ROUTING STRATEGIES IN DIGITAL NETWORKS

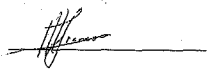
PART I: ROUTING STRATEGIES

Marc Leon Francois

A project report submitted to the Faculty of Engineering, University of the Witwatersrand, Johannesburg, in partial fulfilment of the requirements for the degree of Master of Science in Engineering

Johannesburg, 1988

I declare that this project report is my own, unaided work. It is being submitted for the Degree of Master of Science in Engineering in the University of the Witwatersrand, Johannesburg. It has not been submitted before for any degree or examination in any other University.



14th day of May 1988

ABSTRACT-

The problem of obtaining efficient routing algorithms for the fast delivery of messages to their destinations is of the utmost importance in the design of modern digital networks. The research project deals with the investigation of various types of routing algorithms and the techniques available for evaluating the performance of these algorithms. A minimum delay adaptive routing algorithm, developed by R G Gallager is selected. The overall performance of a particular network consisting of randomly distributed nodes is optimized using the algorithm. The performance of the network is optimized by minimizing the average packet delays. A simulation program is used to determine the performance of the network. The minimum delay routing algorithm provides a 30 to 40 percent improvement on the average packet delays compared to a static least cost algorithm. The improvement is obtained once the adaptive routing algorithm has been carefully adapted to the network in which it is implemented.

ACKNOWLEDGEMENTS

I would like to thank the following, for their assistance:

Mr B I Whitby, my supervisor for his valuable guidance and advice,

The CSIR, (The Council of Scientific and Industrial Research, Foundation for Research and development), for their financial assistance,

The University of the Witwatersrand for their financial assistance and facilities provided, and

Mr L V Russell for the use of his personal computer on which the thesis was compiled.

CONTENTS Page

PART I: ROUTING STRATEGIES

CONTENTS

CONTENTS Page

CONTENTS

CONTENTS

CONTENTS

CONTENTS

CONTENTS Page

CONTENTS

CONTENTS Page

PART II: SIMULATION PROGRAM

CONTENTS

CONTENTS

LIST OF FIGURES

PART I

LIST OF FIGURES

PART II

LIST OF GRAPHS

PART I

LIST OF TABLES

PART I

1 INTRODUCTION

During the past decade the key technology has been information gathering, processing and distribution. This has led to the design and implementation of large complex digital computer networks.

A network consists of a set of nodes which are interconnected by a system of links. Data is transferred from one node to another in the network via the links. In modern networks it is necessary to minimize average packet delay and to maximize the throughput. The project involves the study of routing algorithms which determine the path taken by the data travelling through the network. The particular routing algorithm studied minimizes the total average packet delay in the network.

1.1 Project description

The project examines the various types of network topologies which are most efficient with respect to the routing of traffic. A fixed network topology is selected. Various types of routing algorithms are studied in order to establish an algorithm best suited for the type of network topology selected. Methods for predicting the performance of the routing algorithms are examined. A suitable method is then chosen and used to predict the performance of the network. The adaptability of the minimum delay routing algorithm is optimized and the performance of the network under rapidly varying input traffic loads is studied.

1.2 Layout of the report

The report is divided into two parts. The first part, Part I, deals with the actual research project and includes the following chapters: Chapter 2, introduces the theory used to describe and analyse digital networks. Chapter 3 is devoted to a literature survey on routing algorithms. The features of the more important routing algorithms are described in this chapter. Chapter 4 describes the techniques available for analysing and predicting the performance of large digital networks. Chapter 5 deals with the type and specification of the network and routing algorithm to be tested and optimized. The subsequent chapter, Chapter 6, describes the development of the network routing simulation package.

The last two chapters, Chapters 7 and 8, evaluate the results obtained from the simulation program and the optimization of the network. Chapter 8 concludes the report with the deductions made from the research project, the limitations of the project as well as recommendations for possible further investigation.

Part II of the report deals exclusively with the network routing simulation program developed in the research project. Part II includes the user's manual and design specification of the program. The entire program listing is also provided in part II of the report.

2 BACKGROUND THEORY

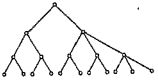
The basic concepts and theory used to define and evaluate the *performance of a network* are introduced. The OSI model of a network is defined. The definition, types and properties of routing algorithms are stated. A list and description of the terminology used in the report is given in Appendix A.

2.1 Digital networks

A digital network consists of a set of nodes which are interconnected by links. Information or data is transferred between the nodes in the network via the links. Nodes which generates date to be transferred to other nodes are known as the source or transmitter nodes. Nodes which consume data from other nodes are referred to as sink or receiver nodes. The manner in which the links interconnect the nodes in the network is known as the topology of the network. Typical network topologies are illustrated below.

a) Ring type network

b) Star network



c) Hierarchical network

d) Irregular network

Figure 2.1 Diagram illustrating various network topologies.

2.2.1 Uses and advantages of digital networks

The uses and benefits of networks are numerous. The main advantage of a network is that it allows separate independent nodes to communicate with each other. This leads to several important uses:

- Information gathering: A network allows information to be gathered over large areas and relayed to a central point permitting large amounts of valuable information to be extracted and correlated.

- Increased reliability: Networks provide alternative sources of supply. The loss of a single node or computer in a network is usually less serious than if all the terminals were connected to a single computer. In applications such as military, industrial process control and banking the loss of computing power, even for a short period of time, can be catastrophic.

- Resource sharing: A network allows a number of users or nodes to share expensive or valuable resources such as data bases and printers.

- Cost of computing: Due to the decreasing cost of computers in general it has become more feasible to process the data where it is collected, rather than transferring the data to a central computer to be processed. This naturally leads to the implementation of a network.

- Rapid communication: Networks allow nodes that may be physically distant from each other to rapidly exchange information. If a node gathers information that is valuable to another node the data can be transferred immediately to that node.

- Simple design of complex systems: Finally, in a network it is possible to dedicate some (or all) of the processes to specialized functions. This leads to the natural decomposition of large complex systems into smaller more manageable systems. Each node in the network can be programmed to perform a single function at any one time. A network allows a system to expanded in small increments and processes may be added to the network as they are required.

The requirements of a network may vary according to the application. In general a network must be able to transfer information from one node to another as quickly and cheaply as possible. Often an additional requirement is that of security where one user may prevant the other users in the network from obtaining privileged information.

There exists an abundant number of possible applications for computer networks which extends from education, industrial process control, banking, automated newspapers and military applications to fully automated libraries and software distribution. The advantages, uses and application possibilities of networks and the need for reliable and efficient networks are escalating extremely rapidly.

2.3 The OSI reference model

To reduce the complexity of designing networks and to allow networks to be designed in a structured manner, the network may be divided into 'layers'. Each layer performs a well defined function and is built upon its predecessor. The purpose of each layer is to provide certain functions to the higher levels. Each layer may be described as a 'black box' with the interface to the adjacent levels clearly defined but

the functions performed within the 'black box' hidden from the higher layers.

The International Systems Organization (ISO) have defined a set of layers which has become widely recognized as a standard to which networks are designed. The set of layers is known as the Reference Model of Open Systems Interconnection (OSI). The OSI model is divided into seven layers as illustrated below.

Application	<----- Protocol ----> <-- Interface	Application	Message
Presentation	<----- Protocol ---->	Presentation	Message
Session	<----- Protocol ---->	Session	Message
Transport	<----- Protocol ---->	Transport	Message
Network	<----- Protocol ---->	Network	Packet
Data Link	<----- Protocol ---->	Data link	Frame
Physical	<----- Protocol ---->	Physical	Bit

Figure 2.2 Schematic representation of the OSI reference model. (Courtesy, A S Tanenbaum [43]).

The project is concerned with the third layer the 'Network layer'. For completeness a brief description of the functions of each of the layers is provided.

2.3.1 The Physical layer

The physical layer is concerned with the transmission of raw bits over a communication channel [43]. The layer is concerned with aspects such as the voltage levels that represent a one and a zero and whether data transmission is allowed in both directions. A simplex channel is one that allows data to travel in one direction only. A channel that

allows data to travel in both directions but not simultaneously is known as a half-duplex channel. A full-duplex channel is one that allows data to be transmitted in both directions simultaneously. The capacity of the link from node i to node k is represented as C_{ik} and defines the maximum number of bits per second that may be transmitted over the link.

2.3.2 The Data link layer

The function of the data link layer is to transform the physical layer into a channel that appears free of transmission errors. This is accomplished by breaking up the data into packets or frames which are transmitted sequentially across the network. The data link layer is also concerned with the transmission of acknowledgement frames which are transmitted to the source node by the receiver. The project is concerned with the next layer, the network layer It is assumed that the 'Data link' layer has been implemented and provides a perfect physical layer which is free of transmission errors. It is however not possible to isolate all of the functions of the data link layer from the network layer. Protocols used by the data link layer may effect the efficiency of the routing algorithm found in the network layer.

2.3.3 The Network layer

The network layer is responsible for determining how packets are routed within a network. The project is devoted to this layer. Congestion control is also handled by the network layer. Accounting functions, such as calculating the cost of transferring a message across the network are performed by this layer.

Two basic models of how packets are routed across the network exist. These include the virtual circuit model and datagram circuit model. In the virtual circuit model a fixed path is set up to a particular destination when the user enters the network. The path then remains fixed for the entire period the user is connected to the network. A single path is available and packets are always received by the destination in the same order they were transmitted. In a datagram network the network layer simply accepts packets from the transport layer and a best attempt is made to transfer the packet to the destination. Different packets for the same destination may take different paths. No guarantee is made that packets arrive at the destination in the correct order or even at all. When designing a network it is important to select the type of model used. This is dependent on the type and properties of the required network.

The type of routing algorithm, congestion protocol and model needed by the network requires careful selection. This aspect of the project is described in chapter 5. A brief description of the functions of the next layers is given in order to verify some of the assumptions made in the report.

2.3.4 The Transport layer

The function of the transport layer is to split the data into smaller packets or frames and to pass these to the network layer. It also ensures that all the packets arrive at the destination in the correct order.

2.3.5 The Session layer

The session layer provides the user interface to the network. A connection between users is often called a session. The setting up of a session is a complicated procedure which is not considered here. In some networks the session and transport layers are merged into a single layer.

2.3.6 The Presentation layer

The presentation layer performs functions such as filing and the translation of ASCII characters to bits that may be transmitted. The layer may be used to translate different file formats to a format understandable by the particular terminal.

2.3.7 The Application layer

This is the uppermost layer in the OSI model. The purpose of this layer is to provide transparency and to hide the physical distribution of the resources from the user. The exact content of the application layer is dependant on the actual user.

2.4 Conclusion: Network Definition

Although the description given is brief, it is clear that a network is a combination of numerous interrelated functions making up a highly complex system. The project focusses on only a small part of the system, and includes the research into routing algorithms and other related topics such as congestion control protocols.

2.5 Queuing theory

The digital network studied in the project is based on queuing network principles. Therefore it is necessary to have a basic understanding of queuing theory. The theory required for the understanding of the report is provided. A more detailed explanation on queuing theory is given in references [24,25].

2.5.1 Introduction to queuing theory

Queuing theory is used to model processes in which packets arrive, wait their turn for service, are serviced and then depart. Queuing systems are characterized by the following:-

- The interarrival-time probability density function. The interarrival-time probability density function represents the distribution of the time delay between two successive arrivals at a queue.
- The service-time probability density function. The service time probability density function represents the distribution of the time period required to service a packet.
- The number of servers. The number of servers present to service the packets from the same input queue.
- The queuing discipline. The queuing discipline specifies how incoming packets are inserted into the input queue.
- The amount of buffer space in the queues.

The theory required to analyse a queuing system is dependent on the type of system used. It is necessary to accurately define the system that will be used in this project in order to derive the theory directly applicable to the project.

2.5.2 Queuing systems in digital networks

In order to simplify the theory a node is assumed to have a buffer of infinite length. The theory is limited to what is known as a M/M/1 queue. a M/M/1 queue is defined as follows:-

- An exponential probability density for the interarrival time,
- An exponential probability density for the service time, and
- a single server system.

A typical M/M/1 queuing system is illustrated below,

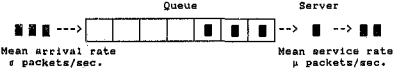


Figure 2.3 Diagram of a simple M/M/1 queuing system.

The incoming packets at a node are served on a first come first serve basis. Graphically a node appears as follows:

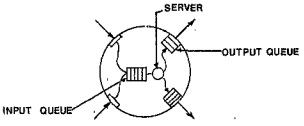


Figure 2.4 Diagram of a typical node in a network.

Assuming the mean arrival rate of packets to the queue to be σ packets per second, then the probability of exactly n packets arriving in a time interval of t is given by the Poisson law:

$$P_n(t) = \frac{(\sigma t)^n e^{-\sigma t}}{n!}$$

and the probability density function can easily be derived as:

$$a(t) = \sigma e^{-\sigma t}$$

The advantage of using Poisson probability density is that the probability of an arrival during time t is independent of the number of packets already in the queue. Assuming a mean packet length to be $1/\mu$ bits per packet and the channel capacity to be C bits per second then the average packet delay T is may be written as:

$$T = 1/(\mu C - \sigma)$$

T includes both the transmission time and queuing time.

This result leads to some important conclusions. The most important conclusion being that as the average packet arrival rate σ tends to the average channel packet rate μC then the delay tends to infinity. That is if one assumes an infinite input buffer length. The average delay experienced by packets transmitted across the network is one of the most important aspects to consider in the design of a digital network.

2.5.3 Conclusion: Queuing theory

Many of the important aspects regarding the queuing systems employed in digital networks may be derived from the simple formula given above. Some of these conclusions are detailed in chapter 4 where the theory is applied to a particular network.

2.6 Routing algorithms

Before designing a digital network the types, properties, advantages and limitations of routing algorithms must be defined.

2.6.1 Properties of routing algorithms

The task of a routing algorithm is to decide on which output path or paths an incoming packet should be transmitted.

Routing algorithms are implemented in digital networks in order to achieve one or more of the following:

- to minimize the total average packet delay,
- to reduce the cost of transferring a packet across the network from a source to a destination node,
- to minimize the number of blocked packets in the network,
- to reduce the effect of link and node failures in the network,
- to maximize the total network throughput, that is to maximize the number of packets that can be transferred across the network in a given time interval, and
- to minimize the number of nodes passed (number of hops) by a packet in moving from a source node to a destination node in the network.

In addition to the above features routing algorithms should also posses the following properties:

- be robust; a system wide failure should not be allowed,
- be able to cope with changes in the topology of the network,
- be simple and stable,
- be able to provide fairness to all the nodes in the network,
- have a fast response; the speed at which the algorithm adapts to changes in traffic must be faster than the average rate at which the traffic changes in the network if the routing algorithm is to be useful,
- Number of control packets transmitted; the amount of routing information exchanged between nodes must be sufficient to indicate changes in traffic and network topology, but if too much information is exchanged then the overhead and bandwidth required to transmit the data may be too large,
- Buffer space required; the amount of storage space required to update and contain the routing tables in a node must not be large, but sufficient to route packets efficiently around the network and,
- should demonstrate loop free properties. The routing algorithm should ensure that packets do not return to nodes already visited in the network.

It is impossible to design an efficient algorithm that simultaneously satisfies all the advantages and properties of routing algorithms described above. It is however possible, by considering the type of network and it's application, to optimize some of the features at the expense of others until a compromise is reached. For example in military applications it is important to have a robust system which minimizes the probability of a packet not reaching it's destination whilst minimizing the cost of running the network may not be

essential. In large global networks, it may be necessary to reduce the cost of running the network at the expense of increased delays. Some networks may allow the user to select a minimum delay or least cost path or to send the packets via a path that will guarantee the packet to reach it's destination.

2.6.2 Types of routing algorithms

The various basic types of routing algorithms and their specific advantages and drawbacks are described.

Static routing algorithms

Static routing algorithms are implemented as follows. Each node contains a table listing the possible output path or paths for a particular destination node. The paths are usually listed in order of priority. One of the simplest way to implement a static routing algorithm is each time a packet is to be forwarded, the node generates a random number, and according to the random number and destination node the packet is sent along one of the output paths. Static routing algorithms are the easiest type to implement and are therefore the most widely used algorithms. The main problem with static routing algorithms is that they cannot adapt to traffic changes in the network. Also if the topology of the network changes, or if a link or node fails the tables in all the nodes must be updated before the network can operate effectively. Static routing algorithms may be used as a backup algorithm to a more complex algorithm since static algorithms are simple and robust.

Adaptive routing algorithms

Adaptive routing algorithms attempt to route packets according to the traffic in the network. For example if the traffic in one part of the network increases dramatically the algorithm attempts to route packets around that part of the network. Adaptive algorithms may also be made to adjust to changes in topology of the network and to react to node and link failures in such a way as to minimize the effect on the traffic. Generally adaptive routing algorithms, sometimes called dynamic routing algorithms, are designed to minimize the average delay experienced by the packets. Adaptive algorithms may however be designed to minimize the average number of hops per packet or to maximize the throughput of the network. The cost of utilizing an adaptive routing algorithm in a network is the added complexity and overhead required at each node. Although adaptive routing algorithms are useful no routing algorithm can cope with a network where the traffic at all the nodes increases simultaneously. *Adaptive algorithms are most useful in networks where the* traffic between two nodes suddenly increases for a short period of time.

Centralized routing systems

In this type of system each node contains a table of the possible paths to each destination node. This is similar to a static routing algorithm. Periodically each node transmits status information such as queue lengths, amount of traffic per line etc. to one node known as the routing control center (RCC). The RCC using the information obtained from all the nodes calculates a new routing table for each node. The RCC then transmits the new table to each particular node and the node then continues to route the packets according to the new routing table received from the RCC. The advantage of such a

system is that the RCC contains information about the entire network and can therefore make perfect decisions on how packets are to be routed. Another advantage is that it relieves each node of the task of calculating the routing tables. The disadvantages of such a system are unfortunately quite severe. If the system is to react rapidly to changes in traffic the RCC must perform routing calculations fairly often requiring a fast sophisticated central processing unit (CPU). Another problem is the heavy flow of traffic near the RCC due to the large amount of data being transferred to and from the RCC. The most serious problem suffered by a centralized routing system is it's vulnerability, since if the RCC fails the entire network fails.

Decentralized routing systems

The problems of centralized routing systems has led to the development of decentralized or distributed routing systems. In decentralized routing systems each node is responsible for calculating it's own routing tables. Information such as queue lengths, packet delays etc. may be transferred between the various nodes in the network, and based on this information each node compute it's own routing tables. The advantages of such a system is that it can be made extremely reliable. For example, if a node fails the rest of the network can continue to operate. The main disadvantage of a distributed system is the time taken for a change in traffic patterns or a node failure to propagate through the network so that each node can alter it's routing tables accordingly. Another problem is that if routing information is exchanged at regular intervals between the nodes in the network a significant amount of channel capacity may be taken up by the routing packets. A large number of digital networks employing a decentralized routing system have been implemented. The most fitting example is the ARPANET network.

2.7 Conclusion: Background theory

The basic properties applicable to routing algorithms in general was detailed in this chapter. The reader is supplied with the basic information required to understand the advantages, uses and drawbacks of the algorithms discussed in the report.

3 LITERATURE SURVEY

A literature survey was undertaken to determine the types of routing algorithms currently proposed or in use in large networks. A table, summarizing the characteristics of each algorithm is provided in Appendix B.

3.1 Routing algorithms

Routing algorithms are divided into two categories, namely the static and adaptive *algorithms*. Static routing algorithms include the flooding, random and least cost algorithms. The flooding and random routing algorithms are simple and require *minimal intelligence. The flooding* and random algorithms are only effective in small networks with a very low constant traffic flow. The least cost routing algorithm minimizes the cost of transferring messages across a network. Least cost or shortest path algorithms may be used to minimize the average packet delay in a network.

Adaptive routing algorithms are more suited to networks with a varying input traffic load. Adaptive routing algorithms attempt to minimize the average packet delay. Minimizing the delay is accomplished by routing packets around congested regions in the network. Numerous adaptive routing algorithms have been developed. Each algorithm suited to a particular type of network. The characteristics and properties of each algorithm is provided in Appendix B.

Several algorithms appear to be suited to a network consisting of randomly interconnected nodes. These algorithms include:

- The delta routing algorithm proposed by H Rudin [37],
- The minimum delay routing algorithm developed by R G Gallager [11],

- The extremal flows algorithm proposed by D C Cantor and M Gerla [8], and
- The second derivative routing algorithm by D P Bertsekas [5].

All these adaptive routing algorithms minimize the packet delay in the network. The minimum delay routing algorithm, developed by R G Gallager appears to a suitable algorithm for several reasons:

- It may be adapted to any type or size of network,
- simple,
- requires a minimum amount of routing information to be exchanged between nodes,
- minimizes the total average packet delay in the network, and
- may be used in 'quasi-static' networks. The performance of the algorithm in dynamic networks requires more study.

3.2 Networks

Some of the networks currently in use are also summarized in Appendix B. The networks studied include the ARPANET, TRANSPAC, TIDAS, TYMNET and DECNET networks. The survey indicates that the size of modern networks extends from 5 nodes to as many as 300 nodes. From the survey it is evident that the most important properties required by a routing algorithm are:

- robustness
- loop freedom,
- be able to cope with high traffic situations,
- be able to cope with link failures, and
- must allow for future expansion of the network.

3.3 Conclusion: The literature survey

In the literature survey, numerous types of routing algorithms have been identified. Only a few of these algorithms have been fully tested and the performance of the algorithm in various types of networks established. Various existing networks have been examined and the basic problems experienced by these networks have been identified. Briefly the major problems identified are:

- poor response of the routing algorithm to changes in traffic patterns in the network,
- poor response to link or node failures in the network,
- large memory required by the routing algorithms,
- the large amount of routing information that must be exchanged between the nodes in order for the routing algorithm to operate effectively and efficiently, and
- poor performance of the routing algorithm in extremely high traffic load situations.

In some of the networks examined some these problems have been alleviated. This has been done by examining the overall performance of the network. The routing algorithm and protocols in the network are altered in order to optimize the performance of the network. Such a technique is often extremely costly.

It is evident, from the literature survey, that an inexpensive technique for accurately predicting the performance of a routing algorithm for a particular network is required. The overall performance of a network must be determined before the network is implemented, due to the large costs involved in implementing and altering large digital networks.

4 PERFORMANCE EVALUATION TECHNIQUES

Evaluating the performance of a network is important for several reasons [22]:

- Determines the performance of the network against the overhead introduced in improving the network performance,
- Assists in tuning of the network parameters in order to obtain an optimum in the *performance of the network*,
- Helps in comparing the performance of two or more networks.

Several techniques exist for evaluating the performance of a network and these include:

- Analytical techniques,
- Experimental techniques, and
- Simulation techniques.

The methods used in evaluating the performance and efficiency of a network is are described in detail in this chapter.

4.1 Performance evaluation criteria

The performance of a network depends not only on the exact implementation, but also on the individual associated with the network. M Ilyas and H T Mouftah [22] have divided the performance criteria of a network into three categories, the user-orientated, manager-orientated and designer-orientated criteria.

4.1.1 User-Orientated criteria

A user is usually concerned with the 'friendliness' of a network. A user requires a network that provides short delays in transferring messages across the network at a minimum cost. In addition a user may require security and privacy of data.

4.1.2 Manager-Orientated criteria

A manager of a network is concerned with the cost-performance tradeoff of the network, and likes to see the best utilization possible of the network resources. A manager wants a reliable network with a high throughput and is interested in the type and adaptability of the routing algorithm used in the network.

4.1.3 Designer-Orientated criteria

A designer is mainly concerned with how accurately the performance of the network can be predicted and the design specifications met. The designer is also concerned with aspects such as flow control protocols, buffer utilization and the efficiency of the routing procedure used.

4.2 Performance measurements

To determine the performance of a network several measurements must be obtained from the network.

4.2.1 Time delays

One of the most important aspects to consider when evaluating the performance of the network is the time taken for a packet to move from a source to destination node in the network. This criteria is dependent on the amount of traffic in the network. Hence the time delays for various traffic loads must obtained before a useful deduction can be made about the effectiveness of the network. Measurement of the time delays often gives a good indication of the efficiency of the routing algorithm used in the network.

4.2.2 Throughput

The throughput of the network indicates the number of packets per second moving through the network with respect to the total capacity of the network. It is often desirable to maximize the throughput of the network. High throughput are obtainable by the use of high capacity links, efficient nodes and highly effective routing algorithms.

4.2.3 Cost

The cost of running the network and the cost involved in transferring a data packet across the network is an important aspect to consider when evaluating the performance of the network. Other cost that may be considered are maintenance costs and the costs involved in adding a new node or link to the network.

4.2.4 Percentage use of the network capacity

The percentage use of the links in the network under various traffic loads need also be considered. At high traffic loads it is desirable to have all or most of the links operating at or near maximum capacity. A measure of the average percentage use of each link reveals the particular link that becomes saturated under heavy load conditions, and by increasing the capacity of this link the throughput of the network may be increased. Often it is necessary to predict exactly how much traffic the network is able to handle in order to plan for possible future expansions of the network.

4.2.5 Number of blocked packets

In some applications, such as banking a blocked packet may have a disastrous effect on the network and it is necessary to be able to determine the possibility of a packet being blocked under various load situations. Determining the number of packets blocked over a particular time interval is an indication of the buffer sizes required at the nodes. A network should also be able to detect blocked packets and to take appropriate action if necessary.

4.2.6 Use of buffers

The average percentage use of the buffers at each node provides an indication of the buffer size required at each node and may also indicate if the node is capable of handling all the input traffic to the node. Under-utilization of buffers at heavy loads indicates that an excess number of buffers have been provided or that the node is particularly efficient. Over-utilization of the buffers indicates that too

few buffers have been allocated and that blocked packets or deadlock may occur.

4.2.7 Average number of hops

The number of hops represents the number of nodes crossed by a packet travelling from a source to a destination node. *Minimizing the number of hops usually, but not always,* minimizes the delay and cost involved in transferring the packet. The number of hops may indicate the presence of 'looping' or circulating packets in the network.

4.2.8 Other performance measurements and conclusions

Other performance measurements may be directly related to the actual implementation of the network. For example if a network employs a distributed adaptive routing algorithm it *is necessary to measure or calculate* the ratio of routing packets to data packets in the network. If the ratio is too high it indicates that too many routing packets are circulating around the network and consuming valuable bandwidth.

There exists numerous measurements that can be made to determine the overall performance of a network and the routing algorithm used. Usually the overall performance is established by considering all of the measurements simultaneously. For example it is not much use minimizing the delays if the number of blocked packets *increases. Obtaining* a quantitative measure of the efficiency of a routing algorithm and computer network as a whole is a difficult and complex task.

4.3 Performance evaluation: Analytical techniques

For this method mathematical models are used to determine the performance of the network. The models are usually based on several assumptions, and the more the assumptions made the easier the method, but the results obtained are less accurate. Highly accurate results require sophisticated and complex models of the system and the accuracy obtained does not usually compensate for the efforts and complexity involved.

Evaluating the performance of a single node employing a single server system is simple and accurate results can be obtained using analytical methods, however the evaluation of multi-node systems becomes extremely difficult if not impossible to analyse. In fact in multi-node systems it is only possible to accurately predict the performance of networks employing a static routing convention.

4.3.1 Queuing theory analysis

Assuming a static routing algorithm valuable insight about delays, costs, percentage use and buffer allocation can be gained using simple queuing theory. Basic link capacity requirements can usually be determined from simple analytical methods, assuming constant traffic and a static routing algorithm.

In a simple network assuming infinite buffer lengths and an M/M/1 queuing principle as discussed in chapter 2 the delay can be estimated from the simple formula:

$$T_i = 1/(\mu C_i - \sigma_i)$$

T_i is the delay in link i, C_i is the capacity of link i and
$1/\mu$ is the average service rate of node i and
σ_i is the average arrival rate of link i.

Summing over all the links the total average delay T, can be expressed as:

$$T = \delta\Sigma(\phi_i/\phi)/(\mu C_i - \phi_i)$$

where δ is expressed as $\phi/\emptyset$ and

ϕ is $\Sigma\phi_i$ the sum of the individual arrival rates and

$\emptyset$ is $\Sigma\emptyset_{ij}$ where $\emptyset_{ij}$ is the total traffic from source i to the destination node j.

The formula was used on a simple network illustrated below.

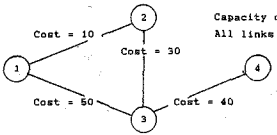


Figure 4.1 A simple digital network under test.

The performance of the network was analytically determined as follows:

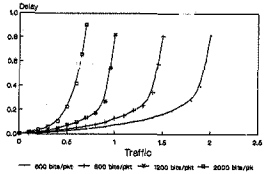
The input to the network is the average traffic rate for all the source destination node pairs in the network. In this example the total traffic is assumed to be 40 packets per second. The total traffic is then multiplied by a scale factor which varies from 0 to 2. That is the input traffic to the network is varied between 0 and 80 packets per second. The average packet delay is then calculated. The average percentage use of the network is also determined. The percentage use of the network is defined as the total number of packets travelling through the network per second divided by the total capacity of the network. The effect of the

packet length, input traffic loads and link capacity values on the average packet delay is observed. In the analysis a simple least cost routing algorithm is assumed. Packet arrival times to the network are calculated using a Poisson probability density function.

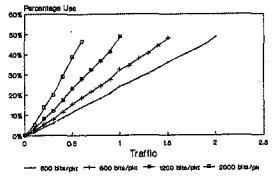
Packet bit lengths

The average packet delay as well as the percentage utilization of the links in the network for various packet lengths are studied.

It is important to note that the infinite delays illustrated in the graphs below are a result of assuming infinite buffer lengths at the nodes. It is assumed that no packets are blocked and that all packets eventually reach their destination, even if it takes forever!



Graph 4.1 The average packet delay as a function of the input traffic to the network for various packet lengths.

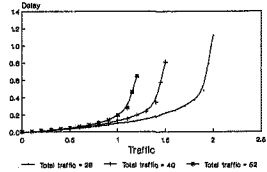


Graph 4.2 The percentage use of the network as a function of the input traffic to the network for various packet lengths.

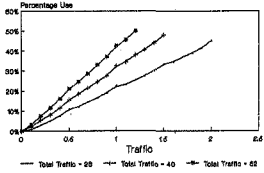
From the graphs it is evident that the shorter packets have a shorter average delay. It can also be noted that the delay increases rapidly at some critical value. This occurs when one of the links becomes saturated. The link that becomes saturated is known as the saturation link. For smaller packets the effect of the saturation link on the delay is less critical compared with that of large packets. Allowing the flow to increase close to the capacity of any link dramatically increases the delay illustrating that high throughput and low delays are difficult to achieve simultaneously. In all cases the peak utilization of the network is approximately 50% and is dependent on the capacity of the saturation link.

Traffic rate results

Increasing the traffic has the result of increasing the average packet delay in the network. For all traffic rates the maximum percentage utilization of the network is 50%. The results obtained are illustrated in the graphs below. The graphs are obtained using a packet length of 800 bits.



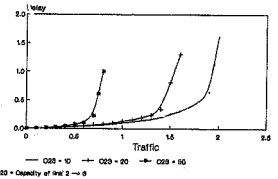
Graph 4.3 The packet delay for various input traffic loads.



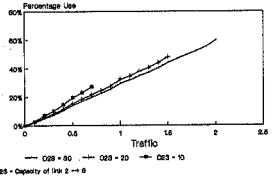
Graph 4.4 The percentage use of the network for various input traffic loads.

Various link capacity results

The capacity of the saturation link (link (2,3)) was increased and the results are portrayed in the graphs shown below. The packet length is assumed to be 800 bits.



Graph 4.5 The packet delay versus the input traffic load for various link capacity values.



Graph 4.6 The percentage use of the network versus the input traffic load for various link capacity values.

Increasing the capacity of the saturation link (link (2,3)) decreases the average packet delays. The decrease in average packet delays becomes less significant as another link becomes saturated. Increasing the capacity of the saturation link also increases the percentage utilization of the network. The problem of a highly saturated link can be solved by increasing the capacity of the link but this solution suffers from catastrophic results if the link fails. A more appropriate solution is to provide several links parallel to the saturation link. In an ideal network it is desirable for all the links to have the same percentage utilization. In a static network with constant traffic flows this can be accomplished by correctly assigning capacities to the various links in the network. For dynamic networks were traffic rates vary the only solution is to employ a sophisticated routing algorithm to distribute the traffic as evenly as possible across all links in the network. Such a routing algorithm is very useful in networks where alternate links to the saturation links are available.

4.3.2 Single node analysis

Analytical techniques are extremely useful in designing and predicting the performance of individual nodes in the network. Analytical methods are useful for determining buffer length requirements, buffer allocation techniques and service rates required by the node.

G B Agnew and J W Mark [1] have developed a model for determining the performance of individual nodes in a network. A certain buffer length, buffer allocation technique and service rate of the node under test is stipulated. The performance of the node is determined by calculating the blocking probabilities and delays of the packets at the node. The percentage buffer utilization may also be calculated. The

model applies to static and adaptive routing algorithms. The modal is useful in determining the stability and convergence of a particular routing algorithm. The overall performance of large complex networks cannot be accurately predicted using this method.

4.3.3 Electrical analog analysis

The method of using an electrical circuit analog for determining the performance of a network was first proposed by J B Dennis and is described by T E Stern in reference [41]. The links in the network are represented as resistances, the traffic flow in the links as current and the each node is given a potential. The method is useful obtaining an optimal routing strategy for the n. Several assumptions do however have to be made in orde able to analyse large multi-destination networks. The metho requires further study before it may be applied to large dynamic digital networks.

4.3.4 Analysis of quasi-static networks

A Segall [38] states that the problem with analysing a dynamic system using analytical techniques is the number of states required to obtained accurate results. A network consisting of N nodes, a outgoing links per node and maximum buffer length of m packets requires approximately $(aNm)^{N-1}$ states. Taking a small network of 10 nodes with 2 outgoing links per node and a buffer length of 10 the number of states required is greater than 5×10^{20}. Segall has proposed a model for the analysis of quasi-static networks which reduces the number of states required. The proposed algorithm is however limited and requires further investigation and improvement.

4.3.5 Conclusion, analytical methods

Various analytical techniques for evaluating the performance of a digital network have been considered. A static network consisting of four nodes, each node modelled by a M/M/1 queuing system was analysed using simple queuing theory. The results obtained indicate that the packet delay is dependent on the input traffic to the network and the total capacity of the network.

Simple analytical methods can be used as a first step in the design of a digital network. The main problem being that for large networks the complexity increases drastically. Analysing a network employing a dynamic routing algorithm is difficult. In conclusion analytical techniques are limited to determining the performance of simple static networks.

4.4 Performance evaluation: Experimental techniques

A network or prototype network is first implemented. Measurements are then taken from the network. Various parameters are now varied in order to obtain the optimum performance of the network.

4.4.1 Measurement techniques

Before any adjustments can be made to improve the performance of the network various measurements must be made such as the measurement of the input traffic, queue lengths and link utilization. The measurements can be classed into three types, namely long term measurements, trace measurements and the snapshot measurements.

- Long term measurements are obtained by recording various network statistics such as queue lengths for a fixed period of time, and using this recorded information long term average statistics are compiled.
- Trace measurements are made by observing a packet as it travels through the network from the source node to the destination node. This method of measurement is used to determine if any deadlocks or loops exist in the network, the number hops, and packet delay.
- Snapshot measurements give an instantaneous state of the network. These are preferable to long term measurement techniques due to the reduced overhead. Snapshot measurement techniques are used to determine queue lengths at the nodes.

4.4.2 Conclusion, Experimental techniques

This method, although the most accurate method suffers from many disadvantages.

- the network must be assembled before measurements can be made. Drastically altering the network after it has been assembled may be costly.
- the time taken for measurements to be made, the data to be analysed and network parameters altered may take several weeks during which the network may not be operating at an optimum.
- it may be difficult to know exactly which parameters must be varied and how these parameters should be altered to improve the performance of the network. Many of the parameters are usually dependant on each other and one parameter cannot be replaced without effecting other parameters in the network.
- measurements must be made constantly requiring a lot of overhead.

A good example of the uses of experimental techniques is the ARPANET network where two hours a week are reserved for software maintenance. This was used to conduct experiments on the network. The experiments were successfully used to 'debug' and optimize the new routing algorithm. In large networks such as the ARPANET network such experiments are however costly and time consuming.

Experimentation techniques are only practical for the fine tuning of the network parameters. The method may also be useful if a network is expanded or altered to such an extent that the performance of the network drops noticeably. The experimental technique is the most costly of the three methods discussed, but is the most accurate method available particularly for large dynamic and complex networks.

4.5 Performance evaluation: Simulation techniques

Simulations are performed with the aid of a computer and can be made extremely flexible. Simulation is often the only feasible method of predicting the performance of a network. As the complexity of a network increases the number of assumptions that must be made in order to analytically predict the performance of the network may render the results useless. Simulations are normally time consuming, but the results obtained are usually far more accurate than those of analytical methods due to the fewer assumptions that have to be made.

The computer mimics the events that occur in a network. Static networks as well as complex dynamic systems may be simulated. The advantage of computer simulations is that it is relatively inexpensive, easy to modify and adaptable. A simulation can be made as accurate as desired at the expense of an increase in simulation time.

4.6 Conclusion: Performance evaluation techniques

It is impossible to single out one performance evaluation technique as being the best. Some of the aspects that must be considered when selecting a performance evaluation technique are:

- Type of network: The type of network, number of nodes, topology of the network and routing algorithm.

- Accuracy: the accuracy of the delays, blocking probabilities and throughput of the network.

- Flexibility: Allowing the performance of a network using various topologies, routing algorithms and protocols to be evaluated.

- Turnaround time: The time taken to obtain measurements from the network, to analyse the results and to successfully improve the performance of the network is often of vital importance.

- Cost: Cost always plays a vital role in the design, improvement and implementation of any system. Often a compromise between cost, accuracy and flexibility must be obtained.

Analytical methods are most useful in small static networks or in the first stage design of a network. Experimentation techniques are often expensive and most useful in the final tuning and optimization of a network. Simulation is the most accurate, flexible and cost efficient method of analysing fairly large dynamic networks.

5 *NETWORK SELECTION*

The purpose of the project is to examine various routing strategies and protocols in a digital network and to optimize the algorithm for the particular network. Before a routing algorithm can be selected a suitable network must be chosen and the specifications and requirements of the network properly defined.

5.1 Network requirements

The network is to be a digital computer based network. Digital networks are extremely important due to the increase in the use of computers throughout the world. The exchange of digital information between computers is becoming a necessity. The network is used to interconnect several nodes or users. The network must allow any user in the network to transfer digital data to any other user connected to the same network. The network must be efficient and reliable under all traffic load situations. The routing algorithm must be able to cope with bursty traffic situations. Bursty traffic is a characteristic of digital networks. The routing algorithm used must be stable, display loop free characteristics and minimize the delay and possibly the cost of transferring a message between two nodes. The network must be free of deadlocks and be able to support broadcasting. Changes to the topology of the network should be possible without having to physically update the routing tables of all of the nodes in the network.

The specifications of the network are typical requirements needed in modern digital networks. The specifications may be slightly idealistic and an optimum network meeting as many of the above requirements is being sought.

5.2 Selected network

Using the network specifications described in the above paragraph a suitable network topology is selected and some of its features are described.

5.2.1 Network topology

The first step in designing a network is to select an appropriate topology. The type of network topology is dependent on several factors:

- the physical distribution and location of the nodes in the network,
- the type of network and the services supplied by the network,
- the reliability required,
- the type of traffic,
- the number of nodes in the network,
- the number of users to node ratio, and
- the cost of connecting a pair of nodes in the network.

If the network is to support numerous users distributed over a large area, a hierarchical network topology is the best suited. If the number of nodes in the network is small then a ring topology or broadcast channel topology may be adequate. If the number of users to node ratio is high then the use of concentrators in the network may be necessary to accommodate all the users.

In our network we will assume that the users, or group of users are distributed randomly over the area spanned by the network. The network is a multi-commodity network consisting of numerous source destination node pairs. This leads to a asymmetrical distributed network with randomly distributed

nodes and were links connect directly adjacent nodes. Links are also provided between nodes that frequently exchange traffic. Distributed networks allow links and nodes to be easily added or removed from the network compared to hierarchical and ring type networks.

To increase the reliability of the network at least two paths must be made available to most of the nodes. The network will employ a decentralized routing strategy in order to eliminate the vulnerability of a centralized system. A hierarchical network is unsuitable since it is vulnerable to a network failure if a node or link at the top of the hierarchy fails.

The figure below indicates an example of a distributed network topology that will be used in the project.

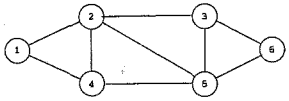


Figure 5.1 Diagram of a typical distributed network.

5.2.2 Network protocols

Now that the topology of the network has been defined it is necessary to decide the type of service that is to be provided by the network. The basic protocol used in transferring data to a destination node need also be considered.

Type of service

In a virtual circuit network the path from the source to destination node remains constant for the period the user is connected to the network. In a datagram network each packet is treated as a separate entity and packets belonging to the same message or user may be routed along different paths to the destination node. In the network selected a datagram circuit will be assumed. A datagram circuit allows a greater adaptability to changes in traffic in the network. This allows for a more efficient use of the bandwidth of the network and permits the user to rapidly transfer data to differing destinations in the network.

Circuit, packet or message switched network

Three techniques exist in transferring messages from a source node to a destination node.

A circuit switched network is a network where a physical connection is made between the source and destination nodes. Circuit switching is most useful in analog circuits and networks were delays between messages are intolerable such as the telephone service.

An alternative to circuit switching is message switching. No physical connection is made in advance between the source and destination nodes. The data is stored at each node along the path *and forwarded to the next node after the data has* arrived at the node. There is no limit on the size of the data that may be transferred. Each node must have disk storage facilities if the messages are extremely long. A disadvantage of such a system is that if transmission errors occur then the whole message must be retransmitted. Also a

long packet may block up a particular link for a long period of time, rendering the line useless to other traffic.

The last method is packet switching. In packet switching long messages are broken up into smaller packets. This method has several advantages:

- Each packet because of it's small size may be stored in the random access memory of each node along the path,
- the packets are small and therefore no single user can monopolize a link for more than a few milliseconds,
- Packets can be individually routed to the destination node, independent of the other packets increasing the throughput of the network, and
- If a packet contains transmission errors only the single packet, and not the whole message is retransmitted.

The basic disadvantage of a packet switched network is that the packets may arrive out of order at the destination node. The problem may be rectified by reordering the packets at the destination node. The network used in the project is assumed to be a packet switched network.

5.2.3 Node specifications

Having defined the network topology and type of service provided by the network the structure of a node in the network is now defined. The node consists of a single server system. The server is a dedicated processor devoted to the routing of packets around the network. Arriving packets are serviced on a first come first serve basis. Arriving packets are inserted into an input queue, the buffers being obtained from a common input buffer pool. After a packet has been serviced the buffer is returned to the common buffer pool. Routing information is stored internally at each node in the

network. All the nodes in the network are identical except that differing nodes in the network may have differing service rates and a differing number of available buffers. A schematic representation of a typical node in the network is illustrated below.

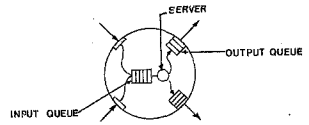


Figure 5.2 Schematic representation of a node.

5.2.4 Link specifications

Full duplex links of typically 9,6 kilo bits per second (kbps) to 48kbps capacity is used in the network. Various links in the network may have varying capacities. Allocating capacities to the various links in the network is important if a large maximum throughput is required. If the average traffic flow rate between each commodity, source destination pair, in the network is known then various algorithms can be used to assign the capacity values to the links in the network. If a reliable high capacity link is required between two nodes it may be preferable to insert two or more parallel links between the nodes.

In order to accurately predict the required capacities of each link it is necessary to determine the actual average traffic in each link. This may depend on the routing algorithm used. In a dynamic algorithm this may be a difficult task. Often a static least cost algorithm is used to predict the average expected traffic in each link of the network. Methods to predict the traffic in each link in the network under dynamic conditions include experimental and simulation methods. The experimental technique is expensive since changing the capacity of a link after the link has been installed is usually costly. Simulation techniques are useful in assigning capacities to the links in the network, since the capacities of the links can easily be altered until an optimum solution is found.

The links in the network are full duplex channels and of fixed capacity. The capacity of a link may be different in each direction. That is the capacity of link (i,k) does not have to equal the capacity of link (k,i).

5.2.5 Packet specifications

Typically a packet or frame consists of several fields, namely:

- a start flag depicting the start of a packet,
- a destination address,
- control data containing sequence numbers, acknowledgements, routing information and perhaps other information,
- data to be transmitted to the destination node,
- a checksum used by the destination node to detect any errors that may have occurred during the transmission of the message and,
- a stop flag depicting the end of the packet.

Start flag	Destination address	Control field	Data field	Checksum	Stop flag

Figure 5.3 Typical packet or frame format.

For the project the packet is assumed to have a similar format to that shown above. The length of the packet may be a constant, but usually a maximum length is stipulated. The maximum length allowed is dependent on a number of factors such as the average size of the messages or data to be transmitted, the size of the buffers at the nodes and the capacity of the links. Short packets are desirable since the packet delay is directly proportional to the packet length. If packets are made too short then too much overhead is required to transfer the small portion of the data. In the project the average data packet lengths vary between 800 bits per packet to 2000 bits per packet.

5.3 Routing requirements

The routing algorithm is required to be efficient, reliable and to minimize the average packet delay. The routing strategy is to be employed in a decentralized manner. The algorithm must be able to adapt to changes in the topology of the network. The algorithm must minimize the cost in transferring a message across the network if necessary. The algorithm is therefore divided into two separate algorithms, a least cost algorithm an a adaptive minimum delay algorithm.

5.3.1 The least cost routing algorithm

The least cost algorithm must minimize the cost in transferring a message across the network. A constant cost per link is to be used to calculate the cost in transferring

a packet to the destination node. Therefore a static least cost algorithm is required. The cost of a link may depend on the physical length of the link, the capacity of the link or the average percentage utilization of the link. Several efficient least cost algorithms exist and a typical least cost (shortest path) algoritnm is given in [43]. The least cost algorithm may be used as a back up algorithm to a more complex adaptive routing algorithm.

5.3.2 Adaptive routing algorithm

The adaptive routing algorithm must be simple and robust. Packets must be transferred to the destination node as quickly as possible. The algorithm to be tested is based on R G Gallager's minimum delay routing algorithm since it is simple, easy to modify and may be adapted to any particular type of network. More elegant algorithms have been proposed such as D G Cantoor's and M Gerla's second derivative routing algorithm [8]. These algorithms are however complex and computationally intensive. A derivation and explanation of R G Gallager's routing algorithm is provided in Appendix C. The algorithm is only intended for 'quasi-static' networks, but the performance of the algorithm under dynamic conditions will also be tested. Improvements will be made to the algorithm and the performance of the algorithm is to be optimized.

5.3.3 Conclusion: Routing algorithms

The requirements of the static and adaptive routing algorithms to be used in the network have been specified. The performance of R G Gallager's adaptive routing algorithm compared to a static least cost algorithm is investigated. The limitations of the algorithms are investigated and

decisions are made as to which type of network each algorithm is best suited.

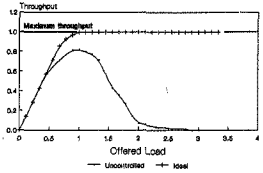
5.4 *Congestion protocols and buffer* allocation techniques

Congestion control protocols form an important part of the network layer of the OSI reference model. The congestion control protocols and buffer allocation techniques used have a direct impact on the efficiency of the routing algorithm. The main function of congestion control is for:

- the prevention of throughput degradation and loss in performance due to the overloading of the network,
- the fair allocation of resources among competing users and prevent the over allocation of the resources in the network,
- deadlock avoidance and,
- flow control, ensuring that the source node does not provide data at a rate exceeding the input capabilities of the destination node.

5.4.1 Congestion control

Congestion results from too many packets being present in the network or part of the network. If the demands made on the network exceed the capacity of the network then congestion may occur. The diagram below illustrates the typical throughput for a network without the implementation of congestion control compared to a network with congestion control.



Graph 5.1 Throughput versus offered load for a uncontrolled network compared with the throughput of an ideal network.

Congestion control is used to keep excess traffic out of the network. Congestion control decides when to drop packets, either at the entrance of the network or within the network. Congestion control is best performed by preventing new packets entering into the network at high loads and favouring the packets already in transit in the network. In order for a network to display good congestion control characteristics it is essential that good flow control techniques are used. Flow control techniques are divided into four classes or levels (M Gerla [13]):

- Hop level flow control,
- Entry-to-exit level flow control,
- Network Access level and,
- Transport level flow control.

The definition and uses of each level of flow control are explained:

Hop Level Flow Control

This level of flow control attempts to keep a smooth flow of traffic between two neighboring nodes. Hop level flow control is used to increase throughput and eliminate deadlocks. The type of flow control scheme used here is dependent on the type of network used. Assuming two nodes i and k are directly interconnected by a link. Packets are transferred from node i to node k as illustrated below.

Figure 5.4 Diagram of two inter-connected nodes.

If node i is able to provide packets at a rate faster than the capacity of link i,k then packets will accumulate at the output queue of node i. If packets arrive at node k at a rate faster than node k can service the packets then packets will accumulate at the input queue of node k. One method to prevent congestion at the input queue of node k, is to allow node k to discard incoming packets at will. Another method is for node k to inform node i that it must slow down the rate at which packets are being sent to node k. This is fine if node i is generating all the packets to node k. If node i is receiving packets from other nodes destined to node k, then node i must in turn inform each of these nodes to decrease the rate at which they are sending the packets to node k. This process is time consuming, especially in the case of unidirectional links. By this time congestion may have already occurred. Discarding packets at will has serious consequences if the packets discarded contain important data or vital routing information.

Buffer allocation techniques play an important role in hop-level flow control and various buffer allocation techniques are provided in section 5.4.2.

Entry to Exit Level Flow Control

The main objective of entry to exit flow control is to prevent buffer congestion and deadlocks at the destination node. If packets arrive out of order at the destination node the destination node must reassemble the packets in the correct order. This may result in the destination node waiting indefinitely for a missing packet and deadlock results. In the project we are only concerned with the transferring of individual packets and defining protocols required for packet reordering is a complex task which is beyond the scope of the project.

Network Access Level Flow Control

The objective of network access flow control is to prevent external traffic from entering the network during periods of high traffic flow. The traffic in the network may be determined from the number of buffers available at each node. In a distributed network this means that nodes must frequently exchange information about the state of the buffers at each node. During congestion buffer information must be exchanged rapidly between the nodes in the network. This takes up valuable network bandwidth.

One method of solving the problem is to allocate a certain number of permits to the network. A packet may only enter the network once it has obtained a permit. The permit is released to the network once the packet has arrived at the destination

node. This method has its problems. Permits must be distributed fairly to each node in the network and lost permits must be detected. A certain overhead and bandwidth is required to distribute the permits around the network.

Often it is sufficient to monitor the number of available buffers at the entry node of the network. Congestion conditions created internally in a network usually propagate back to the entry nodes of the network due to the back pressure effect. This allows simple protocols to be used to limit the number of packets entering the network.

Transport Level Flow Control

The main function of transport level flow control is to efficiently and reliably transmit messages across the network. The protocol attempts to transform a datagram service into an apparent virtual circuit between the two nodes and is also concerned with the reassembly of messages at the destination node.

5.4.2 Buffer allocation techniques

Buffer allocation techniques are important in the prevention of congestion in a network. Numerous protocols exist for the allocation of a buffer to an incoming packet at a node. Usually a node consists of a large buffer pool containing available buffers. Buffers are removed from the buffer pool and used to store incoming or outgoing packets at a node. Buffers are allocated as illustrated on the next page.

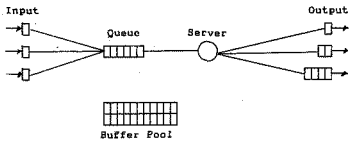


Figure 5.5 Typical buffer allocation at a node.

Various techniques exist to prevent a single input or output queue from monopolizing all the buffers in the buffer pool. The basic buffer allocation techniques are now discussed:

Complete partitioning

If N is the number of queues (input or output) at a node, n_i the number of packets in the ith queue and B the total number of available buffers at the node then the following constraint applies:

$$0 \leq n_i \leq B/N \quad \text{for all } i$$

This implies that the buffers are equally shared between the queues at the node. The advantage of such a system is that no single queue can obtain all the available buffers. If a single queue requires more buffers than B/N then the queue is unable to obtain the buffers even if the other queues are empty. This is a serious disadvantage of this method.

Sharing with maximum queues

Let b_{max} be the maximum queue size allowed where $b_{max} > B/N$ and the following constraints hold.

$$0 \leq n_i \leq b_{max} \quad \text{for all } i \text{ and}$$

$$\Sigma n_i \leq B.$$

This method allows a maximum allowable buffer allocation for each queue. Any one queue is therefore prevented from obtaining all the available buffers in the buffer pool. The disadvantage of the method is that since $b_{max} > B/N$ it is possible for an input queue to be left without any buffers.

Sharing with minimum allocation

Let b_{min} be the minimum buffer allocation guaranteed to each queue, where $b_{min} \leq B/N$ then we have,

$$\Sigma \max(0, n_i - b_{min}) \leq B - Nb_{min}$$

This type of buffer allocation guarantees a minimum number of buffers to each queue. The problem remains that unused buffers may remain in the buffer pool unobtainable by a queue which may require the buffers.

Sharing with minimum allocation and maximum queue

This is a combination sharing with maximum queues and sharing with minimum allocation and provides minimum buffer guarantee and maximum buffer allocation to each queue simultaneously. This may seem to be the best technique since no queue can

obtain *all the available buffers from* buffer pool and each queue is guaranteed a minimum number of buffers.

5.4.3 Conclusion: congestion control and buffer allocation

In the project it is assumed that each node has the liberty of discarding packets at will to prevent excess traffic from entering the network. The probability of a packet being discarded is carefully observed to determine if it is necessary to employ more sophisticated congestion control techniques. Using a simulation to *predict the performance of* the network it is easy to evaluate the performance of the network using various buffer allocation techniques.

5.5 Conclusion: Network selection

In the chapter the basic network topology to be tested has been defined. The routing algorithms, protocols and buffer allocation techniques have been specified. The requirements of the network have been stated and the network will be tested to see if the requirements can be met. *Additional* protocols and routing strategies may be added to the network to optimize the performance of the network.

6 NETWORK SIMULATION

A computer simulation program is used to predict the performance of the network defined in chapter 5. The computer simulation program is used to simulate the network and associated routing algorithm. A simulation technique is used since it is the most accurate, least expensive and flexible method available.

The chapter describes the requirements of the program and the basic network operations needed to accurately simulate a digital network. A detail description of the routing algorithms simulated in the simulation program is provided. A description of the development and implementation of the program is given and this is followed by a list of assumptions and protocols used in the program. A detail description of the design of the program is provided in Part II of the report.

6.1 Program requirements

Before defining the operation of the program it is essential to define the properties required by the simulation program. It is necessary to specify the operation of the software package from a user's point of view. The software package must be able to simulate any type of network topology entered by the user. The program must be capable of simulating more than one routing algorithm so effective comparisons between various types or routing algorithms may be made. The program must be able to simulate the response of a network to various input traffic loads. A user's manual of the simulation package is given in Part II, chapter 2 of the report. The user's manual provides the requirements of the simulation package from a user's point of view.

5.1.1 Program inputs

The basic inputs to the program can be listed as follows:-

- the duration of the simulation,
- the network topology,
- cost and capacities of the links in the network,
- the service rate and buffer lengths of the nodes in the network,
- the input traffic pattern to the network, and
- the type and specifications of the routing algorithm to be used in the network.

Using various combinations of inputs a large number of simulations may be performed. Numerous inputs makes the simulation program more flexible but more complex and difficult to use. Hence a compromise between flexibility and ease of use of the program must be found. The input requirements may also be dependent on the user. For example if the user has a fixed network the user may not want to enter the network topology. The user may only want to alter the parameters of the routing algorithm to optimize the network. If too many input variables are required by the program the probability of entering an incorrect variable increases. The results obtained from the simulation may then be incorrect.

It is often necessary to vary the average expected traffic to the network during a simulation. To facilitate this requirement the duration of the simulation is divided into regular time intervals. The expected traffic for each commodity may be stipulated for each interval. This allows dynamic networks to be simulated. The program is designed to provide maximum flexibility by allowing the user to enter the large amount of input data required by the program via an input file.

6.1.2 Program outputs

The outputs to the program provide the results of the simulation. It is important that useful results are provided by the program. The basic outputs required from the program are:

- The average packet delay,
- the percentage utilization of the network,
- the number of blocked packets during the simulation,
- the generated input traffic to the network,
- the average cost of transferring a packet across the network and,
- the number of routing packets generated.

The results are calculated at regular intervals during the simulation so that the dynamic behavior of the network may be observed. At the end of the simulation long term averages such as the total average packet delay and total costs are calculated. The collaboration of all the outputs of the program must be sufficient as to provide an accurate prediction of the overall performance of the network.

6.2 Simulation of a digital network

A description of the processes in a digital network that must be simulated is provided. The problem of mapping a complex system consisting of numerous interrelated concurrent processes into a serialized single process by the use of an event queue is explained. The operation of a single typical node in the network is considered. This is followed by a discussion on how the operations of the nodes in the network are interrelated.

6.2.1 The event queue

Each node in the network consists of a single dedicated processor. Each node operates concurrently with all the other nodes in the network. The functions performed by the nodes are interrelated by the packets which are transferred between the nodes. Simulating a network on a single processor computer requires the serialization of the processes performed in the network. The processes or tasks performed at a node can be assumed to be initiated by a particular event occurring at the node. To accurately simulate the performance of a network it is sufficient to define three events. These events occur at any node in the network. The events are:

- The arrival of a packet at the node,
- the arrival of a packet at the server of a node and,
- the departure of a packet from the node.

An event occurs at a particular instant of time and has no duration. An event may initiate a process or task which has a finite duration. The events occurring at each node in the network are serialized by inserting the events into a single event queue in chronological order. During the simulation of the network the events are extracted sequentially from the event queue. The processes associated with each event are then executed to completion before the next event is extracted from the event queue. The use of an event queue has the advantage that numerous concurrent processes are serialized. This simplifies the design of the simulation program since only a single event is considered at a time. The disadvantage of such a system is that the time taken to complete a simulation may be extremely lengthy.

6.2.2 Operation of a network node

A node basically consists of an input queue, single server and an output queue for each neighbouring node. The basic operations performed by a node are:

- to generate traffic destined for other nodes,
- to update the routing tables at the node,
- to calculate and transmit routing information to other nodes,
- to provide flow control and
- to consume data packets or to route data packets to other nodes in the network.

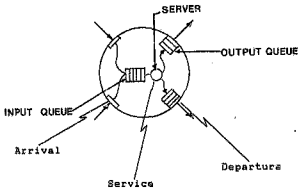


Figure 6.1 Diagram of a typical node

The capacity of the server is assumed to be less than the combined capacity of the input links. This implies that the queues will accumulate at the input of the node rather than at the output. Each input link is able to obtain a buffer from a common input buffer pool. The packets arriving at a node are inserted into the input queue on first come first serve basis. Each output queue is assumed to contain the same number of constant available buffers.

The status of the node can be described by the:-

- service rate,
- number of buffers in the buffer pool,
- number of packets in the input queue,
- number of packets in the output queues, and
- the status of the routing data stored at the node.

The routing data can be thought of as a resource on which two basic operations can be performed. The routing data may be used to determine the path taken by a packet at the node or the routing data itself may be updated. These two basic operations are important in the design of the simulation program.

The three events, namely the arrival, service and departure events are described.

Arrival process

When a packet arrives at a node several operations must be performed. Firstly the input buffer pool is checked to see if it is empty, if it is empty then the incoming packet is simply discarded. If the buffer pool is not empty and the input queue for the link on which the packet arrived is not full, a buffer is removed from the buffer pool for the packet. The packet is then inserted into the input queue.

In the program the task of inserting a packet into the queue is simulated by calculating the time of service and inserting the event into the event queue. Newly generated packets are considered to be arrivals to the source node.

Service process

The service process is a complex op ion. The server must decide whether the incoming packet contains routing information and if so the appropriate action that must be taken to service the routing information. If necessary the routing tables at the node must be updated. If the packet contains data the server must decide whether to consume the packet or to route the packet along one or more outgoing paths. If a packet is to be routed the server must decide which output path the packet is to be sent on. If the packet has looped the server may discard the packet. If a packet is to be sent on an output path, the output path is checked to see if it is not full, if it is not full the departure time is calculated and the event is inserted into the event queue.

Departure process

This process represents the departure of a packet from a node. The arrival time to the next node is calculated from the capacity of the output link and the length of the packet. The event representing the arrival of a packet at the next node is then inserted into the event queue. It is assumed that a packet leaving node i to a neighbouring node k will reach node k free of transmission errors.

By simulating the three events described above the operation of a node may be accurately imitated in the simulation *program*. A flow chart of each of the processes described above is provided in Part II, Appendix A of the report.

5.2.3 *Network operation*

The operation of an entire network may be simulated simply by simulating each node in the network individually. The only impact one node has on another node in the network is via the packets exchanged between the nodes. Once a packet has left a node it no longer influences that node. This reduces the complexity of designing a simulation program. The formulae used to calculate the various times of the events in the network is given in Appendix C.

6.3 Static routing algorithms simulated

In order to test the validity of the simulation program simple static routing algorithms were first tested. The two static algorithms tested are the flooding routing algorithm and the least cost routing algorithm. The static routing algorithms form a standard to which other routing algorithms may be compared. The implementation of the algorithms in the simulation program is described in detail.

6.3.1 Flooding routing algorithm

The flooding algorithm is implemented by sending the packet on each link except the link on which the packet arrived. If a packet has reached its destination it is consumed and not retransmitted. This method of routing leads to numerous packets looping around the network. To remove the circulating packets the following policy has been adopted. Each packet, in its header, stores the number of nodes the packet has passed (number of hops). If this number exceeds the length of the longest path in the network then the packet is discarded. Currently the length of the longest path has been made equal to the number of nodes in the network. This approach ensures

that all circulating packets are eventually removed from the network. Other more sophisticated techniques for removing circulating packets exist. For example each node may keep a record of the identification number (ID) of the packets that have been serviced and retransmitted by the node. If the ID of a packet arriving at a node matches an ID from the record at the node then the packet is discarded. This method requires large tables that must be frequently be modified and updated.

The flooding algorithm is the simplest of all routing algorithms and was therefore the first algorithm implemented in the network simulation program.

6.3.2 Least cost routing algorithm

The least cost routing algorithm is used to route packets along the cheapest path in the network. Two methods for implementing the algorithm exist.

For the first method each node stores only the cost of the links to the neighbouring nodes. Packets are then sent along the output link with the least cost. The advantage of this method is that each node only requires to know the cost of the links directly connected to the node. The node must however know which output links lead to which destination node. The entire path taken by packet may not be the least cost path. Another problem with this method is that packets may loop in the network. Looping of packets occur since each node makes its routing decisions independently of the other nodes in the network.

In the second method each node stores the cost of all the links in the network. Hence the cost of the entire path to each destination node may be minimized. The least cost

routing algorithm used in the simulation is based on this method. The actual least cost algorithm used in the simulation program is detailed in reference [43]. The algorithm calculates the least cost path for each destination node in the network.

6.4 Adaptive routing algorithm

The main purpose of the simulation program is to evaluate the performance of an adaptive routing algorithm. The adaptive routing algorithm to be simulated is based on R G Gallager's minimum delay routing algorithm [11]. The basic protocols, estimations and formulae required to implement the algorithm are discussed. The performance of the algorithm is discussed in the next chapter.

6.4.1 Algorithm protocol

The basic protocols used to implement the adaptive routing algorithm are described. A more detailed explanation of the routing algorithm may be found in Appendix C and in reference [11].

The algorithm attempts to minimize the average packet delay. Each node constructs its own routing tables based on the information obtained from the neighbouring nodes. The algorithm operates by assigning at each node a probability $\phi_{ik(j)}$. This represents the probability that a packet at node i destined for node j is routed via link (i,k). The sum of the probabilities at each node i for a destination node j is equal to 1. The algorithm operates as follows: Each node i in the network waits until it has received all the routing information for destination node j from all its neighbours for which $\phi_{ik(j)} > 0$. The node then decreases the fraction of

traffic sent on the links with a high average packet delay and increases the fraction of traffic sent on the link with the lowest average packet delay.

In order for the algorithm to operate effectively valid starting values for $\phi_{ik(j)}$ must be determined. It is assumed that initially all packets are sent via the least cost paths which implies that $\phi_{ik(j)} = 1$ for the link (i,k) which is on the least cost path and all other $\phi_{ik(j)} = 0$. Various important aspects of the algorithm are discussed in the following paragraphs.

6.4.2 Incremental delays

In R G Gallager's routing algorithm the incremental delays rather than the actual delays are used to calculate the paths along which packets are to be transmitted. The incremental delay is expressed as dD/df_{ik} where D_{ik} is the average packet delay of link (i,k) and f_{ik} is the average traffic flow of link (i,k). An efficient recursive algorithm for calculating the incremental delay has been developed by A Segall [38]. The algorithm is provided in Appendix C and the protocol used to implement the algorithm is briefly discussed. Consider node i in the network connected to a neighbouring node k in the same network as illustrated below.

Figure 6.2 Two neighbouring nodes in a network.

The value of dD/df_{ik} is calculated by considering the arrival and departure times of a packet on link (i,k). The arrival time of a packet on link (i,k) has been taken to be departure time of a packet from node i. To include the queuing time at node k, (which usually makes up a large proportion of the

delay of link (i,k)), the departure time of link (i,k) has been taken to be the service time of the packet at node k.

The procedure for calculating dD/df_{iN} is therefore as follows:
As a packet departs from node i the time is recorded and inserted into the header of the packet. This represents the arrival time of the packet on link (i,k). When the packet is serviced at node k the node calculates the new value of dD/df_{iN}. If necessary node k transmits this value to node i. The value of dD/df_{iN} is transmitted to node i after a certain number of packets have been transferred from node i to node k or when the value of dD/df_{iN} varies significantly from the value of dD/df_{iN} last transmitted to node i. However transmitting the value of dD/df_{iN} regularly to node i ensures that the routing calculations performed at node i are always based on the latest values of dD/df_{iN}. Transmitting the value of dD/df_{iN} regularly uses up valuable channel bandwidth and hence a optimum solution must be found. It is also necessary to transmit the incremental delay if it changes above a certain tolerance so that node i is made aware of the change in delay along link (i,k). It is important to note that to implement this protocol it is necessary for link (i,k) to be a fully bi-directional channel.

6.4.3 Calculation of blocked paths

One of the most important requirements of an adaptive routing algorithm is that it must be loop free. The routing algorithm must ensure that a packet does not transverse any node in the network more than once while travelling from the source node to the destination node. Consider the simple network illustrated on the following page:

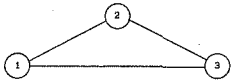


Figure 6.3 A simple network.

Assuming node 1 to be the source node and node 3 to be the destination node. Node 1 may route the packet to node 3 via node 2. Node 2 knows that the packet can be routed to node 3 via node 1 and therefore if the packet is transferred back to node 1 looping occurs.

R G Gallager [11] has developed an algorithm for detecting the presence of potential loops in a network. Each node i in the network contains a blocked path set $B_{i(j)}$ for each destination node j. The set $B_{i(j)}$ then contains the set of output paths k for which the value of $\theta_{ik(j)}$ may not be increased from zero or else looping may occur. The method is based on the fact that the total marginal link delay decreases towards the destination node. Any output path for which the marginal link delay increases may cause a packet to loop around the network. Any output path for which the marginal link delay increases is added to the blocked path set. A more detailed explanation of the formula and protocols used for determining the blocked path set for each node in the network is given in reference [11] and Appendix C.

6.4.4 Estimation of traffic flow at each node

One of the requirements of the routing algorithm is the estimation of the traffic passing through node i destined for node j. The traffic $t_{i(j)}$ includes both the traffic generated

within the node destined to node j as well as incoming traffic from other nodes destined to node j.

The traffic flow may be estimated each time a packet destined to node j is serviced by node i using the following calculation:

$$t_{i(j)} = \frac{\text{Total number of packets destined for node j}}{\text{Total Time}}$$

The problem with the simple formula is that the traffic is only updated each time a packet is serviced by node i. If the traffic passing through node i destined for node j suddenly decreases the value of $t_{i(j)}$ will only be slowly updated. Implementing the formula as it stands determines only the long term average of $t_{i(j)}$ from the beginning of the simulation. This long term average value of $t_{i(j)}$ may differ greatly from the instantaneous value of $t_{i(j)}$.

For this reason the following protocol has been used to estimate the value of $t_{i(j)}$ at each node in the network. Every certain (fixed) number of packets the previous value of $t_{i(j)}$ is discarded and a new value is calculated. This method is useful for traffic rates that vary rapidly. If the number of packets between updates is small a more accurate estimation of the traffic is calculated. The value of $t_{i(j)}$ is not updated if no packets arrive at node i destined for node j, but then a value for $t_{i(j)}$ is not required. A perhaps more accurate method for estimating the traffic flow is to estimate the traffic flow at regular time intervals rather than on the occurrence of a packet arriving at the node.

6.4.5 Conclusion: Adaptive routing algorithm

The basic protocols and procedures used for the implementation of the adaptive routing algorithm have been discussed. R G Gallager's distributed adaptive routing algorithm has been selected since it is a fairly robust and simple algorithm which does not require large and complex computations. The algorithm is used to determine the efficiency of an adaptive routing algorithm compared to a static routing algorithm subject to various traffic conditions. R G Gallager's routing algorithm is studied to determine the following features of the algorithm:

- adaptability of the algorithm to fluctuations in traffic flows in the network,
- convergence of the algorithm to a steady state solution,
- the loop free characteristic of the algorithm and,
- the adaptability of the algorithm to various network topologies.

Various parameters and protocols used by the algorithm are altered to find the optimum performance of the algorithm. The algorithm is tested under various input traffic conditions. The results obtained are discussed in chapter 7.

6.5 Program development

Using the protocols and algorithms defined above the simulation program was designed. The program was developed using an object-orientated design approach [7]. In an object-orientated design the resources or 'data objects' are first defined and the operations on these resources then specified. The resources and processes performed on the resources are then broken down into smaller resources and processes. This is repeated until each process is well defined and operates

on a well defined set of resources. The program was developed using as few assumptions as necessary. This provides a close one to one correspondence between the network and implementation of the simulation package. The design details of the program as well as a full program listing is provided in Part II of the report.

6.5.1 Hardware and software requirements

The main problem with the simulation of large networks is the time required for the simulation. A large number of variables are required to store the status of the network at any point in time during the simulation. It is necessary to use a powerful computer with a fast execution speed and a large amount of available memory. For these reasons it was felt that an IBM PC or compatible was inadequate. The simulation program was programmed on a APOLLO domain computer which has a powerful, 32 bit (68020), central processing unit (CPU).

The simulation program is written in PASCAL due to the availability of the programming language on the APOLLO Domain computer. PASCAL is adequately suited for the development of simulation programs. PASCAL, although not as fast as more primitive programming languages allows the program to be easily read, understood and modified. A flexible modifiable program is an essential feature of any simulation software package.

6.5.2 Program variables

The main problem in designing an accurate digital network simulation program is the large amount of variables required. The variables needed in the program may be divided into three categories:

- Constants: These are the variables that remain constant throughout the duration of the simulation and usually include the input variables such as network topology, node service rates and total number of buffers.
- State variables: State variables define the complete state of the network during the simulation. In order to accurately describe the state of the network at any instant of time during the simulation numerous variables are required.
- Statistics variables: Statistic variables are used to calculate the results of the simulation such as average packet delay and cost of running the network.

A discussion of the variable types used in the program is provided in Part II, Appendix B.

6.5.3 Program procedures

The program may be divided into four main sections namely the input routine, the initialization routine, the simulation routine and the output routine. The principles and purposes of these routines are discussed.

Input routines

The input routines are used to obtain information such as the network topology and traffic flow rates from the user. There are two basic input routines, the first is used to obtain the data from the input file. The second is used to obtain the data entered by the user via the keyboard. Variables such as the duration of the simulation and the names of the input and output files are entered interactively via the keyboard.

Initialization routines

The initialization routines in the program are used to initialize all the nodes in the network and the event queue. The network topology is made known to all the nodes in the network. The routing tables are initialized with the least cost path to all the destination nodes in the network. Random arrivals simulating the generation of packets at the various nodes in the network are generated and inserted into the event queue.

Simulation routines

These routines form the basis of the simulation program. Events are extracted sequentially from the event queue and processed. This portion of the program simulates the routing of packets around the network. A detail explanation of the functions of the procedures used in the simulation program is given in Part II, Chapter 3. During the simulation process facts such as packet delays, number of blocked packets and the percentage utilization of the network are recorded. The least cost, flooding or adaptive routing algorithms may be used to route the packets around the network. The simulation process continues until all the data packets have been removed from the event queue or until the simulation is interrupted by the user.

Output routines

The function of the output routines is to process the data recorded during the simulation. Long term averages and totals are calculated and the results are sent to an output file. The formulae used in calculating the results are detailed in Part II, Appendix C.

6.6 Assumptions made in the simulation program

It essential to describe the protocols used and assumptions made in the implementation of the simulation program. The limitations of the program must be understood so that authentic deductions may be made from the results obtained from the network routing simulation package. The following assumptions and protocols have been used in the simulation program:

- Packets arriving at a node are inserted into a single input queue at the node. Buffers for the input queue are obtained from a common input buffer pool.
- Each output queue at a node is assumed to contain the same constant number of available buffers. This assumption has been made since the number of buffers utilized at each output queue of a node is usually small, particularly if the links have a high capacity, and hence more sophisticated buffer allocation techniques for the output queues are not required.
- Each node in the network may have differing service rates.
- A packet is considered blocked when no buffers are available for the incoming packet. Blocked packets are simply disregarded and lost. The recovery of lost packets has not been considered in the simulation program.
- Packets are initially generated at each node in the network using a Poisson probability density function.
- The simulation program assumes that at the start of the simulation all the queues in the network are empty and that each node contains the correct information about the topology of the network.
- Messages are assumed to consist of single packets only. This assumption may seem severe but multi-packet messages may be simulated by increasing the number of packets or

messages initially generated. All packets in the network are assumed to contain the same number of bits.

- The links are assumed to be ideal and that all the packets are free of transmission errors.
- The number of nodes passed by a packet (number of hops) is used to check if a packet has looped in the network. Looping packets when detected are removed from the network and discarded.
- The calculations used and assumptions made to determine the time at which each event occurs during the simulation are detailed in Appendix C.

The list given above illustrates the more basic assumptions made in the program. It is important to note that some of the assumptions made may be invalid in some particular applications. However the program may be modified if necessary to implement more complex and sophisticated protocols if necessary. Additions made to the simulation program are discussed in the next chapter.

6.7 Conclusion

The requirements of the simulation program have been specified. The chapter discusses the routing algorithms and protocols used in the simulation program. The assumptions made in the implementation of the package have been stated. In the following chapter further modifications made to the program are discussed and the results obtained from the simulation package are investigated.

messages initially generated. All packets in the network are assumed to contain the same number of bits.

- The links are assumed to be ideal and that all the packets are free of transmission errors.
- The number of nodes passed by a packet (number of hops) is used to check if a packet has looped in the network. Looping packets when detected are removed from the network and discarded.
- The calculations used and assumptions made to determine the time at which each event occurs during the simulation are detailed in Appendix C.

The list given above illustrates the more basic assumptions made in the program. It is important to note that some of the assumptions made may be invalid in some particular applications. However the program may be modified if necessary to implement more complex and sophisticated protocols if necessary. Additions made to the simulation program are discussed in the next chapter.

6.7 Conclusion

The requirements of the simulation program have been specified. The chapter discusses the routing algorithms and protocols used in the simulation program. The assumptions made in the implementation of the package have been stated. In the following chapter further modifications made to the program are discussed and the results obtained from the simulation package are investigated.

7 SIMULATION RESULTS

The results acquired from the network routing simulation program are described. Several alterations are made to the program to improve the accuracy of the simulations. Modifications are also made to the adaptive routing algorithm to improve the overall performance of the network. The comparisons made between the various routing algorithms and deductions on the overall performance of the network are described in this chapter.

7.1 Network topologies simulated

Several differing network topologies were simulated using the network simulation package. The results obtained for small networks compared favourably with the results obtained for larger networks. The time required to simulate a network for any reasonable time duration is large and increases drastically with the number of nodes in the network. Therefore most of the results provided in this chapter are obtained using a simple four node network.

To obtain worthwhile results the network topology simulated allows for alternate paths between most of the source and destination node pairs in the network. If no alternate paths are available the implementation of an adaptive routing algorithm does not improve the performance of the network. The network simulated also contains inherent loops so that the loop freedom of each algorithm can be verified.

The diagram below illustrates the simple network used in the simulation package.

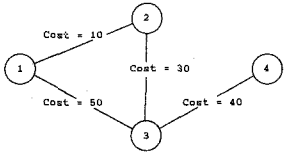


Figure 7.1 Diagram of a simple four node network.

Unless otherwise specified the following network specifications have been assumed:

- All the links in the network are full duplex channels,
- all the links in the network have a capacity of 20kbps in each direction,
- the cost of the links are assigned as illustrated in the figure 7.1,
- the service rate has been assumed to be the same for all the nodes and the service rate is usually set to 50 packets per second,
- the input buffer pool at each node is assumed to contain sufficient buffers for the storage of 50 packets,
- the number of bits per packet is set to 800, and
- a single unit of time in the simulation has been assumed to represent a real time unit of one second (This has been assumed for convenience).

Using the specifications given the capacity of each link may be calculated as 25 packets per second. The capacity of the links in the network have been arbitrarily chosen and more suitable link capacity assignments can be made by observing

the results from the simulation program. The average connectivity of each node in the network is two. This implies a maximum of 50 incoming packets per second at each node. The service rate of each node has been made equal to this value. The costs of the links have been purposely assigned values so that the least cost path is not always the minimum hop path.

The network topology allows only a single path between nodes 3 and 4 in the network. All packets destined to node 4 must be routed via node 3. No packets generated at node 3 destined for node 4 may be routed via node 1 or node 2. This fact helps in evaluating the correctness of the routing algorithm used in the simulation.

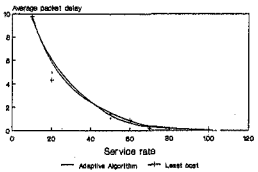
Using the simple network described above numerous results may be obtained on the performance and accuracy of the routing algorithms simulated. The results are described in the subsequent sections in the chapter.

7.2 General results

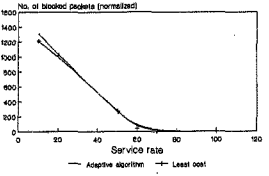
General results typical to all the routing algorithms simulated are discussed and explained. The results are obtained by varying the service rates, input buffer lengths and output buffer lengths of the nodes in the network. These simple tests performed on the network are used to verify the validity of the network specifications selected. The performance of a network may be greatly improved by ensuring that optimum values for the link capacities and node service rates are selected. The input file for the simulations is given in Part II, Chapter 2.

7.2.1 Service rate tests

As is clearly illustrated in graph 7.1, increasing the service rate of the nodes in the network reduces the average packet delay. As the service rates of the nodes are increased the average number of packets present in the input queue at each node decreases rapidly. The decreases in delay becomes less significant as the average packet delay becomes largely dependant on the transmission delays of the packets. Unfortunately increasing the service rate of a node to a large value is expensive and usually several parallel processors must be employed at each node in the network in order to obtain extremely high service rates.



Graph 7.1 Graph illustrating average packet delay for various service rates.



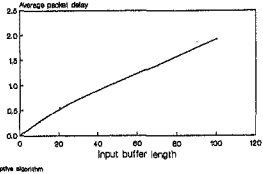
Graph 7.2 Graph illustrating the number of blocked packets for various service rates.

Increasing the service rate of a node decreases the rate at which the input queue of a node accumulates. Therefore the number of packets blocked decreases. This is illustrated in graph 7.2. The number of blocked packets decreases almost linearly with the increase in service rate up to a certain threshold. The threshold is due to the fact that eventually the number of packets blocked on the output of a node becomes greater than the number of packets blocked on the input of the node. The selection of a service rate of 50 packets per second appears to be acceptable and provides a fairly small delay with a small number of blocked packets.

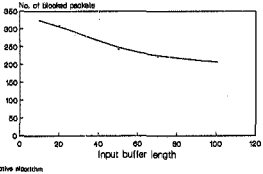
7.2.2 Input buffer length tests

The number of buffers in the input buffer pool of a node determines the number of packets that can be accepted before the packets are discarded. Increasing the number of buffers in the pool decreases the number of blocked packets of a node. The average packet delay increases as the input queue

to a node is allowed to attain a longer length. The increased delays may be acceptable since without the increased number of available buffers the packets would be blocked and discarded anyway. These observations may be seen in the graphs illustrated below. Another factor that must be taken into consideration is the cost of providing the additional buffers at a node. A buffer pool of 50 packets provides a reasonable solution.



Graph 7.3 Average packet delay as a functi' of the length of the input buffer pool.



Graph 7.4 The number of packets blocked as a function of the length of the input buffer pool.

7.2.3 Length of the output buffers at the nodes

Varying the length of the output buffer has no effect on the average packet delay or number of packets blocked in the network. In the network simulated it is assumed that each output link to a node has a capacity greater or equal to the capacity of the server of the node. This means that packets seldom accumulate on the output queues of a node. For this reason a very small number of buffers are required for each output queue in the network.

The effect of the output buffer length becomes significant if the capacity (service rate) of the server at the node exceeds the capacity of the output links to that node. The effect of this will be increased delays, depending on the length of the output queues allowed, as well as an increase in the blocking probability of the packets.

7.3 Optimization of the adaptive routing algorithm

Once a fairly optimum network configuration has been designed it is necessary to fully optimize the performance of the entire network. This is accomplished by maximizing the efficiency of the routing algorithm used. The optimization of the adaptive routing algorithm used in the simulation program is now considered. The performance of the adaptive routing algorithm is improved by finding the variables, protocols and traffic estimation algorithms which optimize the efficiency and reliability of the routing algorithm. Finding the variables that produce an optimum performance of the algorithm is a difficult and time consuming task since many of the variables are inter-related and dependant on the input traffic to the network.

7.2.3 Length of the output buffers at the nodes

Varying the length of the output buffer has no effect on the average packet delay or number of packets blocked in the network. In the network simulated it is assumed that each output link to a node has a capacity greater or equal to the capacity of the server of the node. This means that packets seldom accumulate on the output queues of a node. For this reason a very small number of buffers are required for each output queue in the network.

The affect of the output buffer length becomes significant if the capacity (service rate) of the server at the node exceeds the capacity of the output links to that node. The affect of this will be increased delays, depending on the length of the output queues allowed, as well as an increase in the blocking probability of the packets.

7.3 Optimization of the adaptive routing algorithm

Once a fairly optimum network configuration has been designed it is necessary to fully optimize the performance of the entire network. This is accomplished by maximizing the efficiency of the routing algorithm used. The optimization of the adaptive routing algorithm used in the simulation program is now considered. The performance of the adaptive routing algorithm is improved by finding the variables, protocols and traffic estimation algorithms which optimize the efficiency and reliability of the routing algorithm. Finding the variables that produce an optimum performance of the algorithm is a difficult and time consuming task since many of the variables are inter-related and dependent on the input traffic to the network.

The 'fine tuning' of a routing algorithm can be accomplished by trial and error. A particular variable is altered slightly and the effect on the overall performance of the network observed. This method is costly if the tests are performed directly on the network. For this reason the use of a simulation program to optimize the routing variables appears to be a particularly attractive method.

In this section the variables and protocols considered in improving the performance of the adaptive routing algorithm are explained. This is followed by a list of the limitations, drawbacks and advantages of the particular adaptive routing algorithm.

7.3.1 Exchange rate of routing data between nodes

It is important to consider when and how often routing information is to be exchanged between the nodes in the network. When a packet arrives at node k from node i, node k calculates the new marginal delay dD/df_{iM}. This new value of dD/df_{iM} is then transmitted to node i if one of two conditions holds true:

- The value of dD/df_{iM} changes significantly from the last value of dD/df_{iM} transmitted to node i, or
- A certain fixed number of packets has arrived at node k from node i since the last value of dD/df_{iM} was transmitted to node i.

Exchanging routing information at regular intervals

The effect of altering the number of fixed packets allowed between the transfer of dD/df_{iM} between nodes k and i in the network is considered. If only a small maximum number of

The 'fine tuning' of a routing algorithm can be accomplished by trial and error. A particular variable is altered slightly and the effect on the overall performance of the network observed. This method is costly if the tests are performed directly on the network. For this reason the use of a simulation program to optimize the routing variables appears to be a particularly attractive method.

In this section the variables and protocols considered in improving the performance of the adaptive routing algorithm are explained. This is followed by a list of the limitations, drawbacks and advantages of the particular adaptive routing algorithm.

7.3.1 Exchange rate of routing data between nodes

It is important to consider when and how often routing information is to be exchanged between the nodes in the network. When a packet arrives at node k from node i, node k calculates the new marginal delay dD/df_{iN}. This new value of dD/df_{iN} is then transmitted to node i if one of two conditions holds true:

- The value of dD/df_{iN} changes significantly from the last value of dD/df_{iN} transmitted to node i, or
- A certain fixed number of packets has arrived at node k from node i since the last value of dD/df_{iN} was transmitted to node i.

Exchanging routing information at regular intervals

The effect of altering the number of fixed packets allowed between the transfer of dD/df_{iN} between nodes k and i in the network is considered. If only a small maximum number of

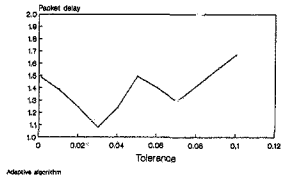
fixed packets is transferred from node i to node k between the transfer of dD/df_{ik} values, then the nodes in the network are continuously updated with the latest value of dD/df_{ik}. The routing algorithm at a node i in the network is updated each time the marginal delays are received from all the neighbouring nodes k. This implies that the routing algorithm is continuously updated at each node. As a result the data packets are routed in the most efficient manner. However the great drawback of exchanging the marginal delays dD/df_{ik} between the nodes regularly is the network bandwidth utilized. This increases the total average packet delay and the total number of packets blocked in the network. Each time the routing variables are updated at a node the new values for $\delta D/\delta r_{ik(t)}$ are also transferred between the nodes in the network. In a real time network, updating the routing tables regularly requires a significant amount of computation time. Marginal delays may also be exchanged between nodes at regular time intervals.

If the time between updates is large the algorithm does not adapt rapidly to changes in traffic. The delays experienced by the data packets may then be large due to the non-optimum routing of packets. However if the marginal delays are exchanged whenever the routing information changes significantly then it is only necessary to ensure that routing information is exchanged at fairly regular intervals. It is necessary to transfer routing information at regular time intervals to ensure that the nodes in the network are also kept informed of slowly varying traffic. The rate at which marginal delays are exchanged is dependent on the capacity of the links, service rates of the nodes and the input traffic of the network.

For the network used in this section it was found that transferring the data approximately every 16 'seconds' is sufficient. This value ensures that the nodes are informed of slowly varying input traffic without the routing information consuming to much bandwidth. The rate of convergence of the routing algorithm is dependent on the frequency with which routing data is exchanged between the nodes in the network.

Informing the network of changes in traffic

Apart from transferring routing information at regular intervals in the network it is also necessary to inform the nodes in the network of any rapid changes in the marginal delays or traffic patterns in the network. To detect significant changes in the marginal delay dD/df_{ik} of link (i,k), node k when it calculates the new value of dD/df_{ik} compares the new value with the last value of dD/df_{ik} transmitted to node i. If the new value differs by more than certain tolerance the new value of dD/df_{ik} is transferred to node i. The question is how to select an appropriate value for the tolerance. If the tolerance is made too large significant changes in the marginal delays of the links will go undetected and the routing tables will not be updated to minimize the delay. If the tolerance is too small new values of dD/df_{ik} will be exchanged frequently wasting valuable network bandwidth. The effect of varying the tolerance is illustrated in the graph on the following page.



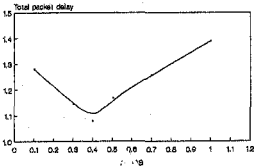
Graph 7.5 Tolerance versus the average packet delay.

The actual values obtained from the graph are dependent on the specific network simulated and may differ for other networks. A maximum tolerance of 0.03 provides the minimum delay as well as the minimum number of blocked packets in this specific case. This value is now used in all future simulations.

7.3.2 Convergence rate of the algorithm

The convergence rate of the algorithm is determined by the variable α of the adaptive routing algorithm as described in Appendix C. The convergence rate determines the number of iterations the routing algorithm requires before a steady state solution is found for a particular traffic flow in the network. The rate of convergence determines the number of routing packets required for the algorithm to adapt to changes to the traffic patterns in the network. A high convergence rate ensures that a minimum number of routing packets are required for the algorithm to adapt to traffic pattern changes in the network. The danger of a high convergence rate is that in some situations, particularly if

the traffic rate in the network changes rapidly, the algorithm may become unstable. Finding the optimum convergence rate is a difficult task which is simplified by the use of a simulation program. Simulations are performed by varying the convergence variable α of the adaptive routing algorithm and keeping all the other variables of the network and routing algorithm constant. A varying input traffic rate was used in the simulation. The results obtained from the simulations are illustrated in graph 7.6. The overall average packet delay and total number of blocked packets of the entire simulation are used to determine the optimum convergence rate of the algorithm.



Graph 7.6 The total average packet delay as a function of the convergence rate of the algorithm.

From the graph the optimum value for alpha is 0.4 for the specific network simulated. The value is dependent on the input traffic to the network. In the real case a slightly lower convergence rate should be used to guarantee convergence of the routing algorithm. A lower convergence rate should be used if the input traffic to the network varies rapidly and is unpredictable.

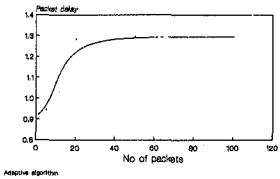
7.3.3 Traffic estimation and prediction

Another aspect to consider when trying to find the optimum performance of the adaptive routing algorithm is the traffic estimation algorithm. The routing decisions performed by the adaptive routing algorithm are based on the estimated traffic in the network. It is therefore important to maximize the accuracy of the traffic estimation algorithm.

Estimation of the traffic in the network

The traffic estimation algorithm used in the simulation program averages the incoming traffic, at a node i, over a certain number of incoming packets n. If n is large then the long term average traffic is calculated. If n is small then the short term average traffic for the network is calculated. If small values of n are used fluctuations in the traffic flow in the network may be detected more rapidly. As a result the adaptive routing algorithm is able to react to the changes in the traffic quickly and efficiently. Small values of n are suitable for the estimation of the traffic flow in highly dynamic networks where the traffic in the network varies rapidly.

Graph 7.7 illustrates various values of n versus the average packet delay in the network. The network described in section 7.1 was used to obtain the results with the traffic flow between two nodes in the network varying rapidly.

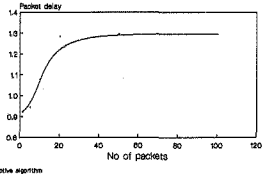


Graph 7.7 Graph showing the effect of n on the average packet delay.

The graph illustrates that short term averages provide the shortest delays. A value of n = 1 gives the smallest delay. A value of n = 1 implies that the instantaneous, rather than average traffic rate is calculated. As n increases the delay increases to a maximum value. Averaging the traffic over 20, 50 or even 100 packets has little effect on the long term average traffic estimation. For small values of n the estimated value of the traffic at the time the routing algorithm is updated is a more accurate representation of the actual traffic present in the network at that time.

Traffic prediction

A formula to predict the future traffic in the network was implemented. The formula was implemented in an attempt to increase the adaptability of the routing algorithm. The purpose of the traffic prediction formula is to attempt to predict the future traffic in the network. Packets may then be routed across the network according to the predicted



Graph 7.7 Graph showing the effect of n on the average packet delay.

The graph illustrates that short term averages provide the shortest delays. A value of n = 1 gives the smallest delay. A value of n = 1 implies that the instantaneous, rather than average traffic rate is calculated. As n increases the delay increases to a maximum value. Averaging the traffic over 20, 50 or even 100 packets has little effect on the long term average traffic estimation. For small values of n the estimated value of the traffic at the time the routing algorithm is updated is a more accurate representation of the actual traffic present in the network at that time.

Traffic prediction

A formula to predict the future traffic in the network was implemented. The formula was implemented in an attempt to increase the adaptability of the routing algorithm. The purpose of the traffic prediction formula is to attempt to predict the future traffic in the network. Packets may then be routed across the network according to the predicted

future traffic in the network rather than to the present or past traffic in the network.

A simple prediction formula was used to predict the traffic in the simulation program:

$$T_{n+1} = 2T_n - T_{n-1}$$

Using the formula above, the predicted traffic (T_{n+1}) is only dependent on the present estimated traffic (T_n) and the previous estimated traffic value (T_{n-1}) at the node. More accurate prediction formulae do exist which base the prediction of the future traffic on numerous previous traffic estimation values. However a *simple prediction formula* is sufficient to determine whether the use of a prediction formula does in fact improve the overall performance of the network.

Several simulations were performed using the traffic prediction formula in the network. It was discovered that if the traffic at a node was estimated or averaged over a large number of packets then the implementation of the traffic prediction formula resulted in slightly shorter delays in the network. This implies that the long term average traffic in a network is usually *relatively predictable*. If the traffic was estimated over a short number of packets or short interval of time then the implementation of a prediction formula resulted in slightly larger packet delays in the network. In networks the traffic is often bursty and this makes the prediction of the short term or instantaneous traffic extremely difficult if not impossible.

Finally, the smallest average packet delay is obtained by estimating the instantaneous traffic in the network without any prediction of the future traffic in the network.

7.3.4 Implementation of 'piggy-backing'

The implementation of an adaptive routing algorithm into the network simulated provided little improvement to the overall performance of the network. At high traffic loads the adaptive routing algorithm usually resulted in a higher average packet delay compared to the static least cost routing algorithm. From the results obtained from various simulations it was evident that a large proportion of the total network capacity was being utilized in transferring routing information between the nodes in the network. To overcome this problem it was decided that the routing information, where possible would be appended to a packet containing only data. Appending routing information to a data packet is commonly known as 'piggy-backing'.

The 'piggy-backing' protocol

It is impossible to piggy-back all the routing information since a data packet must be available before the routing information can be piggy-backed. The following protocol has been adopted to implement piggy-backing of routing information in the network routing simulation program. Assuming routing information is to be sent from node i to neighbouring node k, remembering that routing information is only exchanged between neighbouring nodes in the network. Firstly, the output queue at node i leading to node k (output queue k) is checked to see if the queue contains any pure data packets (A pure data packet is a packet that contains only data). If the output queue does contain one or more pure data packets then the routing information is appended to the pure data packet in front of the queue. If no data packet is found in the output queue then a separate routing packet is created and inserted into the output queue. When a data packet containing routing information arrives at node k, the

routing data is removed from the data packet and processed. The data packet then becomes a pure data packet.

The routing information exchanged between the nodes are the marginal delays dD/df_{ik} and $\partial D/\partial r_{i(j)}$. The marginal delays dD/df_{ik} and $\partial D/\partial r_{i(j)}$ are never transferred to node k simultaneously. It is therefore assumed that the routing information to be transferred between nodes is short and can be inserted into the header of a data packet without altering the length of the data packet. The protocol defined above ensures that the routing information always arrives at the destination node in the correct order.

The simple protocol defined above has been adopted to verify the advantages of appending routing information to data packets. Other more sophisticated protocols may be used to implement piggy-backing. For example if no pure data packets are found in the output queue at a node, the routing information may be stored until a data packet becomes available to which the routing information could be appended. In this case precautions must be taken to ensure that the routing information does not become out-dated whilst waiting for a data packet.

The updated 'piggy-backing' protocol

Due to the fact that the output queues to a node are usually short only a few of the routing packets are piggy-backed (Usually less than 1% of the total number of routing packets). The input queues to the node are however usually long. It is therefore more suitable to piggy-back routing packets arriving at a node. The routing info arriving at a node is appended to a data packet present in the input queue.

routing data is removed from the data packet and processed. The data packet then becomes a pure data packet.

The routing information exchanged between the nodes are the marginal delays dD/df_{iN} and $dD/dr_{i(s)}$. The marginal delays dD/df_{iN} and $dD/dr_{i(s)}$ are never transferred to node k simultaneously. It is therefore assumed that the routing information to be transferred between nodes is short and can be inserted into the header of a data packet without altering the length of the data packet. The protocol defined above ensures that the routing information always arrives at the destination node in the correct order.

The simple protocol defined above has been adopted to verify the advantages of appending routing information to data packets. Other more sophisticated protocols may be used to implement piggy-backing. For example if no pure data packets are found in the output queue at a node, the routing information may be stored until a data packet becomes available to which the routing information could be appended. In this case precautions must be taken to ensure that the routing information does not become out-dated whilst waiting for a data packet.

The updated 'piggy-backing' protocol

Due to the fact that the output queues to a node are usually short only a few of the routing packets are piggy-backed (Usually less than 1% of the total number of routing packets). The input queues to the node are however usually long. It is therefore more suitable to piggy-back routing packets arriving at a node. The routing info arriving at a node is appended to a data packet present in the input queue.

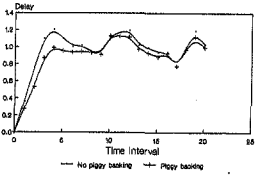
This has several advantages:-

- The input queue is usually large, particularly at high traffic rates. Therefore the probability of the packet being piggy-backed is large.
- Appending routing packets to data packets in the input queue reduces the number of buffers required to store the incoming packet.
- The routing information is appended to the data packets in the front of the queue. This effectively gives priority to the routing information, without blocking any data packets.

If no data packet is present in the input queue then the routing packet is inserted into the queue as a separate packet. All the packets arriving at a node are inserted into the same input queue. A routing packet arriving at node i from node k may not be appended to a data packet arriving from the same node k. Therefore it is essential to ensure that the routing information contains the node number from which it was generated so the routing information is correctly serviced by the node.

When a node services a data packet containing appended routing information, the routing information is removed from the data packet. The routing information and data packet are then serviced separately.

Piggy-backing the routing packets at the input of a node results in a slight improvement in the average packet delays in the network. Almost a 20% improvement in the number of blocked packets is achieved. The reduction in the number of blocked packets is due to the decrease in buffer utilization at the nodes. The graph below illustrates the packet delay versus time for a network with and without piggy-backing.



Graph 7.8 Graph of the average packet delay for network with and without piggy-backing.

'Piggy-backing' conclusion

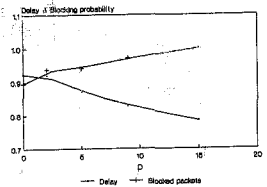
The overhead required to transfer a packet from one node to another in the network is usually large. This is particularly true in the case of packet containing routing information, since these packets are extremely short. Piggy-backing is therefore an essential feature in networks that operate at or near maximum capacity for even a small portion of the time.

7.3.5 Limiting the input traffic to the network

To reduce the average packet delays in the network it is necessary to limit the traffic entering the network and to give priority to the data packets already present in the network. Deleting existing packets in the network increases the average packet delay of the successfully transferred data packets. This is due to the wasted network bandwidth utilized by the blocked packets before the packets are removed from the network. At high traffic loads it is preferable to block

the data packets before the packets actually enter the network.

To block a certain percentage of the data packets entering the network the following procedure has been adopted. When a packet enters the network at a node the number of remaining available input buffers at the node is determined. If the number of available input buffers is less than a specified number p, then the incoming packet is discarded. It is necessary to determine the value of p which results in the optimum performance of the network. Once again the simulation program was used to determine the optimum value of p, and the results obtained are illustrated in the following graph.

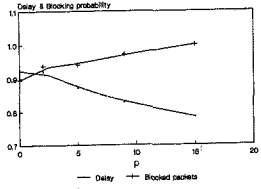


Graph 7.9 Graph illustrating delay and number of blocked packets as a function of p.

As the value of p increases the average packet delay decreases, but the number of blocked packets in the network increases slightly due to the effectively reduced buffer lengths at each node in the network. Setting p equal to five, (that is assuming only 10% of the total number of buffers to remain in the buffer pool before incoming packets are discarded), provides a 6% improvement on the average packet

the data packets before the packets actually enter the network.

To block a certain percentage of the data packets entering the network the following procedure has been adopted. When a packet enters the network at a node the number of remaining available input buffers at the node is determined. If the number of available input buffers is less than a specified number p, then the incoming packet is discarded. It is necessary to determine the value of p which results in the optimum performance of the network. Once again the simulation program was used to determine the optimum value of p, and the results obtained are illustrated in the following graph.



Graph 7.9 Graph illustrating delay and number of blocked packets as a function of p.

As the value of p increases the average packet delay decreases, but the number of blocked packets in the network increases slightly due to the effectively reduced buffer lengths at each node in the network. Setting p equal to five, (that is assuming only 10% of the total number of buffers to remain in the buffer pool before incoming packets are discarded), provides a 6% improvement on the average packet

delay, with a 8% increase in the blocking probability of a packet. The slight increase in the blocking probability may be acceptable. The results are obtained for a heavily overloaded network, when a large percentage of the packets in the network are discarded anyway. At low traffic rates, limiting the input traffic to the network, has no effect on the overall performance of the network. It is more desirable to discard packets at the input of a network since the source node is immediately made aware of the situation and may attempt to retransmit the packet at a later stage. If a packet is blocked at a node, other than the source node, it may be necessary to inform the source node of such an event, increasing the number of packets required in an already overloaded network.

7.3.6 The final product

The modifications and adjustments made to the adaptive routing algorithm are to improve the overall performance of the adaptive routing algorithm. The goal is to minimize the average packet delays and to maximize the overall throughput of the network. Using the techniques described above an improvement of almost 40% on the average packet delays was obtained. The improvement was obtained by finding the optimum performance of the algorithm to a particular type of network with a specific input traffic load. Changing any of the network parameters or the traffic to the network may result in a degradation of the overall network performance. Some of the problems and limitations of the algorithm are discussed in the next section.

7.4 *Problems and limitations of the adaptive routing algorithm*

Any routing algorithm implemented in a network, does have limitations. In this section some of the more prominent problems experienced in the implementation of R G Gallager's routing algorithm as well as the limitations of the algorithm are identified and discussed.

7.4.1 Generation of routing paths

Occasionally the set of output probabilities $\phi_{im(j)}$ generated by the adaptive routing algorithm did not add up exactly to 1 as required. The problem was a result of rounding off errors. The error increased drastically as the simulation progressed eventually resulting in a totally unacceptable set of output probabilities at the nodes.

Initially when the error is first created, the error is very small (usually $< 1\%$). It is therefore simple to compensate for the error. Once a new set of $\phi_{im(j)}$ is calculated at node i, it is checked that the sum of the individual $\phi_{im(j)}$ add up exactly to one (up to five decimal places). If not the slight error is determined and this value is added (or subtracted) from $\phi_{k1(j)}$, where (i,1) is the least cost path to destination j, ensuring that $\phi_{k1(j)} \leq 1$. Using this simple technique errors are quickly detected and corrected for.

7.4.2 Loop elimination

In some simulations the blocked path set generated by the routing algorithm was found to be insufficient to ensure that packets do not loop in the network. Looping of packets occurred in networks larger than four to five nodes. Each

node in the network contains information about the topology of the entire network. Using this information each node may compile a set of blocked paths that guarantees loop freedom.

The following principle has been used to eliminate all possible loops in the network: The sink tree defines a set of paths for a particular source destination node pair. The set of paths all converge to the destination node. The paths defined by the sink tree are loop free. The sink tree for source destination node pair (1,6) is illustrated below:

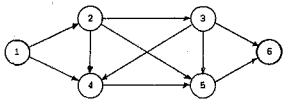


Figure 7.2 An example of a sink tree for source node 1 and destination node 6.

If each node k in the network, for a destination node j, calculates the same sink tree then the paths to that destination are loop free. It is essential that each node in the network for each destination node j assumes the same source node i if loop freedom is to be guaranteed. That is each node must use the same sink tree as all the other nodes in the network, for each destination node.

The algorithm used by each node in determining the sink tree is provided in Part II, Chapter 2. From the sink tree the blocked path set may be determined. The source node i used in determining the sink tree for each destination node j may be selected using one of the following principles:

node in the network contains information about the topology of the entire network. Using this information each node may compile a set of blocked paths that guarantees loop freedom.

The following principle has been used to eliminate all possible loops in the network: The sink tree defines a set of paths for a particular source destination node pair. The set of paths all converge to the destination node. The paths defined by the sink tree are loop free. The sink tree for source destination node pair (1,6) is illustrated below:

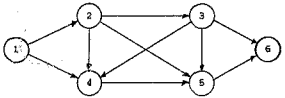


Figure 7.2 An example of a sink tree for source node 1 and destination node 6.

If each node k in the network, for a destination node j, calculates the same sink tree then the paths to that destination are loop free. It is essential that each node in the network for each destination node j assumes the same source node i if loop freedom is to be guaranteed. That is each node must use the same sink tree as all the other nodes in the network, for each destination node.

The algorithm used by each node in determining the sink tree is provided in Part II, Chapter 2. From the sink tree the blocked path set may be determined. The source node i used in determining the sink tree for each destination node j may be selected using one of the following principles:

- random selection of node i. Here it is necessary to ensure that each node selects the same 'random' node i,
- selecting node i to be the furthest from node j,
- selecting node i to be the node with the most traffic to destination node j, or
- selecting node i to be a constant node but ensuring that node i is not equal to node j.

The best performance is obtained by selecting a source destination node pair that allows a maximum bifurcation of traffic. Using the adaptive routing algorithm this ensures that a minimum delay is achieved. Generally the source node with the highest traffic rate to the destination node provides the best solution. However it is often difficult to predict which node will provide the highest traffic rate to the particular destination node. For this reason selecting the source node to be the node furthest from node j provides an adequate solution.

The set of blocked paths are calculated at the beginning of the simulation and then remain constant. If the topology of the network changes each node in the network must update the blocked path set. The new blocked path set must be consistent with the new topology of the network. Using the above principle for determining the blocked path set, loop freedom is guaranteed.

7.4.3 The transfer of routing data in the network

To successfully implement a distributed adaptive routing algorithm into a network it is essential that routing information be exchanged between the nodes. The amount of routing information exchanged in the network must be minimized in order to reduce the network capacity required to transfer the information. At the same time the amount of

- random selection of node i. Here it is necessary to ensure that each node selects the same 'random' node i,
- selecting node i to be the furthest from node j,
- selecting node i to be the node with the most traffic to destination node j, or
- selecting node i to be a constant node but ensuring that node i is not equal to node j.

The best performance is obtained by selecting a source destination node pair that allows a maximum bifurcation of traffic. Using the adaptive routing algorithm this ensures that a minimum delay is achieved. Generally the source node with the highest traffic rate to the destination node provides the best solution. However it is often difficult to predict which node will provide the highest traffic rate to the particular destination node. For this reason selecting the source node to be the node furthest from node j provides an adequate solution.

The set of blocked paths are calculated at the beginning of the simulation and then remain constant. If the topology of the network changes each node in the network must update the blocked path set. The new blocked path set must be consistent with the new topology of the network. Using the above principle for determining the blocked path set, loop freedom is guaranteed.

7.4.3 The transfer of routing data in the network

To successfully implement a distributed adaptive routing algorithm into a network it is essential that routing information be exchanged between the nodes. The amount of routing information exchanged in the network must be minimized in order to reduce the network capacity required to transfer the information. At the same time the amount of

information must be sufficient to ensure that all the nodes in the network are accurately informed about changes in the traffic patterns and delays in the network.

In R G Gallager's minimum delay adaptive algorithm the routing information exchanged between the nodes are the marginal delays dD/df_{im} and $\partial D/\partial r_{m(j)}$. The routing information dD/df_{im} is transferred from node k to node i whenever node i detects a significant change in the value of dD/df_{im}. The marginal delay $\partial D/\partial r_{m(j)}$ is transferred from node i to all neighbouring nodes k each time the routing data at node i is updated, and this occurs each time sufficient routing data has been received by node i from the neighbouring nodes k for which $\theta_{im(j)} > 0$. Originally the new value of $\partial D/\partial r_{i(j)}$ at node i was transmitted to all the neighbouring nodes. This resulted in the 'looping' and rapid accumulation of routing packets in the network. To overcome this problem the new calculated values of $\partial D/\partial r_{i(j)}$ are transferred to all the neighbouring nodes k, except for those nodes k which are an element of the blocked path set $B_{i(j)}$.

The routing tables at a node i in the network are only updated once all the marginal delays from each neighbouring node k for which $\theta_{im(j)} > 0$ have been received. This ensures that the amount of routing information exchanged between the nodes in the network is minimized and the number of routing packets generated during peak traffic conditions is kept to a minimum. The results obtained from simulations for which routing tables are only updated once the routing data had been received from all the neighbouring nodes show a small increase in average packet delay. That is compared to the results obtained for a network where all the routing tables are updated each time an individual routing packet is received by a node.

On average the percentage number of routing packets generated compared to the total number of packets in the network is approximately 10 to 20 percent. This may appear to be large as it implies that 10 to 20 percent of the total network capacity is utilized in transferring routing information across the network. However in the simulation program the length of the routing and data packets are assumed to be the same. Generally it can be assumed that:

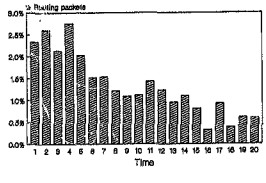
- the length of routing packets are one tenth of the length of data packets. The routing information is usually short and routing packets do not require as much header information as a data packet since the packet are only transferred across a single link in the network, and
- routing packets are only exchanged across a single link in the network, whereas data packets are often transferred over several links as the packet moves from the source node to the destination node.

Considering these assumptions the amount of network capacity utilized in transferring routing packets between the nodes in the network is between 1 to 3 percent. The improvement in the average packet delays due to the implementation of the adaptive routing algorithm is far greater than the increase in packet delay introduced by the routing packets. Using the assumptions that routing packets are far shorter than data packets further reduction in the overall average packet delay may be obtained. The number of routing packets transferred between the nodes in the network decreases rapidly as the algorithm converges, as illustrated in the graph on the next page.

On average the percentage number of routing packets generated compared to the total number of packets in the network is approximately 10 to 20 percent. This may appear to be large as it implies that 10 to 20 percent of the total network capacity is utilized in transferring routing information across the network. However in the simulation program the length of the routing and data packets are assumed to be the same. Generally it can be assumed that:

- the length of routing packets are one tenth of the length of data packets. The routing information is usually short and routing packets do not require as much header information as a data packet since the packet are only transferred across a single link in the network, and
- routing packets are only exchanged across a single link in the network, whereas data packets are often transferred over several links as the packet moves from the source node to the destination node.

Considering these assumptions the amount of network capacity utilized in transferring routing packets between the nodes in the network is between 1 to 3 percent. The improvement in the average packet delays due to the implementation of the adaptive routing algorithm is far greater than the increase in packet delay introduced by the routing packets. Using the assumptions that routing packets are far shorter than data packets further reduction in the overall average packet delay may be obtained. The number of routing packets transferred between the nodes in the network decreases rapidly as the algorithm converges, as illustrated in the graph on the next page.

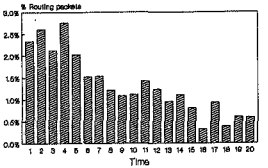


Graph 7.10 The percentage number of routing packets transmitted as the algorithm converges.

7.4.4 General performance of the adaptive routing algorithm

The average packet delays in the network fluctuate before the algorithm converges. The fluctuations are only slight and vary relatively slowly with time. The fluctuations therefore do not degrade the performance of the algorithm by any measurable amount.

The limitations of the current implementation of the adaptive routing algorithm may appear to impose restrictions on the results obtained. Most of the assumptions made in the program actually decreases the performance of the algorithm and hence the results obtained from a simulation may be assumed to be the worst case result. The performance of the algorithm may be further improved by alleviating some of the assumptions made in the program. Many of the restrictions and assumptions in the program may be removed at the expense of increased program complexity and a possible increase in the actual duration of the simulations. Many useful and accurate results



Graph 7.10 The percentage number of routing packets transmitted as the algorithm converges.

7.4.4 General performance of the adaptive routing algorithm

The average packet delays in the network fluctuate before the algorithm converges. The fluctuations are only slight and vary relatively slowly with time. The fluctuations therefore do not degrade the performance of the algorithm by any measurable amount.

The limitations of the current implementation of the adaptive routing algorithm may appear to impose restrictions on the results obtained. Most of the assumptions made in the program actually decreases the performance of the algorithm and hence *the results obtained from a simulation may be assumed* to be the worst case result. The performance of the algorithm may be further improved by alleviating some of the assumptions made in the program. Many of the restrictions and assumptions in the program may be removed at the expense of increased program complexity and a possible increase in the actual duration of the simulations. Many useful and accurate results

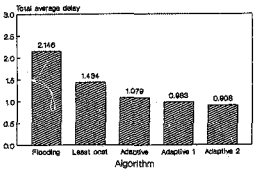
may be obtained from the current implementation of the network routing simulation program.

7.5 Comparison of the various routing algorithms

It is of little use finding the optimum performance of a particular routing algorithm if a simpler more robust algorithm displaying the same or similar performance characteristics can be employed. It is essential when defining the overall performance of a particular routing algorithm that the results be compared to other routing algorithms. This is done to determine if and when the implementation of a more complex routing algorithm is most suitable. It is the relative results obtained from the various routing algorithms implemented in the simulation program that are important, rather than the absolute results obtained from a single simulation.

7.5.1 Average packet delays

One of the most important comparisons to make when comparing two or more routing algorithms is the total average packet delay due to each routing algorithm. To make valid comparisons each of the routing algorithms are simulated using the same identical network with the same input traffic. Graph 7.11 illustrates the total average packet for an entire simulation for each routing algorithm.



Graph 7.11 The total average packet delay for each routing algorithm.

In graph 7.11 the following abbreviations have been used:

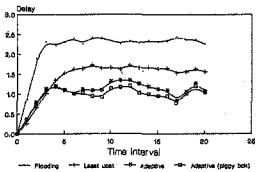
- Flooding: Flooding routing algorithm.
- Least cost: Least cost routing algorithm.
- Adaptive: Adaptive routing algorithm using no piggy backing or traffic prediction.
- Adaptive 1: Adaptive routing algorithm implementing piggy backing and traffic prediction, where the traffic rate is averaged over 10 packets.
- Adaptive 2: Adaptive routing algorithm implementing piggy backing on both the input and output queues. No traffic prediction and the instantaneous traffic rate is used.

From the graph, the least cost routing algorithm provides an almost 36% improvement on the overall packet delay in the network compared to the flooding algorithm, and the adaptive algorithm an improvement of almost 50%.

The performance of the algorithm may be improved slightly (5%) by 'piggy-backing' the routing data in the network. A

further 5% improvement can be obtained by the adaptive routing algorithm if the instantaneous traffic rate is used rather than the average traffic rate.

The graph illustrates the overall average packet delay. It is often necessary to observe the average packet delay during the simulation as illustrated in the graph below. The results are obtained for a network where the input traffic rate is constant for all commodities, except for source destination pair 1,3, where the traffic rapidly increases and then decreases during the simulation.



Graph 7.12 Graph illustrating average packet delays in a network.

It can be seen that the adaptive routing algorithm provides the smallest average packet delay throughout the duration of the simulation. At the beginning of the simulation both the adaptive and least cost algorithms display the same delay due to the fact that the adaptive routing algorithm commences by routing packets via the least cost paths. The delay increases and then remains almost constant for the entire duration of the simulation, even though the input traffic changes. During

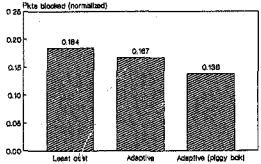
peak traffic the input queues to each node increases drastically. As the input traffic decreases a certain period of time elapses before the network is able to consume the packets in the input queues. The levelling off of the packet delays in the network illustrates that saturation has been reached. Saturation occurs when all or most the input buffers in the network have been utilized, and any additional packets to the network are discarded. The maximum delay experienced by any packet travelling across the network is approximately the same for both the least cost and adaptive routing algorithms, but almost double for the flooding routing algorithm.

The delays in graph 7.11 illustrates the optimum performance of the adaptive routing algorithm. All the algorithm parameters have been carefully selected to produce a minimum average packet delay. Depending on the application an approximate reduction of 40% on the average packet delay by the adaptive routing algorithm compared to the least cost routing algorithm may be achieved. This may be sufficient to verify the increase cost and complexity of implementing an adaptive routing algorithm into the network. However other aspects such as the cost of running the network may also have to be considered.

7.5.2 Number of blocked packets

It is vitally important to consider the probability of a packet being blocked in a network when evaluating the performance of a routing algorithm. It is useless to minimize the average packet delay if a large percentage of the packets are discarded by the routing algorithm anyway. The graph below illustrates the normalized number of blocked packets for the least cost and the adaptive routing algorithms. The normalized number of blocked packets is defined as the total

number of blocked packets in the network divided by the total number of packets generated in the network, excluding routing packets.



Graph 7.13 The number of blocked packets for the adaptive and least cost routing algorithms.

The number of blocked packets for the flooding algorithm have not been included in the graph. The values obtained from the simulation program, for the flooding algorithm, is not an accurate representation of the number of packets not reaching their destinations. In the flooding algorithm numerous copies of a packet are created and transferred across the network. A blocked packet in the network may not mean that the data in the packet has or will not reach the desired destination. It can be assumed that the number of packets not reaching their destinations is larger than for the least cost or adaptive routing algorithms due to the large number of duplicate packets in the network consuming valuable buffer space.

The number of blocked packets for the adaptive and least cost routing algorithms are comparable, with the adaptive algorithm showing only a slight improvement (10%) in the number of blocked packets. The results obtained in the graph are for a network operating under saturation conditions where

the input traffic to the network exceeds the total capacity of the network. Under normal operating conditions the input traffic to the network must be less than the capacity of the network. Providing a large number of input buffers at each node ensures that the blocking probability of a packet is very small if not negligible.

7.5.3 Average percentage use of the network

In determining the overall performance of a network it is essential to observe the percentage utilization of the links in the network. For the flooding algorithm, since traffic is routed across all the links in the network, the overall utilization of the links in the network is almost 100%. A almost 100% utilization occurs since the service rate of the nodes in the network is set to be a 50 packets per second. The capacity of each link in the network is 25 packets per second and on average each node in the network is connected to two adjacent nodes. Therefore on average the capacity of each node is approximately equal to the total capacity of the links connected to the node.

For the least cost routing algorithm packets are routed along a single path and the average percentage use is approximately 50%. The average percentage use of the adaptive routing algorithm is slightly higher due to the number of routing packets transferred across the network and the routing of packets around saturated links in the network. The maximum percentage use is determined by the *saturation link*. A *link* (i,k) may become saturated for one of two reasons:

- the nodes i and k connected to the link cannot supply or consume packets at a rate exceeding the capacity of the link and the nodes become saturated, or

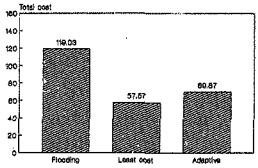
- the nodes connected to link (i,k) are capable of supplying packets to the link at a rate greater than the capacity of the link and the link becomes saturated.

To overcome the problems above it may be necessary to increase the service rate of the nodes connected to the link or to increase the capacity of the link respectively. In a *ideal network we require* that all the *links (or nodes)* in the network become saturated simultaneously allowing for a maximum utilization of the entire network capacity.

7.5.4 Total cost of running the network

The cost of running the network is calculated by adding the cost of transferring individual packets across the links in the network. The cost of running the network is therefore directly related to the number of packets generated in the *network. For this reason the cost of running the network* using the flooding routing algorithm is extremely costly. The least cost routing algorithm provides the minimum cost of transferring packet across the network with a 50% improvement on the cost compared to the flooding routing algorithm while the adaptive algorithm provides a 40% improvement. The improvement in the cost of running the network is largely dependent on the topology of the network and the cost allocation of the links in the network.

For the adaptive routing algorithm not all the data packets *are routed via the least cost path. The* cost of transferring routing packets between nodes is also considered in the total cost of running the network. Hence the adaptive routing algorithm displays a larger total cost compared to the least cost algorithm.

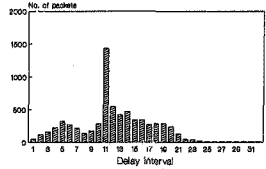


Graph 7.14 Graph illustrating total cost of each algorithm.

In general the price of decreasing the average packet delay of a network is the increased cost of running the network. Only the flooding routing algorithm illustrates an increase in both cost and average packet delay in the network. The final choice of which routing algorithm to utilize may be largely dependent on the application of the network. In some applications it may be necessary to minimize the cost of transferring packets across the network and to optimize the throughput, while in other applications it may be necessary to minimize the packet delay or blocking probabilities in the network. Usually a compromise between average packet delay and the average cost in transferring a packet across the network must be obtained.

7.5.5 Delay distributions

In is both interesting and useful to observe the distribution of the individual packet delays in a network. The graph below illustrates the average delay distribution obtained from several simulations using the minimum delay or adaptive routing algorithm.



Graph 7.15 Delay distribution for the minimum delay routing algorithm.

From the graph it is evident that approximately 20% of the packets in the network experience a delay of roughly 1 time unit, *which is approximately equal to half the maximum delay* experienced by the majority of the packets in the network. Less than 5% of the packets experience a delay greater than 2 time units. In the best case this value drops to less than 0,01%.

In order to understand the significance of the results, it is necessary to compare the delay distribution of the least cost algorithm with that of the minimum delay routing algorithm, as given in the table on the following page. From the table it is clear that the majority of the packets for the minimum *delay routing algorithm experience a smaller delay compared* to the least cost routing algorithm. For the minimum delay algorithm the percentage number of packets experiencing a delay greater than 2 time units is less than 0,01, whilst for the least cost algorithm almost 20% of the packet have a delay greater than 2 time units.

Delay interval (T time) (units)	Least cost algorit (% Number (packets)	Minimum delay algoritha (% Number of) (packets)
0,0 < T ≤ 0,5	7,5	19,5
0,5 < T ≤ 1,0	16,0	12,1
1,0 < T ≤ 1,5	24,4	62,1
1,5 < T ≤ 2,0	34,5	6,3
2,0 < T ≤ ∞	17,6	< 0,01

Table 7.1 Delay distribution for the least cost and minimum delay routing algorithms.

In the case of the least cost routing algorithm the packet delays are more evenly distributed over the entire range, but for the minimum delay algorithm a large percentage of the packets (over 60%) experience a delay of 1 to 1,5 time units. This is due to the fact that the minimum delay routing algorithm distributes the traffic evenly across the network and the packet delay is less sensitive to the input traffic to the network.

From these observations, the advantages of the minimum delay or adaptive routing algorithm are obvious. Both the average and maximum packet delay for the minimum cost algorithm are significantly less than for the least cost routing algorithm.

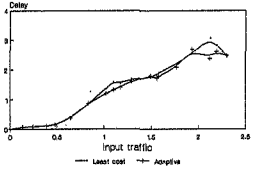
7.5.6 Network performance at extreme conditions

It may be necessary to be able to predict the performance of a routing algorithm under extreme conditions. For example it may be essential to know how a routing algorithm adapts if the input traffic rate suddenly increases simultaneously at all the nodes in the network. Under such conditions it is

imperative that the performance of the network degrades gradually rather than suddenly and that deadlock does not occur in the network. Testing the network under extreme conditions is necessary to determine the limits and operating tolerances of the network and the associated routing algorithm.

Network performance at high traffic loads

The situation where the traffic increases suddenly and simultaneously for all commodities in the network is considered. Under such conditions it is necessary to ensure that the routing algorithm does not fail or become unstable.

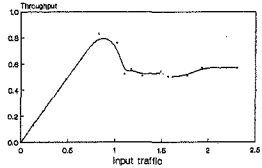


Graph 7.16 The average packet delay as a function of the input traffic to the network.

The graph illustrates the average packet delay as a function of the input traffic to the network. As the input traffic to the network increases the average packet delay increases until the network becomes saturated and all the queues in the network are fully utilized. Under such conditions the adaptive routing algorithm actually produces a slightly

higher average packet delay compared to the least cost algorithm. The use of an adaptive algorithm in such conditions is not practical and only increases the cost of running the network, without decreasing the average packet delay in the network.

As the input traffic increases the throughput of the network increases until the input traffic to the network approaches the total capacity of the network, where the throughput suddenly decreases. The throughput eventually stabilizes as all additional packets to the network are discarded.

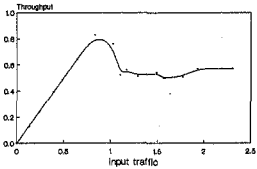


Graph 7.17 Graph illustrating the overall throughput of the network as a function of the input traffic.

Operating a network where the input traffic rate exceeds the overall capacity of the network is usually totally unacceptable due to the large delays and large number of packets that are blocked. In digital computer networks the probability of all the nodes simultaneously transmitting information to all other nodes in the network is extremely small. The traffic is usually bursty where the traffic between two nodes in the network suddenly increases and only lasts a very short period of time. It is however essential to

higher average packet delay compared to the least cost algorithm. The use of an adaptive algorithm in such conditions is not practical and only increases the cost of running the network, without decreasing the average packet delay in the network.

As the input traffic increases the throughput of the network increases until the input traffic to the network approaches the total capacity of the network, where the throughput suddenly decreases. The throughput eventually stabilizes as all additional packets to the network are discarded.



Graph 7.17 Graph illustrating the overall throughput of the network as a function of the input traffic.

Operating a network where the input traffic rate exceeds the overall capacity of the network is usually totally unacceptable due to the large delays and large number of packets that are blocked. In digital computer networks the probability of all the nodes simultaneously transmitting information to all other nodes in the network is extremely small. The traffic is usually bursty where the traffic between two nodes in the network suddenly increases and only lasts a very short period of time. It is however essential to

guarantee that the network will not fail if the input traffic to the network exceeds the capacity of the network. Under such conditions the adaptive routing algorithm is able to produce a throughput of approximately 60%.

Performance of the network at high service rates

The performance of the least cost and adaptive routing algorithms at extremely high and low node service rates are discussed. From various simulation results obtained it is evident that at a extremely low service rate of 10 packets per second the adaptive routing algorithm displays a smaller average packet delay compared to the least cost algorithm. However the delays experienced by both algorithms are extremely large and the number of packets blocked is more than 90% of the total number of packets generated and this is obviously unacceptable. At extremely high service rates of around 100 packets per second the average packet delay of both the least cost and adaptive routing algorithms are identical, indicating that the implementation of a complex adaptive algorithm in networks with extremely high service rates is unnecessary. The blocking probabilities of packets in such a network is negligible. However implementing a network with high service rates, particularly for a large network is costly.

7.5.7 Memory and computational requirements

It is essential to consider the amount of memory and processing time required by a routing algorithm when designing a network. The increased processing power of microprocessors and relatively inexpensive memory allows fairly sophisticated routing algorithms to be implemented in modern networks. However extremely complex routing algorithms may

require to much processing time and memory to be successfully and efficiently employed in large complex networks.

Memory requirements

A description of the memory and computational requirements of each of the algorithms used in the simulation program given in Appendix D. The amount of memory required by both the least cost and adaptive routing algorithms is directly proportional to N^2, where N is the number of nodes in the network. For large networks ($N \geq 100$) the amount of memory required by the adaptive routing algorithm is approximately five times that required by the least cost routing algorithm. The problem with most routing algorithms is that the amount of memory required by the algorithm increases rapidly with an increase in the number of nodes in the network.

Computational requirements

The computational requirements and the time taken for a routing algorithm to attain an acceptable set of routing paths for each destination node in the network is often difficult to evaluate, particularly in the case of dynamic routing algorithms. For the static least cost routing algorithm the number of computations to be performed at each node in the network is of the order N^2, ($O(N^2)$). This is acceptable since the static routing algorithm must only define a *new set of routes if the topology of the network* changes or if the cost of any link in the network changes. In networks were the topology of the network changes rapidly, for example in mobile radio networks, the computational time required by the least cost algorithm may become a limiting factor on the overall performance of the network.

The adaptive routing algorithm is made up of smaller independent algorithms and evaluating the computational requirements is difficult. For a single iteration of the algorithm in determining a new set of routing paths to a particular destination is {O(N)}, and for (N-1) destination nodes in the network the number of computations is {O(N^2)}. A single iteration of the algorithm is performed at a node each time sufficient data is obtained from all the neighbouring nodes. Therefore the time spent by each node in the network in computing a new set of paths for each destination node in the network is dependent on the frequency at which routing information is exchanged between the nodes in the network. The computational time required by R G Gallager's adaptive routing algorithm does not appear to be a serious restriction to the performance of the network, providing that the algorithm is not updated too frequently.

Simulation program computational and memory requirements

In the simulation program all the routing calculations performed at each node in the network is performed by a single processor. All the routing information stored in all the nodes in the simulated network must be contained in the available memory of the computer. This seriously limits the size of the networks that can be efficiently simulated. Currently a maximum of a 20 node network may be simulated.

7.6 Other network topologies simulated

Other larger networks were also simulated using the network routing simulation program. A six node network was simulated. It was noted that increasing the number of direct links between nodes in the network drastically reduces the average packet delay in the network independent of the routing

algorithm used. The results obtained for larger networks compared favourably with the results obtained using a simple four node network.

The routing algorithms were also simulated using a 10 node hierarchical network. The use of an adaptive routing algorithm in such a network is of little use since little traffic bifurcation is possible. A least cost algorithm provided reasonable results with the same average packet delay as for the adaptive routing algorithm. The flooding algorithm illustrates very poor performance in hierarchical networks since packets are duplicated and transferred back and forth across the network before being discarded.

Using the network routing simulation program numerous network topologies with differing characteristics may be simulated in order to establish an optimum network for a particular application.

7.7 Topology changes, link and node failures in the network

An aspect that has not been dealt with in the simulation program is the aspect of link or node failures in the network or the case where links or nodes are added to or removed from the network. If a link (i,k) fails in the network the following operations must be performed:

- The nodes i and k must detect the *failed link* rapidly to avoid packets being transferred down the link, and
- the fact that a link has failed must be made known to all the nodes in the network as quickly as possible.

The following protocol may be used to handle the event of a link failure in the network:

- A node will detect a failed link by periodically transmitting a packet to each neighbouring node. If the neighbouring node does not respond by retransmitting the packet back to the node within a certain time period the link will be assumed to have failed.
- A node having detected the failure of a link will inform all the other nodes in the network of the event by using a reliable and r t flooding algorithm.
- As each node receives the information the routing tables will, in the case of the adaptive routing algorithm, be updated using the least cost routing algorithm, after which the minimum delay routing algorithm may proceed as usual.

The main problem with a link failing in the network occurs in the time interval between the failure of the link and the time when all the nodes in the network are made aware of the failure. During this interval numerous packets may be lost so it is important to inform all the nodes in the network of the link failure as quickly as possible by giving these packets highest priority. Each node makes it's own decision on which path an incoming packet is to be routed so the network is able to adapt to a link failure more promptly. The problem is more severe in large networks since the time taken to inform all the nodes in the network of link failure may be very large. The average packet delay immediately after the failure of a link is expected to increase slightly until the minimum delay algorithm is able to minimize the average packet delay again. The increase in delay is directly dependent on the average utilization of the failed link.

The event of a node failing in the network can be handled in a similar way. The event of a node failure is usually more serious since a failure of a node is equivalent to the failure of all the links connected to that node. The severity of a link failure in the network is dependent on a number of factors:

- the general utilization of the link,
- the number of packets present in the link or destined to be transferred across the link at the time of failure of the link,
- the capacity of the link, and
- the importance of the link to the network (ie. the effect removal of the link has on the connectivity of the network).

Assuming a node or group of nodes are connected to the rest of the network via a single link. If that link fails the node, or group of nodes, will become totally disconnected from the rest of the network. This may lead to catastrophic results. Thus to improve the reliability of the network and to decrease the effect of a link failure in the network it is necessary to ensure that each node is connected to at least two other nodes in the network. The question of adding links or nodes to the network may be handled in a similar manner to link failures. Adding a link or node to the network has a far less detrimental effect on the overall performance of the network compared to a link failure. However before all the nodes are made aware of the change to the network packets may loop and be discarded in the network.

It is important to be able to add or remove links or nodes to or from a network without having to reconfigure each node in the network. This allows the network to be modified and expanded at a minimum cost and with the least interference to the other users present in the network.

The event of a node failing in the network can be handled in a similar way. The event of a node failure is usually more serious since a failure of a node is equivalent to the failure of all the links connected to that node. The severity of a link failure in the network is dependent on a number of factors:

- the general utilization of the link,
- the number of packets present in the link or destined to be transferred across the link at the time of failure of the link,
- the capacity of the link, and
- the importance of the link to the network (ie. the effect removal of the link has on the connectivity of the network).

Assuming a node or group of nodes are connected to the rest of the network via a single link. If that link fails the node, or group of nodes, will become totally disconnected from the rest of the network. This may lead to catastrophic results. Thus to improve the reliability of the network and to decrease the effect of a link failure in the network it is necessary to ensure that each node is connected to at least two other nodes in the network. The question of adding links or nodes to the network may be handled in a similar manner to link failures. Adding a link or node to the network has a far less detrimental effect on the overall performance of the network compared to a link failure. However before all the nodes are made aware of the change to the network packets may loop and be discarded in the network.

It is important to be able to add or remove links or nodes to or from a network without having to reconfigure each node in the network. This allows the network to be modified and expanded at a minimum cost and with the least interference to the other users present in the network.

7.8 Conclusion: Network simulations

From the numerous simulations performed using the network routing simulation package many results and conclusions on the accuracy and effectiveness of the simulation program may be made. Valid deductions on the performance of each routing algorithms used in the simulation program can be made.

7.8.1 The network routing simulation package

The results obtained from the simulation program appear to be accurate and compare favourably, in the case of the static routing algorithms, with the results obtained from analytical methods. The results of the simulation program are expected to be more accurate due to fewer assumptions made in the program compared to analytical evaluation techniques. For example the program does not base the simulations on the assumption that an infinite number of buffers are available at each node as does the analytical method discussed in chapter 4.

The randomly generated traffic, using a Poisson distribution, compares very closely to the expected traffic in the network. This indicates that although the simulation of a continuous real time network is actually simulated at discrete time intervals, the increments between the discrete time intervals are small enough to ensure accurate results. The accuracy of the results obtained from a simulation is largely dependant on the simulated duration. If a network is simulated for a long period of time then the results obtained are likely to be more accurate. More packets will be transferred across the network and the final results of the simulation will be averaged over a larger number of packets giving a better indication of the overall performance of the network. The cost of obtaining more accurate results is the increase in

7.8 Conclusion: Network simulations

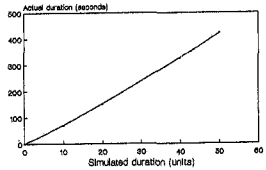
From the numerous simulations performed using the network routing simulation package many results and conclusions on the accuracy and effectiveness of the simulation program may be made. Valid deductions on the performance of each routing algorithms used in the simulation program can be made.

7.8.1 The network routing simulation package

The results obtained from the simulation program appear to be accurate and compares favourably, in the case of the static routing algorithms, with the results obtained from analytical methods. The results of the simulation program are expected to be more accurate due to fewer assumptions made in the program compared to analytical evaluation techniques. For example the program does not base the simulations on the assumption that an infinite number of buffers are available at each node as does the analytical method discussed in chapter 4.

The randomly generated traffic, using a Poisson distribution, compares very closely to the expected traffic in the network. This indicates that although the simulation of a continuous real time network is actually simulated at discrete time intervals, the increments between the discrete time intervals are small enough to ensure accurate results. The accuracy of the results obtained from a simulation is largely dependant on the simulated duration. If a network is simulated for a long period of time then the results obtained are likely to be more accurate. More packets will be transferred across the network and the final results of the simulation will be averaged over a larger number of packets giving a better indication of the overall performance of the network. The cost of obtaining more accurate results is the increase in

the actual duration of the simulation. The actual duration of the simulation varies linearly with the simulated duration as is illustrated in graph 7.18.



Graph 7.18 The actual duration of a simulation versus the simulated duration of a network.

The advantage of the simulation program is that the results obtained from the simulation may be made as accurate as required by removing some of the assumptions made in the program. The assumptions made in the program are to reduce the complexity of the program and to minimize the average duration required for a simulation. Various protocols used in the program may also be modified in order to obtain an optimum set of rules for the routing of data packets in a particular network.

The current version of the simulation program is fairly reliable. The program consists of well over 3500 lines, but this does include the abundant number of comments that have been included in the program to make the program more understandable and easily modifiable.

7.8.2 Routing algorithms

The chapter is concluded with a discussion on the effectiveness of the routing algorithms simulated. The flooding algorithm consumes too much valuable network capacity to be utilized in any reasonably busy network. The only advantages of the flooding routing algorithm is that it may be used to broadcast packets in the network. The flooding algorithm is robust and may be used as a last resort to transfer vitally important packets in the network.

The least cost or shortest path algorithm provides a great improvement on the performance of the network compared to the flooding routing algorithm. The least cost routing algorithm is most useful in networks with a low connectivity and where it is essential to minimize the cost of transferring packets across the network. The least cost algorithm is also most useful in stable networks where the topology of the network remains constant for long periods of time. Typical networks include global networks where nodes are separated by large distances and it expensive to transfer messages from one node to another in the network.

The adaptive routing algorithm provides shorter average packets delays and a smaller overall packet blocking probability compared with the least cost routing algorithm. The average cost of transferring packets across the network is slightly higher. The disadvantage of the adaptive algorithm is the increased complexity of the network and the increased computational facilities required at each node in the network. In the current implementation of R G Gallager's minimum delay adaptive routing algorithm it is essential that all the links in the network are full duplex channels if the algorithm is to operate efficiently. If a link fails in one direction only it must be assumed that the entire duplex link has failed.

The buffer allocation techniques used in the simulation package provides a fairly proficient method of allocating buffers to incoming packets at a node.

It is clear from various simulation results that the adaptive routing algorithm is the best suited for networks consisting of autonomous randomly distributed nodes with a high connectivity. The adaptive algorithm is also more suited to bursty traffic situations compared to the least cost routing algorithm. The adaptive routing algorithm also works best in smaller networks due to the shorter delays required to transfer routing information across the network. The adaptive routing algorithm is therefore well suited to digital computer networks where the nodes are randomly distributed over a relatively small area. It is essential to carefully select the parameters of the adaptive routing algorithm so that the algorithm operates at maximum efficiency.

Both the least cost and adaptive routing algorithms are best suited to smaller networks of less than 50 nodes due to the large amount of information that must be stored at each node in the network. Finally the network routing simulation package provides interesting and valuable information on the performance of various routing algorithms in different types of networks.

8 FINAL CONCLUSIONS AND RESULTS

In the thesis, the performance of an adaptive and static routing algorithm have been compared with the aid of a simulation program. Various routing parameters, protocols and congestion control techniques have been studied. An optimum for the performance of the adaptive routing algorithm for a particular network has been found. The uses, advantages and limitations of the use of a simulation program as a means of effectively analysing a digital computer network have been carefully studied. In this chapter the most important results and conclusions made from the research project are summarized.

8.1 The adaptive routing algorithm

The adaptive routing algorithm studied was based on R G Gallager's minimum delay routing algorithm. Various protocols, parameters and measurement techniques used in the algorithm were varied in order to obtain an optimum routing algorithm for a particular network topology. A packet switched datagram type network was selected. Such a network provides maximum adaptability to changes in the traffic patterns in the network. In analysing the overall performance of any routing algorithm in a network it is necessary to consider the cost-performance trade off of the algorithm.

8.1.1 Uses and limitations of the adaptive routing algorithm

The adaptive routing algorithm has been optimized for a simple four node network with an average connectivity of two. It is important to realize that the implementation of an adaptive routing algorithm into a particular network provides

little improvement on the overall performance of the network if:

- the input traffic load into the network is light and far less than the total throughput of the network,
- the input traffic to the network remains constant,
- the input traffic increases at all the nodes in the network simultaneously,
- the connectivity of the network is less than two, or
- a large proportion of the packets generated in the network are to be broadcast to all the destination nodes in the network.

The main advantage of using an adaptive routing algorithm into a network is to increase the throughput of the network and to decrease the average delay and blocking probabilities of the packets. The cost of implementing a routing algorithm into a network is the increased cost of running the network and the increase in computational and memory requirements of each node in the network. A limitation of the adaptive routing algorithm is that packets cannot be broadcasted to all the destination nodes in the network. However packets may be broadcasted using a flooding algorithm or by generating individual packets for each destination node and routing these packets individually to each destination.

Generally adaptive routing algorithms are best suited to highly interconnected digital computer networks operating at a fairly light loads, but where the traffic between any two nodes in the network may suddenly increase for a short period of time.

little improvement on the overall performance of the network if:

- the input traffic load into the network is light and far less than the total throughput of the network,
- the input traffic to the network remains constant,
- the input traffic increases at all the nodes in the network simultaneously,
- the connectivity of the network is less than two, or
- a large proportion of the packets generated in the network are to be broadcast to all the destination nodes in the network.

The main advantage of using an adaptive routing algorithm into a network is to increase the throughput of the network and to decrease the average delay and blocking probabilities of the packets. The cost of implementing a routing algorithm into a network is the increased cost of running the network and the increase in computational and memory requirements of each node in the network. A limitation of the adaptive routing algorithm is that packets cannot be broadcasted to all the destination nodes in the network. However packets may be broadcasted using a flooding algorithm or by generating individual packets for each destination node and routing these packets individually to each destination.

Generally adaptive routing algorithms are best suited to highly interconnected digital computer networks operating at a fairly light loads, but where the traffic between any two nodes in the network may suddenly increase for a short period of time.

8.1.2 Comparison with the ARPANET network

The ARPANET network is one of the most well known networks in the world. A brief comparison between the routing algorithm used in the ARPANET network with the routing algorithm used in the simulation program is made. The adaptive routing algorithm used in the simulation program is very similar to the ARPANET routing algorithm. The major difference being that in the ARPANET network the delay of individual packets is minimized where as the adaptive routing algorithm studied attempts to minimize the overall average packet delay.

It is difficult to compare the performance of two different routing algorithms unless the algorithms are tested using the same network topology with the same input traffic rate to the network. Therefore only general comparisons are made. The amount of routing information exchanged between the nodes in the network is very similar for both routing algorithms and the ARPANET network requires approximately 2% of a 50kbps line to transfer routing information between nodes in the network. This is comparable to bandwidth required for the routing information in R G Gallager's routing algorithm. In the ARPANET network routing information is exchanged at least every 10 seconds between the nodes in the network. This is similar to the maximum time period allowed in the simulation program between the exchange of routing information. In the ARPANET network routing packets are short and have a length of 176 bits and are given highest priority. In the ARPANET routing algorithm if the instantaneous traffic is used to update the routing variables then the oscillations produced by the ARPANET routing algorithm are large and therefore the traffic is averaged over a finite time interval. In the adaptive routing algorithm used in the simulation program the oscillations produced by using the instantaneous traffic rates are small.

The ARPANET network attempts to optimize the performance of the network from a user point of view by minimizing the packet delay of each packet in the network individually. The simulated routing algorithm optimize the network performance from a manager orientated point of view by minimizing the overall average packet delay and performance of the network.

8.1.3 Conclusion: The adaptive routing algorithm

In general the adaptive routing algorithm operates efficiently and provides a definite improvement in the average delay and blocking probabilities of packets compared to the least cost algorithm. The main problem in designing or establishing an optimum routing algorithm in a network is that it is difficult to isolate the algorithm or parts of the algorithm from the rest of the network. The overall performance of the algorithm is dependent on other aspects such as buffer allocation techniques, congestion control protocols, computational efficiency, input traffic patterns and topology of the network.

As the distance, or number of hops, between the nodes in the network increases the adaptability of the adaptive routing algorithm decreases. The adaptability of the routing algorithm is also dependent on the rate at which routing information is exchanged between the nodes in the network. If routing information is exchanged frequently the adaptability of the algorithm increases but then the number of routing packets present in the network increases. This may increase the total average packet delay in the network. An optimum between the rate at which routing information is exchanged and the number of routing packets in the network must be found. The adaptive routing algorithm used in the simulation program attempts to achieve a maximum adaptability using a minimum amount of routing information.

The ARPANET network attempts to optimize the performance of the network from a user point of view by minimizing the packet delay of each packet in the network individually. The simulated routing algorithm optimize the network performance from a manager orientated point of view by minimizing the overall average packet delay and performance of the network.

8.1.3 Conclusion: The adaptive routing algorithm

In general the adaptive routing algorithm operates efficiently and provides a definite improvement in the average delay and blocking probabilities of packets compared to the least cost algorithm. The main problem in designing or establishing an optimum routing algorithm in a network is that it is difficult to isolate the algorithm or parts of the algorithm from the rest of the network. The overall performance of the algorithm is dependent on other aspects such as buffer allocation techniques, congestion control protocols, computational efficiency, input traffic patterns and topology of the network.

As the distance, or number of hops, between the nodes in the network increases the adaptability of the adaptive routing algorithm decreases. The adaptability of the routing algorithm is also dependent on the rate at which routing information is exchanged between the nodes in the network. If routing information is exchanged frequently the adaptability of the algorithm increases but then the number of routing packets present in the network increases. This may increase the total average packet delay in the network. An optimum between the rate at which routing information is exchanged and the number of routing packets in the network must be found. The adaptive routing algorithm used in the simulation program attempts to achieve a maximum adaptability using a minimum amount of routing information.

An adaptive routing algorithm is only useful in a network that has a need for such an algorithm and that will allow the algorithm to operate effectively. A simulation program is a valuable 'tool' that may be used to establish the optimum performance of a routing algorithm.

8.2 The simulation of networks

The design and implementation of the network routing simulation program formed a large part of the research project. Considering the limitations and advantages of a simulation program the effectiveness of a simulation program to accurately imitate the routing of packets in a network is verified.

8.2.1 Features and limitations of the simulation program

The most important aspect of a simulation program is accuracy and depends on the precision with which the a system is modelled and simulated. The accuracy of a system is dependent on

- the cost and time taken to design, implement and test a simulation program,
- the time taken for an actual simulation and
- the computational and memory requirements.

A simulation program can be made as accurate as desired at the expense of the criteria described above. For this reason and due to the limited time allocated to the design and implementation of the simulation package several assumptions have been made and are listed in Chapter 6. The assumptions made do not drastically effect the overall accuracy of the simulation program, although they should be noted when analysing the results from the program.

Despite the various assumptions made in the implementation of the simulation program, accurate and meaningful results may be obtained. The simulation program is flexible and the basic features included in the simulation program are:

- any randomly interconnected network consisting of up to 20 nodes may be simulated,
- *a flooding, least cost and adaptive* routing algorithm may be simulated,
- accurate results may be obtained in reasonably short time intervals,
- numerous network parameters such as network topology, service rates, link costs, link capacities, input traffic, buffer lengths and the number of bits per packet may be entered by the user and
- the program may be modified to implement additional network protocols.

Some of the assumptions made in the simulation program may be alleviated by modifying the program. The program was originally designed, implemented and as far as possible tested, in small increments in order to guarantee the overall accuracy of the network.

8.2.2 Conclusion: Simulation of networks

The basic advantages of using a simulation program to predict the performance of a digital network is that it is relatively *inexpensive*, powerful and accurate. Normally any type of network may be simulated, whether it be a distributed or centralized network, or a static or dynamic network. A simulation program eliminates many of the assumptions required in analytically predicting the performance of a network.

In designing, implementing and using the network routing simulation program a greater understandability of the *principles of networks was* obtained. Also using a simulation program there is a close one to one correspondence of the simulated network with a real network. In a simulation program the 'entire' network is effectively concentrated at one point. Thus abundant information on the entire network at any point in time can be obtainad at a low cost compared to a real network where the information must be gathered over the entire network and processed at a later stage.

Generally a computer *simulation provides an effective way of* analysing and determining the optimum performance of networks up to approximately 20 nodes accurately and cheaply.

8.4 Recommendations for future work

Some of the more important aspects which require additional investigation are described.

8.4.1 Topology changes

One of the most important aspects that has not been implemented into the network routing simulation package is the behaviour of the network if a link or node in the network fails. It is important to determine the response time of the network to such an event. The response time is the time interval from the time at which the link or node failed to the time at which each node in the network is made aware of the link or node failure. The degradation of the performance of the network, increase in packets delays and number of *blocked packets resulting* from a link failure should be investigated.

8.4.2 Removal of the limitations of the program

The most obvious modification that may be made to the simulation program are the removal, where possible, of some of the limitations of the program discussed in chapter 6. The limitations of the simulation program are however not severe and alleviating some of the limitations may only increase the complexity of the program without significantly improving the accuracy of the simulation program and may even decrease the reliability of the overall simulation package.

8.4.3 Congestion control and buffer allocation techniques

In the project the features of various congestion control and buffer allocation techniques have been considered. More advanced congestion control protocols and buffer allocation techniques may be inserted into the program and simulated to determine the overall effect on the performance of the network.

8.4.4 Data link protocols

The functions of the data link layer as defined by the OSI reference model have been overlooked in the project. The main function of the data link layer is to provide an error free transmission medium. This is usually accomplished by the use of 'handshaking' packets transferred between the source and destination nodes in the network.

In a distributed network the problem is far from trivial. For example if acknowledgement frames are used to inform the source node that a packet has successfully reached it's destination, then problems may arise due to the fact that the source node may have to wait a long period of time between

8.4.2 Removal of the limitations of the program

The most obvious modification that may be made to the simulation program are the removal, where possible, of some of the limitations of the program discussed in chapter 6. The limitations of the simulation program are however not severe and alleviating some of the limitations may only increase the complexity of the program without significantly improving the accuracy of the simulation program and may even decrease the reliability of the overall simulation package.

8.4.3 Congestion control and buffer allocation techniques

In the project the features of various congestion control and buffer allocation techniques have been considered. More advanced congestion control protocols and buffer allocation techniques may be inserted into the program and simulated to determine the overall effect on the performance of the network.

8.4.4 Data link protocols

The functions of the data link layer as defined by the OSI reference modal have been overlooked in the project. The main function of the data link layer is to provide an error free transmission medium. This is usually accomplished by the use of 'handshaking' packets transferred between the source and destination nodes in the network.

In a distributed network the problem is far from trivial. For example if acknowledgement frames are used to inform the source node that a packet has successfully reached it's destination, then problems may arise due to the fact that the source node may have to wait a long period of time between

the transmission of the data packet and the reception of the associated acknowledgement frame. Acknowledgement packets may become lost resulting in duplicate copies of packets being transferred across the network. Another techniques is to use a request retransmission frame as apposed to an acknowledgement frame. In this method the source node appends a sequential packet number to each packet destined to a particular destination node. The destination node, after receiving a certain number of packets, checks to see if all the packets in the sequence have been received. If not the destination node sends a request retransmission packet to the source node requesting the source node to retransmit that particular packet. The problem with this method is that in a datagram network packets may arrive at the destination node not in a strict sequential order. Another problem is that the source node must store duplicate copies of the packets for long periods of time before they can be discarded

The suitability of various data link protocols to a particular network simulated requires further analysis.

8.4.5 Simulation of additional routing algorithms

Other routing algorithms whether static or dynamic algorithms may be implemented into the network routing simulation program. An additional interesting algorithm that could be simulated is the 'Second derivative minimum delay routing algorithm' proposed by D P Bertsekas et al. P[5]. The algorithm should provide a slightly greater adaptability to traffic changes compared with the minimum delay routing algorithm used in the simulation program. The Second derivative routing algorithm is however complex, requiring complex calculations and additional memory requirements. The suitability of the algorithm in large networks is questionable and requires further inquiry.

8.5 Final words

In the project various routing algorithms and network protocols were examined in a literature survey. A suitable adaptive minimum delay routing algorithm was selected. The specifications for a typical network were then stipulated. A simulation program was designed and implemented in order to determine the overall performance of the routing algorithm when used in the network topology specified.

Many interesting and valuable results were obtained from the simulation package. The simulation package was used to find the optimum performance of the minimum delay routing algorithm.

In general it can be concluded that the minimum delay routing algorithm compares favourably with the least cost routing algorithm and provides a significant decrease in the average packet delay with only a slight increase in the total cost of running the network. It can also be concluded that simulation is a practical and powerful method for accurately predicting the overall performance of a digital computer network.

APPENDICES

APPENDIX A

TERMINOLOGY AND SYMBOLS

APPENDIX A

TERMINOLOGY and SYMBOLS

The terminology used in the report is provided together with a summary and explanation of all the symbols used in the report.

Arc: Connects two adjacent nodes, and allows packets to be transferred between the nodes.

Arc connectivity: The minimum number of links or arcs that must be removed to disconnect two nodes in the network.

Bandwidth: This normally refers to the total available capacity of the network, that is the maximum number of packets per second the network is capable transferring.

Bifurcation: Bifurcation implies that two or more paths to a single destination node are made available by the routing algorithm.

Blocked packet: A packet that is unable to reach the required destination node.

Broadcast: Broadcast occurs when a packet is transmitted to every possible destination node in the network.

Bursty Traffic: Bursty traffic occurs when the traffic between a pair of nodes in the network suddenly increases for a short period of time.

Capacity: The maximum number of bits per second that a link is capable of transferring from one node to another.

Commodity: A commodity refers to a source destination pair. A multi-commodity network exists when numerous source destination pairs occur in the network.

Congestion: Congestion occurs when too many packets are present in the network, or part of the network, resulting in the degradation of the network performance.

Connectivity: The number of links or nodes that must be removed in order to disconnect node i from node j in the network.

Datagram circuit: Each packet is treated as a separate entity and each packet is routed individually to the destination node. In a datagram network only a best attempt is made in transferring the packet to it's destination.

Deadlock: Deadlock occurs due to cyclic wait of resources to become available.

Destination: The node for which the data is intended, that is the node to which the data packet must be transferred.

Flow control: Protocols that ensure that the source node does not supply data to a destination node at a rate exceeding the input capabilities of the destination node.

Frame: A ensemble of data (represented in bits) that is transferred along a link.

Hop count: The number of nodes passed by a packet travelling from the source node to the destination node.

Author Francois Marc Leon
Name of thesis Routing Strategies In Digital Networks. 1988

PUBLISHER:
University of the Witwatersrand, Johannesburg
©2013

LEGAL NOTICES:

Copyright Notice: All materials on the University of the Witwatersrand, Johannesburg Library website are protected by South African copyright law and may not be distributed, transmitted, displayed, or otherwise published in any format, without the prior written permission of the copyright owner.

Disclaimer and Terms of Use: Provided that you maintain all copyright and other notices contained therein, you may download material (one machine readable copy and one print copy per page) for your personal and/or educational non-commercial use only.

The University of the Witwatersrand, Johannesburg, is not responsible for any errors or omissions and excludes any and all liability for any errors in or omissions from the information on the Library website.