ROUTING ALGORITHM FOR TORUS NETWORKS

BY

KOFI OSEI BONSU (B.Sc. COMPUTER SCIENCE)

A THESIS SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE

KWAME NKRUMAH UNIVERSITY OF SCIENCE AND TECHNOLOGY

IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE

DEGREE OF MPHIL INFORMATION TECHNOLOGY

OFEBRUARY 2014

DECLARATION

thesis is a presentation of my perspectives on the subject matter and contains to the best of my knowledge; no material published by another for an award. Wherever the contributions of others are involved, every effort is made to indicate this clearly, with due reference to the literature, and acknowledgement of collaborative research and discussions. The work was done under the guidance of Mr. Dominic Asamoah of the Computer Science Department, KNUST.

KOFI OSEI BONSU ..... 28/04/2014 .....

(PG5094010) (SIGNATURE) (DATE)

CERTIFIED BY:

Dominic Asamoah ..... 14-05-14 .....

(SUPERVISOR'S NAME) (SIGNATURE) (DATE)

CERTIFIED BY:

Dr. M Asante ..... 15/05/14 .....

(HEAD OF DEPARTMENT) (SIGNATURE) (DATE)

ABSTRACT

The inevitable shift towards parallel computing systems has accentuated the need for more reliable and high performing networks. This is because as the number of components in the network increases, so too does the failure rate of the system. Torus network which is a type of interconnection network has become very important in the deve!opment of digital and/or parallel computer systems. Torus is an n-dimensional network topology, with each dimension having k nodes. A routing algorithm determines the sequence of channels for a packet to traverse from the source to destination node in torus networks. Several routing algorithms have been proposed in literature to route packet in torus networks. This work develops a performance-based network simulator which uses a routing algorithm to route packets from source to destination and determines the terminal reliability and performance of the network in order to improve network throughput. Terminal reliability, a commonly used measure of ccnr.ectivity is the ability of a network to carry out the desired operation such as comrnuhication. In this thesis, the simulator calculates the terminal reliability and performance of the torus network having either unidirectional links or bidirectional links using source routing by generating packet(s) from source node and transmitting it

to destination nodes. The results obtained from the simulation conducted on several torus networks with different sizes (N) were analyzed and discussed. In the end, the results shows that terminal reliability does not depend on the number of nodes (N) but rather in order to improse reliability and performance of torus networks the nodes in the must have high reliability.

DEDICATION

This work is dedicated to my supervisor, Mr. Dominic Asamoah.

To my wife Mrs. Bonsu Adomah Duah without whom I would not have come this far.

ACKNOWLEDGEMENT

The success of this project could not have happened without the excellent supervision and support of Mr. Dominic Asamoah, who not only served as my supervisor and advisor, but also encouraged me and inspired me to strive towards my goals. His selfless attitude and dedication towards academics is definitely appreciated.

I would also like to take this opportunity to thank Mr. Aboagye and Master Adu Poku. Both of these individuals have given excellent guidance and support throughout my work.

ACRONYMS

Selected acronyms used in this thesis are defined below.

2-1) - 2- Dimensional

I/O Network - Input /Output Network

CPU - Central Processing Unit

WANs - Wide Area Networks

VCT - Virtual Cut Through

DOR - Dimension Order Routing

S-XY - Surrounding XY

SH-XY - Surrounding Horizontal XY

SV-XY - Surrounding Vertical XY

OSI - Open System Interconnect

FIFO - First-In-First-Out

GNS-3 - General Network Simulator 3

NS 2 - Network Simulator 2

OPNET - Optimized Network Engineering Tool

- Number of Nodes in a Row

N * N -föfiTfi%er ofNodes in a Row

- Reliability ofNodes

N-XY - Normal XY

IVAL — Improved Valient Routing RI and R2 — Virtual Channel 1 and Virtual Channel 2

TCP — Transmission Control Protocol

VolP — Voice Over Internet Protocol

IPV6 — Internet Protocol Version 6

IBM's Blue Gene/ L International Business Machines Blue Gene/L

IBM's Blue Gene/ P — International Business Machines Blue Gene/P

Cisco IOS — Cisco Internetwork Operating System

MAC OS X — Macintosh Operating System X

OSPF V3 — Open Shortest Path First Version 3

MPLS Multiprotocol Label Switching

OF

LIST TABLES

OF

LIST FIGURES

OF

TABLE CONTENTS

CHAPTER 1

INTRODUCTION

1.0 INTRODUCTION

An interconnection network is a programmable system that enables fast data communication between components of a digital system. The network is programmable in the sense that it enables different connections at different points in time. The network is a system because it is composed of many components: buffers, channels, switches, and controls that work together to deliver data (Dally & Towles, 2004).

The area of interconnection networks ranges enormously, and so do their applications. Interconnection networks can be viewed in two different ways. First, by taking a bigger look, the entire Internet can be viewed as an interconnection network with end-users and servers as the nodes, and the links, routers, and switches as the fabric connecting them. At a more microscopic level, interconnection networks are used in a variety of applications such as switch and router fabrics, processor - memory interconnect, and on-chip network to name a few. Though these two examples vary greatly in both the size of the communication fabric and the numbers of nodes within the network, both are solid example of current applications of interconnection networks. Interconnection networks enablefast datæynsmication between components of a digital system.

The design of an interconnection network has three aspects, they are; the topology, the flow control mechanism employed for the network and the routing algorithm used. Topology is usually the schematic description of the arrangement of a network, including its nodes and connecting lines.

The topology of a network is selected to exploit the characteristics of the available packaging technology to meet requirements such as bandwidth, latency, etc. of the application, at a minimum cost.

Flow control is the management of data flow between computers or devices or between nodes in a network so that the data can be handled at an efficient pace.

The function of the routing algorithm is to select a path to route a packet from its source to its destination. There are various types of interconnection networks; some of these are; fully connected network, mesh network, ring network, torus network, tree interconnection network, etc.

Central to this thesis, is the torus network (i.e. 2-Dimensional Torus Network) and the routing algorithms used to route data in the network. This thesis focuses on improving the reliability and performance of torus networks which invariably improves throughput in the network by considering the routing algorithms used in this network. Torus network is an interconnection network that belongs to the family of k-ary ncubes and is one of the most popular networks in the last generation of multicomputer (Kessler and Schwarzmeier, 1993). According to (Seitz, 1985) torus has appeared as an alternative to other networks, like hypercube. Due to its modularity, it can be expanded simply by adding nodes and channels without any change to the existing node structure. Torus networks can also be thought of as rings of rings in multiple dimensions. A kary- I-cube is simply a ring network or I-dimensional torus of k nodes. All nodes in torus håvs_thesame n*æof---neighbours, being a direct network, each of these n nodes serves as the input terminal, output terminal and switching node of the network at the same time.

Each node in the network is assigned an n-digit radix k address and is connected by a pair of channels one in each direction to all nodes with addresses that differ by plus or minus one in exactly one address digit. Torus is regular and is also edge symmetric which helps to improve load balance across all the channels.

A torus may be unidirectional, with channels in only one direction in each dimension or bidirectional with channels in both directions between connected nodes. A Torus network is an improved version of basic mesh network. A simple torus network is a mesh in which the heads of the columns are connected to the tails of the columns and the left sides of the rows are connected to the right sides of the rows. Torus network has better path diversity than mesh network, and it also has more minimal routes (Rantala et al, 2006). Figure 1.1 is a 2-D torus network with 16 nodes. Torus network is use for connecting nodes in systems like parallel computers. Supercomputers like IBM's Blue Gene/L and Blue Gene/P use torus network.

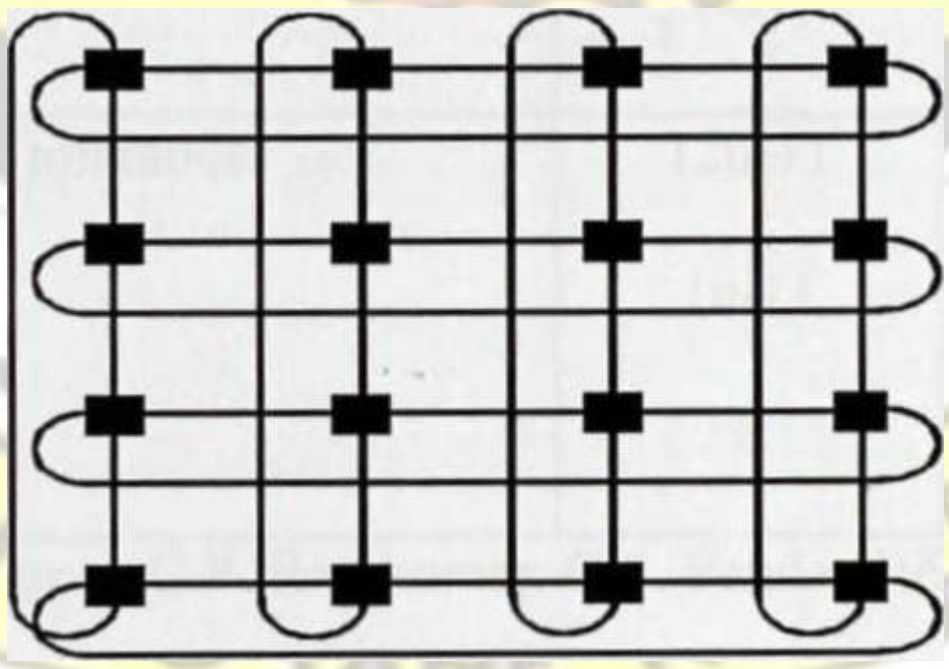


Figure 1.1 Torus Network

Söüöce: Rantala et al, (2006)

Table 1.1

Properties of k-ary n-cube Interconnection Network (Torus Network)

No. of nodes		
No. of edges	nkn nkn 2	Ifk 0 2 Ifk=2 Ifk= 1
Degree	2n n	Ifk 0 2 Ifk=2 Ifk= 1
Diameter	kn 2 (k-l)n 2	If k is even If k is odd
Node fault tolerance	(211-1) (n-l)	Ifk 0 2 Ifk=2 Ifk= 1

Source: C.P. Ravikumar, C.S. Panda (1997)

Table 1. Iconiains the properties of torus networks, n is the dimension of the network and k is the number of nodes along a dimension. For a 2-D network with four nodes in each dimension, the total number of nodes will be sixteen nodes.

Routing is the process by which the network determines what path(s) the messages should take through the network (Vasseur et al, 2011). Along the way, at least one intermediate node typically is encountered. Routing within a network is done by a routing algorithm. A routing algorithm is the logic that is used to decide for each incoming packet which output link the packet should be transmitted on (Pinkston,

2006). A routing algorithm maps a source-destination pair to a path through the network from the source to the destination. Routing algorithms use routing metric(s) in the making decisions during routing. A metric is a standard of measurement, such as path bandwidth, that is used by routing algorithms to determine the optimal path to a destination (A. Balchunas, 2007). There are two main classes of routing algorithms; they are; nonadaptive or static routing algorithms and adaptive routing algorithms.

Nonadaptive or static algorithms select the path using only the identity of the source and destination nodes.

In static routing, routing decisions are pre-determined and not based on measurements of the current network topology and traffic load. Static routing algorithm is simple and fast since there is no flexibility in routing packets. Adaptive algorithms also base routing decisions on the current state of the network when network topology and/or traffic load changes. Adaptive routing may re-determine routes periodically. It usually requires additional network resource.

Several routing algorithms have been proposed for routing in torus networks. The Clue algorithm, Turn model and X-Y routing to name a few are all routing algorithms that have been proposed for routing in the torus network. The focus of this research is to improve pe ormance an%g.liahlllty-l•n torus networks which intend improves network throughput by considering the routing algorithm used in the network.

1.1 STATEMENT OF THE PROBLEM

A routing algorithm determines the sequence of channels for a packet to traverse from the source to the destination (Xiang and Luo, 2012). Network throughput is affected by

algorithms which have low performance and reliability especially when packets are easily lost. Routing algorithms that do not perform well may lead to network delay and

packet loss. Several routing algorithms proposed for torus networks have or may cause the network to have the following short falls;

- ✓ Poor performance and low network reliability.
- ✓ Low network throughput.

Although the routing algorithms for torus network may have other vulnerabilities, the scope of this thesis will be limited to the above mentioned problems.

1.2 AIMS AND OBJECTIVES

This thesis is to find out answers to the problems raised in the scope of the problem statement mentioned above by improving network reliability and performance which intend improves network throughput.

The following are the specific targets of this work;

- ✓ To improve network reliability.
- To improve performance.
- ✓ To propose conditions that satisfies the reliability of packet delivery between nodes to increase network throughput.

1.3 SIGNIFICANCE OEN4E-STUDY

Attaining the aims and objectives of the research as mentioned above will go a long way to improve routing in torus networks. Communication between interconnected systems such as multiprocessors, chips, etc. are built using torus networks and therefore an improved routing within these systems will greatly enhance the capability and efficiency of these systems.

1.4 DESCRIPTION OF CHAPTERS

This thesis is divided into five main chapters. This subsection provides an overview of each chapter as presented in the thesis.

Chapter one gives a brief background and introduction of the topic. It also talks about torus networks and routing algorithms. The problem statement, aims and objectives and significance of the study are all included in chapter one.

Chapter two deals with literature review. The routing algorithms proposed in literature are critically analyzed and discussed and the relevance of the intended research established by pointing out the loop holes in the existing algorithms proposed by other researchers.

Chapter three focuses on the methods to be used in conducting the research. It sets out the type of research to be conducted as well as the tools for testing and analyzing the results of the research. Chapter four presents and discusses the results on the study.

Chapter five concludes the work based on the findings of the research and suggests a number of recommendations. Suggestions for future work are also included in this section.

CHAPTER 2

LITERATURE REVIEW

2.0 OVERVIEW

Today, interconnection networks are used in almost all digital systems that have two or more components to connect. Torus network which is a type of interconnection network has therefore become very important in the development of digital and/or computer systems. The use of torus networks in devices and its associated routing algorithms used for routing packets cannot therefore be overemphasized. Torus network is therefore attractive because of the following reason; at low dimensions, torus networks have uniformly short wires allowing high-speed operation without repeaters.

The routing method employed by a network determines the path taken by a packet from a source node to a destination node. Networks with high path diversity offer many alternative paths between a source and destination. A good routing algorithm makes its route selection in a manner that exploits locality to provide low latency and high throughput on benign traffic (Singh, 2005). Routing algorithms are generally categorized into two groups, they are; oblivious routing algorithms (i.e. static/ nonadaptive routing algorithms) and adaptive routing algorithms.

Oblivious algorithms select a path from a source to a destination using only the identity of the source and destination nodes. In other words, the routing decision is made "oblivious" of the state of the network (Singh, 2005).In static routing, routing decisions are pre-determined and not based on measurements of the current network topology and traffic load. Adaptive algorithms also base routing decisions on the current state of the network when network topology and/or traffic load changes.

Problems of deadlock, livelock and packet loss persist in routing algorithms that are used in torus network and these problems could lead to poor network performance and even network failure.

Presently, research on routing algorithms for torus networks aims to reduce or completely eliminate packet loss, deadlock, starvation, livelock and increase network throughput by using adaptive routing algorithms to route packets adaptively.

The focus of this chapter is to compare the various routing algorithms for torus networks by comparing its strength and weaknesses whiles bringing to light the purpose of this research.

2.1 INTERCONNECTION NETWORKS

An •interconnection network is a programmable system that enables fast data communication between components of a digital system. The network is programmable in the sense that it enables different connections at different points in time. The network is a system because it is composed of many components: buffers, channels, switches, and controls that work together to deliver data (Dally &Towles, 2004).There are different types of interconnection networks; some of these are; hypercube, fully connected network; nesh network, ring network, torus network, tree interconnection network, etc.

Interconnection network can be viewed both microscopically and macroscopically, that is, interconnection network is use to connect individual devices together into a community of communicating devices, where the term device is generically used to signify anything from a component or set of components within a computer to a single computer to a system of computers. Interconnection networks are used in most digital/computer systems.

The performance of most digital/computer systems today is limited by their communication or interconnection, not by their logic or memory. It is therefore important that the underlying interconnection network perform efficiently to improve the efficacy of the entire system. In addition to providing external connectivity, interconnection networks are used to connect sensors and actuators to processors in control systems, host and disk nodes in I/O networks and on-chip cores in chip multiprocessors. In a computer system, the interconnection network between processor and memory determines key performance factors such as the memory latency and memory bandwidth. Again, the performance of the interconnection network in a communication switch largely determines the capacity of the switch.

The performance of an interconnection network can be measured by using a set of metrics employed by the routing algorithm used by the network. To meet the performance specifications of a particular application, the topology, routing algorithm, and flow control of the interconnection network must be considered.

2.2 UNDELIVERED DATA

In every network, the number of resources is finite. This creates a situation where packets compete for resources which results in problems like deadlock, livelock and starvation. In a network certain assumptions are made under which communication conflicts are considered. This assumption is called "fairness" assumptions. The "fairness" assumptions under which communication conflicts are considered are as follows;

- ✓ Routing is distributed and deterministic.
- ✓ A node cannot send a packet to itself.

- ✓ A packet that reached its destination is always consumed and ejected in finite time by this node.
- ✓ Physical channels are bidirectional full-duplex and can be divided into virtual channels.
- ✓ Routing and arbitration unit in router can only process one packet header at a time.
- ✓ If incoming packets contend for resources, it is allocated in a round robin way. ✓ If a packet acquires the routing and arbitration unit, but it cannot be routed since the required output channels are busy, it waits in the corresponding input buffer until its next round.
- ✓ Virtual channels are assigned the physical channel cyclically only if they are ready to route a flit (demand-slotted round robin).
- ✓ Once a buffer accepts the first flit of a packet, it must accept the remaining flits of the same packet before accepting any flits from another packet.

With the high path diversity and the large number of routing decisions within torus networks, the issues of deadlock, livelock and starvation are of great concern and must be considered within a specific routing algorithm.

Deadlock occurs when a set of agents holding resources are waiting on another set of resources such that a cycle of waiting agents is formed (Dally et al, 2004).When data reaches a node or link and it cannot traverse through, it stops its progress, and is usually dropped (Duato et al, 2003). Deadlock is usually an issue to be considered within routing algorithms because of finite number of buffers and random failures. Deadlock arises due to the finiteness of network resources, the probability of its occurrence increases with increased network traffic and decreased availability of network resources.

For the network to function properly, the routing algorithm must guard against this anomaly which can occur in various forms such as routing deadlock, request-reply or protocol deadlock, and fault-induced or reconfiguration deadlock. At the same time, for the network to provide the highest possible performance, the routing algorithm must be efficient allowing as many routing options to packets as there are paths provided by the topology, in the best case.

Most networks are designed to avoid deadlock, but deadlocks do occur. It is also possible to recover from deadlock by detecting and breaking cyclic wait-for relationships. Duato et al, 2003 suggested that the problem of deadlock can be subdued by implementing policies such as deadlock prevention, avoidance and recovery techniques.

Livelock, as it is defined within (Dally et al, 2004), occurs when a packet is not able to make progress in the network and is never delivered to its destination. Unlike deadlock, though, a livelock packet continues to move through the network. Livelock tends to occur most frequently when data is allowed to be misrouted, or routed non-minimally (Duato et al, 2003).

More specifically, livelock becomes a reality when data is forever in transit towards its destination, and continues to move within the network, but does not reach the destination within finite time.

Starvation occurs when a node with a packet is never allowed to inject it into the network or a packet already in the network, is permanently stopped if traffic is intense and the resources requested by it are always granted to other packets also requesting them.

2.2.1 HANDLING UNDELIVERED DATA

Deadlock is the most difficult problem to solve among these three problems. A dead!ock has an adverse performance effect that offsets the advantages of resource sharing and processing concurrency (Ling et al, 2006). Deadlock can be handled in three ways: deadlock prevention, deadlock avoidance and deadlock recovery. It is a long-held consensus that both the deadlock prevention and deadlock avoidance strategies are conservative and less feasible in handling the deadlock problem in general, whereas the deadlock detection/resolution strategy (deadlock recovery) is widely accepted as an optimistic and feasible solution to the deadlock problem.

In deadlock prevention resources are granted to packets in such a way that a request can never lead to a deadlock. It is a very conservative approach and may lead to very low resource utilization. Prevention averts deadlock by providing sufficient in-process buffer spaces (Venkatesh and Smith, 2003).

Deadlock avoidance is a less conservative approach. In deadlock avoidance resources are requested only when they are really needed. That is, resources are requested as a packet advances through the network so that the resulting global state is deadlock-free. A coinmon technique used in deadlock avoidance is to establish an ordering between resources and granting —to—each packet in descending order. Most existing multieomputers use deterministic routing for deadlock avoidance, which is based on the concept of virtual channels (Dally and Seitz, 1987).

Deadlock recovery can be used only if deadlocks are rare and the results of aborting packet transmissions can be tolerated. In deadlock recovery, deadlock is allowed and resources are granted to packets without any check. When a deadlock is detected, some resources that were granted to other packets are freed and granted to other packets.

Starvation can be solved easily if a correct and fair resource allocation scheme is used. A simple demand-slotted round-robin scheme can be used to handle starvation. If some packets must have higher priority than the others, then some part of the bandwidth must be reserved and guaranteed for lower-priority packets. Virtual channels and buffers can be used to do that.

The simplest way to handle livelock problem is to use greedy routing (Venkatesh and Smith, 2003). If no greedy routing is used, then the number of misrouting operation must be bounded.

In other words, the simplest way of guarding against livelock is to restrict routing such that only minimal paths from sources to destinations are allowed or less restrictively, only a limited number of non-minimal hops (Pinkston et al, 2006). The strictest form has the added benefit of consuming the minimal amount of network bandwidth, but it prevents packets from being able to use alternative non-minimal paths in case of contention or faults along the shortest or minimal paths.

2.3 ALGORITHM TYPES

There are two main classes of routing algorithms; they are; nonadaptive / static routing algorithms an adaptive ro—galgonthms.

Nonadaptive/ static routing algorithms select the path using only the identity of the source and destination nodes. In static routing, routing decisions are pre-determined and not based on measurements of the current network topology and traffic load. Static routing algorithm is simple and fast since there is no flexibility in routing packets.

Adaptive routing algorithms also base routing decisions on the current state of the network when network topology and/or traffic load changes. Adaptive routing may redetermine routes periodically. It usually requires additional network resource.

L 1
BHA

KWAME NKRUMAH
14 'N'VERS'TY OF SCtENCE e TECHNOLOro
KUMAS

Routing algorithms can also be classified as follows;

Static and Dynamic

Single-Path and Multipath

✓
Flat and Hierarchical

✓
Host-Intelligent and Router-Intelligent

✓
Intra-Domain and Inter-Domain

✓
Link-State and Distance Vector

2.3.1 Static and Dynamic Routing Algorithms

Static routing algorithms are hardly algorithms at all, but are table mappings established by the network administrator before the beginning of routing. These mappings do not change unless the network administrator alters them.

Algorithms that use static routes are simple to design and work well in environments where network traffic is relatively predictable and where network design is relatively simple. Because static routing systems cannot react to network changes, they generally are considered unsuitable for today's large, constantly changing networks.

Most of the dominant routing algorithms today are dynamic routing algorithms, which adjust to changing netw>gircumstances by analyzing incoming routing update messages. If the message indicates that a network change has occurred, the routing software recalculates routes and sends out new routing update messages. These messages permeate the network, stimulating routers to rerun their algorithms and

change their routing tables accordingly. Dynamic routing algorithms can be supplemented with static routes where appropriate.

2.3.2 Single-Path and Multipath Routing Algorithms

Some sophisticated routing protocols support multiple paths to the same destination. Unlike single-path routing algorithms, these multipath routing algorithms permit traffic multiplexing over multiple lines. The advantages of multipath routing algorithms are obvious: that is, they can provide substantially better throughput and reliability. This is generally called load sharing.

2.3.3 Flat and Hierarchical Routing Algorithms

Some routing algorithms operate in a flat space, while others use routing hierarchies. In a flat routing system, the routers are peers of all others. In a hierarchical routing system, some routers form what amounts to a routing backbone. Packets from non-backbone routers travel to the backbone routers, where they are sent through the backbone until they reach the general area of the destination. At this point, they travel from the last backbone router through one or more non-backbone routers to the final destination. Routing systems often designate logical groups of nodes, called domains, autonomous systems, or areas.

In hierarchical systems, some routers in a domain can communicate with routers in other domains, while otheæg_communicate only with routers within their domain. In very large networks, additional hierarchical levels may exist, with routers at the highest hierarchical level forming the routing backbone. The primary advantage of hierarchical

routing is that it mimics the organization of most companies and therefore supports their traffic patterns well. Most network communication occurs within small company groups (domains). Because intradomain routers need to know only about other routers within their domain, their routing algorithms can be simplified, and depending on the routing algorithm being used, routing update traffic can be reduced accordingly.

2.3.4 Host-Intelligent and Router-Intelligent Routing Algorithms

Some routing algorithms assume that the source end node will determine the entire route. This is usually referred to as source routing. In source-routing systems, routers merely act as store-and-forward devices, mindlessly sending the packet to the next stop. Other algorithms assume that hosts know nothing about routes. In these algorithms, routers determine the path through the internetwork based on their own calculations. In the first system, the hosts have the routing intelligence. In the latter system, routers have the routing intelligence.

2.3.5 Intra-Domain and Inter-Domain Routing Algorithms

Some routing algorithms work only within domains; others work within and between domains. The nature of these two algorithm types is different. It stands to reason, therefore, that an optimal intra-domain routing algorithm would not necessarily be an optimal inter-domain routing algorithm.

2.3.6 Link-State and Distance Vector

Link-state algorithms (also known as shortest path first algorithms) flood routing information to all nodes in—ternetwork. Each router, however, sends only the

portion of the routing table that describes the state of its own links. In link-state algorithms, each router builds a picture of the entire network in its routing tables. Distance vector algorithms (also known as Bellman-Ford algorithms) call for each router to send all or some portion of its routing table, but only to its neighbours. In essence, link-state algorithms send small updates everywhere, while distance vector algorithms send larger updates only to neighbouring routers.

Distance vector algorithms know only about their neighbours. Because they converge more quickly, link-state algorithms are somewhat less prone to routing loops than distance vector algorithms.

On the other hand, link-state algorithms require more CPU power and memory than distance vector algorithms. Link-state algorithms, therefore, can be more expensive to implement and support. Link-state algorithms are generally more scalable than distance vector algorithms.

2.4 SWITCHING/ROUTING TECHNIQUES

Switching technique defines how connections are established in the network. Ideally, connections between network resources are established or "switched in" only for as long as they are actually needed and exactly at the point that they are ready and needed to be used, considering both time and space. This allows efficient use of available network bandwidth by ensuring efficient traffic flows and minimal latency.

In a network, connections at each node along the topological path allowed by the routing algorithm can be established in three basic ways: prior to packet arrival using circuit

switching, upon receipt of the entire packet using store-and-forward packet switching, or upon-receipt of only portions of the packet with unit size no smaller than that of the packet header using cut-through packet switching.

Circuit switching establishes a circuit such that network bandwidth is allocated for packet transmissions along an entire source-destination path.

Packet switching enables network bandwidth to be shared and used more efficiently when packets are transmitted. There are two basic approaches to routing packets, based on what a switch does with a packet as its flits begins to arrive; they are;

- ✓ Store-and-forward switching
- ✓ Cut-through switching

These two switching techniques allow network link bandwidth to be multiplexed on packet-sized or smaller units of information. This better enables bandwidth sharing by packets originating from different sources.

Store-and-forward packet switching establishes connections such that a packet is forwarded to the next hop in sequence along its source-destination path only after the entire packet is first stored (staged) at the receiving switch.

As packets are completely stored at every switch before being transmitted, links are completely decoupled, allowing full link bandwidth utilization even if links have very different bandwidths. This property is very important in WANs, but the price to pay is packet latency. The total delay is multiplicative with the number of hops.

Cut-through packet switching establishes connections such that a packet can "cut through" switches in a pipelined manner once the header portion of the packet or equivalent amount of payload trailing the header is stored at receiving switches. That is, the rest of the packet need not arrive before switching in the granted resources. This allows delay to G@additive with the number of hops rather than multiplicative to reduce total packet latency.

Cut-through comes in two varieties; they are; Virtual Cut-through (VCT) and Wormhole routing.

The concept of virtual cut-through was originally proposed in the interconnection network of computer system to reduce packet latency (Shin and Daniel, 1996). In VCT

switching, one packet can be forwarded to the next stage before its entirety is received by the current switch. Therefore, VCT switching reduces the store-and-forward delays. However, when the next stage switch is not available, the entire packet still needs to be stored in the buffers of the current switch. VCT switching is proposed to reduce the packet delays at each routing stage.

Wormhole routing was developed from the earlier idea of cut through switching (Kermani and Kleinrock, 1979) and was first introduced in hypercube communication chips ((Seitz et al, 1985). In wormhole routing; a message is broken into flits for transmission and flow control.

The header flit containing routing information governs the route, and the remaining data flits follow in a pipelined fashion. If the header is blocked, the data flits are also blocked. Wormhole routing has also promoted the use of high diameter networks, like the torus, as it makes latency independent of the message distance in the absence of blocking (Seitz, 1985).Since wormhole switches need to be capable of storing only a small portion of a packet, packets that block in the network may span several switches. This can cause other packets to block on the links they occupy, leading to premature network saturation and reduced effective bandwidth unless some centralized buffer is used within the switch to store them. Premature network saturation caused by wormhole switching can be mitigated by allowing several packets to share the physical bandwidth of a tilik simultaneousl via e-multiplexed switching at the flit level. This requires physical links to have a set of virtual channels at each end, into which packets are switched.

The main difference between the two varieties is the size of the unit of information on which flow control is applied and consequently, the buffer requirements at switches. Virtual cut-through switching implements flow control at the packet level, whereas

wormhole switching implements it on flow units or flits, which are smaller than the maximum packet size but usually at least as large as the packet header.

As chips can implement relatively large buffers in current technology, virtual cutthrough is the more commonly used switching technique. However, wormhole switching may still be preferred in network on-chips designed to minimize silicon resources.

2.5 ROUTING ALGORITHMS IN TORUS NETWORK

Several routing algorithms have been proposed in literature for torus networks.

2.5.1 Dimension Order Routing

Dimension order routing (DOR) is a typical minimal turn algorithm. The algorithm determines to what direction packets are routed during every stage of the routing (Dally and Towels, 2004).

Dimension order routing (DOR) is a type of algorithmic routing for meshes and torus. It exploits the orthogonality of each dimension in these topologies and works by nullifying the offset of a message from its destination, in each dimension in turns. DOR for meshes is deadlock free as no cycles can be created, it is also livelock — free and simple in implementation but in torus networks with extra wrap-around links, deadlock can occur within a dimension.

2.5.2 xy Routing

XY routing is a dimension order routing which routes packets first in x or horizontal direction to the correct column and then in y or vertical direction to the receiver. XY routing suits well on a network using mesh or torus topology. Addresses of the routers are their x and y coordinates.

XY routing never runs into deadlock or livelock (Dehyadgari et al, 2005). Figure 2.1 illustrates how XY routing routes packet from a source node A to a destination node B. There are some problems in the traditional XY routing. The traffic does not extend regularly over the whole network because the algorithm causes the biggest load in the middle of the network. Dimension Order Routing (DOR) is called XY-routing algorithm for 2D-meshes. However, XY routing tends to send packets toward the centre of the mesh when the contention is high (De Micheli and Benini, 2006).

Hence, the central part of the mesh becomes overloaded. It leads to early network saturation and performance degradation. XY-routing performs well just for uniformly distributed traffic. XY routing never runs into deadlock or livelock (Dehyadgari et al, 2005).

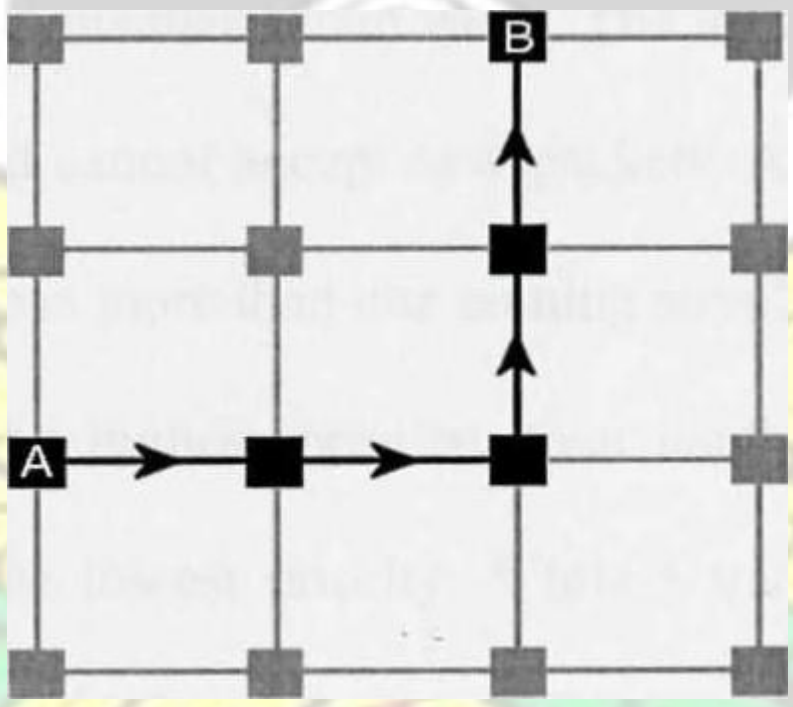


Figure 2.1 XY Routing

Source: Rantala et al, (2006)

2.5.3 Pseudo Adaptive XY Routing

Pseudo adaptive xy routing works in deterministic or adaptive mode depending on the state of the network. The algorithm works in deterministic mode when the network is not or only slightly congested. When network becomes blocked, the algorithm switches to the adaptive mode and starts to search routes that are not congested. Pseudo adaptive xy routing works on network which consists of routers, wires and IP-blocks. In pseudo

adaptive xy routing every router has five bidirectional ports: north, south, east, west and local. Local port connects router to its local core while the other ports are connected to neighbouring routers. Each port has a small temporary storage buffer and a 2-bit status identifier called quantized load value. The identifier tells the other routers if the router is congested and cannot accept new packets. A router assigns priorities to incoming packets when there is more than one coming simultaneously.

Packets from north have the highest priority, then south, east and at last packets incoming from west have the lowest priority. While a traditional XY routing causes network loads more in the middle of the network than to lateral areas, the pseudo adaptive algorithm divides the traffic more equally over the whole network (Dehyadgari et al, 2005).

2.5.4 Surrounding XY Routing

Surrounding xy routing (S-XY) has three different routing modes. N-XY (Normal XY) mode works just like the basic XY routing. It routes packets first along x-axis and then along y-axis. Routing stays on NXY mode as long as network is not blocked and routing does not meet inactive routers. SH-XY (Surround horizontal XY) mode is used when the router's left or right neighbour is deactivated. The third mode SV-XY (Surround vertical XY) is used when the upper or lower neighbour of the router is inactive. The SH-XY mode routes packets to the correct column on the grounds of coordinates of the destination. The algorithm bypasses packets around the inactive routers along the shortest possible path. The situation is a little bit different in the SVXY mode because the packets are already in the right column. Packets can be routed to left or right. It is a method that supports communication between modules which are dynamically placed on a device (Bobda et al, 2005).

2.5.5 Turn Model

The turn model as proposed by (Glass and Ni, 1994) was also designed for partially deadlock-free networks. What makes this algorithm unique is that it achieves partial adaptivity without adding any virtual or physical channel to the network. It prevents the minimum number of turns to prevent cyclic channel dependencies. The turn model algorithm for routing packet is based on the following four steps;

- ✓ Identify all possible turns from one direction to another.
- ✓ Identify all possible cycles that the turns can form.

v/ Prohibit the minimum number of turns so that at least one turn is prohibited in each cycle.

Incorporæ_manyturns as possible without reintroducing cycles (Glass and Ni, 1992).

There are variations in the turn model which includes the following,

2.5.6 West-First Routing

A west-first routing algorithm prevents all turns to west. So the packets going to west must be first transmitted as far to west as necessary. Routing packets to west is not possible later.

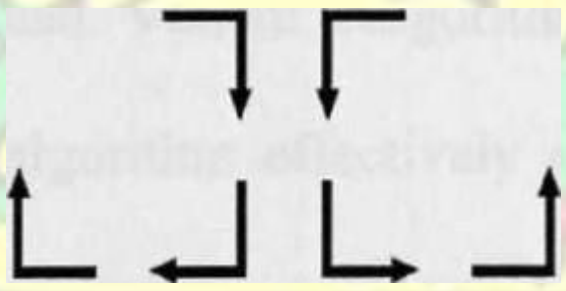
2.5.7 North-Last Routing

Turns away from north are not possible in a north-last routing algorithm. Thus the packets which need to be routed to north must be transferred there at last.

2.5.8 Negative-First Routing

Negative-first routing algorithm allows all other turns except turns from positive direction to negative direction. Packet routings to negative directions must be done before anything else (Kariniemi and Nurmi, 2004).

rar



L ei L -t

North-Last

Figure 2.2 Wgst-First Negative First Routing Algorithms.

Source: Rantala et al, (2006)

Figure 2.2 above illustrates how West-First, North-Last and Negative-First routing algorithms work.

2.5.9 Valiant's Algorithm

This algorithm was pioneered by L. Valiant in 1981. He proposed that routing should be split into two phases. Valiant's random algorithm is a partly stochastic routing algorithm. One main problem in the oblivious routing algorithms is that they affect an irregular load on the network. Valiant's algorithm can balance the load for any traffic pattern to any connected topology. The load is especially high in the middle areas of the network. Valiant's random algorithm equalizes traffic load on networks that have good path diversity. First the algorithm randomly picks one intermediate node and routes packets to it. Then the packets are simply routed to their destination. Thus the delivery of packet is divided into two phases. Both phases must have an algorithm that balances the load for uniform traffic. Routing from beginning to the intermediate node and then to the destination are done using some of oblivious algorithms. Valiant's algorithm gives a good worst case performance on k-ary n cubes (torus).

In Valiant's algorithm, good performance on worst-case traffic patterns comes at expense of locality and overhead. Valiant's algorithm performs very badly for nearest neighbour traffics. Valiant's algorithm effectively equalizes network's load over the whole network regardless of network's topology (Dally and Towles, 2004).

2.5.10 Improved Valiant Routing (IVAL)

Improved valiant's routing is an improved version of the oblivious Valiant's algorithm. It is a bit similar to turn around routing. On the algorithms first stage, packets are routed to a randomly chosen point between the sender and the receiver by using an oblivious dimension order routing. The second stage of the algorithm works almost equally, but this time the dimensions of the network are gone through in reversed order.

Deadlocks are avoided in IVAL routing by dividing router's channels to virtual channels. Full deadlock avoidance requires a total of four virtual channels per one physical channel.

2.5.11 Clue Algorithm

The Clue algorithm is a fully adaptive routing algorithm in virtual cut through. This algorithm is use for torus network. Two virtual channels, RI and R2 are used to provide deadlock free routing in a virtual cut through switch with n-dimensional torus (Xiang and Luo, 2012). Three rules are used to illustrate the algorithm;

Rule I: Virtual channel RI is fully adaptive. A packet can request RI channels at any time.

Rule 2: A packet need not traverse any wraparound link from the current node to destination which is delivered in a mesh sub network. It could request a R2 channel of the mesh sub-network link. However it must follow a deadlock free routing algorithm for meshes such as negative- first or dimension order routing when being delivered via R2.

Rule 3: The next hop of a packet is to traverse a wraparound link of dimension d, and d is the lowest of the dimension in which the packet need traverse the wraparound links from the current link to theaestifiätiöiil The packet can request the R2 channel of that wraparound link.

2.6 VIRTUAL CHANNEL

A Virtual channel consists of a buffer that can hold one or more flits of a packet and associated state information (Dally, 1992). Several virtual channels may share the bandwidth of a single physical channel. Virtual channels decouple the allocation of buffers from allocation ofchannels by providing multiple buffers for each channel.

The virtual channel technique can be used to avoid new physical channel by splitting one physical channel into several virtual channels.

2.7 CONCLUSION

This research is conducted in an attempt to improve nemork performance and reliability in torus networks intends increases throughput. An improvement in netvvork throughput will go a long way in improving packet transmission in torus networks.

CHAPTER 3

RESEARCH METHODOLOGY

3.0 INTRODUCTION

This chapter describes the approach adopted for the intended study. This chapter provides some background information on routing, the design and operation of routing algorithms whiles providing a general guideline for achieving the aims and objective of this research. The chapter provides specific descriptions of methods; techniques and tools employed in the research. The methodologies therefore employed in this thesis are guided by the specific problem the research is seeking to address.

3.1 ROUTING COMPONENTS

Routing is often contrasted with bridging, which might seem to accomplish precisely the same thing to the casual observer. The primary difference between the two is that bridging occurs at Layer 2 (the link layer) of the OSI reference model, whereas routing occurs at Layer 3 (the network layer). This distinction provides routing and bridging with different information to use in the process of moving information from source to destination, so the two functions accomplish their tasks in different ways.

Routing involvesÄ'0 basic aetiTtfiéS7ifrey are:

- ✓ Determining Optimal Routing Paths (Path Determination)
- ✓ Transporting Information (typically called packets) through an internetwork.

Although packet switching is relatively straightforward, path determination can be very complex.

3.1.1 SOURCE ROUTING

In source routing the information about the whole path from the source to the destination is pre-computed and provided in packet header as opposed to distributed routing, where

packet header contains destination address only and the path is computed dynamically by the participation of nodes on the path (Dally and Towles, 2004). In source routing, all routing decisions are made inside the source node before transmitting any packet in the network. For this purpose, each source node contains lists or tables that contain complete route information to reach all other nodes (destination) in the network. Instead of storing tables in source, it is also possible to add extra logic or software in the source resources that implements any adaptive routing algorithm and dynamically computes paths for source routing (Mubeen, 2009).In order to route a packet through the network using source routing, a sender node consults its routing table to get a complete path to the required destination. The packet must follow the path while traversing through the network towards its destination. Each node that receives this packet forwards it to the destined node.

3.1.2 PATH DETERMINATION

Path determination involves reviewing all paths to a destination and choosing the optimal route. The optimal route is determined by putting information into the routing table. The routing algorithms for the network then use metrics to evaluate what path will be the best for a packet to travel.

3.1.3 ROUTING TABLE

Routing table contains information used by algorithms to select the best route. It can also be said to be a set of rules, often viewed in table format that is used to determine where data packets travelling over a network will be directed (A. Balchunas, 2007). To aid the process of path determination, routing algorithms initialize and maintain routing tables, which contain route information. Routing algorithms fill routing tables with a variety of information called route information which varies depending on the routing algorithm used.

Routing table is either manually built or are automatically created and updated. Manually built routing table contains entries in which an administrator has entered a series of routes, each of which contains information about how to get to a specific destination. This is usually used in static routing. In dynamic routing, devices build and maintain their routing tables automatically by using routing protocols to exchange information about the surrounding network topology. The routing table will contain at least one entry. All IP-enabled devices, including routers use routing tables.

A basic routing table includes the following information:

- ✓ Destination: The IP addrSöfihe packet's final destination.
- ✓ Next hop: The IP address to which the packet is forwarded.
- ✓ Interface: The outgoing network interface the device should use when forwarding the packet to the next hop or final destination.
- ✓ Metric: Assigns a cost to each available route so that the most cost-effective path can be chosen.
- ✓ Routes: This includes directly-attached subnets and indirect subnets that are not attached to the device but can be accessed through one or more hops.

It also includes default routes to use for certain types of traffic or when information is lacking.

Each packet travelling in a network contains information about its origin and destination. When a packet is received, a network device examines the packet and matches it to the routing table entry providing the best match for its destination. The table then provides the device with instructions for sending the packet to the next hop on its route across the network. Routing tables can also contain other information, such as data about the desirability of a path. Routers compare metrics to determine optimal routes, and these metrics differ depending on the design of the routing algorithm used. Routers communicate with one another and maintain their routing tables through the transmission of a variety of messages. The routing update message is one such message that generally consists of all or a portion of a routing table. By analyzing routing updates from all other routers, a router can build a detailed picture of network topology. A link-state advertisement, another example of a message sent between routers, informs other' routers of the state of the sender's links. Link information also can be used to build a complete picture of network topology to enable routers to determine optimal routes to network destinations.

3.2 ROUTING METRICS

Routing algorithms use many different metrics to determine the best route.

Sophisticated routing algorithms can base route selection on multiple metrics, combining them in a single (hybrid) metric. Some routing metrics includes the following:

- ✓ Path length
- ✓ Reliability

Delay

Bandwidth

☑ Load

Communication cost

3.2.1 Path Length

It is the most common routing metric. Path length is the sum of the costs associated with each link traversed. Some routing protocols allow network administrators to assign arbitrary costs to each network link. Other routing protocols define hop count, a metric that specifies the number of passes through internetworking products, such as routers, that a packet must take en route from a source to a destination.

3.2.2 Reliability

In the context of routing algorithms, reliability refers to the dependability (usually described in terms of the bit-error rate) of each network link. Some network links might go down more often than others. After a network fails, certain network links might be repaired more easily or more quickly than other links. Reliability factors of the links can be taken into-account n the assignment ofthe reliability ratings, which are arbitrary ic values usually assigned to network links by network administrators.

3.2.3 Routing Delay

It refers to the length of time required to move a packet from source to destination through the internetwork. Delay depends on many factors, including the bandwidth of

intermediate network links, the port queues at each router along the way, network congestion on all intermediate network links, and the physical distance to be travelled. Routing delay is a common and useful metric because delay is a conglomeration of several important variables.

3.2.4 Bandwidth

It refers to the available traffic capacity of a link. Although bandwidth is a rating of the maximum attainable throughput on a link, routes through links with greater bandwidth do not necessarily provide better routes than routes through slower links. For example, if a faster link is busier, the actual time required to send a packet to the destination could be greater.

3.2.5 Load

-It refers to the degree to which a network resource, such as a router, is busy. Load can be calculated in a variety of ways, including CPU utilization and packets processed per second. Monitoring these parameters on a continual basis can be resource-intensive itself.

3.2.6 Communication Cost

It is another important ortant metric, especially because some companies may not care about

performance as much as they care about operating expenditures. Although line delay may be longer, they will send packets over their own lines rather than through the public lines that cost money for usage time.

3.3 ROUTING ALGORITMS DESIGN

Routing algorithms are used to transport packets through an internetwork. Routing algorithms can be differentiated based on several key characteristics. Various types of routing algorithms exist, and each algorithm has a different impact on network resources. The particular goals of the algorithm designer affect the operation of the resulting routing algorithm. Also, routing algorithms use a variety of metrics that affect calculation of optimal routes.

3.3.1 Design Goals

Routing algorithms often have one or more of the following design goals:

- ✓ Optimality
- ✓ Simplicity and Low Overhead
- ✓ Robustness and Stability
- ✓ Rapid Convergence

  Flexibility

3.3.2 Optimality

Optimality refers to the capäfifififif¯the routing algorithm to select the best route, which depends on the metrics and metric weightings used to make the calculation.

For example, one routing algorithm may use a number of hops and delays, but it may weigh delay more heavily in the calculation. Naturally, routing algorithms must define their metric calculation algorithms strictly.

3.3.3 Simplicity and Low Overhead

Routing algorithms also are designed to be as simple as possible.

In other words, the routing algorithm must offer its functionality efficiently, with a minimum of software and utilization overhead. Efficiency is particularly important when the software implementing the routing algorithm must run on a computer with limited physical resources.

3.3.4 Robustness and Stability

Routing algorithms must be robust, which means that they should perform correctly in the face of unusual or unforeseen circumstances, such as hardware failures, high load conditions, and incorrect implementations. Because routers are located at network junction points, they can cause considerable problems when they fail. The best routing algorithms are often those that have withstood the test of time and that have proven stable under a variety of network conditions.

3.3.5 Rapid Convergence

Convergence is the process of agreement, by all routers, on optimal routes. When a network event causes or become available, routers distribute routing-update messages that permeate networks, stimulating recalculation of optimal routes and eventually causing all routers to agree on these routes. Routing algorithms that converge slowly can cause routing loops or network outages.

3.3.6 Flexibility

Routing algorithms should also be flexible, which means that they should quickly and accurately adapt to a variety of network circumstances. Assume, for example, that a network segment has gone down. As many routing algorithms become aware of the problem, they will quickly select the next-best path for all routes normally using that segment.

Routing algorithms can be programmed to adapt to changes in network bandwidth, router queue size, and network delay, among other variables.

3.4 CORRECTNESS OF A ROUTING ALGORITHM

A distributed routing algorithm needs time in order to converge on the computation of its paths. This convergence time depends on factors such as the actual routing algorithm, network topology, link transmission speed, and router computation speed. For example, in the link state algorithm, convergence time depends on the time needed to detect topology changes, broadcast topology information, and recomputed shortest paths.

When the network state changes, one expects that after a time, the routing algorithm will converge and calculate the appropriate paths based on the new network state.

Assume that the convergence time for a link state algorithm is five seconds. This means that, five seconds after a link failure, the algorithm will calculate shortest paths such that the fäiTed link is not used. In this case, the algorithm is correct because it calculates the expected paths and does so within five seconds.

However, the notion of correctness and convergence is less clear in scenarios where the netwcrk state changes faster than a routing algorithm's convergence time. This type of network scenario typically occurs in dynamic metric routing because link costs can change much more frequently than a routing algorithm's convergence time. Assume that the network topology or link costs of a network changes every second, then the link state algorithm can never catch up to the network changes and can never compute paths that reflect the actual shortest path in the current network.

Rather, the paths that the link state algorithm computes will reflect the shortest paths in an out-dated network state, where the degree of out-datedness depends on the algorithm's convergence time.

In other words, as the network changes, link state tracks the changes and computes paths according to its most up-to-date view of the network, this may be the actual network state sometime in the past.

Also in network scenarios where a routing algorithm does not have an up-to-date view of the current network state, one can still reason about the correctness of a routing algorithm.

A routing algorithm is still correct if it computes the expected path within its convergence time even though rate of change is slower than its convergence time. Since routing algorithm computes paths based on its current view of the network, such algorithms are correct if

- ✓ Their view of the network is consistent with the actual network.
- ✓ The expected paths are computed based on that view.

In situations whéie a routi a gorithm's network view is not consistent with the current network, a correctness criterion must ensure that the routing algorithm does not diverge in its shortest path computation.

That is; the paths that a routing algorithm actually calculates should track the paths that the routing algorithm would calculate given it has the instantaneous network state. This property can be ensured by two criteria:

The algorithm's view of the network must track the actual state of the network.

v/ Correct paths are calculated based on the algorithm's view of the network state.

Routing algorithm might not have an up-to-date network view either because the network state changes too frequently or because the algorithm does not attempt to obtain the exact, moment-to-moment state of the network due to efficiency issues.

Mechanisms such as thresholding and hold-downs allow routing algorithms to reduce routing cost at the expense of less accurate network view.

In practice, this implies that the accuracy with which a dynamic metric routing algorithm tracks physical link costs directly affects the quality of the paths calculated. The closer the tracking, the closer the calculated paths reflect the current state of the network.

3.5 RESEARCH DESIGN

This thesis attempts to address two specific issues that relates to torus network, namely;

- ✓ To improve network reliability.
- ✓ To improvyerformance.
- ✓ To propose conditions11fGfWes the reliability of packet delivery to improve - =network throughput.

To address these three issues, a stepwise approach that involves building a simulator and estimating the reliability through experimentation by way of simulation was used to analyze the torus network to determine the results.

3.6 VIRTUAL CHANNELS IN ROUTING ALGORITHMS

A Virtual channel consists of a buffer that can hold one or more flits of a packet and associated state information (Dally, 1992). Several virtual channels may share the bandwidth of a single physical channel. Virtual channels decouple the allocation of buffers from allocation of channels by providing multiple buffers for each channel. The virtual channel technique can be used to avoid new physical channel by splitting one physical channel into several virtual channels.

The throughput of interconnection network is limited to a fraction typically 20% to 50% of the network's capacity because of coupled resource allocation. Interconnection networks are composed of two types of resources: buffers and channels. Network throughput can be increased by dividing the buffer storage associated with each network channel into several virtual channels (Dally and Seitz, 1987). Each physical channel is associated with several virtual channels rather than a single deep queue. The virtual channels associated with one physical channel are allocated independently but compete with each other for physical bandwidth. Virtual channels decouple buffer resources from transmission resources. This decoupling allows active messages to pass blocked messages using network bandwidth that would otherwise be left idle. Adding virtual channels-rean intercont<Gétwork is analogous to adding lanes to a street netwyK._A network without virtual channels is composed of a one-lane street, in such a network a single blocked packet blocks all following packets. Adding virtual channels to the network is like adding lanes to the street. This allows blocked packets to be passéd. In addition to increasing throughput, virtual channels provide an additional degree of freedom in allocating resources to packets in the network.

This flexibility permits the use of scheduling strategies that reduce the variance of network latency. Adding virtual channels to a network makes more effective use of both physical channel and buffer memory by decoupling their allocation. Virtual channels were first introduced for the purposes of deadlock avoidance (Dally and Seitz, 1987).

3.6.1 FLOW CONTROL

Communication between nodes is performed by sending messages. A message may be broken into one or more packets for transmission. A packet is the smallest unit of

information that contains routing and sequencing information. A packet contains one or more flow control digits or flits. A flit is the smallest unit on which flow control is performed. The flow control protocol of the network determines how resources are allocated and how packet collision over resources are resolved. A resource collision occurs when a packet is unable to proceed because some resource it needs usually a buffer, is held by another packet. Collisions may be resolved by misrouting the packet to a channel other than the one it requires, buffering the packet in a node prior to where the collision occurs. Flow control strategies allocates buffers and channel bandwidth to flits. This is beeause flits haf¯fFÖuting or sequencing information and so the allocation must be done in a manner that keeps the flits associated with a particular packet together. This may be done by associating a set of buffers and some control state together into a virtual channel. A virtual channel is allocated to a packet and buffers of the virtual channel are allocated in a First-in-First-out (FIFO) manner to the flits of that packet.

In a network using virtual channel flow control, flow control is performed at two levels. Virtual channel assignment is made at the packet level whiles physical channel bandwidth is allocated at the flit level. When a packet arrives at a node it is assigned to a virtual channel.

This assignment remains fixed for the duration of the packet. The virtual channels associated with the physical channel arbitrate for physical channel bandwidth on a flit by flit basis.

3.7 SIMULATORS

As a result of the scalability and flexibility offered by simulators as well as the number of nodes required in the experiment, the simulation approach was thought to be more

feasible as opposed to using real life. A number of Network simulators were evaluated for the experiment. These include GNS 3, NS-2 and OPNET

3.7.1 GNS-3 (Graphical Network Simulator-3)

GNS-3 is a graphical simulator that allows users to design complex network topologies. It allows for the creation of simulations and the configuration of devices such as workstations and powerful simulations, GNS 3 is strongly linked with.

- ✓ Dynamic, a CISCO IOS emulator v/ DYiiagen,

  a text$äGdGRtend for Dynamics

- ✓ Qeme, a generic and open source machine emulator.

- ✓ Virtual Box which is free and powerful virtualization software

GNS-3 is an open-source, free program that may be used on multiple operating systems including Windows, LINUX and MAC OS X.

3.7.2 NS-2 (Network Simulator 2)

NS-2 is a discrete event simulator targeted at networking research NS-2 supports the simulation of TCP, routing and multicast protocols over wired and wireless networks. Despite the confidence deposed in NS-2 by researchers, NS-2 still remains a work in progress. NS-2 therefore is not a polished and finished product, but the result of an ongoing effort of research and development.

Particularly, bugs in the software are still being discovered and corrected. Users of NS2 are responsible for verifying for themselves that their simulations are not invalidated by bytes. Similarly, users of NS-2 are entirely responsible for verifying for themselves that their simulations are not invalidated because the model implemented in the simulator is not the model that they were expecting a situation that does not exist with

GNS„3.

3.7.3 OPNET — Optimized Network Engineering Tools

OPNET is a commercially available network engineering simulator marketed by OPNET technologies, INC. It provides performance analysis for computer networks, analysis and design of communication networks, devices, protocols and applications. OPNET alters users to analy -simulated networks to compare the impact of different technJogy designs on end-to-end behaviour. OPNET incorporates a broad suite of protocols and technologies and includes a development environment to enable modeling of all network types and technologies including: VOIP, TCP, OSPF V3,

MPLS, and IPV6.

OPNET provides:

v" One of the fastest event simulation engines among leading industry solutions.

v/ Hundreds of protocol and render device models with source code. ✓ Object-oriented modeling along with key features.

3.8 SINWLATION

Figure 3.1 shows the interface of a performance•based network simulator that was specifically built for simulation for this research. The simulator was built using the java programming language and runs on a browser. It also requires a java run-time environment. The simulator was made to run on a browser so that it can be easily used by anybody who wants to use it. The simulator works by calculating the terminal reliability and estimating performance of the torus network. The simulator uses source routing to route packets from source to destination nodes. In source routing, the information about the whole path from source to destination is pre-computed and provided in the packet header (Flich et al, 2000). The simulator first constructs the torus network and a source and destination pair is selected to transmit the packet(s). During

simulation, the simulator generates random packets from source through other nodes to the destination node, so that it gets the list of all possible path(s) from source to destination without collision. The system then calculates the terminal reliability and finds the list of all shortest possible paths from the source node to the destination node.

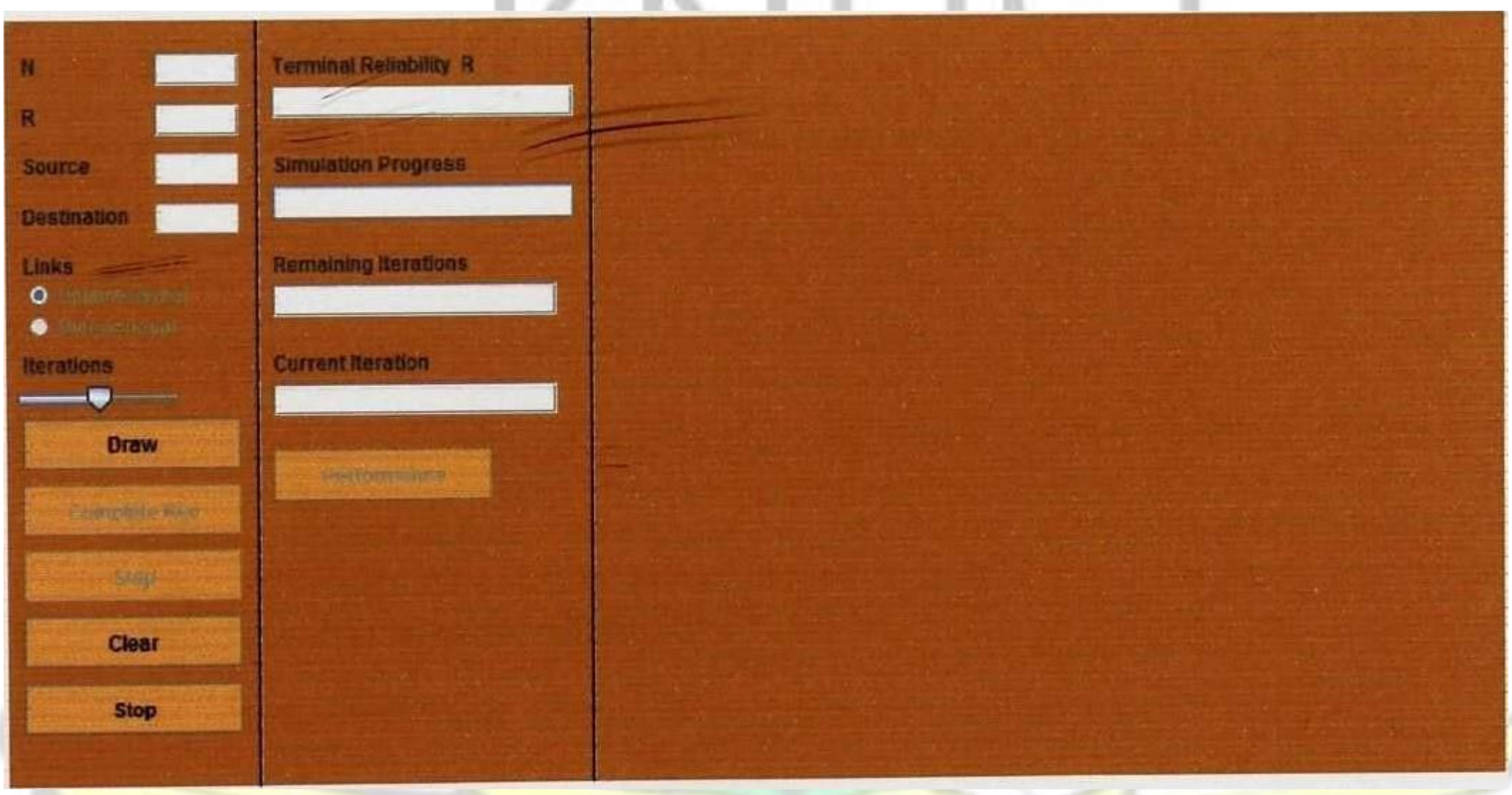


Figure 3.1 Network Simulator

3.8.1 OPERATION OF SIMULATOR

The number of nodes in a row (N) is entered when the simulator is started. The torus network will have total number of nodes as N * N. The value of N should be less than 10. The reliability R of each node is also entered. The value of R should be less than or equal to 1.0. After these values have been entered the user clicks on the "Draw" button to draw the torus network with the specified number of nodes. The nodes are numbered from left to right and top to bottom. The nodes in the network are identified by the numbers that appear against them. In order to calculate terminal reliability, a source and destination pair needs to be specified. The user must therefore enter the source and the destination nodes, by using the number that appears against each node. The link that connects any two nodes in the network must also be selected. The links can be unidirectional or bidirectional and are also assumed to be fault-free. Once all the parameters are entered, simulation can be started. Two different options are available for simulation;

a. Complete Run: On pressing this button, the simulation will run to its end. The total number of iterations executed is determined by the position of slider. For all simulations, a default value of 500000 is set.
b. Step: On-pressing 'Step!-tFSihåWation will execute the first 5000 iterations. To execute next set of iterations, press "Step" again. This can be executed till simulation runs to completion. At any time of simulation run, stop can be pressed. This will exit the simulation and calculate the terminal reliability also indicating the number of iterations in 'Current Iteration' that have been used to determine the terminal reliability.

During simulation, the following colours denote the node characteristics:

- ✓ RED: A red node denotes a faulty node.
- ✓ BLACK: All nodes that remain black, are up or fault-free
- ✓ GREEN: A path between a source and destination pair is illustrated with all nodes and links in path as green. This obviously means that all green nodes are also fault free.

In a simulation, if no green nodes or links appear, this symbolizes absence of path from source to destination. One such scenario is when the source or the destination itself is faulty. After completion, the impact of performance can be viewed by clicking on the 'Performance' button. This shows the average distance in presence of faults. The 'Clear' button clears all fields. New values can be entered for another simulation.

CHAPTER 4

RESULTS AND DISCUSSION

4.0 INTRODUCTION

The inevitable shift towards parallel computing systems has accentuated the need for more reliable networks. Network topologies like torus, mesh and hypercube provide reliable networks in the form of multiple paths for delivering packets from source nodes to destination nodes. This research improves the throughput of torus networks as measured by the terminal reliability (network reliability) and performance of the torus network.

Terminal reliability is a major concern when dealing with interconnection networks especially torus networks because of its use in parallel computer systems. This is due to the fact that faults in these systems greatly degrade reliability and performance which affects their throughput. Therefore the effect of reliability is of much concern in torus networks.

This chapter focuses on the results and discussion of the torus network simulation; this involves networks with different number of nodes which were analyzed using simulation. The result of the simulation that was done with respect to this research is discussed belowg

4.1 TERMINAL RELIABILITY

Terminal reliability, a commonly used measure of connectivity, is the probability that a processor, called the "source node," can successfully communicate with another processor, called the "destination node," in the network (Tein et al, 1994). That is, reliability is concerned with the ability of a network to carry out a desired operation such as "communication".

Reliability of a network addresses the probability that a given source-destination pair has at least one fault-free path between them (Sarwar et al, 2000).

To compute the terminal reliability in a torus network, the following assumptions are made:

- ✓ The network does not have any self-loops.
- ✓ Failure of a link is independent of other link failures.
- ✓ A link has only two states operational or faulty.

v/ Nodes are assumed to be fault-free.

The terminal reliability is computed by finding all possible paths from the source to the destination node (Tein et al, 1994). The average terminal reliability of a network can also be obtained by fixing a source node and averaging the terminal reliabilities obtained for all possible destination nodes in the network.

4.2 SIMULATION ENVIRONMENT AND RESULTS

The degree to which a network is able to transmit data needs to be quantitatively assessed by using the network reliability measure. For the purpose of this research, a network simulator was built to calculate the terminal reliability and estimate performance of Etorus networkfifif"imulation is based on the assumption that all the nodes-are perfect and the communication links are static and irreplaceable. That is; there is a correlation between the reliability of node(s) and the terminal reliability. In this thesis, two terminal reliability computation is used. Two nodes are distinguished as source and destination nodes. The rest of the nodes function as relays to provide communication paths between the source and destination nodes. The simulator provides the fleXibility to simulate a torus network with the number of nodes (N), less than ten.

During •simulation the torus network will have a total number of nodes as N * N with each node having a reliability (R) which is a value less than or equal to 1.0. For the sake

of clarity and simplicity, a few torus networks with different number of nodes (N) were simulated for this research.

4.2.1 SIMULATION RESULTS AND ANALYSIS

The number of nodes (N) used for this simulation are as follows; N=3, N=5and N=7. During simulation the torus network will have a total number of nodes as N * N with each node having a reliability (R) which is a value less than or equal to 1.0. Each node in the network generated for simulation has the same reliability(R). In order to obtain correct results during simulation, packets are transferred from the same source node to the same destination node either by using bidirectional or unidirectional links for all the different networks. Figure 4.1 shows an on-going simulation of a torus network with nine nodes. Figure 4.2 also shows the terminal reliability and performance of a torus network with a total of twenty five nodes. Figure 4.3 also shows the results obtained for terminal reliability of a network with forty nine nodes which uses unidirectional links. Table 4.1 and Table 4.2 show the result obtained for terminal reliability when simulation was conducted.

Table 4.1

Simulation Results of Torus Networks with each Node having the Same Reliability (R) with Bidirectional Links.

Number of Nodes(N)	Reliability of Each Node(R)	Terminal Reliability of Network
3	0.8	0.6375
5	0.8	0.6374
7	0.8	0.6378

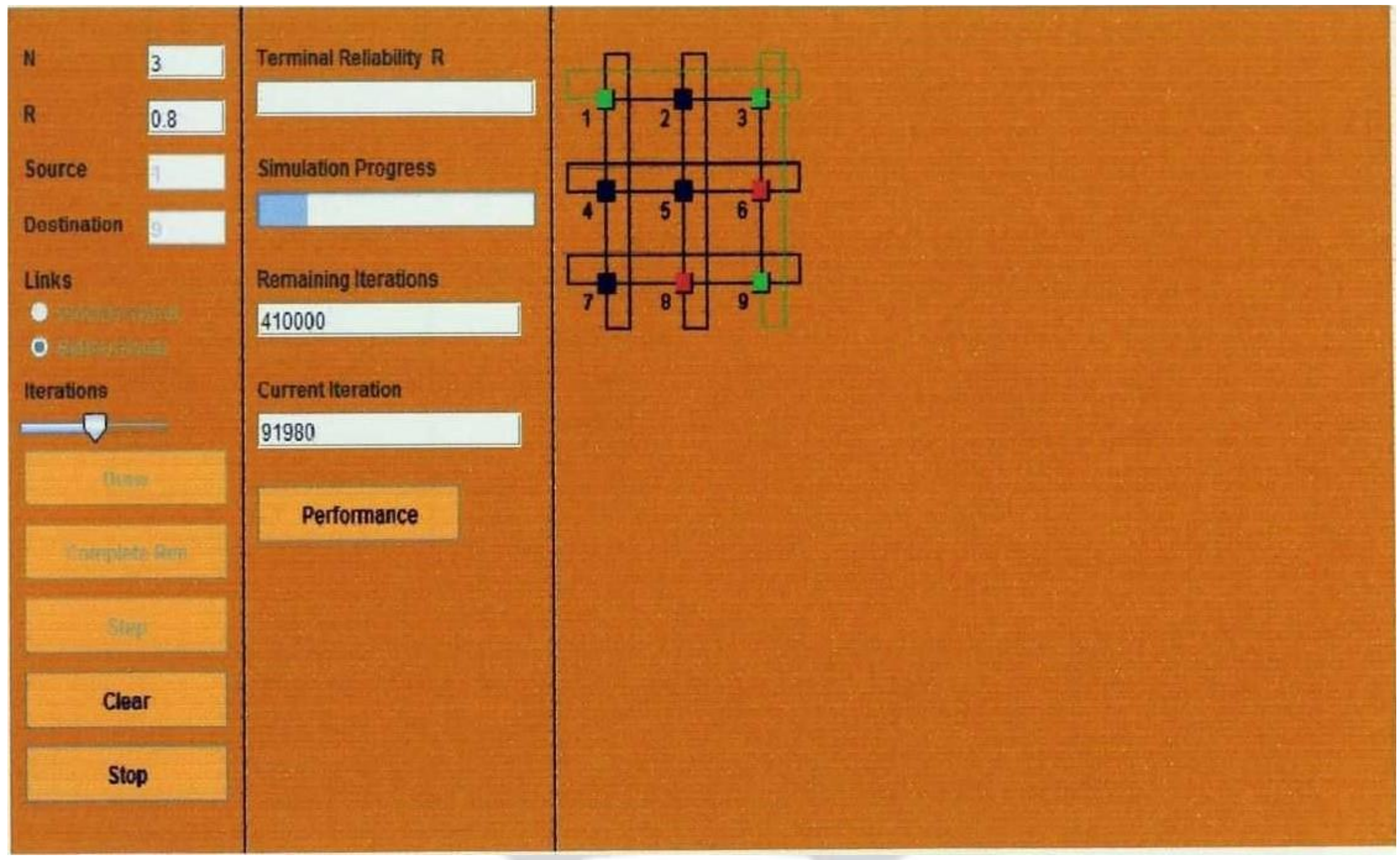


Figure 4. I On-going Simulation of a Torus Network

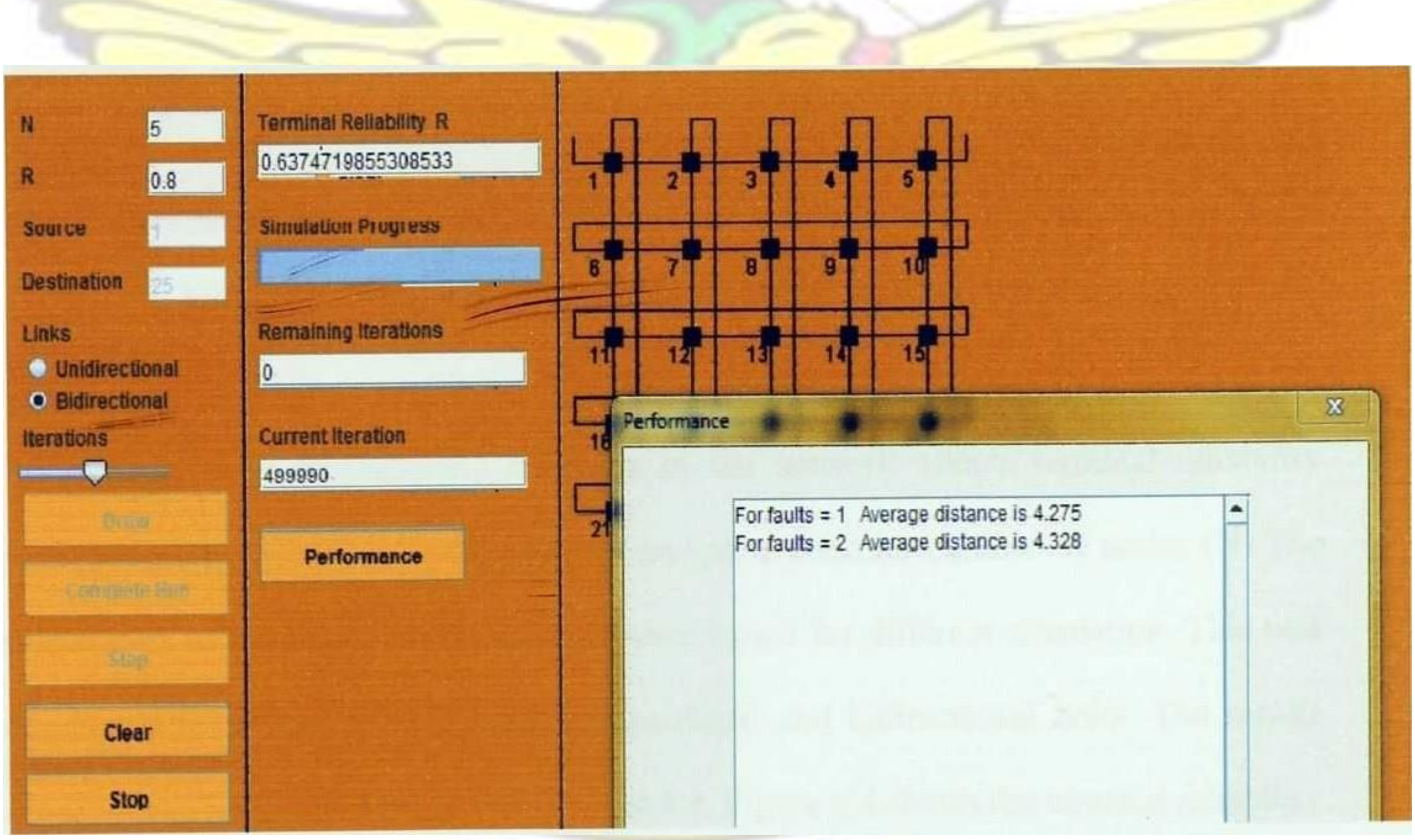


Figure 4.2 Completed Simulation Showing Terminal Reliability and Performance

Table 4.2

Simulation Results of Torus Networks with each Node having the Same Reliability (R) with Unidirectional Links.

Number of Nodes(N)	Reliability of Each Node(R)	Terminal Reliability of Network
3	0.8	0.6147
5	0.8	0.6149
7	0.8	0.6136

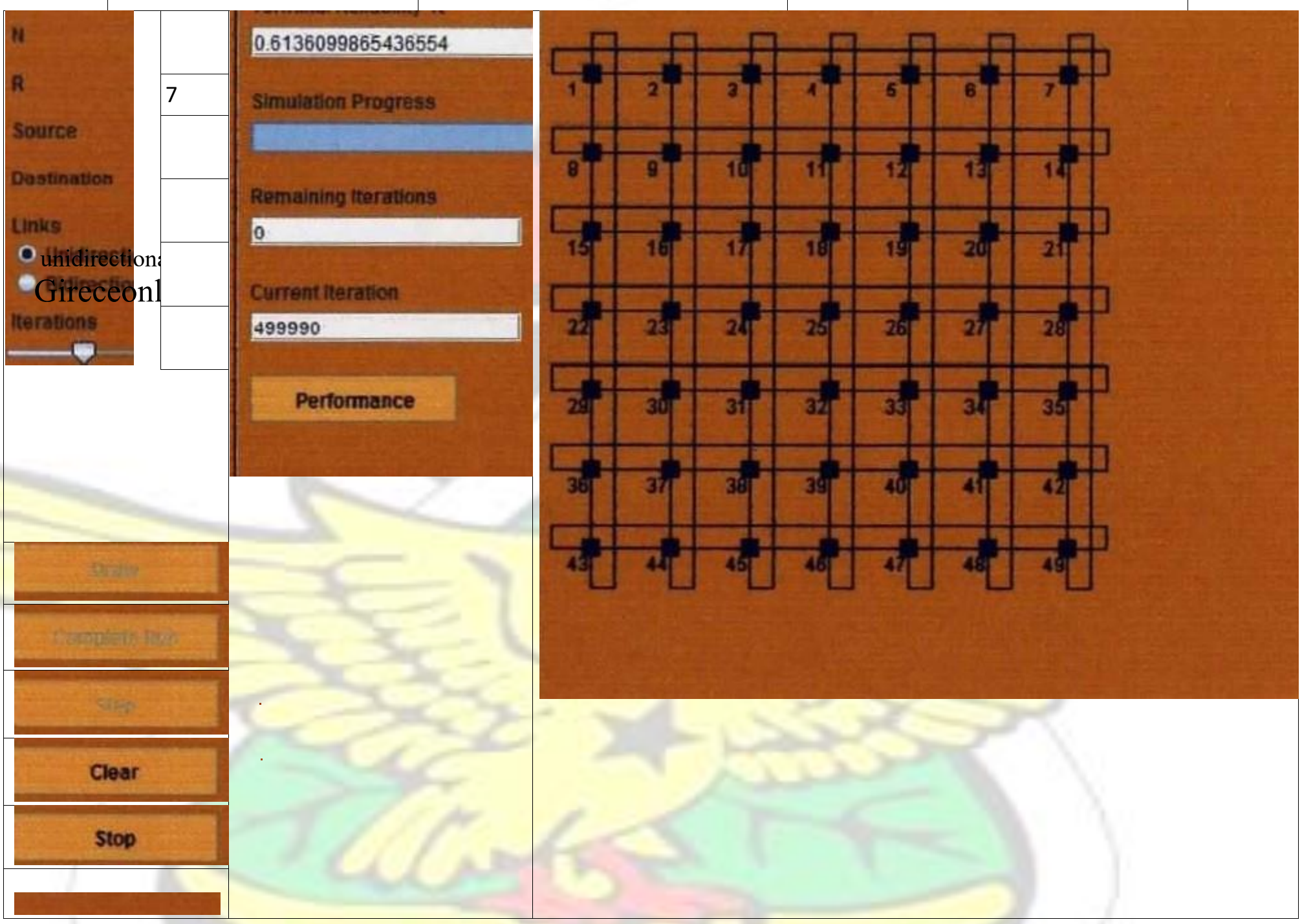


Figure4.3 Complet+imulation Showing Terminal Reliability

To evaluate how reliability(R) of nodes in the network affects terminal reliability simulation was conducted on a network having a constant number of nodes (N). The reliabilities of the nodes in the network were varied for different simulation This was conducted for networks with both unidirectional and bidirectional links. The results obtained can be found in the tables 4.3 and 4.4. Figure 4.4 shows the terminal reliability result obtained for a network with total number ofnode as nine with reliability of 0.9.

Table 4.3

Simulation Results of Torus Networks with each Node having Different Reliability (R) with Bidirectional Links.

Number of Nodes(N)	Reliability of Each Node(R)	Terminal Reliability of Network
3	0.1	0.0019
3	0.3	0.0546
3	0.5	0.2185
3	0.7	0.4803
3	0.9	0.8086

Table 4.4

Simulation Results of Torus Networks with each Node having Different Reliability (R) with Unidirectional links.

Number of Nodes(N)	Reliability of Each Node(R)	Terminal Reliability of Network
3	0.1	0.0019
3	0.3	0.0459
3	0.5	0.1869
-3	0.7	0.4460
3	0.9	0.8013

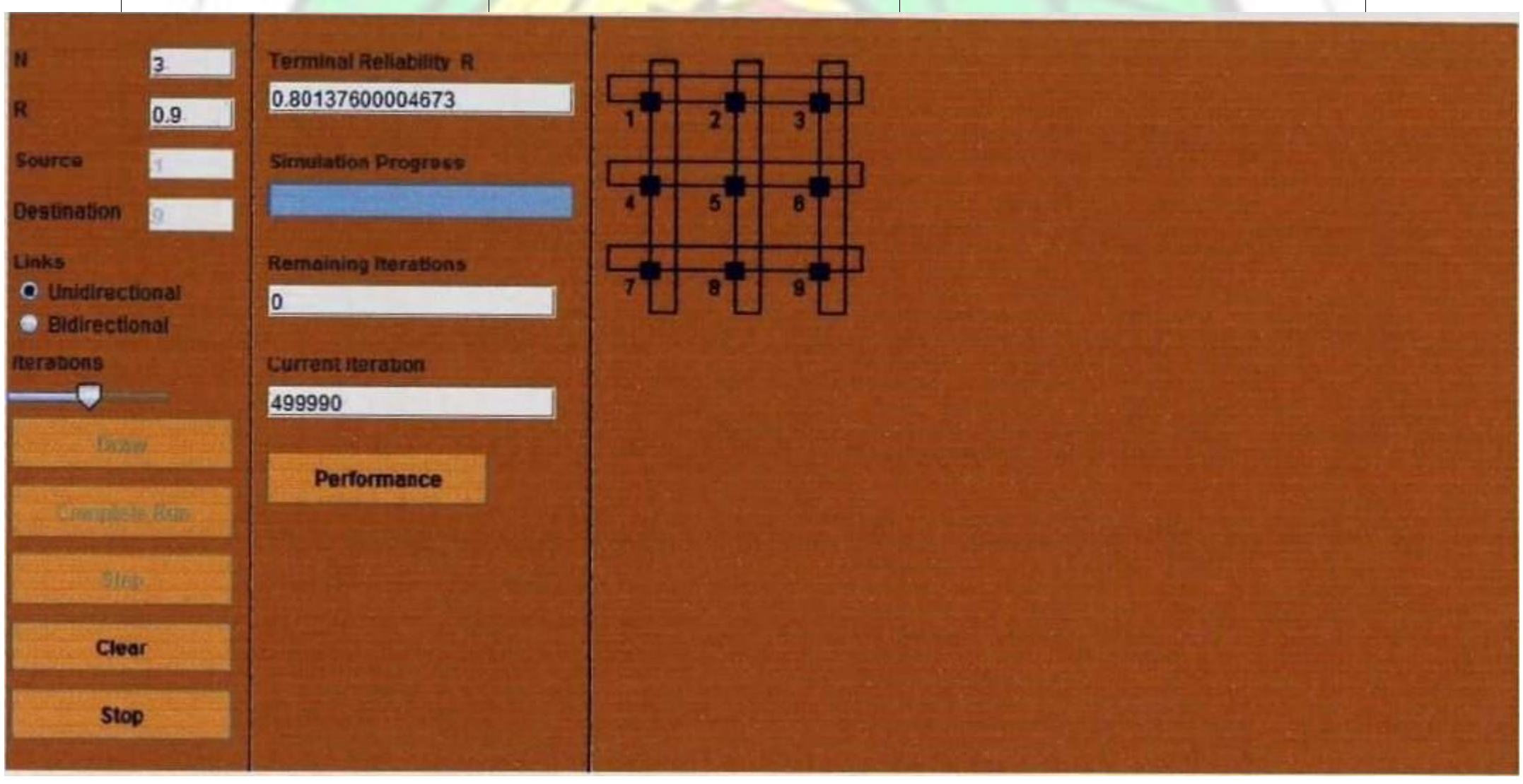


Figure 4.4 Completed Simulation showing Terminal Reliability

4.3 DISCUSSION OF RESULTS

Terminal reliability is computed by finding all possible paths from the source node to the destination node (Tein et al, 1994). Tables 4.1 and 4.2 illustrate situations where simulation was performed on three different networks; each network has different number of nodes (N), but with each node in a network having the same node reliability (R). Table 4.1 shows results for tori networks with bidirectional links whiles table 4.2 shows results of networks with unidirectional links. Analyzing the results obtained from the simulatiön in table 4.1 and 4.2, it is clear that the terminal reliability of each network in table 4.1 and 4.2 respectively is virtually the same even though the networks had different number of nodes (N). For instance, in table 4.1, networks with number of nodes, N = 3 and 5 had terminal reliability of0.6375 and 0.6374 respectively.

It can therefore be deduced that, terminal reliability is the same for torus networks with different number of nodes (N) but with each node in a network having the same node reliability(R). That is, terminal reliability is not dependent on the network size or the total number of nodes in the network but rather on the reliability (R) ofeach node in the network. That is; reliability of node(s) R has a direct effect on the terminal reliability of the network. This means that networks with high terminal reliability are able to transmit packets from source node(s) to destination node(s) efficiently which invariably leads to increase in network throughput. This is because packets can be transmitted through alternative paths even when there are faulty nodes in the network during transmission. This assertion represents several such trends that can be found in torus networks.

Again, comparing the terminal reliability values in table 4.1 and table 4.2, it can be observ2d that the values in table 4.1 are slightly higher than the values in table 4.2. This shows that, there is an improvement in terminal reliability of torus networks with bidirectional links over torus networks with unidirectional links.

These differences in results are particularly significant as they demonstrate that improvement in terminal reliability can be achieved by doubling the number of links in

the network without any radical reorganization of the network topology. It must therefore be noted that implementing a bidirectional torus network over a unidirectional torus network with the same topology generally results in an increase in terminal reliabi ity of the-network whieh&ä€fo better network throughput. It must also be noted hat torus networks with bidirectional links generally results in an improvement in the effective bisection bandwidth and latency. On the other hand, unidirectional torus networks are becoming increasingly popular (Chou and Du, 1991). This is due to the fact that the use of paths in only one direction has also been recommended to avoid deadlocks in torus networks (Dally and Seitz, 1987).

Tables 4.3 and 4.4, shows simulation results of different torus networks having the same number of nodes (N) but each node(s) in a network has different node

.1

reliability (R). Table 4.3 has bidirectional links whiles table 4.4 has unidirectional links. the results of the simulation obtained in both table shows that terminal reliability of the networks increases as the node reliability (R) of each network also increases. 'Ihese results obtained in tables 4.3 and 4.4 supports the conclusion drawn earlier that terminal reliability of torus networks does not depend on the total number of nodes in the network but rather on the reliability of node(s) in the network It is also clear that the terminal reliability of bidirectional torus network is higher than that of unidirectional torus networks. The simulation also calculates the performance of the network. The performance of the network is the average shortest distance that a packet travels from the source node to the destination node in the presence of faults.

CHAPTER 5

CONCLUSION

5.0 CONCLUSION

The inevitable shift towards parallel computing systems (torus network) has accentuated the need for more reliable networks. This is because as the number of components in the network increases, so too does the failure rate of the system. Routing algorithms that are used in these systems for transmitting packets must therefore be able to route packets reliably and efficiently. There are several routing algorithms that are used to route packets in torus systems and each has its own drawbacks. Dimension order routing (DOR) is one of such algorithms which tends to send packets towards the centre of the network where the congestion is high which causes the network to be overloaded. This leads to early network saturation and performance degradation (Dally, 1992). The turn model has a problem of complex routing logic whiles the odd even turn model does not even have a fault tolerant implementation yet (Ge-Ming, 2000). The planar algorithm is nonadaptive and the only improvement in non- planar networks (Chien and Kim, 1992). This thesis introduces intercoryecüen-network, torus network, the types of algorithm used in torus networks and goes on to show how reliability and throughput in this network can greatly be improved. This thesis analyze the issue of reliable data transmission in torus based interconnection as an important aspect in increasing network reliability and throughput by performing simulations on 2-D torus (2Dimentional torus) by using a simulation- based performance evaluator to determine terminal reliability and performance. A network simulator which uses source routing was specifically built for this purpose.

In source routing, the information about the whole path from source to destination is pre-computed and provided in the packet header (Flich et al, 2000). The simulator first constructs the torus network and a source and destination pair is selected to transmit the

packet. The torus network constructed can either have a unidirectional links or bidirectional links. During simulation, the system (simulator) generates random packets from source through other nodes to the destination node. The system then calculates the terminal reliability and finds the shortest possible paths from the source node to the destination nodes. The results obtained from the simulation prove that torus networks with high node reliability achieved high terminal reliability and high performance because packets are able to reach their destination through alternative path(s) in the network. That is; terminal reliability of torus network does not depend on the network size but on the reliability of nodes in the network. This situation invariably also improves network throughput of tori networks.

5.1 RECOMMENDATIONS

Performance evaluation and reliability prediction are two important factors in the study of mukiprocessor-and interconneeéon-systems (torus networks). In this regard, a lot more rc•search is required to unearth the cutting edge innovations in this area of research. Torus networks need to be more powerful in terms ofreliability and performance to guarantee better computer systems for the future.

By way of recommendations:

- ✓ Reliability of torus networks can be improved by making sure that nodes in these networks have high reliability.
- ✓ Routing algorithms should be designed to transmit packets through the shortest paths in these networks but not through random nodes which increases delay.

5.2 FUTURE WORK

There exist several possible directions for future research. One possible area for future research is the application of the techniques described in this paper to other emerging high-performance networks such as photonic networks, guassian networks and optical interconnects.

REFERENCES

Alex Ho, Steven Smith and Steven Hand "On Deadlock, Livelock and Forward Progress" University of Cambridge Computer Laboratory, 2005 [Online] Accessed 2012, May 10. Available at http://www.cl.cam.ac.uWTechReports/

Balchunas A. "The Routing Table" , [Online] Accessed on : May 10,2012. Available at http://www.routeralley.com/rad/docs/static_dynamic.pdf

Bobda C., Ahmadinia A., Fekete S., Teich J., " Packet Routing in Dynamically Changing Network on Chips," Proceedings, 19th IEEE International Parallel and Distributed Processing Symposium, pp. 154b, April 2005.

Chien, A.A., Kim, J.H., "Planar-Adaptive Routing: Low-Cost Adaptive Networks for Multiprocessors", Proceedings of the International Symposium on Computer Architecture, pp. 268-277, 1992.

Chou C. and Du D. H. C., "Hierarchical Unidirectional Hypercubes," In-Prcceedings International Conference on Parallel Processing, vol. 1, pp. 53, 1991.

Christopher J. Glass and Lionel M. Ni, " The Turn Model for Adaptive Routing", Journal for theÄssociation of Com uting Machinery, vol. 45, No 5, pp. 874 — 902, September, 1994.

Dally W. J., and Towles B. P. "Principles and Practices of Interconnection Networks" Elsevier, March, 2004.

Dally W. J., Seitz C. L., "The Torus Routing Chip" Department of Computer Scipnce, California Institute of Technology, January 1986.

Dany W.J., "Virtual-Channel Flow Control", IEEE Transaction on Parallel and Distributed Systems, vol 3, No 2, March, 1992.

Daily W.J., and Seitz C.L., "Deadlock-free Message Routing in Multiprocessor Interconnection Networks," IEEE Trans Comput. C-6 (547-553), May, 1987.

De Micheli G., Benini L., "Network on Chips, Technology and Tools", Morgan KaUfmann Publishers, 2006.

Dehyadgari M., Mohsen N., Ali A., Zainalabein N., " Evaluation of Pseudo Adaptive XY Routing Using an Object Oriented Model for Network on Chip" IEEE, pp. 204-206,2005.

Duato J., Yalamanchili S., Ni L. M., "Interconnection Network: An Engineering Approach", Morgan Kaufmann Publishers, 2003.

Faraz Shamim, Zaheer Aziz, Johnson Liu, "Troubleshooting IP Routing Protocols" Cisco Press, May, 2002. [Online] Accessed on: May 10, 2012. Available at: http://docstore.mik.ua/cisco/pdf/routing/Cisco.Press.Troubleshooting.IP.Routing.P r otocols.pdf

Flich J., Lopez P., Malumbers, M. P. and Duato J., "Improving the Performance of Regular Networks with Source Routing", Proceeding of IEEE International Conference on Parallel Processing, pp. 353-361, August, 2000.

Ge-Ming Chius"The Odd-Even Turn Model for Adaptive Routing," Parallel and Distributed Systems, IEE ransactions on Computing, vol. 11, no. 7, pp. 729-738, July 2000.

Glass C.J. and Ni L.M. " The Turn Model for Adaptive Routing" Journal for the Association of Computing Machinery, vol. 51, no. 5, pp. 874-902, September 1994.

Kariniemi H., Nurmi J., "Arbitration and Routing Schemes for On-chip Packet Networks", Interconnect-Centric Design for Advanced SOC and NOC, Kluwer Academic Publishers, pp. 253-282,2004.

Kermani P. and L. Klienrock "Virtual-Cut Through: A New Computer Communication Switching Technique", Computer Networks, vol 3, pp. 267-286, 1979.

Kessler R. E. and Schwarzmeier J. L. " Cray T3D: A new Dimension for Cray Research, In Proceeding of COMPCON, pp. 176-182, February, 1993.

Krishnan M.N., Raghunath S., Ajith Pravin Dhas D., Benny Raj A.M. and Pounambal M. "A Deadlock — Free Routing Algorithm for Torus Network" School of Information Technology and Engineering, VIT University, India. [Online] Accessed on: 8 June, 2012. Available at: http://www.iiste.org/Journals/index.php[NCS/article/view/2371/2370

Lawrence G. R. , "Communications and System Architecture" Advanced Research Projects Agency, May 1970. [Online] Accessed on: 8 May, 2012. Available at: http://www.packet.cc/files/com-sys-arch.html

Ling Y., Shigang C., Cho-Yu J. C., "Optimal Deadlock Detection Scheduling" 2006 [Online] Accessed on March 20, 2012. Available at http://arxiv.org/pdf/1008.0451 .pdf

Mubeen S. "Evaluation of Source Routing for Mesh Topology Network on Chip Platform" Thesis, Jonkoping Institute of Technology, 2009.

Pinkston T. MÆifd Duato J., "Interconnection Networks" 2006[Online] Accessed on February 28, 2013. Available at http://web.mit.edu/6.173/www/currentsemester/readings/R07-interconnection networks-hennessy-patterson.pdf

Rartala V., Teijo L., Juha P., "Network on Chip Routing Algorithms" Turku Centre for Computer Science Technical Report, No 779, August,2006.

Ravikumar C. P. and C. S. Panda "Adaptive Routing in k-ary n-cubes using Incomplete Diagnostic Information" Microprocessors and Microsystems, vol. 20, pp. 351-360, 1997.

Sander 1., "Oblivious and Adaptive Routing" [Online] Accessed on 12 June, 2013. Available at http://www.ict.kth.se/courses/IL2207/0708/Lectures/IL2207 L6_NoC_Routing2pdf

Seitz C. L., "The Hypercube Communications Chip" Department of Computer Science, California Institute of Technology, March, 1985.

Seitz, C. L. " The Cosmic Cube", CACM, vol,28(1),pp. 22-33, January, 1985.

Shin, K. G. , Daniel, G. W., " Analysis and Implementation of Hybrid Switching", . IEEE Trans. on Computers, vol. 45(6), pp. 684-692, 1996.

Singh A. "Load Balanced Routing in Interconnection Networks" [Online] Accessed 2013, February 5. Available at http://cva.stanford.edu/publications/2005/thesis_arjuns.pdf

Tein Y. Chung, Nita Sharma and Dharma P. Agrawal, "Cost-Performance Tradeoffs in Manhattan Street Network versus 2-D Torus" IEEE Transactions on Computers, vol 43, No 2, February, 1994.

Valiant L. G. anWBrebner G. L" Universal Schemes for Parallel Communication" In Proceedings of the 13th Annual ACM Symposium on Theory of Computing, pp. 263—277, 1981.

Vasseur, J.P., Kim, M., Pister, K., Dejean, N., and D. Barthel, "Routing Metrics Used for Path Calculation in Low — Power and Lossy Networks", RFC 6551, March 2012.

Venkatesh,S. and J.S.Smith, "A Graph-Theoretic, Linear-time Scheme to Detect and Resolve Deadlocks in Flexible Manufacturing Cell" Journal of Manufacturing

Systems ,V01.22, No.3, pp. 220-238, 2003.

Xiang D. and Luo W., "An Efficient Adaptive Deadlock-Free Routing Algorithm for Torus Networks" IEEE Transactions on Parallel and Distributed Systems, vol. 23, No. 5, May 2012.