passibility

Besides, design and reliability considerations there is other important parameter i.e. permutation passibility. Permutation Passibility is an important parameter to study the behavior of any multistage interconnection networks. Permutation Passibility means how many input requests are simultaneously able to pass through a given network and reach successfully at the intended destination. Some of the requests will pass through the most favorable path, others have to be routed through an alternative paths are available or paths are busy in serving some other requests.

Cost

The system should also be cost effective in nature. The major advantage of irregular MINs is that they provide reasonably good performance at a considerably low cost.

Comparison

Compare the proposed networks with existing network.

CHAPTER 4 Construction and Routing Scheme

4.1 Construction of FT Network

Four Tree network is a dynamic irregular MIN which provides multiple paths of varying lengths between a Source-Destination pair. Four Tree network of size NxN is constructed with two identical groups G^i (i=0, 1), each consisting of MDOT network of size N/2 x 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 the others with MSB 1 falling into G^1. A FT network of size 2^n x 2^n consists of (2m-1) stages and (2^{m+2} – 6) switches, out of which 2^{n-1} are of size 2 x 2 and the rest are of size 3 x 3 (where m = $\log_2 N/2$, N = 2^n). There are 2^n number of 2 x 1 multiplexers and an equal number of 1 x 2 demultiplexers [4]. Both stage i and stage (2m-i) have 2^{n-i} switches [19] where i=1, 2, 3, ---, N-1. The switches in a stage having the same number in both the groups form a loop [4]. Such loops of switches are formed in all the stages except the last. Each source and destination is connected to both the groups with the help of multiplexers and demultiplexers as shown in Figure 4.1.

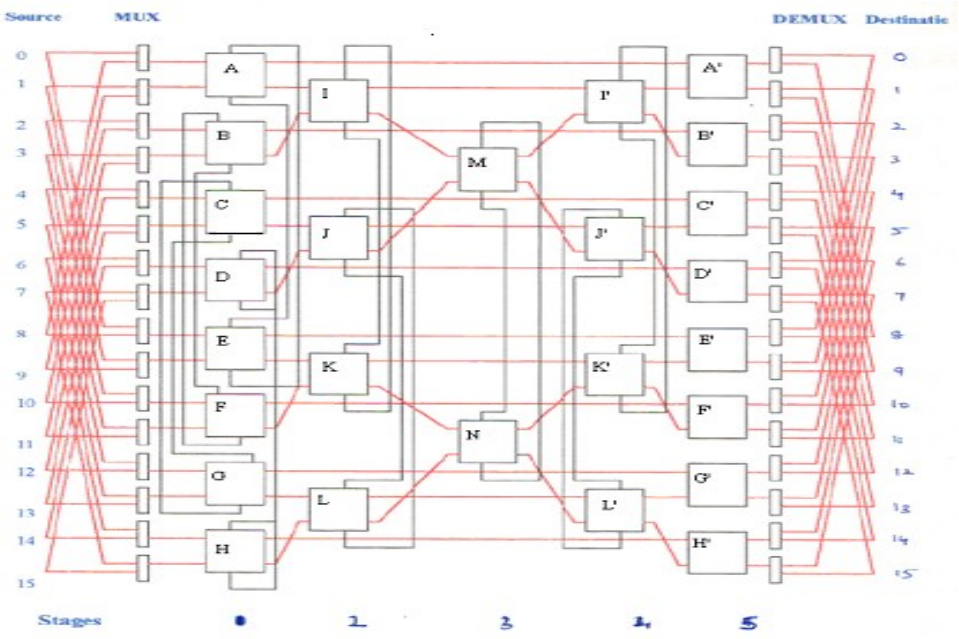


Figure 4.1: Four Tree Network (FT) of 16 x 16

4.1.1 Redundancy Graph: A redundancy graph offers a convenient way to study the properties of a multi-path MIN, such as the number of faults tolerated or the type of

rerouting possible. A redundancy graph depicts all the available paths between a source and a destination in a MIN. It consists of two distinguished nodes-the source S and the destination D-and the rest of the nodes correspond to the switches that lie along the paths between S and D as shown in Figure 4.2.

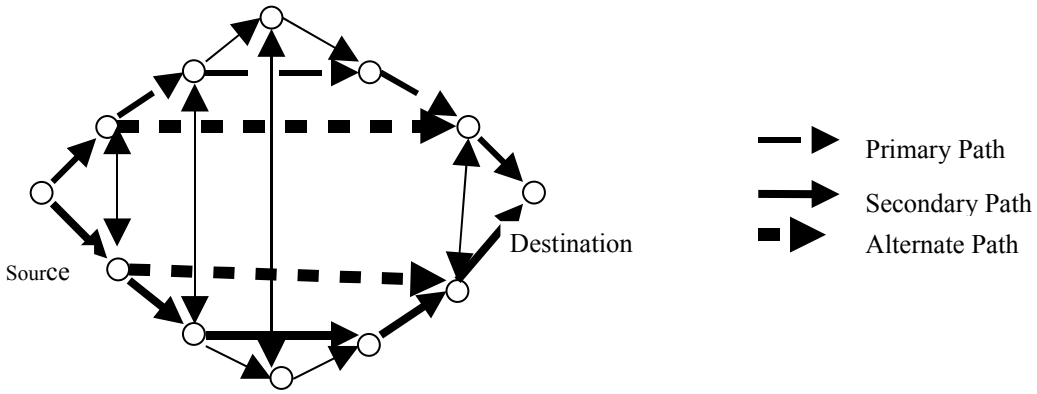


Figure 4.2: Redundancy Graph for FT Network

4.1.2 Routing Scheme

4.1.2.1 Path Length algorithm for FT

The possible path lengths between a particular source destination pair varies from 2 to (2n- 1) for an NxN network, depending upon the addresses of the source and destination terminals.

The Path Length algorithm is:

If

$[(S_{n-2} \oplus D_{n-2}) + (S_{n-3} \oplus D_{n-3}) + \ldots\ldots\ldots\ldots + (S_1 \oplus D_1)]$

Is zero

($\oplus$ Represents an exclusive-OR and + represents a logical OR operator)

Then

Minimum path length is 2 and all paths of different lengths are possible

i.e. Paths of length 2, 4, 6… (2_{m-2}), (2_{m-1}).

Else

If

$[(S_{n-2} \oplus D_{n-2}) + (S_{n-3} \oplus D_{n-3}) + \ldots\ldots\ldots\ldots + (S_2 \oplus D_2)]$

Is zero

Then

All paths of length equal to or greater than 4 are possible

Else

.

If

$[(S_{n-2} \oplus D_{n-2}) + (S_{n-3} \oplus D_{n-3}) + \ldots\ldots\ldots\ldots + (Sj \oplus D_j)]$

Is zero {where $1 \leq j \leq (n-2)$}

Then

All paths of length equal to or greater than 2j are possible.

Else

Path of length 2_{m-1} (i.e longest path) is possible only.

4.1.2.2 Routing Tag Algorithm

If

$2 \leq x \leq (2_{m-1})$

Then

Routing tag= $S_{n-1}.\ (1.1\ldots\ldots\ldots1)_{(\lfloor x/2 \rfloor -1)}.0.(d_{(\lfloor x/2 \rfloor -1)}\ldots d_0).d_{n-1}$

(Where x is the path length which varies in steps of 2)

Else

If

$x = (2_{m-1})$

Then

Routing tag= $S_{n-1}.(1.1\ldots\ldots\ldots1)_{(\lfloor x/2 \rfloor)}.0.(d_{(\lfloor x/2 \rfloor)}\ldots d_0).\ d_{n-1}$

Else

No tag is possible.

4.1.2.3 Routing Procedure

This algorithm assumes that sources and switches have the ability to detect faults in the switches to which they are connected. For any source-destination pair, find the minimum possible path length and then the corresponding routing tag.

The request is routed as follows:

Submit the request for connection to the primary path. If the request is routed through the same group to which the source belongs (i.e MSB of the source = group

number), then the path is called the primary path otherwise the path is secondary. If the primary path is faulty (i.e multiplexer or the switch or both are faulty) then submit the request to the secondary path as shown in Figure 4.3. If the secondary path is also faulty, then drop the request [4]. The most significant bit (MSB) of the routing tag, S_{n-1} will route the request through the multiplexer. For each switch in stage i ($i < 2m - 1$) (where $m = \log_2 N/2$, $N = 2^n$) [19] request may arrive at any of the three input links. For each request, use the appropriate routing tag bit and connect to the output link correspondingly. If the required output link is busy or cannot be used because of a fault in the next stage, route the request via the third output link known as auxiliary link to the other switch in the loop. If the auxiliary link is also unusable, because it is busy or faulty, then drop the request. A fault in the demultiplexer at the output of a switch in a stage (2m - 1) is regarded as a fault in that switch. From the demultiplexer, the request is routed to the upper or lower destinations according to the least significant bit of the tag (i.e d_{n-1}).

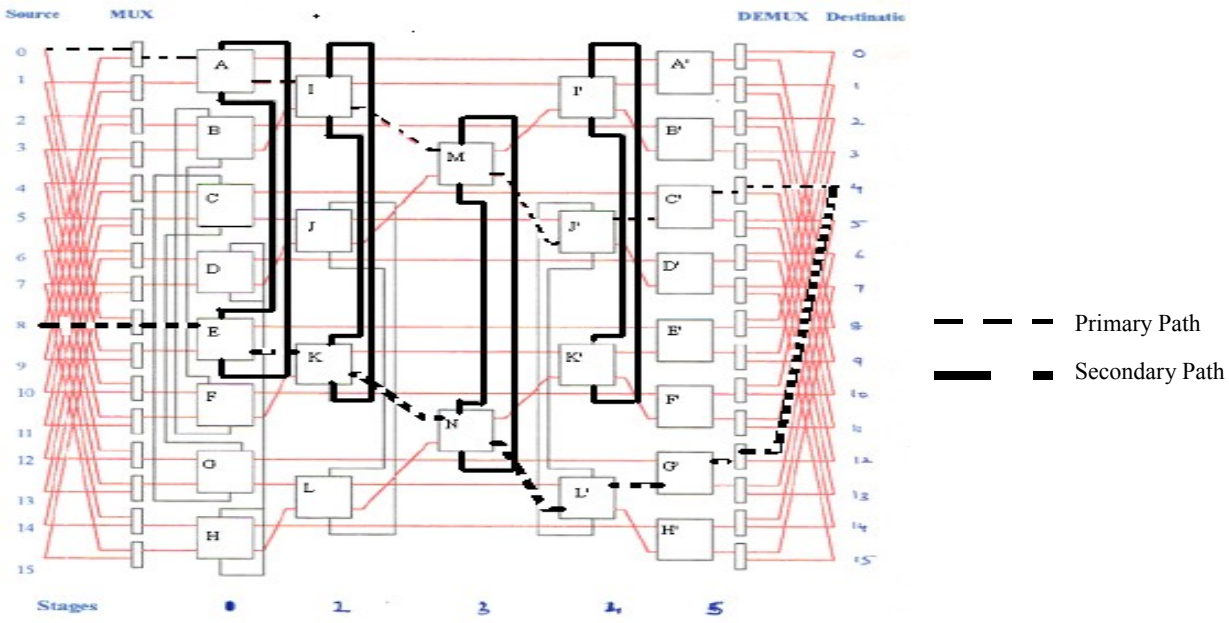


Figure 4.3: Routing in FT

In the FT network if the Source=0000 and Destination=0100 then routing tag =0.1.1.1.0.0.0. and following are the possible paths from where the request can go as shown in Figure 4.3.

Primary Path: 0-> MUX(0)->A->I->M->J' ->C' ->DEMUX(4)->4

Secondary Path: 0-> MUX(8)->E->K->N->L' ->G' ->DEMUX(12)->4

CHAPTER 5 Performance Analysis

5.1 Performance Parameters

Following are the main parameters used to measure the performance of a MIN.

Permutation Passibility

Permutation Passibility tells out of N number of requests occurring simultaneously at the input side and how many of them will successfully mature i.e. will reach their destination 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 may not be served at all [27].

Path Length

The path length p, of a path between an input terminal and an output terminal in a MIN is the number of SEs encountered by a request sent along that path [16]. It gives the information about the different path lengths possible between the source-destination pair [6].

Reliability

Reliability, of a system is the probability that it will successfully perform its intended operations for a given time under stated operating conditions [9]. 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.

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 [6]. 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.

Fault Tolerance

A fault tolerance multistage interconnection network provides service even under the faults. Fault can be permanent or transient in nature. Fault tolerance is a criterion that must be met for the network which has tolerated a given fault or faults [13]. 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 [15].

5.1.1 Permutation Passibility for FT, FT-1 and FT-2 Network

In unique path multistage interconnection networks there exists a unique path between a source and a destination. While multipath multistage interconnection networks provide more than one paths from a source to a destination. In multipath multistage interconnection networks, the request is routed through an alternative path, if the most favorable path is not amiable due to the following reasons:

Some switch(es) in that path may be faulty

Some other request is occupying that path at that instant of time

In the multipath MINs a very important performance parameter is the Permutation passibility (which means that 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) [27]. The network tries to send the request firstly through the most favorable path if such path is not available then the request is routed through the alternative path. If there is no alternative path available, the request is dropped. The desirable character of any network is that the network should be such that it allows maximum requests through it with minimum path length (PL).

The messages issued by one user should not interfere with the rest of the network traffic. To find out the permutations passable in a network, it is assumed that Si, (i=0, l ... N-1) and Di, (i=0, I... N-1) represents the source and destination of a network respectively [25]. The permutations realized by the FT, FT-1 and FT-2 in the absence, as well as in the presence of faults have been computed and portrayed in this section.

5.1.1.1 Permutation Passibility 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.

5.1.1.1.1 FT Network

Case I:

Set of ten source destination pairs is [1,2] , [2,3] , [12,3], [14,12], [5,9], [9,13], [6,8], [4,5], [7,5], [15,4]

The various paths that will be followed for this set are:

[1,2] 1-->MUX(1)-->A-->I-->I'-->B'-->DEMUX(2)-->2…….P L = 6

[2,3] 2-->MUX(2)-->B-->B'-->DEMUX(3)-->3……. P L = 4

[12,3] 12-->MUX(4)-->C-->J->N-->I'-->B'-->DEMUX(3)-->3……....P L= 7

[14,12] 14-->MUX(6)-->D-->J-->J'-->C'-->DEMUX(4)-->12……PL = 6

[5,9] 5-->MUX(5)-->C-->G-->L-->N-->K'-->E'-->DEMUX(9)-->9……..P L= 8

[9,13] 9-->MUX(1)-->CLASH……PL= 0

[6,8] 6-->MUX(6)-->CLASH……PL = 0

[4,5] 4-->MUX(4)-->CLASH……PL = 0

[7,5] 7-->MUX(7)-->D-->H-->L-->L'-->G'-->DEMUX(13)-->5……...PL= 7

[15,4] 15-->MUX(7)-->CLASH……PL= 0

No. of Requests = 10

Total number of requests matured successfully = 6

Total PL = 38

Average Path Length = [38/6] =6.333333

Case II:

Set of eight source destination pairs is [4,2], [1,3], [9,6], [3,10], [8,2], [7,10], [0,7], [13,2]

The various paths that will be followed for this set are:

[4,2] 4-->MUX(4)-->C-->J-->M-->I'-->B'-->DEMUX(2)-->2……PL = 7

[1,3] 1-->MUX(1)-->A-->I-->I'-->B'-->DEMUX(3)-->3……..PL = 6

[9,6] 9-->MUX(1)-->CLASH……PL = 0

[3,10] 3-->MUX(3)-->B-->B'-->DEMUX(2)-->10…….PL= 4

[8,2] 8-->MUX(0)-->A-->E-->K-->K'-->F'-->DEMUX(10)-->2…….PL = 7

[7,10] 7-->MUX(7)-->D-->J -->L-->N-->K'-->F'-->DEMUX(10)-->10……PL = 8

[0,7] 0-->MUX(0)-->CLASH…….PL= 0

[13,2] 13-->MUX(5)-->C-->G-->L-->J-->M-->I'-->B'-->DEMUX(2)-->2…...PL= 9

No. of Requests = 8

Total number of requests matured successfully = 6

Total PL = 41

Average Path Length = [41/6] =6.833333

Case III:

Set of fourteen source destination pairs is [1,3], [2,4], [3,5], [6,5], [12,4], [13,2], [9,6], [4,12], [5,14], [7,10], [8,3], [10,2], [11,15], [14,3]

The various paths that will be followed for this set are:

[1,3] 1-->MUX(1)-->A-->I-->I'-->B'-->DEMUX(3)-->3……PL= 6

[2,4] 2-->MUX(2)-->B-->I-->M-->J'-->C'-->DEMUX(4)-->4……PL = 7

[3,5] 3-->MUX(3)-->B-->F-->K-->N-->L'-->G'-->DEMUX(13)-->5……..PL= 8

[6,5] 6-->MUX(6)-->D-->J-->J'-->C'-->DEMUX(5)-->5…….PL = 6

[12,4] 12-->MUX(4)-->C-->C'-->DEMUX(4)-->4…….PL = 4

[13,2] 13-->MUX(5)-->C-->J-->M-->I'-->B'-->DEMUX(2)-->2……PL = 7

[9,6] 9-->MUX(1)-->CLASH……PL= 0

[4,12] 4-->MUX(4)-->CLASH……PL = 0

[5,14] 5-->MUX(5)-->CLASH……PL= 0

[7,10] 7-->MUX(7)-->D-->H-->L-->N-->K'-->F'-->DEMUX(10)-->10……PL = 8

[8,3] 8-->MUX(0)-->A-->E-->I-->I'-->F'-->DEMUX(11)-->3…….PL = 7

[10,2] 10-->MUX(2)-->CLASH……...PL = 0

[11,15] 11-->MUX(3)-->CLASH…….PL= 0

[14,3] 14-->MUX(6)-->CLASH……..PL= 0

No. of Requests = 14

Total number of requests matured successfully = 8

Total PL = 53

Average Path Length = [53/8] =6.625000

Case IV:

Set of nine source destination pairs is [4,2], [11,3], [9,6], [3,10], [8,2], [7,10], [0,7] , [14,6], [13,2]

The various paths that will be followed for this set are :

[4,2] 4-->MUX(4)-->C-->J-->M-->I'-->B'-->DEMUX(2)-->2……PL= 7

[11,3] 11-->MUX(3)-->B-->B'-->DEMUX(3)-->3……PL= 4

[9,6] 9-->MUX(1)-->A-->I-->M-->J'-->D'-->DEMUX(6)-->6……PL = 7

[3,10] 3-->MUX(3)-->CLASH……..PL = 0

[8,2] 8-->MUX(0)-->A-->E-->K-->K'-->F'-->DEMUX(10)-->2……PL= 7

[7,10] 7-->MUX(7)-->D-->J-->L-->N-->K'-->F'-->DEMUX(10)-->10……PL = 8

[0,7] 0-->MUX(0)-->CLASH……...PL = 0

[14,6] 14-->MUX(6)-->D-->D'-->DEMUX(6)-->6……PL= 4

[13,2] 13-->MUX(5)-->C-->G-->L-->J-->M-->I'-->B'-->DEMUX(2)-->2...PL= 9

No. of Requests = 9

Total number of requests matured successfully = 7

Total PL = 46

Average Path Length = [46/7] =6.571429

Case V:

Set of nine source destination pairs is [14,11], [15,0], [1,13], [13,12], [5,4], [11,10], [10,12]

The various paths that will be followed for this set are :

[14,11] 14-->MUX(6)-->D-->J-->M-->I'-->B'-->DEMUX(3)-->11…… PL= 7

[15,0] 15-->MUX(7)-->D-->H-->L-->N-->K'-->E'-->DEMUX(8)-->0……PL = 8

[1,13] 1-->MUX(1)-->A-->I-->M-->J'-->C'-->DEMUX(5)-->13……PL= 7

[13,12] 13-->MUX(5)-->C-->C'-->DEMUX(4)-->12……PL= 4

[5,4] 5-->MUX(5)-->CLASH……PL = 0

[11,10] 11-->MUX(3)-->B-->B'-->DEMUX(2)-->10……PL = 4

[10,12] 10-->MUX(2)-->B-->I-->K-->N-->L'-->G'-->DEMUX(12)-->12…...PL= 8

No. of Requests = 7

Total number of requests matured successfully = 6

Total PL = 38

Average Path Length = [38/6] =6.333333

Case VI:

Set of eight source destination pairs is [5,11], [4,2], [12,9], [11,8], [6,12], [3,5], [8,12], [1,14]

The various paths that will be followed for this set are :

[5,11] 5-->MUX(5)-->C-->J-->M-->I'-->B'-->DEMUX(3)-->11……PL= 7

[4,2] 4-->MUX(4)-->C-->G-->L-->N-->K'-->F'-->DEMUX(10)-->2……...PL= 8

[12,9] 12-->MUX(4)-->CLASH…….PL = 0

[11,8] 11-->MUX(3)-->B-->I-->I'-->A'-->DEMUX(0)-->8……..PL = 6

[6,12] 6-->MUX(6)-->D-->J-->J'-->C'-->DEMUX(4)-->12……..PL= 6

[3,5] 3-->MUX(3)-->CLASH……..PL= 0

[8,12] 8-->MUX(0)-->A-->I-->M-->J'-->C'-->DEMUX(4)-->12……PL = 7

[1,14] 1-->MUX(1)-->A-->E-->K-->N-->L'-->H'-->DEMUX(14)-->14…..PL = 8

No. of Requests = 8

Total number of requests matured successfully = 6

Total PL = 42

Average Path Length = [42/6] =7.000000

Case VII:

Set of ten source destination pairs is [1,2], [4,13], [14,8], [3,0], [5,6], [6,14], [9,4], [0,12], [7,11], [11,8]

The various paths that will be followed for this set are:

[1,2] 1-->MUX(1)-->A-->I-->I'-->B'-->DEMUX(2)-->2……...PL = 6

[4,13] 4-->MUX(4)-->C-->C'-->DEMUX(5)-->13…….PL = 4

[14,8] 14-->MUX(6)-->D-->J-->M-->I'-->A'-->DEMUX(0)-->8……PL= 7

[3,0] 3-->MUX(3)-->B-->I-->M-->N-->K'-->E'-->DEMUX(8)-->0……...PL= 8

[5,6] 5-->MUX(5)-->C-->J-->J'-->D'-->DEMUX(6)-->6……...PL = 6

[6,14] 6-->MUX(6)-->CLASH……PL= 0

[9,4] 9-->MUX(1)-->CLASH……...PL = 0

[0,12] 0-->MUX(0)-->A-->E-->K-->N-->L'-->G'-->DEMUX(12)-->12…….PL= 8

[7,11] 7-->MUX(7)-->D-->H-->L-->N-->M-->I'-->B'-->DEMUX(3)-->11...PL = 9

[11,8] 1-->MUX(3)-->CLASH……..…PL= 0

No. of Requests = 10

Total number of requests matured successfully = 7

Total PL = 48

Average Path Length = [48/7] =6.857143

Case VIII:

Set of twelve source destination pairs is [15,15], [5,11], [0,0], [4,2], [10,9], [11,8], [6,12], [3,5], [12,12], [9,0], [8,12], [1,14]

The various paths that will be followed for this set are:

[15,15] 15-->MUX(7)-->D-->D'-->DEMUX(7)-->15……PL = 4

[5,11] 5-->MUX(5)-->C-->J-->M-->I'-->B'-->DEMUX(3)-->11……PL= 7

[0,0] 0-->MUX(0)-->A-->A'-->DEMUX(0)-->0……..PL = 4

[4,2] 4-->MUX(4)-->C-->G-->L-->N-->K'-->F'-->DEMUX(10)-->2……PL = 8

[10,9] 10-->MUX(2)-->B-->I-->I'-->A'-->DEMUX(1)-->9……PL = 6

[11,8] 11-->MUX(3)-->B-->F-->K-->K'-->E'-->DEMUX(8)-->8……..PL = 7

[6,12] 6-->MUX(6)-->D-->F-->F'-->C'-->DEMUX(4)-->12……PL = 6

[3,5] 3-->MUX(3)-->CLASH……PL = 0

[12,12] 12-->MUX(4)-->CLASH……PL = 0

[9,0] 9-->MUX(1)-->A-->E-->E'-->DEMUX(8)-->0……PL = 5

[8,12] 8-->MUX(0)-->CLASH……...PL = 0

[1,14] 1-->MUX(1)-->CLASH……PL = 0

No. of Requests = 12

Total number of requests matured successfully = 8

Total PL = 47

Average Path Length = [47/8] =5.875000

5.1.1.2 Permutation Passibility with faults

As multiple paths of varying lengths are available, faulty network will pass most of the permutations unless the fault is a critical fault; in that case no path is available for the data to reach its final destination. Conjugate pairs provide redundant path for all the SEs as well as the multiplexers and demultiplexers.

5.1.1.2.1 FT Network

Case I:

Set of ten source destination pairs is [1,2] , [2,3] , [12,3], [14,12], [5,9], [9,13], [6,8], [4,5], [7,5], [15,4]

The various paths that will be followed for this set are:

Failed switch is 32

[1,2] 1-->MUX(9)-->E-->K-->K'-->F'-->DEMUX(10)-->2……...PL = 6

[2,3] 2-->MUX(2)-->B-->B'-->DEMUX(3)-->3……...PL = 4

[12,3] 12-->MUX(4)-->C-->J-->M-->I'-->B'-->DEMUX(3)-->3……PL= 7

[14,12] 14-->MUX(6)-->D-->J->J'-->C'-->DEMUX(4)-->12……..PL= 6

[5,9] 5-->MUX(5)-->C-->G-->L-->N-->K'-->E'-->DEMUX(9)-->9……..PL= 8

[9,3] 9-->MUX(9)-->CLASH…….PL= 0

[6,8] 6-->MUX(6)-->CLASH…….PL = 0

[4,5] 4-->MUX(4)-->CLASH…….PL= 0

[7,5] 7-->MUX(7)-->D-->H-->L-->L'-->G'-->DEMUX(13)-->5……PL = 7

[15,4] 15-->MUX(7)-->CLASH……PL= 0

No. of Requests = 10

Total number of requests matured successfully = 6

Total PL = 38

Average Path Length = [38/6] =6.333333

Case II:

Set of eight source destination pairs is [4,2] [1,3] [9,6], [3,10], [8,2], [7,10], [0,7], [13,2]

The various paths that will be followed for this set are:

Failed switch is 37

[4,2] 4-->MUX(4)-->C-->J->M-->I'-->B'-->DEMUX(2)-->2……...PL= 7

[1,3] 1-->MUX(1)-->A-->I-->I'-->B'-->DEMUX(3)-->3…….PL= 6

[9,6] 9-->MUX(1)-->CLASH……...PL = 0

[3,10] 3-->MUX(3)-->B-->B'-->DEMUX(2)-->10……..PL= 4

[8,2] 8-->MUX(0)-->A-->E-->K-->K'-->F'-->DEMUX(10)-->2……..PL= 7

[7,10] 7-->MUX(7)-->D-->J-->L-->N-->K'-->F'-->DEMUX(10)-->10……...PL= 8

[0,7] 0-->MUX(0)-->CLASH……PL= 0

[13,2] 13-->MUX(5)-->C-->G-->L-->J-->M-->I'-->B'-->DEMUX(2)-->2….PL= 9

No. of Requests = 8

Total number of requests matured successfully = 6

Total PL = 41

Average Path Length = [41/6] =6.833333

Case III:

Set of fourteen source destination pairs is [1,3], [2,4], [3,5], [6,5], [12,4], [13,2], [9,6], [4,12], [5,14], [7,10], [8,3], [10,2], [11,15], [14,3]

The various paths that will be followed for this set are:

Failed switch is 44

[1,3] 1-->MUX(1)-->A-->E-->E'-->B'-->DEMUX(3)-->3……..PL = 6

[2,4] 2-->MUX(10)-->B1-->E1-->G1-->F1'-->C1'-->DEMUX(12)-->4……PL = 7

[3,5] 3->MUX(11)-->B1-->B-->E-->E1-->G1->F1'->C1'->DEMUX(13)->5…PL= 9

[6,5] 6-->MUX(6)-->D-->F-->F'-->C'-->DEMUX(5)-->5……PL = 6

[12,4] 12-->MUX(4)-->C-->C'-->DEMUX(4)-->4…….PL = 4

[13,2] 13-->MUX(13)-->C1-->F1-->G1-->E1'-->B1'-->DEMUX(10)-->2….PL = 7

[9,6] 9-->MUX(9)-->A1-->E1-->CLASH……..PL = 2

[4,12] 4-->MUX(4)-->CLASH……...PL= 0

[5,14] 5-->MUX(5)-->C-->F-->F1-->F1'-->D1'-->DEMUX(14)-->14……..PL= 7

[7,10] 7-->MUX(15)-->D1-->F1-->CLASH……..PL= 2

[8,3] 8-->MUX(0)-->A-->A1-->E1-->E1'-->E'-->B'-->DEMUX(3)-->3…..PL= 8

[10,2] 10-->MUX(2)-->B-->B'-->DEMUX(2)-->2……PL= 4

[11,15] 11-->MUX(11)-->CLASH……..PL= 0

[14,3] 14-->MUX(14)->D1->D->F->F1->G1->E1'-->B1'-->DEMUX(11)->3...PL= 9

No. of Requests = 14

Total number of requests matured successfully = 10

Total PL = 71

Average Path Length = [71/10] =7.100000

Case IV:

Set of nine source destination pairs is [4,2], [11,3], [9,6], [3,10], [8,2], [7,10], [0,7] , [14,6], [13,2]

The various paths that will be followed for this set are:

Failed switch is 35

[4,2] 4-->MUX(4)-->C-->J-->M-->I'-->B'-->DEMUX(2)-->2……...PL = 7

[11,3] 11-->MUX(3)-->B-->B'-->DEMUX(3)-->3……PL = 4

[9,6] 9-->MUX(1)-->A-->I-->M-->J'-->D'-->DEMUX(6)-->6……..…...PL = 7

[3,10] 3-->MUX(3)-->CLASH……..PL= 0

[8,2] 8-->MUX(0)-->A-->E-->K-->K'-->F'-->DEMUX(10)-->2……..….PL= 7

[7,10] 7-->MUX(15)-->H-->L-->N-->K'-->I'-->B'-->DEMUX(2)-->10…...PL = 8

[0,7] 0-->MUX(0)-->CLASH……PL = 0

[14,6] 14-->MUX(14)-->H-->H'-->DEMUX(14)-->6……PL= 4

[13,2] 13-->MUX(5)-->C-->G-->L-->N-->K'-->F'-->DEMUX(10)-->2…...PL = 8

No. of Requests = 9

Total number of requests matured successfully = 7

Total PL = 45

Average Path Length = [45/7] =6.428571

Case V:

Set of nine source destination pairs is [14,11], [15,0], [1,13], [13,12], [5,4], [11,10], [10,12]

The various paths that will be followed for this set are:

Failed switch is 42

[14,11] 14-->MUX(6)-->D-->J-->M-->I'-->B'-->DEMUX(3)-->11…….PL= 7

[15,0] 15-->MUX(7)-->D-->H-->L-->N-->K'-->E'-->DEMUX(8)-->0……PL= 8

[1,13] 1-->MUX(1)-->A-->I-->M-->J'-->C'-->DEMUX(5)-->13…….PL= 7

[13,12] 13-->MUX(5)-->C-->C'-->DEMUX(4)-->12……PL = 4

[5,4] 5-->MUX(5)-->CLASH……PL= 0

[11,10] 11-->MUX(3)-->B-->B'-->DEMUX(2)-->10……PL= 4

[10,12] 10-->MUX(2)-->B-->I-->CLASH……PL= 0

No. of Requests = 7

Total number of requests matured successfully = 5

Total PL = 30

Average Path Length = [30/5] =6.00000

Case VI:

Set of eight source destination pairs is [5,11], [4,2], [12,9], [11,8], [6,12], [3,5], [8,12], [1,14]

The various paths that will be followed for this set are:

Failed switch is 33

[5,11] 5-->MUX(5)-->C-->J-->M-->I'-->B'-->DEMUX(3)-->11……PL = 7

[4,2] 4-->MUX(4)-->C-->G-->L-->N-->K'-->F'-->DEMUX(10)-->2…….PL = 8

[12,9] 12-->MUX(4)-->CLASH……PL= 0

[11,8] 11-->MUX(11)-->F-->K-->K'-->E'-->DEMUX(8)-->8……PL = 6

[6,12] 6-->MUX(6)-->D-->J-->J'-->C'-->DEMUX(4)-->12……PL= 6

[3,5] 3-->MUX(11)-->CLASH……PL= 0

[8,12] 8-->MUX(0)-->A-->I-->M-->J'-->C'-->DEMUX(4)-->12……PL= 7

[1,14] 1-->MUX(1)-->A-->E-->K-->N-->L'-->H'-->DEMUX(14)-->14……PL= 8

No. of Requests = 8

Total number of requests matured successfully = 6

Total PL = 42

Average Path Length = [42/6] =7.000000

Case VII:

Set of ten source destination pairs is [1,2], [4,13], [14,8], [3,0], [5,6], [6,14], [9,4], [0,12], [7,11], [11,8]

The various paths that will be followed for this set are:

Failed switch is 48

[1,2] 1-->MUX(1)-->A-->I-->I'-->B'-->DEMUX(2)-->2……PL = 6

[4,13] 4-->MUX(4)-->C-->C'-->DEMUX(5)-->13……PL= 4

[14,8] 14-->MUX(6)-->D-->J-->M-->I'-->A'-->DEMUX(0)-->8……...PL= 7

[3,0] 3-->MUX(3)-->B-->I-->M-->CLASH…….PL= 3

[5,6] 5-->MUX(5)-->C-->J-->J'-->D'-->DEMUX(6)-->6……PL= 6

[6,14] 6-->MUX(6)-->CLASH……...PL= 0

[9,4] 9-->MUX(1)-->CLASH……PL= 0

[0,12] 0-->MUX(0)-->A-->E-->K-->N-->L'-->G'-->DEMUX(12)-->12…...PL= 8

[7,11] 7-->MUX(7)-->D-->H-->L-->J-->M-->I'-->B'-->DEMUX(3)-->11…PL= 9

[11,8] 11-->MUX(3)-->CLASH……PL = 0

No. of Requests = 10

Total number of requests matured successfully = 6

Total PL = 48

Average Path Length = [43/6] =7.166667

Case VIII:

Set of twelve source destination pairs is [15,15], [5,11], [0,0], [4,2], [10,9], [11,8], [6,12], [3,5], [12,12], [9,0], [8,12], [1,14]

The various paths that will be followed for this set are:

Failed switch is 49

[15,15] 15-->MUX(7)-->D-->D'-->DEMUX(7)-->15……PL = 4

[5,11] 5-->MUX(5)-->C-->J-->M-->I'-->B'-->DEMUX(3)-->11……...PL = 7

[0,0] 0-->MUX(0)-->A-->A'-->DEMUX(0)-->0……..PL= 4

[4,2] 4-->MUX(4)-->C-->G-->L-->N-->K'-->F'-->DEMUX(10)-->2…...PL = 8

[10,9] 10-->MUX(2)-->B-->I-->I'-->A'-->DEMUX(1)-->9……PL = 6

[11,8] 11-->MUX(3)-->B-->F-->K-->K'-->E'-->DEMUX(8)-->8…….PL= 7

[6,12] 6-->MUX(6)-->D-->J-->J'-->C'-->DEMUX(4)-->12…….PL= 6

[3,5] 3-->MUX(3)-->CLASH…….PL = 0

[12,12] 12-->MUX(4)-->CLASH……PL= 0

[9,0] 9-->MUX(1)-->A-->E-->E'-->DEMUX(8)-->0……PL= 5

[8,12] 8-->MUX(0)-->CLASH…….PL = 0

[1,14] 1-->MUX(1)-->CLASH……PL= 0

No. of Requests = 12

Total number of requests matured successfully = 8

Total PL = 47

Average Path Length = [47/8] =5.875000

5.1.2 Path Length for FT, FT-1 and FT-2 Network

Different combinations of different sets of source destination pairs are selected and their path length is calculated for these networks, only if requests mature successfully. Unsuccessful requests are indicated by CLASH. For a set of different source destination pairs, average path length is calculated. The average path length is defined as the ratio of sum of the path lengths used in the successful requests to the total number of requests.

5.1.3 Reliability for FT, FT-1 and FT-2 Network

Reliability of FT, FT-1 and FT-2 networks are analyzed in terms of Mean time to Failure (MTTF).

MTTF of MINs is evaluated using a 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 can make whole of the system useless. The reliability of a series configuration model is always worse than the poorest component in it. Series configuration model shown in Figure 5.1

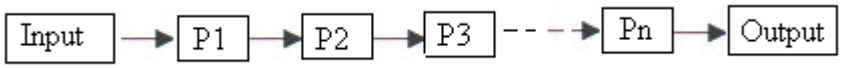


Figure 5.1: Series configuration model

Parallel configuration model

In this all components are connected parallel to each other. In this if one component fails than it can follow another path and keeps system active. This type of model is shown in Figure 5.2.

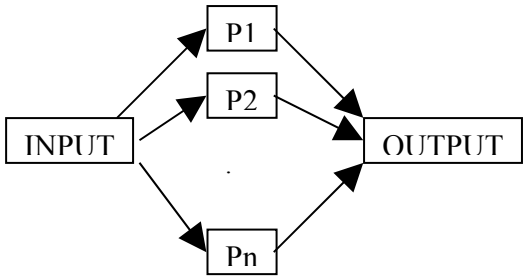


Figure 5.2: Parallel configuration model

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.

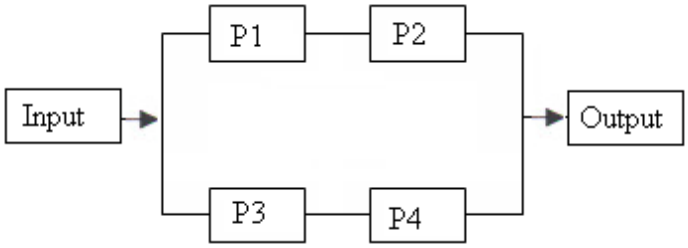


Figure 5.3: Series-Parallel configuration model

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

1. Switch failures occur independently in a network with a failure rate of for 2 x 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 $_d$ receptively. 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. Based on the gate counts of crossbar switches, the number of gates in a 2 x 2 crossbar switch is approximately equal to that in a 2 x 1 MUX or a 1 x 2 DEMUX. Thus to simplify the analysis we can assume that $_m$ = m /2 for a m x 1 MUX, where $_m$ failure rate of MUX or $_d$ (= $_m$) for 1 x m DEMUX, where $_d$ failure rate of DEMUX.
3. 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.

5.1.3.1 Reliability of FT Network

FT network has fault tolerance feature due to the presence of fork at every stage except the last in the network. Therefore, this network has got good reliability. Reliability is determined in terms of MTTF. The actual value of MTTF lies between these two values:

5.1.3.1.1 Optimistic (Upper Bound) Analysis of FT network

We assume that the FT network is operational as long as one of the two multiplexers attached to a source (in a particular sub network) is operational and as long as a conjugate pair (loop or switch) is not faulty, then we will permit as many as half components to fail and the FT network may still be operational.

5.1.3.1.1.1 Reliability Block Diagram

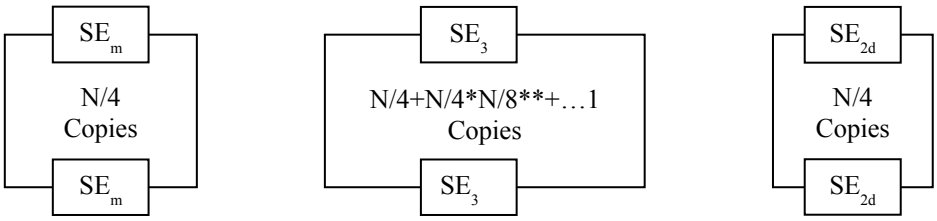


Figure 5.4: Upper Bound for FT

5.1.3.1.1.2 Reliability Equations

$R_{optimistic}(t) = f1 * f2 * f3$ where,

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

$$f2 = \left[\ 1 - \left(1 - e^{-\lambda_3 t}\right)^2\right]^{\left\{(N/4)+(N/4)^{*}+(N/8)^{**}+\ldots+2^{\circ}\right\}^{\Delta}}$$

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

$R_{optimistic}(t)$ .dt $\qquad TTF = \int_0^{\infty}$

5.1.3.1.2 Pessimistic (Lower Bound) Analysis of FT network

At the input side of FT, the routing scheme does not consider the multiplexers to be an integral part of a 3x3 switch. For example, as long as one of the two multiplexers attached to a particular switch is operational, the switch can still be used for routing. Hence, if we group two multiplexers with each switch in input side and

consider them as a series system (SE_{3m}), then we will have a conservative estimate of the reliability of these components shown as factor f1 in the reliability calculation. The aggregated failure rate will be $_{3m}$ = 4.25 [25]. Finally, these aggregated components and the switches in the intermediate stages can be arranged in pairs of conjugate loops, shown as factor f2 in the reliability calculation. To obtain the pessimistic lower bound o the reliability of FT we assume that the network has failed whenever more than one conjugate loop has a faulty element or more than one conjugate switch in the last stage fails.

5.1.3.1.2.1 Reliability Block Diagram

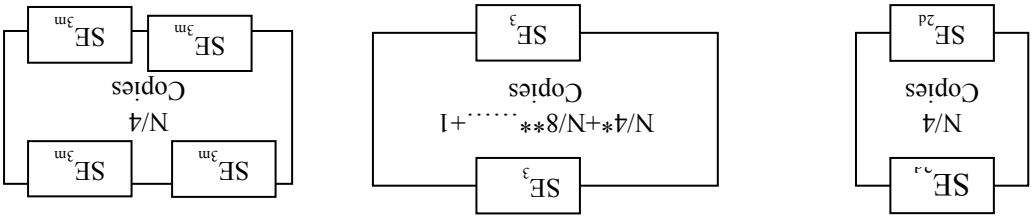


Figure 5.5: Lower Bound for FT

5.1.3.1.2.2 Reliability Equations

$$MTTF = \int_0^{\infty} R_{pessimistic}\, dt$$

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

$$f2 = \left[1 - \left(1 - e^{-\lambda_3 t}\right)^2 \right]^{\left\{(N/4)^{*} + (N/8)^{**} + \ldots 1^{\circ}\right\}^{\Delta}}$$

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

$R_{pessimistic}(t) = f1 * f2 * f3$

*	is the term added for path-length>2
**	is the term added for path-length>4
	is the term added for the largest path- length
△	is the term added for path-length>2

5.1.4 Cost Effectiveness for FT, FT-1 and FT-2 Network

To estimate the cost of a network, one common method is 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 x 2 switch has four units of hardware cost, whereas a 3 x 3 switch has nine units. For the multiplexers and demultiplexers, we roughly assume that each of m x 1 multiplexers or 1 x m demultiplexers has m units of cost [19].

Table 5.1: Cost Functions

MINs	Cost
FT	$(9.75\ 2^{n+1}-54)$
FT-1	$(9.75\ 2^{n+1}-72)$
FT-2	$(9.75\ 2^{n+1}-40)$

5.1.5 Fault-tolerance for FT, FT-1 and FT-2 Network

Fault-tolerance in an interconnection network is very important for its continuous operation over a relatively long period of time. Fault-tolerance is the ability of the network to operate even in the presence of faults, although at a degraded performance. There are many ways to increase fault-tolerance of a network [5].

Increasing the number of stages

Providing multiple links between stages

Increasing size of the switch

Incorporating multiple copies of a basic network

Under this criterion, a network is faulty if there is any source-destination pair that cannot be connected because of faulty components in the network [7]. To achieve fault tolerance, we exploit the fact that there are subsets of switches in each stage which lie on paths leading to the destinations. All the switches in a given stage which lead to the same subset of destinations comprise a conjugate subset of partitioned into several conjugate subsets. The partial routing tag required to set up a connection to a reachable destination from either switch of a conjugate subset is also the same. In each conjugate subset of switches, there are several pairs of switches called conjugate pairs of switches. The switches in such a pair are connected to the same switches in the next stage. Conjugate subsets and conjugate pairs of switches play a fundamental

role. If a switch is not able to process a request for connection because of a faulty switch in the next stage or because of a busy link, it can route that request via its auxiliary output link to the next switch in the loop. The next switch can then make a connection to a different (non-faulty) switch in the following stage. In fact, using an auxiliary link whenever a fault is encountered allows any source to be connected to any destination while tolerating any single faulty switch in any stage other than the initial stage and the final stage.

A major advantage of implementing the loops as modules is that such implementation makes MINs easier to maintain and repair. In other words, the removal of a loop as a whole from a network need not disrupt the continued operation of the network. Thus, when a switch fault is identified, the loop containing the faulty switch can be removed from the network and a replacement loop inserted without interrupting the operation of the network.

The reliability and performance improvement obtained from a multi-path network depend upon how effectively the alternate paths available are used by the routing algorithm. One can use a backtracking routing algorithm that exhaustively searches for an available fault-free path. However, implementation of backtracking is relatively expensive in terms of hardware and backtracking can, in some situations; take an inordinately long time to set up connections. The non-backtracking algorithm assumes that sources and switches have the ability to detect faults in the switches to which they are connected. A network is single-fault tolerant if it can function as specified by its fault-tolerance criterion despite any single fault conforming to its fault model. More generally, if any set of i faults can be tolerated, then a network is i-fault tolerant. A network that can tolerate some instances of i faults is robust although not i-fault tolerant.

In the FT -1 network, number of stages are less than the FT network so, the fault tolerance ability of the network increases. In FT-1 network the requests first routes through the primary path. If any fault occurs then the request is routed through the secondary path as shown in Figure 5.10.

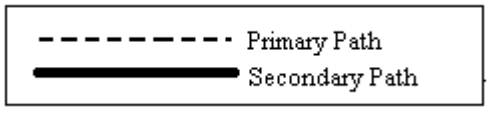


Figure 5.10: Alternate Path in FT -1 network

Following are the possible paths from where the request can go:

Primary Path: 0->MUX (0)->A->A' ->DEMUX (0) ->0

Secondary Path:

0->MUX (0)->A->I->I' ->A' ->DEMUX (0) ->0

0->MUX(8)->E->K->K' ->E' ->DEMUX(8)->0

0->MUX (8) ->E->E' ->DEMUX (8) ->0

CHAPTER 6 Experimental Results

6.1 Permutation Passibility

In this section FT, FT-1 and FT-2 network is analyzed in terms of permutation passibility and average path length of this network is calculated. The result of permutation passibility is shown in Table 6.1 and Figure 6.1 shows the comparison on the basis of requests matured. Table 6.2 shows average path length of these networks and Figure 6.2 shows comparison of average path length.

Table 6.1: Comparison on the basis of Requests Matured

No. of Requests	FT	FT-1	FT-2
Total No. of Requests	78	78	78
No. of Requests matured without failing switch	54	68	73
No. of Requests matured after failing switch	54	63	69

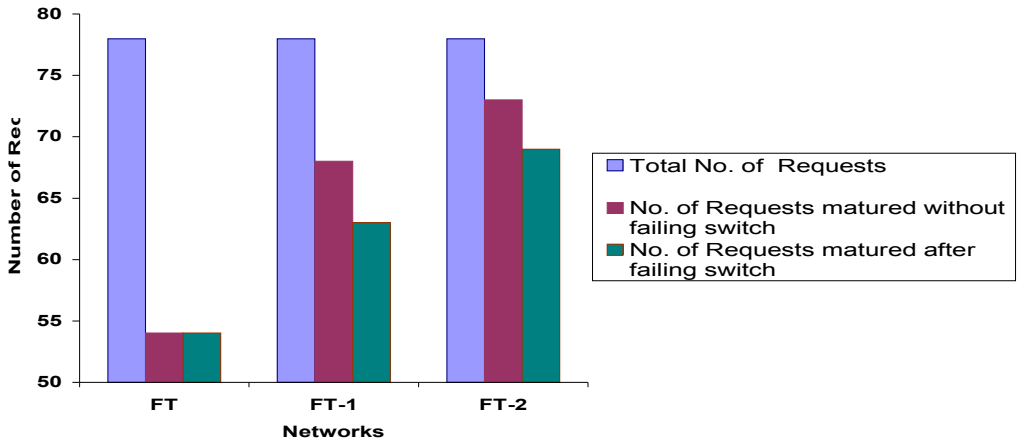


Figure 6.1: Comparison on the basis of Requests Matured

Table 6.2: Comparison on the basis of Average Path Length

Criteria	FT	FT-1	FT-2
Without Failing Switch	52.4	50.35	51.82
After Failing Switch	52.71	50.41	52.35

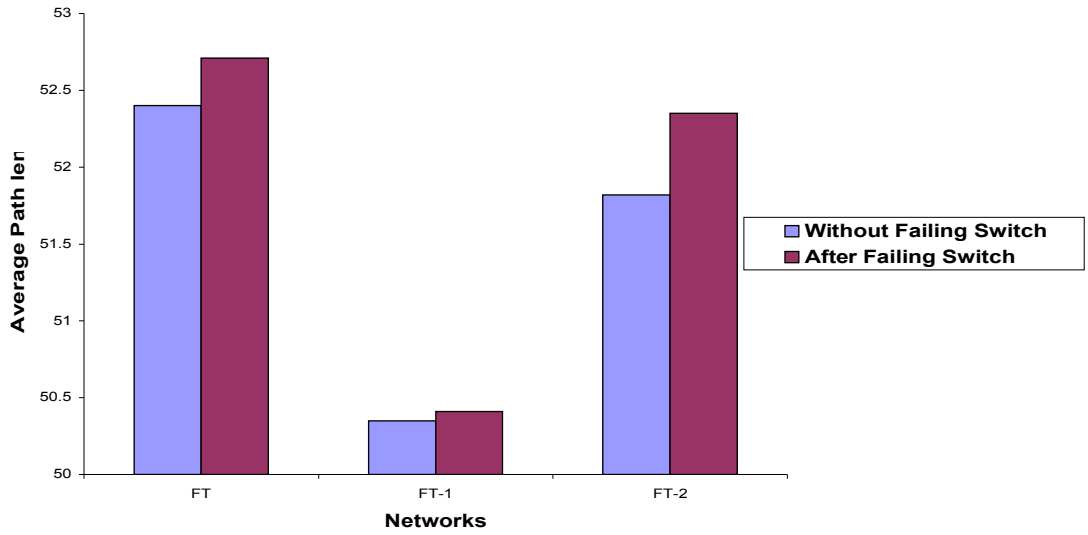


Figure 6.2: Comparison on the basis of Average Path Length

6.2 Reliability

Reliability of FT, FT-1 and FT-2 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:

6.2.1 Comparative Analysis of Upper Bound MTTF

Optimistic approach is used for calculating the Upper bound MTTF (explained in chapter 5). And the result is shown in Table 6.3 and in Figure 6.3.

6.2.2. 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 6.4 and in Figure 6.4.

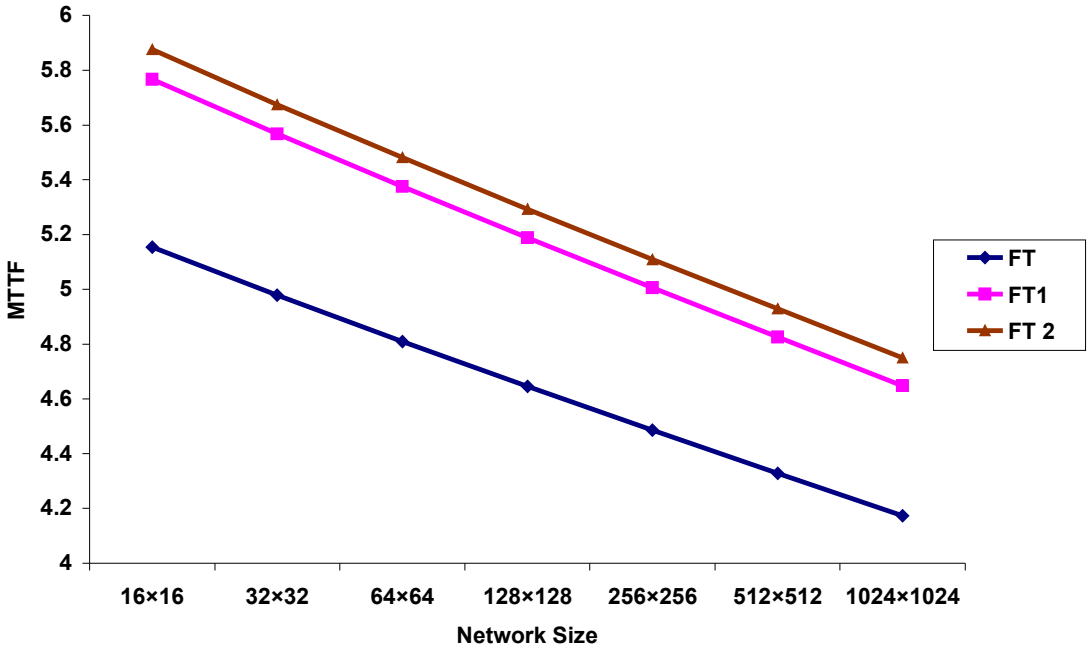


Figure 6.4: Comparative Analysis of Lower Bound MTTF

6.3 Cost Effectiveness

One common method is used to calculate the cost of FT, FT-1 and FT-2 networks (method is explained in Chapter 5). Comparison of these networks is shown below in Table 6.5 and in Figure 6.5.

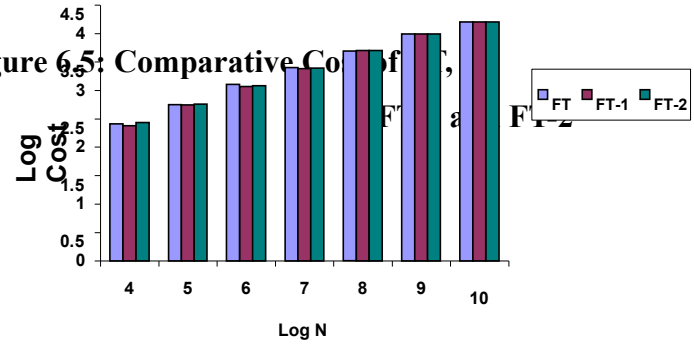


Figure 6.5: Comparative Co[illegible]f [illegible], [illegible] a[illegible] F[illegible]

CHAPTER 7 Conclusions and Future Scope

7.1 Conclusions

In this thesis irregular networks are analyzed in terms of permutation Passibility, reliability, fault tolerance, path length and cost. A detail of different multistage interconnection networks is provided.

Conclusions drawn from the work done in this thesis is summarized as:

Permutation Passibility of proposed irregular network FT-1 and FT-2 is better than the existing irregular FT network.

Reliability is analyzed in terms of MTTF (Mean Time to Failure) or ability of a network to perform even in the presence of faults. Reliability of FT-1 is higher than FT and reliability of FT-2 is higher than FT-1 and FT both.

Path length of FT-1 and FT-2 decreases in the FT-1 and FT-2 network than FT network.

Fault tolerance increases in the FT-1 and FT-2 network.

7.2 Summary of Thesis Work

Work presented in this thesis is summarized as:

Survey of existing regular and irregular multistage interconnection networks

Two new networks FT-1 and FT-2 are proposed, which are derived from existing FT network.

Existing FT and two proposed networks FT-1 and FT-2 networks are analyzed for following performance measures:

Fault tolerance

Permutation Passibility

Reliability

Cost

7.3 Future Scope

The field of irregular networks can be further explored in the light of the following suggestions:

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

The designing of more irregular networks having better reliability and permutation passibility can be explored.

The detailed performance analysis can be done for all the Irregular MINs with respect to other parameters such as Probability of acceptance and Bandwidth etc.

Concept of optical MINs also be used.

References

[1] 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].

[2] 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].

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

[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 J. Computer c. 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”, Proceeding 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., (1985), “Computer architecture and parallel processing”, McGraw Hill book company, New York, 1985.

[15]Kamiura Naotake, Kodera Takashi and Matsui Nobuyuki,” Fault Tolerant Multistage Interconnection Networks with Widely Dispersed Paths”, Proceeding of International Conference IEEE, 2000, pp. [423-428].

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

[18]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].

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

[20]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].

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

[22]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].

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

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

[25]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].

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

[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].

[30]Wu Xingfu, Sun Xian-He,” Performance Modeling for Interconnection Networks”, Proceeding of International Conference IEEE, 2000, pp. [380-385].

Appendix

Pseudo code for Permutation Passibilty

Global Parameters:

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

(ii) Two variable for storing number of source destination pairs and number of faults. n, n_f.

(iii) An array for storing source and destination values. s_d_pair[].

(iv) An array for storing the faulty nodes. faulty_node[].

(v) An array to stores the binary of the destination. dest_a [].

(vi) An array for storing the nodes traversed in a path. path [][].

(vii) An array that keeps track of all visited nodes. visit [][].

Function Name: main ()

Called by: operating system

Calling: (i) enter_faults()

(ii) find_path()

(iii) display ()

Parameters passed: No

Purpose: This function includes the overall logic of the algorithm.

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: nothing

Parameters passed: Number of faulty nodes

Purpose: To get the number of faulty nodes and return to main function.

Function Body:

(i) Enter the faulty nodes

(ii) Return to main () function.

Function Name: find_path()

Called by: main function

Calling:

(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()

(ix) stage_4()

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 (), stage_3 () and stage_4 () are called for finding the next switches in the path.

(vii) Return to the main ().

Function Name: cal_rout_tag(variable that tells the path length, destination)

Called by: find_path()

Calling: nothing

Parameters passed: Value of destinations,

Purpose: To find out the routing tag according to path length.

Function Body:

(i) Find the path length

(ii) If path length is minimum routing tag = $0.D_1.D_{n-1}$

(iii) Else routing tag =$1.D2. D_1.D_0.D_{n-1}$

Function Name: find_mux()

Called by: find_path()

Calling: chek_clash()

Parameters passed: Value of source and destinations

Purpose: To find out the MUX in the path

Function Body:

(iv) Find out the appropriate MUX

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

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

Function Name: find_mux1 ()

Called by: find_path()

Calling: 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: 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:

(iv) Find out the next alternate MUX

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

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

Function Name: mux_to_stage1 ()

Called By: find_path()

Calling: 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: 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: 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: 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_4 ()

Called By: find_path()

Calling: chek_clash()

Parameters passed: The next switch value.

Purpose: Find out the next switch in the path

Function Body:

(iv) Find out next switch

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

(vi) 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()

(viii) stage_4 ()

Calling: nothing

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: 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: nothing

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 cost Evaluation

Function Name: main ()

Called by: operating system

Calling: (i) FT ()

(ii) FT-1()

(iii) FT-2()

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 FT() or FT-1() or FT-2() or all three.

Function Name: FT **()**

Called by: main ()

Calling: Nothing

Parameters passed: No

Purpose: This function calculate the cost of FT network

Function Body:

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

(ii) Finally, display the evaluated cost.

Pseudo Code for Reliability calculation

Function Name: main ()

Called by: operating system

Calling: (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 calculating upper or lower bound or both.

(ii) Call the appropriate function.

Function Name: **upper ()**

Called by: main ()

Calling: Nothing

Parameters passed: No

Purpose: This function calculate the upper bound of FT network

Function Body:

(i) Evaluate the upper bound values according to the reliability equations.

(ii) Apply trapezoidal rule for calculating the upper bound MTTF.

(iii) Finally, display the evaluated MTTF.

Function Name: **lower ()**

Called by: main ()

Calling: Nothing

Parameters passed: No

Purpose: This function calculate the lower bound of FT network

Function Body:

(i) Evaluate the lower bound values according to the reliability equations.

(ii) Apply trapezoidal rule for calculating the lower bound MTTF.

(iii) Finally, display the evaluated MTTF.

List of Publications

[1] Manpreet Kaur and Rinkle Aggarwal, **"Modified Double Tree Network and its Variants: An analysis"**, National Conference on Advancements in Computer Engineering (ACE-08), Fatehgarh Sahib (Punjab), India, April3-4 2008, pp.[99-104]. **[Presented and Published].**