MIT/LCS/TR-341

# ROUTING NETWORKS FOR PACKET COMMUNICATION SYSTEMS

George Andrew Boughton

# Routing Networks for Packet Communication Systems

*by*

George Andrew Boughton

June, 1985

Copyright © 1985, Massachusetts Institute of Technology

This research was supported in part by the Department of Energy under grant number DE-AC02-79ER10473 and the National Science Foundation under grant number MCS-7915255.

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
Laboratory for Computer Science

Cambridge Massachusetts

# Routing Networks for Packet Communication Systems

by

George Andrew Boughton

Submitted to the Department of Electrical Engineering and Computer Science
on August 15, 1984 in partial fulfillment of the requirements
for the Degree of Doctor of Philosophy

## Abstract

This thesis examines the design of geographically centralized high performance packet switched networks called routing networks. Each of these networks is intended to be used to interconnect the modules of a highly parallel computer system. The design of such networks is considered in present (1984) technology where only a small number of network nodes can be placed on a single chip and in VLSI technology where a large number of nodes can be placed on a chip.

In both technologies, the design of routing networks for uniform patterns of communication is considered. In each technology, it is shown that the characteristics of these patterns imply a minimum cost for networks capable of supporting them. In present technology, the performance of a particular network that is well suited for uniform communication, the indirect n-cube routing network, is studied. The strongest constraint on the performance of the indirect n-cube network that is found still allows the the throughput of the network for uniform patterns of communication to grow linearly with the size of the network. In VLSI, the use of networks such as the crossbar and the indirect n-cube to support uniform patterns of communication is considered.

The design of routing networks for a few localized patterns of communication is briefly considered in both technologies. In each technology, networks that are well suited for these localized communication patterns are discussed.

Thesis Supervisor: Peter Elias
Title: Edwin S. Webster Professor of Electrical Engineering

**Keywords:** Routing Networks, Packet Switched Networks, Interconnection Networks, Indirect n-cube Networks, Packet Communication Systems, Data Flow Computers, VLSI

## Acknowledgments

I would like to thank my thesis supervisor, Peter Elias. Prof. Elias has provided important guidance both in the technical issues of this thesis and in their presentation. I am very fortunate to have had Prof. Elias involved in this thesis.

I would like to thank Jack Dennis. This thesis is based in part on the work on packet communication systems and data flow machines done by Prof. Dennis and his group. Many of the questions studied in this thesis were inspired by discussions with Prof. Dennis. I sincerely appreciate Prof. Dennis' sustained interest, without which this research would not have been possible.

I would like to thank Charles Leiserson for his willing review of drafts of this thesis and his constructive suggestions.

I would also like to thank some of my other friends. While it is impossible to list everyone who has encouraged me during the period of my thesis research, I would like to mention a few special people. I thank Gita Mithal and Arvind. I thank the past and present members of the CSG and FLA research groups especially Dean Brock, Bill Ackerman, Clement Leung, Keshav Pingali, Nena Bauman, Gaung Rong Gao, William Lim, Vinod Kathail, Bob Iannucci, Tom Wanuga, Steve Heller, David Culler, Bhaskar Guharoy, and Greg Papadopoulos. I thank my friends in Tech Squares especially Sue Curtis, Phil King, and Anne Mahoney. I thank the past and present members of the various Project Mac teams especially Gene Stark, Bill Weihl, Jeannette Wing, Joe Zachary, Judy Zinnikas, and Jim Restivo.

Finally, I would like thank the members of my family for their encouragement and support.

# CONTENTS

**1. Introduction ..... 5**

1.1 Overview ..... 5
1.2 Packet Communication Systems ..... 5
1.3 Routing Networks ..... 6
1.4 Research Topics ..... 8
1.5 Notation for Asymptotic Bounds ..... 12

**2. Design of Routing Networks Ignoring Wire Length ..... 14**

2.1 Network Restrictions ..... 14
2.2 Networks for Systems with Uniform Communication ..... 16
2.2.1 Model of the Problem ..... 16
2.2.2 Minimum Network Cost ..... 17
2.2.3 Routing Networks Using an Indirect n-Cube Topology ..... 20
2.2.3.1 Introduction ..... 20
2.2.3.2 Tree Buffering ..... 27
2.2.3.3 Effect of the Last Stage ..... 38
2.2.3.4 Interaction of Stages ..... 59
2.3 Networks for Systems with Localized Communication ..... 83

**3. Design of Routing Networks Considering the Cost of Wires ..... 87**

3.1 Introduction ..... 87
3.2 VLSI Model ..... 89
3.3 Networks for Systems with Uniform Communication ..... 90
3.3.1 Wire Cost ..... 90
3.3.2 Network Structures ..... 95
3.3.3 Multiple Chip Networks ..... 109
3.4 Networks for Systems with Localized Communication ..... 111

**4. Conclusion ..... 115**

4.1 Summary ..... 115
4.2 Suggestions for Further Work ..... 118

# 1. Introduction

## 1.1 Overview

Data communication considerations are becoming increasingly important in the design of high performance computers. Conventional single sequence computer designs have been refined to the point where only limited further improvements in performance can be expected without a dramatic improvement in the speed of available circuits. Levels of performance significantly higher than those of present computers can only be achieved by machines capable of substantial concurrency. To achieve high concurrency a machine must support a large flow of data and control signals. As we shall see below, a class of digital systems which seems well suited for the implementation of such machines in light of their communication requirements is the class of packet communication systems [6]. A packet communication system is composed of a number of subsystems interconnected by one or more communication networks called routing networks. Data transfer between two subsystems is accomplished by passing a packet over a network path from one to the other. This thesis examines several aspects of routing network design.

## 1.2 Packet Communication Systems

By definition, any digital system with the following properties is a packet communication system. A packet communication system is composed of modules and a set of links where each link connects one or more modules. A module can be any form of digital system, and may be capable of storing data and performing various operations on that data. The transmission of data from one module to another can only be accomplished by passing a packet along a path of connected intermediate modules. Thus the behavior of a module can depend only on its internal state and packets it receives from modules connected to it.

Packet communication systems seem well suited for implementing concurrent computers that need to support a large flow of data. Consideration of data communication enters at a very early stage in the

design of a packet communication system. In particular, the design of the component modules and the manner in which they are interconnected should reflect the inherent structure of the computations to be performed by the system. By patterning the structure of the system to the structure of the problem the designer can develop a system which supports the required flow of data with a minimum of hardware.

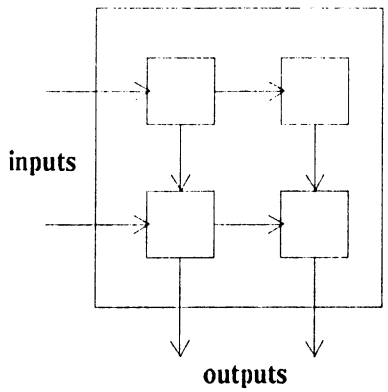
In addition, data communication by packet passing as done in packet communication systems can facilitate the efficient use of data paths. This is particularly true for systems such as the Dennis data flow machine [7] that require only short messages to be transferred among their component units. For these machines packet communication has advantages over the alternative circuit switching. In circuit switching a complete path between two units must be set up before a message can be transferred from one to the other and the entire path must remain allocated for the duration of the transfer. In a packet communication system a message is transferred in the form of a packet that is passed from module to module along its desired path. Thus, the message transfer only requires one link at any given time. Only the next link in a packet's desired path need be available for the packet to proceed.

## 1.3 Routing Networks

A routing network is by definition a packet communication system with designated input and output links (as shown in Figure 1) that has the ability to accept tagged packets on each of its designated inputs and to route each packet to the output corresponding to its tag. A routing network may be connected by its input and output links to other packet communication systems and thus provide intercommunication among them. A routing network as a packet communication system is composed of packet communication modules that are interconnected by links. For the purpose of discussion, the internal modules of a network will be called nodes.

Routing networks can be used to interconnect several small packet communication systems into a larger packet communication system. In such a system, the routing networks handle the required transfer

**Fig. 1. Example Routing Network**




---

of data and control packets among the various subsystems. Thus, an important part of the design of large packet communication systems is the design of their routing networks.

Routing networks are packet networks; data is transferred in the form of packets that are passed from node to node in the network. Since routing networks are also packet communication systems and since the control of a packet communication system can not be centralized, the routing of packets must be done in a decentralized fashion. The routing decisions made by a node for a given packet can depend only on the state information of the node and the label of the packet. There are, however, a number of differences between these networks which are intended for use in localized high performance systems and distributed packet networks such as the ARPA network. In contrast to the ARPA network where the cost of data paths dominates, both the cost of data paths and the cost of nodes must be considered in the design of a routing network. Thus, in the design of a routing network it is important to minimize the complexity of network nodes. Very large buffers for queuing packets or very large tables for storing routing information cannot be used. Unlike the ARPA network, a routing network has data paths and nodes of comparable speed and reliability. This suggests that a simple routing algorithm should be used by each node in a routing network in order to minimize the total time required for a packet to pass

through the network.

Routing networks differ from the majority of networks that have been studied in classical switching theory. While routing networks use packet switching, switching theory has been primarily concerned with networks that use circuit switching. Further, in contrast to routing networks that use decentralized control, most of the networks that have been studied in switching theory use centralized control. Finally, switching theory has assumed that the cost of wires is negligible in comparison to the cost of active switch elements. This assumption, as we see below, is not valid for some of the technologies that may be used to implement routing networks. Thus, while classical switching theory provides a good starting point for research on routing networks, most of the results that have been obtained do not directly apply to routing networks. In this thesis, we examine cost and performance issues for routing networks that are similar to the cost and performance issues that have been studied for circuit switched networks in classical switching theory. We will use cost measures that are appropriate for present and future integrated circuit technologies, and performance measures that are appropriate for the intended network applications.

## 1.4 Research Topics

This thesis examines the design of routing networks under two different sets of assumptions that correspond to two points on the apparent path of integrated circuit technology evolution. One set of assumptions corresponds to present technology where only a small number of network nodes can be implemented on a single integrated circuit. The other set of assumptions corresponds to a technology where a large number of network nodes can be implemented on a single integrated circuit.

In 1984 technology, consideration of the length of wire needed to implement a given link seems unimportant. It seems unlikely that more than a small number of modules can be implemented on a single integrated circuit. Links between modules can be implemented for the most part as printed circuit board wires. The length (as opposed to the number) of such wires is not a significant factor in the total

cost of the system. Similarly, the effect of wire length on system speed is small since even the propagation time of a long wire is less than the delay of the circuit required to drive it from an on-chip signal.

In Chapter 2, we examine the design of routing networks for present technology. We assume that the length of wire required to implement each network link does not affect either the cost or the performance of the network. Further, we place certain additional restrictions on the behavior and complexity of network nodes in order to narrow down the number of design parameters that must be considered. These restrictions, which are described in Section 2.2, are motivated by the current state of technology and the nature of the systems in which the networks may be used.

Given these restrictions, we examine in Section 2.3 the design of networks for a class of systems characterized by uniform communication; each source of packets in such a system generates packets for all the possible destinations and over the long run generates a comparable number for each destination. For the purpose of analysis, we introduce simple probabilistic models of the packet sources and sinks of a system with uniform communication. We examine the minimum number of nodes required by any network that is capable of high throughput when it is connected to the model sources and sinks. We study a particular network, the indirect n-cube routing network, that seems well suited for uniform communication and has a number of nodes within a constant factor of the lower bound. Below, we use the term indirect n-cube network to refer to the indirect n-cube routing network.

It should be noted that networks related to routing networks in general and the indirect n-cube network in particular have been studied in the literature. Sorting networks, networks capable of sorting N data items in parallel where N is the number of network inputs, are clearly not the same as routing networks, but intuition would suggest that these two types of networks have similar complexities. Sorting networks using $O(N \log^2 N)$ nodes have been known for some time [4]. More recently, $O(N \log N)$ node sorting networks have been described [3], although these networks may not be of practical interest because of the very large constant factor. The indirect n-cube network has a comparable complexity; it

uses O(N log N) nodes. Earlier work on routing networks with structures similar to that of the indirect n-cube network has been done [5]. Other networks with related structures have been described in the literature. Some of these networks have been proposed to perform permutations on large vectors of data. In general, these permutation networks have been proposed for systems in which the elements of a given vector are processed synchronously and then permuted. These permutation networks include shuffle-exchange, omega, Pease's indirect n-cube permutation network, and cube-connected cycles [24, 14, 15, 16, 21, 23]. Other networks with related structures have been proposed to interconnect the processors and memories of other types of multiple processor systems. Circuit switched and packet switched banyan networks have been proposed for single instruction stream multiple processor systems [10, 26]. Circuit switched and packet switched delta networks have been proposed for multiprocessors in which each processor makes independent and random memory accesses [20, 8]. The relationship between the indirect n-cube routing network and previously studied networks is discussed in more detail in Section 2.2.3.1.

We consider in Section 2.3 the operation of large indirect n-cube routing networks when connected to the model sources and sinks and examine certain important characteristics of the operation of these networks. We examine the influence of these characteristics by using network models that accurately model these characteristics and that are considerably simpler than the actual network. By analyzing and simulating the models, we examine the influence of these characteristics on certain aspects of the performance of the networks. The performance predicted by these models is compared to the performance of the actual network which we measure by simulation.

One important aspect of performance that we study is throughput. We examine the relationship between network throughput and network size. We would like the throughput of the network to scale linearly with the number of network inputs since if we form a composite packet communication system by using a routing network to interconnect several subsystems, we would like the performance of the

system to scale linearly with the number of subsystems.

Another important aspect of performance that we study is the speed of slow inputs. If a particular network input becomes extremely slow due to network congestion then packets from the module connected to that input can be delayed. If the congestion continues for a long period, a large number of modules that are either directly or indirectly dependent on the blocked module in a highly parallel computation can be affected.

Our study suggests that very large indirect n-cube networks can support high performance for uniform communication patterns. The strongest constraint on network throughput that we find in our study still allows throughput to grow linearly with network size. However, our study also indicates that some of the inputs of a very large network can be slow for a very long period of time.

We also briefly examine in Section 2.3 the design of routing networks for a class of systems characterized by localized communication; the majority of packets generated by a particular source in such a system are tagged for a small group of destinations. Many localized communication patterns can be supported with networks that are less complex than the indirect n-cube network. We discuss one obvious family of networks that seem appropriate for some important localized communication patterns. One of the characteristics of networks of this family is a number of nodes equal to the sum of the number of network inputs and the number of network outputs. This family includes grid structured networks and tree structured networks.

In Chapter 3, we examine the design of routing networks in the technology of five to ten years from now.

As technology changes and the number of network nodes that can be placed on a given integrated circuit increases, the importance of wire length in network cost will increase. Within a few years it should

be possible to implement many modules and the links that interconnect them on a single chip. The length of wire required to implement each link will then be a significant factor in the cost of silicon required to implement a system, since the chip acreage used by each wire is proportional to its length. However, it is likely that for the immediate future, the next five to ten years, even long wires can be driven quickly if drivers of the appropriate size are used.

In Chapter 3, we make some assumptions about the characteristics of the integrated circuit technology of the next five to ten years and study the design of routing networks under these assumptions. For the purpose of discussion, we refer to the technology that will exist at the end of this period as very large scale integration (VLSI). We describe a model of VLSI based on some assumptions about the characteristics of VLSI. We examine in the VLSI model the fundamental cost of a single chip network to support a certain level of performance for uniform patterns of communication. We examine a few structures that seem appropriate for implementing a single chip uniform communication network in VLSI. These structures include a crossbar structure, and an indirect n-cube structure. We discuss a technique for interconnecting such single chip networks to form larger uniform communication networks. We also briefly examine the design in VLSI of networks for localized patterns of communication. We examine a few example network structures and describe the communication patterns that they can support.

## 1.5 Notation for Asymptotic Bounds

We use the following notation to describe asymptotic bounds.

We say that f(N) is $\Omega(g(N))$ if and only if there exists $N_0$ and $c$ greater than zero such that f(N) is greater than or equal to $c\,g(N)$ for all N greater than $N_0$.

We say that f(N) is O(g(N)) if and only if there exists $N_0$, $c_1$, and $c_2$ greater than zero such that $c_1 g(N) \leq c_2 g(N)$ for all N greater than $N_0$.

We say that f(N) is O(g(N)) if and only if there exists $N_0$ and $c$ greater than zero such that f(N) is less than or equal to $c\,g(N)$ for all N greater than $N_0$.

# 2. Design of Routing Networks Ignoring Wire Length

## 2.1 Network Restrictions

In general, we will be concerned only with routing networks that obey the restrictions described in the following paragraphs.

We assume that time is divided into units and that any operation in the network starts on a transition between two time units and finishes before the next transition. It is important to note in passing that this assumption of purely synchronous behavior is only for the convenience of analysis and that the networks that we shall present can easily be designed to function asynchronously.

We assume that there is some limit $k$ on the total number of input and output links that can be attached to a particular network node, and we assume that there is a limit $o$ on the total number of packets that may be buffered at any particular network node at any particular time. These restrictions are motivated by our desire to bound the amount of chip space and number of external connections required to implement a network node as a portion of an integrated circuit.

For the most part, we assume that only two nodes are connected to a given link. The only exception is our discussion of the minimum cost of a network to support high throughput for uniform patterns of communication. In that discussion, we get a more general lower bound by assuming that an arbitrary number of nodes can be connected to a given link. In all cases, we assume that one link can transfer at most one packet per unit time.

In the case that only two nodes are connected by a link, we assume that the two nodes observe a ready/acknowledge protocol for transferring packets. In particular, each link contains *acknowlege* and *ready* control lines in addition to the *data* lines used to transfer the packets as shown in Figure 2. The protocol has 4 phases as shown in Figure 3. A sending node can place a new packet on the *data* lines of

Fig. 2. Lines of a Link

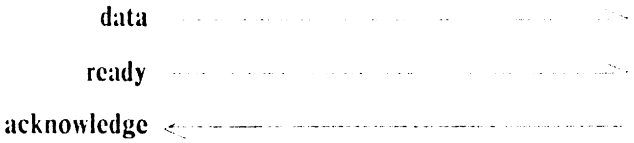
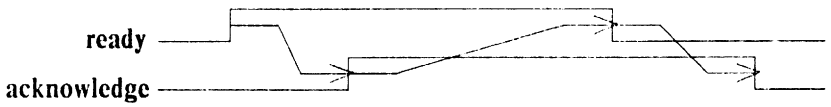



Fig. 3. 4 Phase Protocol




the link only if all 4 phases of the transmission of the previous packet have been completed. The sending node asserts the *ready* line after placing the packet on the *data* lines and will continue to assert the *ready* and *data* lines until the *acknowlege* line is asserted by the receiving node. The receiving node can accept a packet available on the *data* lines only after the *ready* line has been asserted, and the receiving node will assert the *acknowlege* line only after it has safely stored the packet in one of its buffers. This protocol was chosen since it is widely known and provides in a straightforward manner the necessary coordination for packet passing.

We assume that the behavior of a node during a given time unit can depend only on the information available on its links and the contents of packets stored in its buffers, and we assume that the node's behavior can depend only on the destination label and not the data portion of any packet stored in the node. These restrictions are in part motivated by our desire to implement each network node as a portion of an integrated circuit. By limiting the complexity of each node's behavior we limit the amount of space required to implement the control circuits of each node. These restrictions are also motivated by

our desire to minimize the time required for a packet to traverse the network since by limiting the complexity of the control algorithm of each node we indirectly limit the time required by each node to process a packet. These restrictions seem natural and can be easily observed in the design of networks. However, it should be noted that there are many alternative sets of restrictions that could be placed on the node's behavior, and that a different set of restrictions might well lead to different network structures.

Finally, we assume that a packet that enters a node in a given unit of time can not leave that node until the next unit of time, and that a link can transfer at most one packet per unit time. Although we are ignoring the time required to propagate a packet over a link, we are not ignoring the time required to gate a packet through a node or store it in a buffer.

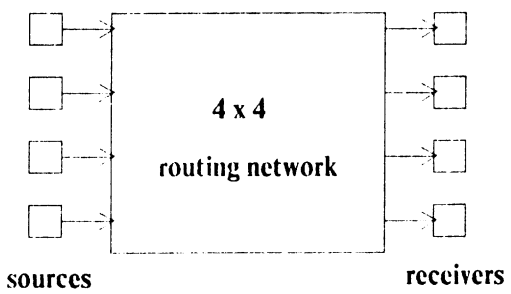
## 2.2 Networks for Systems with Uniform Communication

### 2.2.1 Model of the Problem

In general, a routing network will be used as shown in Figure 4 to connect a group of packet sources to a group of packet receivers. Each source will produce labeled packets and each packet must be

---

**Fig. 4. Use of a Routing Network**



delivered to the receiver which corresponds to its label.

We describe here a simple model of a system with uniform communication. For our model, we assume that the number of receivers equals the number of sources and that receivers and sources behave in a very simple manner. We assume that a receiver will process a message as soon as it arrives and can thus accept a packet every unit of time. We assume that if the network has accepted the last packet generated by a particular source, then the chance that the source will produce a new packet in a unit of time is P where P is a parameter of the model. We assume that the label of each packet is independently selected, and that all of the possible receiver labels are equally likely.

### 2.2.2 Minimum Network Cost

In this section, we find a lower bound on the complexity of any N-input N-output routing network capable of $\Omega(N)$ average throughput when each of its inputs is connected to a model source and each of its outputs is connected to a model receiver. We measure throughput as packets per unit time and complexity as the number of nodes in the network. We show that such a network requires $\Omega(N \log N)$ nodes. This result gives some motivation for the fact that the network that we study for such applications, the indirect n-cube network, and most related networks require $\Omega(\log N)$ stages of $\Omega(N)$ nodes.

For the purpose of this discussion, we allow an arbitrary number of nodes to be connected to a given link. As before, we allow only one packet to be transferred over a given link in a given unit of time. Clearly, the lower bound on network cost that we obtain by allowing an arbitrary number of nodes to be connected to a given link also holds if only two nodes are allowed to be connected to a given link.

**Proposition.** $\Omega(N \log N)$ nodes are required by any N input N-output routing network capable of $\Omega(N)$ average throughput when each of its inputs is connected to a model source and each of its outputs is connected to a model receiver.

*Proof.* Since we assume that a network node can be connected to at most $k$ links, we can get a lower bound on the number of network nodes by finding a lower bound on the sum, over all links in the network, of the number of nodes connected to each link. For the purpose of this discussion, we use the term connection to refer to the juncture between a network link and a network node. Thus, the sum, over all links in the network, of the number of nodes connected to each link is equal to the number of connections in the network. We get a lower bound on the number of connections in the network by considering the sum, over all the packets processed during a long period, of the number of connections used by each packet and by making use of the fact that a connection can be involved in at most one operation per unit time.

A lower bound on the number of connections in the network can be obtained by considering the operation of the network over a long period of time and examining the use of network connections during such a period. Let us consider a period of T time units for some very large T. Since we assume that only one packet can be transferred on a link in one unit of time, it follows that a connection can be used for only one packet in a given unit of time. The total number of connections in the network must be at least as great as (1/T) times the number of connection operations during the period where the number of connection operations is defined to be the sum, over all packets processed, of the number of connections used by each packet. It should be noted that we count each connection of a link separately, and that the the number of connections used by a packet includes each connection of each link used by the packet.

A lower bound on the expected number of connection operations during the T time unit period can be obtained by considering the possible paths through the network. For the purpose of this discussion, we assume that each network link is logically composed of some number of link segments. In particular,

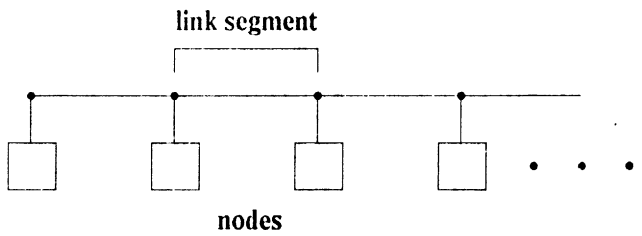
we assume that the nodes connected to a given link are connected according to some linear order, and that a link segment is a portion of a link between two adjacent nodes as shown in Figure 5. A link which is connected to $i$ nodes has $i$ - 1 segments. By this definition, a connection between a link and a node can involve at most two link segments, and at most 2 $k$ link segments in total can be connected to a node. Less than

$$(2k\text{-}1)^0 + (2k\text{-}1)^1 + \cdots (2k\text{-}1)^i = \frac{(2k\text{-}1)^{i+1}\text{-}1}{(2k\text{-}1)\text{-}1} \qquad (1)$$

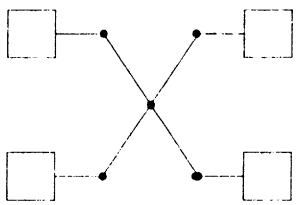
outputs can be reached from a particular input by a path containing $i$ or fewer link segments. Thus, there are less than N/2 outputs that can be reached from a given input using a path containing no more than $(\log_{(2k\ -\ 1)}(N/2))$ - 1 link segments. Since a model source randomly selects a destination for each packet it generates and since all destinations are equally likely, there is at least a 50% chance that a packet generated by a model source must travel over a path of greater than $(\log_{(2k\ -\ 1)}(N/2))$ - 1 link segments.

---

**Fig. 5. Conceptual Model of Nodes Connected to a Link**




**note: Actual implementation need not correspond to the conceptual model. For example,**



It follows that the expected number of link segments used by such a packet must be $\Omega(\log N)$. Since the number of nodes connected to a given link is equal to the number of segments in the link plus one, the total number of nodes connected to the links in the path of a particular packet must be larger than the number of link segments in the path. Thus, a packet generated by a model source is expected to use at least $\Omega(\log N)$ connections. Since by assumption the expected number of packets processed in the T time unit period is $\Omega(TN)$, the expected number of connection operations during the period must be greater than $\Omega(TN \log N)$.

A lower bound on the number of nodes in the network follows. Since the expected number of connection operations during the T time unit period is $\Omega(TN \log N)$ and since a connection can be involved in at most one operation in a given unit of time, the network must have $\Omega(N \log N)$ separate connections between links and nodes. Since each node can be connected to at most $k$ links and since $k$ is fixed and is independent of N, the network must have $\Omega(N \log N)$ nodes.

Thus, there must be $\Omega(N \log N)$ nodes in a N-input N-output routing network capable of $\Omega(N)$ average throughput when each of its inputs is connected to a model source and each of its outputs is connected to a model receiver.

This ends the discussion of the proposition. ∎

### 2.2.3 Routing Networks Using an Indirect n-Cube Topology

#### 2.2.3.1 Introduction

The network shown in Figure 6, the indirect n-cube network, seems well suited for applications with uniform communication and has a cost of the same order as the lower bound derived in the previous section.

In this introduction, we describe the indirect n-cube network, we discuss the relationship between previously studied networks and the indirect n-cube network, and we give a brief overview of our work on the indirect n-cube network.

An indirect n-cube network is constructed as shown in Figure 6. A N-input network is composed from two N/2-input networks and N/2 nodes (called routers). Each node has two input and two output links. This construction yields an interconnection which is topologically equivalent to the interconnection of butterflies in the radix two fast Fourier transform [11]. In total (N/2) $\log_2$ N nodes are required. One and only one path exists from a given input to a given output. If network outputs, stages, and node outputs are numbered as shown in Figure 7, then at the *i*th stage the appropriate path follows the node output that corresponds to the *i*th most significant bit of the binary representation of the number of the destination output.

Each node of the network can be structured as shown in Figure 8. The node has a fifo buffer capable of storing some number of packets on each of its input links. If at the beginning of a time unit a

---

**Fig. 6. NxN Indirect n-Cube Network Construction**

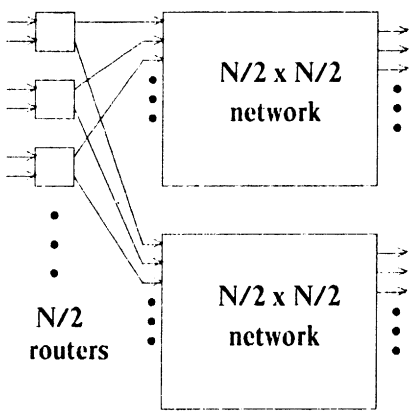


Fig. 7. Example Indirect n-Cube Network

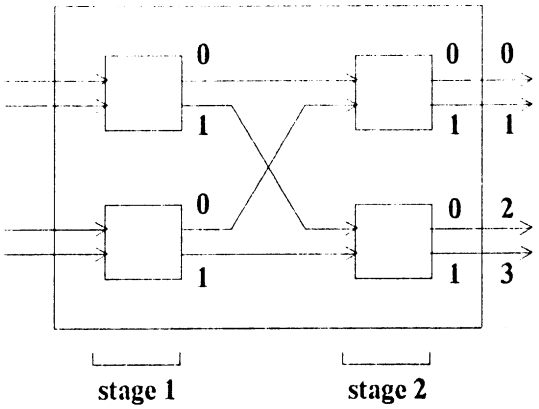
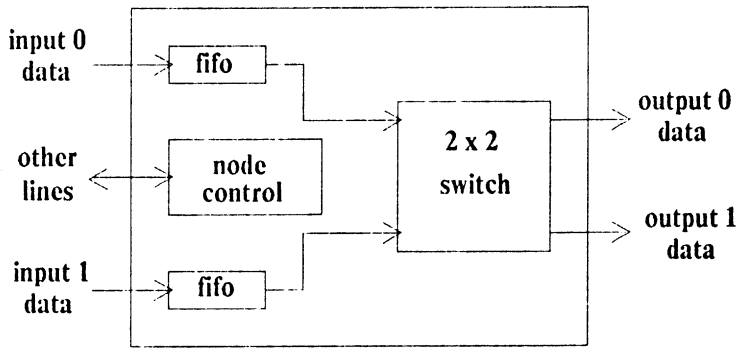



Fig. 8. 2x2 Node




buffer is not full and the corresponding input *ready* line is asserted then the node control places the packet available on the *data* lines in the buffer and asserts the returning *acknowlege* line before the end of the time unit. If a buffer is not empty at the beginning of a time unit then the node control attempts to place the packet which entered the buffer first on the output link corresponding to its destination. If the node control can do this, either because no conflict exists or because of arbitration of the conflict, then it

asserts the corresponding output *ready* line. If an *acknowledge* is returned on that link before the end of the time unit, then the node control removes the *ready* and removes the packet from the buffer. Otherwise, *ready* and the packet are removed during the first time unit when *acknowledge* is returned.

As was mentioned in the first chapter, there are a large number of networks that have structures that are related to that of the indirect n-cube network. The topology of the indirect n-cube routing network is identical to that of several other networks including omega, s=f=2 banyan, and Pease's indirect n-cube permutation network [16, 10, 21]. Networks with the same topology have also been called delta networks [20]. The topology of the shuffle-exchange network [24] is also related since an omega network is simply a cascade of $\log_2$ N shuffle-exchange networks. As was mentioned in the first chapter, these related networks have been proposed for a variety of uses. Some of these networks have been proposed to perform permutations on large vectors of data. In general, these permutation networks have been proposed for systems in which the elements of a given vector are processed synchronously and then permuted. Other related networks have been proposed to interconnect the processors and memories of other types of multiple processor systems.

The work on networks for the synchronous permutation of large vectors of data has been mostly concerned with the types of permutations that can be realized by a given number of passes through such a network, and thus that work does not directly address the question of how well such networks perform when interconnecting the modules of a packet communication system.

Some of the work on interconnection networks for other types of multiple processor systems is more closely related to our study of the indirect n-cube network. We discuss a few pieces of this work. The first is the work of Valiant [28]. Valiant has suggested the use of networks such as the packet switched n-cube for interconnection of processors and memories in a synchronous multiprocessor system. Valiant introduces the concept of an idealistic computer composed of processors that operate synchronously and a memory that the processors share. He considers algorithms such that no memory

location is accessed in a given computational step by more than some constant number of processors. He assumes that the idealistic computer can implement each computational step in a single unit of time. He considers the simulation of the idealistic computer on a realistic computer with a packet switched n-cube network. Each node of the network has the capacity to buffer a number of packets proportional to the log of the number of processors. Valiant shows that with high probability the realistic computer can implement a computational step in time proportional to the log of the number of processors. While it seems plausible that the memory accesses corresponding to several computational steps can be pipelined in the realistic computer, Valiant does not show this. Thus, Valiant's work differs from ours in at least three ways. First, he considers systems of synchronous processors and we consider systems of largely independent asynchronous processors. Second, in each network node he allows buffering proportional to the log of the number of processors and we allow only buffering of fixed size. Finally, he does not consider the pipelining of packets of different computational steps through his network and we consider a continual flow of packets through our networks.

Upfal [27] has shown similar results for networks of fixed degree. He uses the d-way digit-exchange graph. A processor is associated with each network node and each network node is assumed to have O(log N) buffers where N is the number of processors. It is assumed that a packet is initially at each processor and that each packet is destined for some other processor. No two packets are destined for the same processor. Upfal shows that with high probability all the packets can be delivered to their destinations in O(log N) time.

Patel has suggested the use of circuit switched delta networks for multiprocessors in which each processor makes independent and random memory accesses [20]. For his analysis, Patel assumes that memory requests in a multiprocessor are generated in a manner similar to the manner in which packets are generated by our model sources. The primary distinction between our work and that of Patel is the fact that our routing networks are packet switched. In Patel's circuit switched network, the transmission

of a message requires the use of a circuit through all of the stages of the network. In our routing networks, at any given time a packet only uses one link to go from its present stage to the next stage. The throughput of Patel's networks do not grow linearly with their size. As we shall see, there is reason to believe that the throughput of the indirect n-cube network for uniform patterns of communication does grow linearly with its size.

Dias and Jump [8] have done work on the use of packet switched delta networks as interconnection networks for multiprocessors and this work is very closely related to our study of the indirect n-cube network. Their network is topologically identical to the indirect n-cube. Their analysis assumes that the packets and the labels on the packets are generated in a manner similar to the way that packets and packet labels are generated by our model sources. They analyze their networks using network models in a manner similar to the way that we analyze the indirect n-cube network using network models. However, their models differ from ours. They use a Markov model to develop approximate equations for the state probabilities of a router in a given stage in terms of the state probabilities of the routers connected to it. They simultaneously solve the equations for all the stages. Their analysis makes several approximations. The analysis assumes that the routers of a given stage are independent. Also, the analysis of a given router assumes that the state probabilities of routers connected to it are independent of the state and history of the given router. Some of the characteristics of network behavior that we study in our models violate these assumptions of independence. For modest sized networks, the network throughput predicted by our models is consistent with that predicted by their models. However, for very large networks our throughput predictions differ from theirs. Since their model assumes more independence than ours one might expect it to predict higher throughput, but in fact the way that their assumptions are used in their model leads to a prediction of lower throughput. Their model predicts that the normalized throughput goes to zero as the network size goes to infinity [9] where normalized throughput is defined to be network throughput divided by network size. All of the constraints on network throughput that we study allow a non zero asymptote. We believe that our study considers all of the constraints represented

by their model and several that are not. We believe that the asymptotic prediction of their model is the result of the way that their assumptions are used in their model and does not reflect any real constraint on the throughput of the network. Another difference between their work and ours comes from the fact that in addition to studying the average network throughput we also consider the speed of slow network inputs. Their work is primarily concerned with the average network throughput and delay.

Recently, Pippenger [22] has extended the results of Valiant and Upfal to a network with a fixed amount of buffering at each node. In his work, Pippenger uses the d-way digit-exchange graph. A processor is associated with each network node. Only a fixed amount of buffering is assumed at each node. It is assumed that a packet is initially at each processor and that each packet is destined for some other processor. No two packets are destined for the same processor. Pippenger assumes that each node obeys certain rules concerning the order in which it processes the packets that it receives, and shows that if the rules are obeyed then with high probability all the packets can be delivered to their destinations in O(log N) time.

While there are differences between Pippenger's work and ours, Pippenger's results are significant and have some bearing on our work. Pippenger's network differs from the indirect n-cube network; the indirect n-cube network would be more closely related to Pippenger's network if the inputs of the indirect n-cube network were connected to the outputs and if a processor were associated with each network node of the indirect n-cube network. The type of network operation that Pippenger considers differs from the type that we consider; Pippenger does not consider the pipelining of waves of packets through his network and we consider a continual flow of packets through the indirect n-cube network. However, by establishing certain additional rules for the operation of the nodes of the indirect n-cube it may be possible to extend Pippenger's approach to provide results on the performance of the indirect n-cube network for uniform communication. The additional rules would concern the order in which a network node processes the packets that it receives, and possibly the removal of unusual blockages. It may be

possible to show that the normalized throughput of the indirect n-cube network with the additional rules approaches a non zero constant. Such a result is plausible, but even if the result holds it may be very difficult to prove. In any case, such a result is consistent with our work. Our work considers the performance of the indirect n-cube without rules such as those mentioned above. Even without such rules, the strongest constraint on network throughput that we find still allows normalized throughput to approach a non zero constant.

In the following pages, we examine the effect of certain important characteristics of the operation of very large indirect n-cube networks. In particular, we study in 2.2.3.2 the effect of congestion at a single router. In 2.2.3.3, we study the effect of congestion in a single stage of routers. In 2.2.3.4, we study the effect of the interaction of routers of different stages. As was mentioned earlier, we examine the effect of these characteristics of network behavior by using network models that accurately model these characteristics and that are considerably simpler than the actual network. By analyzing and simulating the models, we examine the effect of these characteristics on network throughput and the speed of slow network inputs. As was mentioned earlier, our study suggests that very large indirect n-cube networks can support high performance for uniform communication patterns. The strongest constraint on network throughput that we find in our study is caused by the interaction of routers of different stages and it still allows throughput to grow linearly with network size. However, our study of the interaction of routers of different stages also indicates that some of the inputs of a very large network can be slow for a very long period of time.

### 2.2.3.2 Tree Buffering

The first characteristic of network operation that we examine is tree buffering. We use the term tree buffering to refer to the buffering of packets that occurs in front of a congested router. Such buffering involves a tree structure of buffers. As a result, congestion at one router in a given stage can affect a large number of other routers.

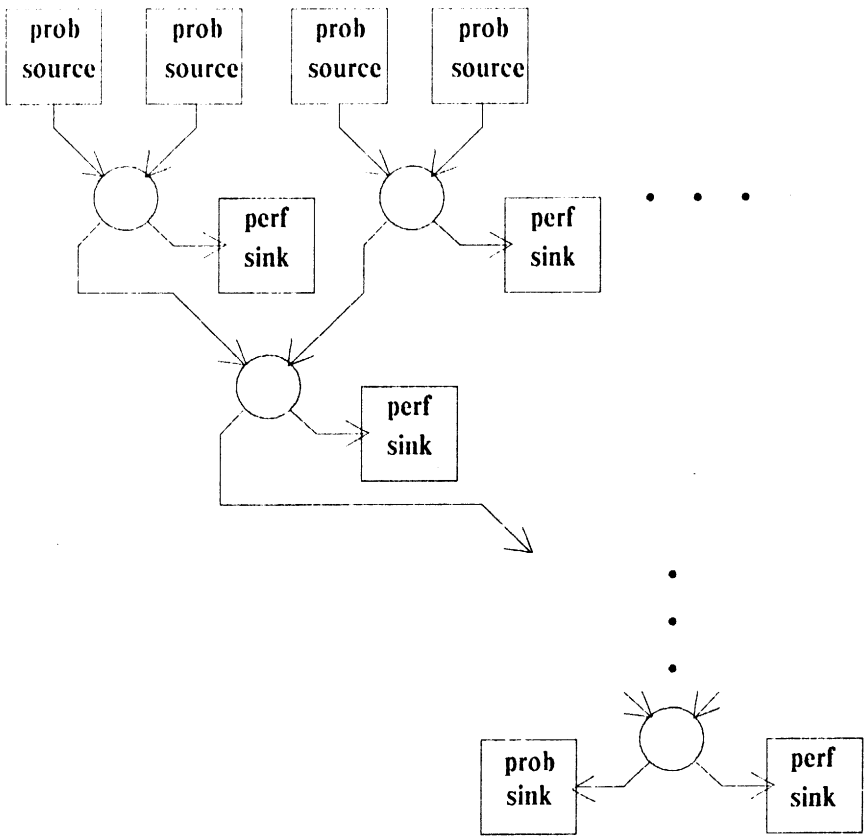
In this section, we examine tree buffering in front of a single congested router. In the next section, we examine a stage of routers and we examine the probability that at least one of the routers of the stage has a deep tree buffered in front of it.

In this section, we use a network model to examine how much tree buffering can occur in front of a congested router. In the model congestion can occur only at one router. We will study the model in order to determine the amount of buffering that occurs as a function of the amount of congestion and the overall network input rate.

As we shall see, the model suggests that deep tree buffering in front of a slow router occurs if the rate at which the router can accept packets is close to the rate at which packets that must go through that router are generated. In particular, the model suggests that if the rate at which packets are generated on each network input is $IN$ and if the rate at which the router can accept packets on its input is $OUT$ then the expected number of packets buffered in front of that input is greater than $2^{(\frac{IN}{2(OUT-IN)(B+1)}-2)}-1$ where $B$ is the size of the buffers. It should be noted that some aspects of this expression are intuitive. The $\frac{IN}{OUT-IN}$ or $\frac{IN/OUT}{1-IN/OUT}$ factor in the exponent is similar to the expected occupancy of certain types of queues in classical queueing theory, and reflects the queueing of packets that must be passed through the congested router. The exponential growth in the total number of packets buffered, most of which do not have to pass through the congested router, comes from the tree structure of routers involved. The $\frac{1}{B+1}$ factor in the exponent reflects the influence of the size of the input buffers of the routers.

The model that we use is shown in Figure 9. The model is composed of a tree of routers as shown in the figure. The depth of the tree is a parameter of the model. The first output of the router at the root of the tree is connected to a probabilistic packet sink and the second output is connected to a perfect packet sink. All other routers have their first output connected to a router in the following stage and have their second output connected to a perfect packet sink as shown in the diagram. Each input of each

Fig. 9. Tree Buffering Model




router at a leaf of the tree is connected to a probabilistic packet source.

The tree of routers of the model represents the tree of routers in front of a congested router in the network. The probabilistic sink represents the congested router. The perfect sinks represent the routers directly connected in the network to the tree of routers being studied.

The routers in this model, unlike the routers of the network, operate instantly. Thus, a packet will ripple through the model in one time unit. It will either be output at a perfect sink or it will run into

other packets buffered as a result of congestion at the probabilistic sink. This assumption allows us to easily study buffering due to congestion at the probabilistic sink since it is the only type of buffering that occurs.

The fifo buffers in the routers all have the same size $B$. $B$ is a parameter of the model.

The probabilistic packet sink contains a fifo buffer whose size is the same as that of the buffers in the routers. Packets input to the sink are placed in the fifo buffer. If at the beginning of a time unit the fifo is not empty then with probability $OUT$ the sink removes one packet from the fifo buffer. $OUT$ is a parameter of the model.

The perfect sinks never block and accept packets at whatever rate they are presented.

The probabilistic packet sources produce packets. If the input buffer connected to a probabilistic source is not full at the beginning of a time unit, then with probability $IN$ the probabilistic source places an additional packet in the buffer. $IN$ is a parameter of the model. The tag for each packet has as many bits as the depth of the network. Each bit of each tag is independently and randomly selected with one and zero being equally likely.

We can obtain a rough understanding of the operation of the model without much effort by considering the packets that are buffered in the model and that are tagged for the probabilistic sink, and by examining the expected number of such packets as a function of $IN$ and $OUT$. For the purpose of discussion, we call these *bps* (buffered probabilistic sink) packets.

*bps* packets can only leave the model at the probabilistic sink. From the operation of the model we can conclude that if at least one *bps* packet exists then the buffer in the probabilistic sink must contain at least one packet. Thus, in a given unit of time if any *bps* packets exist then with probability $OUT$ one will be consumed by the probabilistic sink.

A *bps* packet can enter the network from any one of the probabilistic sources. If the depth of the model is $d$ then the number of sources is N where N is equal to $2^d$. The chance that an unblocked source produces a *bps* packet in given unit of time is $IN/N$. If N is large and if none of the sources is blocked then the chance that $k$ *bps* packets are produced in a given unit of time may be approximated by $\frac{(IN)^k e^{-IN}}{k!}$. The accuracy of this approximation increases with the size of N and is exact for infinite N.

In the following paragraphs we will examine for a very large network model the expected number of *bps* packets as a function of $IN$ and $OUT$. We will assume that $OUT > IN$. We will assume that the network is large enough that the chance of a blocked source is small and can be ignored. We will assume that the network starts at time zero with no *bps* packets.

**Proposition.** The expected value of the limiting distribution for the number of *bps* packets is equal to

$$\frac{2IN - IN^2}{2(OUT - IN)}. \tag{2}$$

*Proof.* We find the average number of *bps* packets using an approach similar to that used by Kleinrock for the M/G/1 queue [12]. For the purpose of discussion, we use the notation $q_n$ to represent the number of *bps* packets at time $n$. We use $\Delta_{n+1}$ to represent the number of *bps* packets served between $n$ and $n+1$. $\Delta_{n+1}$ is of course equal to either zero or one. We use $v_{n+1}$ to represent the number of *bps* packets generated between $n$ and $n+1$.

From these definitions it follows that $q_{n+1}$, the number of *bps* packets at time $n+1$, is given by the equation

$$q_{n+1} = q_n - \Delta_{n+1} + v_{n+1}. \tag{3}$$

If we square both sides we get

$$q_{n+1}^2 = q_n^2 + \Delta_{n+1}^2 + v_{n+1}^2 - 2q_n\Delta_{n+1} + 2q_n v_{n+1} - 2\Delta_{n+1}v_{n+1}. \tag{4}$$

Let us form the expectation of both sides. We use the notation $E[x]$ to represent the expected value of $x$. Also we make use of the fact that $\Delta_{n+1}^2$, the square of the number of *bps* packets served between $n$

and $n+1$, is either one or zero and is equal to $\Delta_{n+1}$. $E[q_{n+1}{}^2]$, the expected value of the square of the number of *bps* packets at time $n+1$, is given by the equation

$$E[q_{n+1}{}^2] = E[q_n{}^2] + E[\Delta_{n+1}] + E[v_{n+1}{}^2] - 2E[q_n \Delta_{n+1}] + 2E[q_n v_{n+1}] - 2E[\Delta_{n+1} v_{n+1}]. \quad (5)$$

Since $v_{n+1}$, the number of *bps* packets generated between $n$ and $n+1$, is independent of $q_n$ and $\Delta_{n+1}$,

$$E[q_{n+1}{}^2] = E[q_n{}^2] + E[\Delta_{n+1}] + E[v_{n+1}{}^2] - 2E[q_n \Delta_{n+1}] + 2E[q_n]E[v_{n+1}] - 2E[\Delta_{n+1}]E[v_{n+1}]. \quad (6)$$

We are interested in the limit as $n$ goes to infinity. We are interested in the limiting distribution for the random variable $q_n$, the number of *bps* packets at time $n$. We denote the limiting distribution by $\tilde{q}$. Similarly we use the limiting distribution for the random variable $v_n$, the number of *bps* packets generated between $n$ and $n+1$. We denote the limiting distribution by $\tilde{v}$. We assume that the $j$th moment of $q_n$ exists in the limit as $n$ goes to infinity independent of $n$, namely,

$$\lim_{n\to\infty} E[q_n{}^j] = E[\tilde{q}^j]. \quad (7)$$

We make a similar assumption about the $j$th moment of $v_n$. We make use of the fact that $\lim_{n\to\infty} E[\Delta_{n+1}]$ must equal the average input rate, $IN$. Thus,

$$E[\tilde{q}^2] = E[\tilde{q}^2] + IN + E[\tilde{v}^2] + 2E[\tilde{q}]E[\tilde{v}] - 2(IN)E[\tilde{v}] - \lim_{n\to\infty} 2E[q_n \Delta_{n+1}]. \quad (8)$$

The probability that $\Delta_{n+1} = 1$ given that $q_n > 0$ is $OUT$. Thus,

$$\lim_{n\to\infty} E[q_n \Delta_{n+1}] = (OUT)E[\tilde{q}] \quad (9)$$

and

$$0 = IN + E[\tilde{v}^2] + 2E[\tilde{q}]E[\tilde{v}] - 2(IN)E[\tilde{v}] - 2(OUT)E[\tilde{q}]. \quad (10)$$

The probability that $\tilde{v} = k$ is $\frac{(IN)^k e^{-IN}}{k!}$. This of course is the Poisson distribution. Thus,

$$E[\tilde{v}] = IN, \quad (11)$$

$$E[\tilde{v}^2] = IN^2 + IN, \quad (12)$$

and

$$0 = IN + IN^2 + IN + 2(IN)E[\tilde{q}] - 2IN^2 - 2(OUT)E[\tilde{q}] \quad (13)$$

or

$$2IN - IN^2 = 2(OUT - IN)E[\tilde{q}]. \tag{14}$$

Therefore, $E[\tilde{q}]$, the expected value of the limiting distribution for the number of *bps* packets, is equal to

$$\frac{2IN - IN^2}{2(OUT - IN)}. \tag{15}$$

This ends the discussion of the proposition. ∎

Thus, the expected number of *bps* packets is very large if and only if $OUT$ is very close to $IN$. For example, if $OUT - IN$ is equal to $1/a$ for some constant $a$ then the expected number of *bps* packets is less than $a\,IN$.

Other measures of the amount of buffering in the network model can be deduced from this result for *bps* packets.

For the purpose of discussion, we define some notation. We use the notation $p_n$ to denote the total number of packets buffered at time $n$. We use $\tilde{p}$ to represent the limiting distribution for the random variable $p_n$. We define $f_1(i)$ as follows.

$$f_1(i) = 0, \qquad \text{if } i = 0,$$

and

$$f_1(i) = \log_2 i, \qquad \text{if } i > 0. \tag{16}$$

We use $l_n$ to represent $f_1(p_n)$. We use $\tilde{l}$ to represent the limiting distribution for the random variable $l_n$.

**Proposition.** The expected value of $\tilde{p}$, the limiting distribution for the total number of packets buffered, is greater than

$$2^{\left(\frac{IN}{2(OUT-IN)(B+1)} - 2\right)} - 1. \tag{17}$$

*Proof.* The desired result can be obtained from a lower bound on the expected value of $\tilde{l}$, the limiting distribution for $f_l(p_n)$. And $E[\tilde{l}]$ in turn can be obtained from the expected value of $\tilde{q}$, the limiting distribution for the number of *bps* packets.

$l_n$, $f_l(p_n)$, can be related to $q_n$, the number of *bps* packets at time $n$. We use $P[x = i]$ to represent the probability that $x$ equals $i$ and $P[x = i | y = j]$ to represent the probability that $x$ equals $i$ given that $y$ equals $j$. Clearly, $E[q_n]$, the expected number of *bps* packets at time $n$, is equal to

$$\Sigma_{i \geq 0} \Sigma_{j \geq 0} j P[q_n = j | l_n = f_l(i)] P[l_n = f_l(i)]. \tag{18}$$

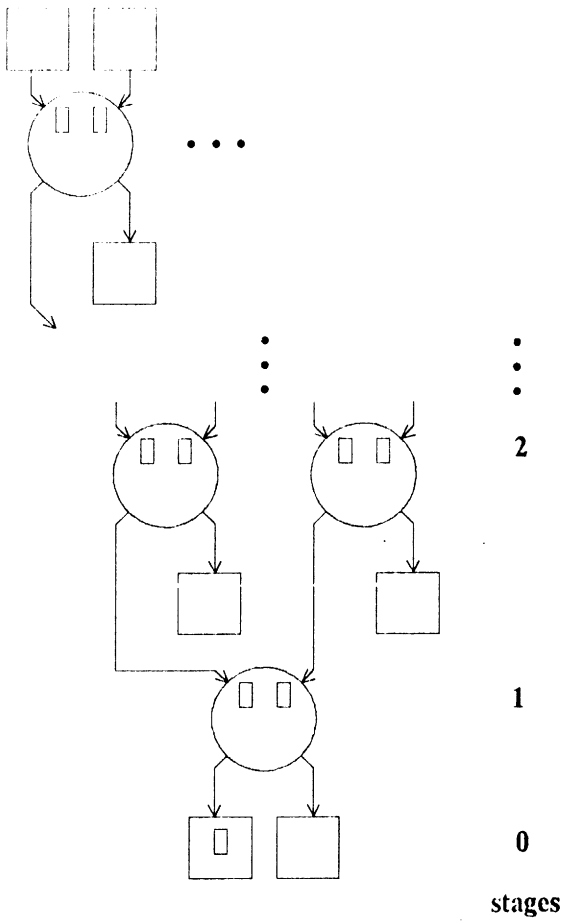
We use the notation $E[x | y = i]$ to represent the expected value of $x$ given that $y$ equals $i$. Thus,

$$E[q_n] = \Sigma_{i \geq 0} (E[q_n | l_n = f_l(i)]) P[l_n = f_l(i)]. \tag{19}$$

An upper bound on $E[q_n | l_n = f_l(i)]$, the expected number of *bps* packets at time $n$ given that the total number of packets buffered at time $n$ is equal to $i$, can be obtained by examining the model in more detail.

Packets buffered in stages close to the probabilistic sink are more likely to be *bps* packets than packets buffered in distant stages. Consider Figure 10. For the purpose of discussion we number the stages of the routers in the model as shown. All the packets in the fifo buffer of the probabilistic sink must be *bps* packets. If a fifo of the router in stage one contains one or more packets then the packet at the output of that fifo must be a *bps* packet. If that fifo contains more than one packet, any packet that is not at the output may be a *bps* packet. The probability that such a packet is a *bps* packet is 1/2. Similar statements can be made about packets buffered in higher stages. The probability that a packet at the output of a fifo in stage $k$ is a *bps* packet is $1/2^{(k-1)}$. If that fifo contains other packets, the probability

Fig. 10. Model Buffers Involved in Tree Buffering




---

that such a packet is a *bps* packet is $1/2^k$.

Thus, we can get an upper bound on $E[q_n | l_n = f_1(i)]$ by assuming that the buffered packets are packed in the lower stages. We assume that at time $n$ all of the $i$ buffered packets are in the $(f_2(i))+1$ lowest stages where $f_2(i)$ is the smallest non negative integer such that the capacity of the lowest $(f_2(i))+1$ stages is greater than or equal to $i$. The $(f_2(i))+1$ lowest stages include stage 0 through stage $f_2(i)$. Notice that

$$\Sigma_{k=0 \text{ to } ((f_2(i))-1)}\ B2^k < i \leq \Sigma_{k=0 \text{ to } f_2(i)}\ B2^k \qquad \text{for } i>0 \tag{20}$$

and

$$B(2^{(f_2(i))}-1) < i \leq B(2^{((f_2(i))+1)}-1) \qquad \text{for } i>0. \tag{21}$$

As a result,

$$f_2(i) \leq \log_2(i/B+1) \qquad \text{for } i \geq 0. \tag{22}$$

Given the discussion of the previous paragraph, the definition of $f_2(i)$, and the fact that the $k$ th stage contains $2^k$ fifos, $(E[q_n | l_n = f_1(i)])$, the expected number of *bps* packets at time $n$ given that the total number of packets buffered at time $n$ is equal to $i$, is less than or equal to

$$B + \Sigma_{k=1 \text{ to } f_2(i)}\ 2^k\left(\frac{1}{2^{(k-1)}}+\frac{B-1}{2^k}\right) \tag{23}$$

or

$$(E[q_n | l_n = f_1(i)]) \leq B + (f_2(i))(B+1). \tag{24}$$

Given the relation (22), we can conclude that

$$(E[q_n | l_n = f_1(i)]) \leq B + (\log_2(i/B+1))(B+1). \tag{25}$$

This implies a relationship between $E[q_n]$, the expected number of *bps* packets at time $n$, and $E[l_n]$, the expected value of $f_1$ (the total number of packets buffered at time $n$). If we substitute (25) into (19), we get

$$E[q_n] \leq \Sigma_{i \geq 0}\ (B+(\log_2(i/B+1))(B+1))P[l_n = f_1(i)]. \tag{26}$$

Thus,

$$E[q_n] \leq (B + 0(B+1))P[l_n = f_1(0)] + \Sigma_{i>0}\ ((B+(\log_2(i/B+1))(B+1))P[l_n = f_1(i)]). \tag{27}$$

Since $\log_2(i/B+1)$ is equal to $\log_2(i(1/B+1/i))$ and is thus equal to $(\log_2 i)+\log_2(1/B+1/i)$,

$$\begin{aligned} E[q_n] \leq \quad & (B + 0(B+1))P[l_n = f_1(0)] \\ & + \Sigma_{i>0}\ ((B+((\log_2 i)+\log_2(1/B+1/i))(B+1))P[l_n = f_1(i)]). \end{aligned} \tag{28}$$

By (16), the definition of $f_1$,

$$E[q_n] \leq \Sigma_{i \geq 0}\ ((B+(B+1)f_1(i)+(B+1))P[l_n = f_1(i)]). \tag{29}$$

Thus,

$$E[q_n] \leq (B+1)E[l_n]+2B+1. \tag{30}$$

We assume that $E[l_n]$ also exists in the limit as $n$ goes to infinity. Thus taking the limit,

$$E[\tilde{l}] \geq \frac{E[\tilde{q}]}{B+1} - \frac{2B+1}{B+1}. \tag{31}$$

Since $2IN - IN^2 > IN$, (15) and (31) imply that $E[\tilde{l}]$, the expected value of the limiting distribution for $f_l(p_n)$, $f_l$ (the total number of packets buffered at time $n$), is greater than

$$\frac{IN}{2(OUT-IN)(B+1)} - 2. \tag{32}$$

This can be used to obtain a lower bound on $E[\tilde{p}]$, the expected value of the limiting distribution for the total number of packets buffered. Since $p_n \geq 2^{(l_n)} - 1$,

$$E[\tilde{p}] \geq E[2^{\tilde{l}}] - 1 \tag{33}$$

where $\tilde{p}$ is the limiting distribution for $p_n$ and $p_n$ is the total number of packets buffered at time $n$. Since exponentiation is a convex function, (33) implies that

$$E[\tilde{p}] \geq 2^{E[\tilde{l}]} - 1. \tag{34}$$

Thus, from (32) we can conclude that $E[\tilde{p}]$, the expected value of the limiting distribution for the total number of packets buffered, is greater than

$$2^{\left(\frac{IN}{2(OUT-IN)(B+1)} - 2\right)} - 1. \tag{35}$$

This ends the discussion of the proposition. ∎

Thus, the results of the model suggest that deep tree buffering will occur in front of a slow router if the rate at which the router can accept packets is close to the rate at which packets that must go through that router are generated. If the rate at which packets are generated on each of the network inputs is $IN$ and if the rate at which the router can accept packets on that input is $OUT$ then the model suggests that the expected value of $f_l$ (the number of packets buffered in front of that input) is greater than $\frac{IN}{2(OUT-IN)(B+1)} - 2$. Similarly, the model suggests that the expected number of packets buffered in

front of that input is greater than $2^{(\frac{IN}{2(OUT-IN)(B+1)} - 2)} - 1$. In the next section, we examine a stage of routers and we examine the probability that at least one of the routers of the stage has a deep tree buffered in front of it.

### 2.2.3.3 Effect of the Last Stage

In this section, we examine the effect that congestion in routers of the last stage of an indirect n-cube has on the network's operation. As we have seen in the previous section, congestion in a router of a given stage can easily affect many routers of an earlier stage. We now consider congestion in all the routers of a given stage. We use the last stage because analysis of the last stage is somewhat easier than analysis of other stages. There is a path from each network input to each router of the last stage. Thus, congestion in any router of the last stage may affect all of the network inputs.

We use a network model to study the effect of the last stage. This model represents an indirect n-cube network with its outputs connected to perfect packet sinks. The model considers only buffering caused by congestion in routers of the last stage of the network. We use the model to study the limit that such buffering places on the throughput of an indirect n-cube network.

Rather than analyze the model directly, we choose to transform the model, analyze the transformed model, and use the results to draw conclusions about the original model.

Based on the results of the model, we conclude that the effect of the last stage of routers in a network does not place a severe constraint on the throughput of the network.

*Model of the Effect of the Last Stage*

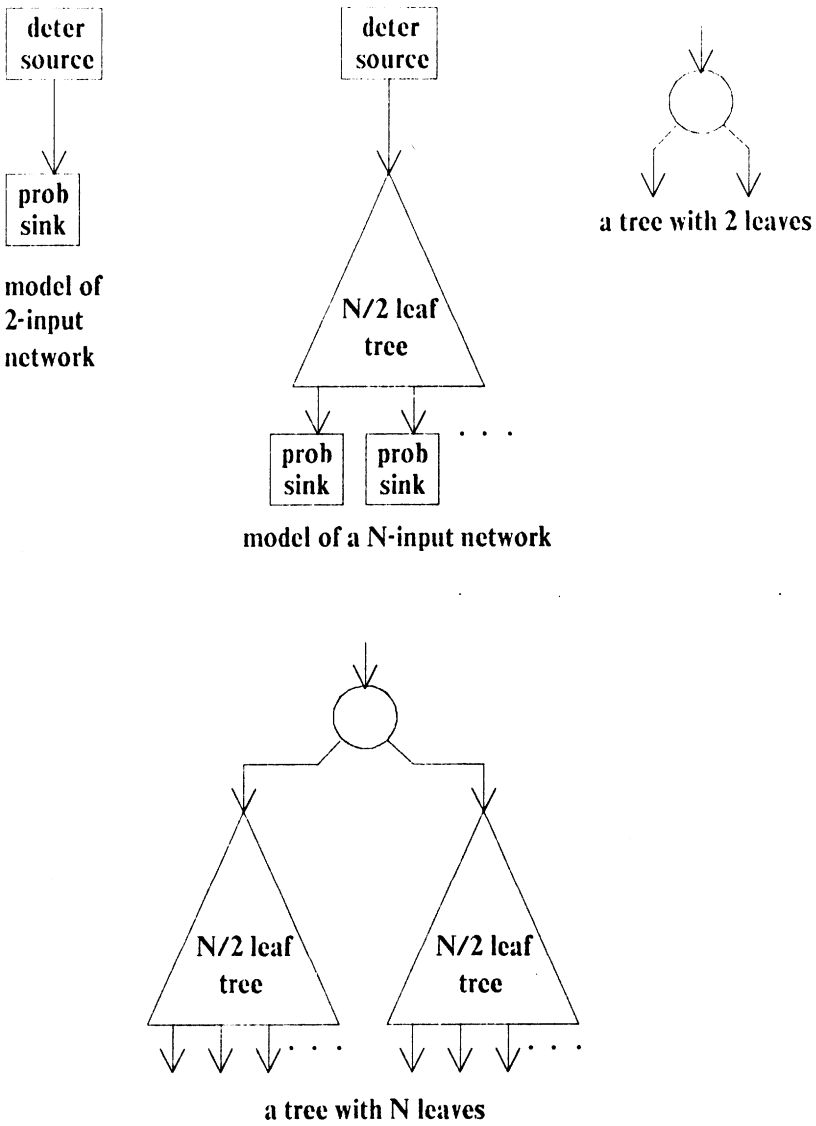
In the following paragraphs, we discuss the model; we describe the model, discuss the relationship between the model and the indirect n-cube network, discuss the characteristics of network buffering that will be studied using the model, and describe the details of the components of the model.

The model is composed of nodes, probabilistic sinks, and a deterministic source. The model is constructed in the manner shown in Figure 11. The model for a 2-input network is constructed from a deterministic source and a probabilistic sink. The model for a N-input network is constructed from a deterministic source, N/2 probabilistic sinks, and a tree of nodes with N/2 leaves. A tree with 2 leaves is one node. A tree with N leaves is constructed from two trees with N/2 leaves.

The model reflects primarily two features of an indirect n-cube network: the probabilistic input rate of routers of the last stage, and the buffering capacity between each router of the last stage and the network inputs. The probabilistic sinks of the model represent the routers of the last stage of the network. The nodes of the model represent the routers of the other stages of the network. The nodes of the model are connected in a tree structure as shown in Figure 12. If the stages of the model are numbered from the root to the probabilistic sinks and if the total number of stages is $d$ then each packet in stage $i$ of the model represents $2^{d+1-i}$ packets in stage $i$ of the network. Each node of the model has an input buffer of size $B$ where $B$ is the size of the buffers in the network. Thus, the buffering capacity of the model between a probabilistic sink and the model input represents the buffering capacity of the network between a router of the last stage and the network inputs.

We use the model to study the buffering of packets in an indirect n-cube network caused by routers of the last stage. The nodes of the model do not operate in the same manner as the routers of the network. Each node of the model evenly splits between its two outputs the flow of packets from its input. A node will instantly process a packet in its input buffer unless one of the buffers connected to its outputs

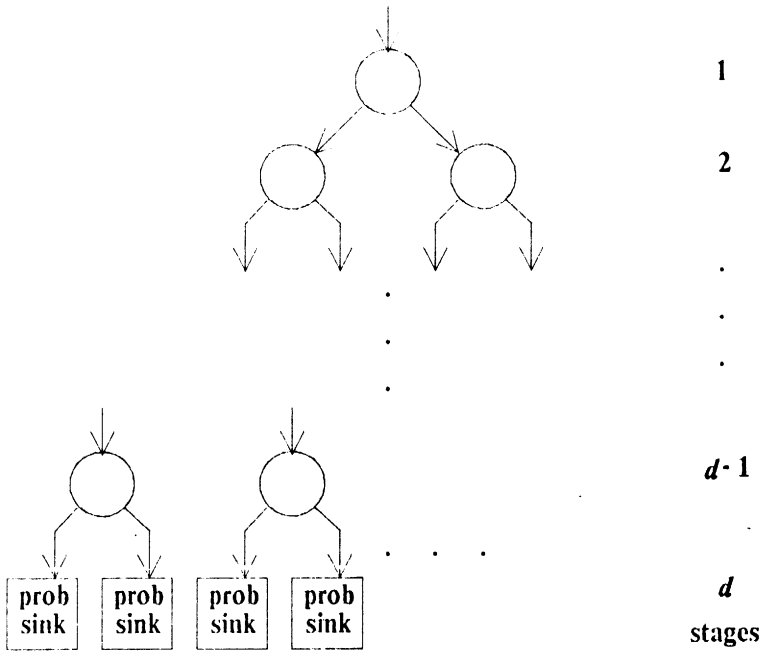
Fig. 11. Model of the Effect of the Last Stage




is full. Thus, the buffering of packets in the model can be caused only by the probabilistic sinks and represents the buffering in the network caused by the routers of the last stage.

We also use the model to examine the limit that conflict in the last stage of network routers places on network performance. The model accurately reflects the buffering capacity of the network between a

Fig. 12. Nodes of the Model of the Effect of the Last Stage




last stage router and the network inputs. Thus, the maximum input rate that can be supported in the model without source blocking is an indication of the limit that conflict in the last stage of network routers places on network performance.

The probabilistic sinks of the model are similar to the probabilistic sinks used in the previous section. Each sink contains a buffer of size $B$ where $B$ is the size of the buffers in the network. If at the beginning of a time unit the buffer is not empty then with probability .75 the sink removes one packet from the buffer. The average rate at which the sink can remove packets from its buffer corresponds to half the average rate at which a router of the last stage of the network can accept packets since each packet removed by the sink represents two packets of the network. It should be noted that the probabilistic sinks are a pessimistic model of the routers of the last stage. The probabilistic sinks of the model can fail to

accept a packet for an arbitrarily long period of time. The outers of the last stage of the network are capable of accepting at least one packet during each time unit. We use this pessimistic model because it makes the discussion simpler and because even with such a model, our analysis suggests that congestion in the last stage of routers does not place a severe constraint on the throughput of the network.

Each node of the model, in effect, evenly splits between its two outputs the flow of packets from its input. A node is assumed to operate instantly. If a node's input buffer is not empty and if both buffers connected to the outputs of the node are not full then the node removes one packet from its input buffer and places a copy of the packet in each of the two buffers connected to its outputs.

The deterministic source generates packets at a constant rate. The deterministic source generates each packet in the form of 100 subpackets. The source generates *IN* subpackets in each unit of time that it is not blocked. *IN* is a parameter. The source buffers the subpackets internally until the first unit of time in which it can output a whole packet. Thus, the source only outputs whole packets.

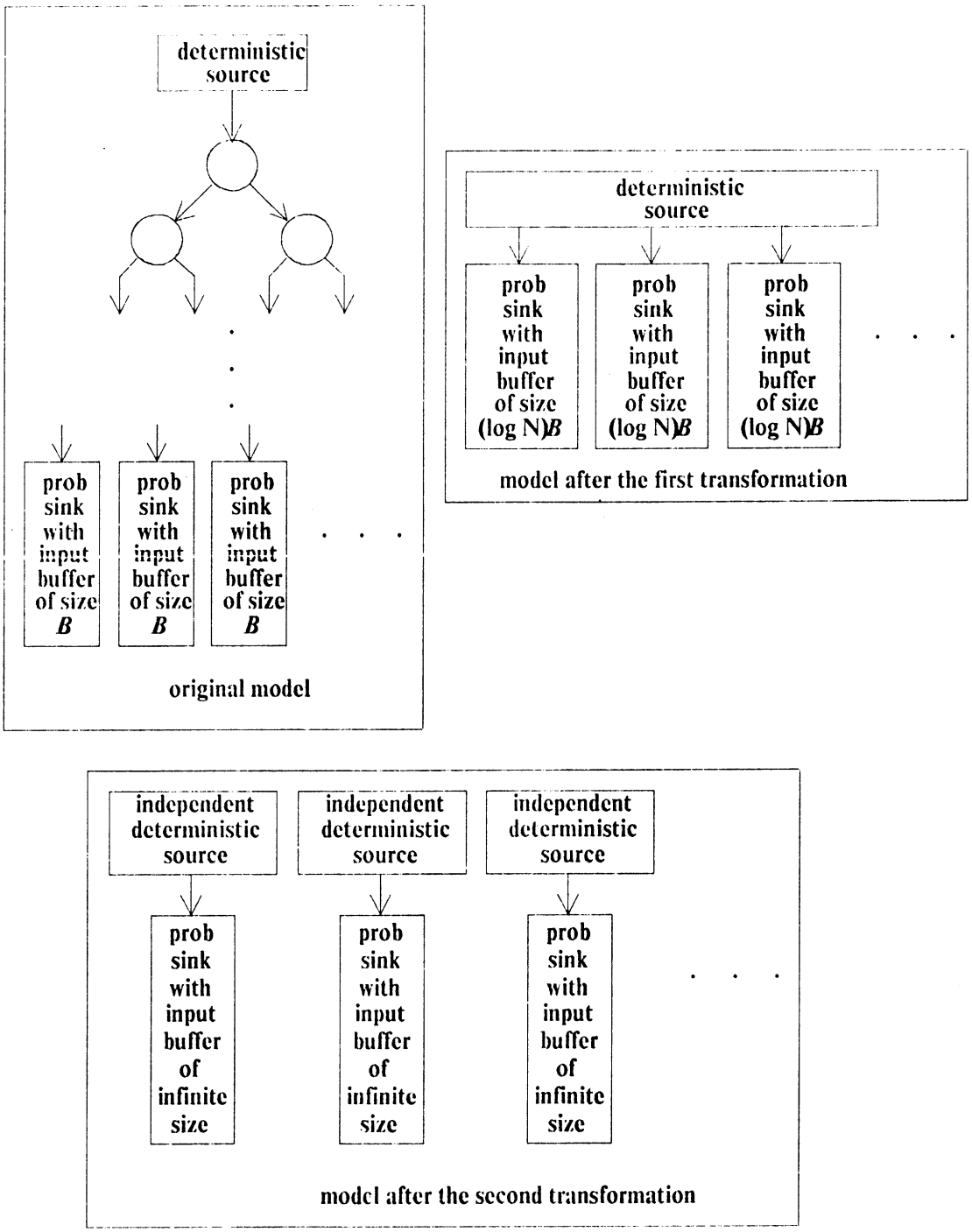
*Transformed Model*

Rather than analyze the model directly, we choose to transform the model and analyze the transformed model. We discuss the relationship between results for the transformed model and results for the original model.

The transformations that we make are shown in Figure 13.

The first transformation takes the buffers that were distributed in the tree of nodes and aggregates them at the probabilistic sinks. The probabilistic sinks in the transformed model contain buffers of size $(\log_2 N)B$ where N/2 is the number of probabilistic sinks and $B$ is the size of the buffers in the original nodes. The deterministic source of the transformed model is directly connected to each of the probabilistic sinks. The deterministic source is blocked if any of the buffers in the probabilistic sinks are

Fig. 13. Transformations on the Model of the Effect of the Last Stage



full. In each time unit that it is not blocked, the deterministic source simultaneously generates $IN$ subpackets for each probabilistic sink. When the source has formed whole packets, it simultaneously outputs one packet to each of the probabilistic sinks.

The buffers in the transformed model have a greater independence than the buffers in the original model. In the transformed model, the buffering of packets caused by a particular probabilistic sink can affect the flow of packets into another probabilistic sink only by blocking the deterministic source. In the original model, the buffering of packets caused by a particular sink can affect other sinks by blocking nodes of the tree.

It can be shown that the maximum input rate of the transformed model is at least as great as the maximum input rate of the original model. Thus, limits on the performance of the transformed model also apply to the original model. An upper bound on the performance of the transformed model implies an upper bound on the performance of the original model.

The second transformation turns the buffer of size $(\log_2 N)B$ in each probabilistic sink into an infinite buffer in which we will look for occupancy of $(\log_2 N)B$ packets, and the second transformation also turns the single deterministic source into a large number of deterministic sources with one associated with each probabilistic sink. We assume that each of the new deterministic sources operates in a manner similar to that of the original deterministic source. Each source generates $IN$ subpackets in each unit of time. The source buffers subpackets until it has produced more than 100 subpackets. The source outputs each group of 100 subpackets as a whole packet to its associated probabilistic sink. However, we assume that the state of each source-sink pair is independent of the state of the other source-sink pairs. We also assume that the sources are never blocked.

In the following paragraphs, we refer to the model after the first transformation as the model of the first transformation, and we refer to the model after both transformations as the divided model since the

model is divided into independent source-sink pairs.

Our analysis of the constraints on the input rate of the divided model also applies to the input rate of the original model. We examine the divided model to determine input rates that imply a high probability that at a randomly selected point in time at least one of the buffers contains at least $(\log_2 N)B$ packets. It seems reasonable to assume that such input rates can not be supported in the model of the first transformation since that model is similar to the divided model but has buffers of size $(\log_2 N)B$. Thus, it seems reasonable to assume that such input rates can not be supported in the original model since the maximum input rate of the original model is less than the maximum input rate of the model of the first transformation.

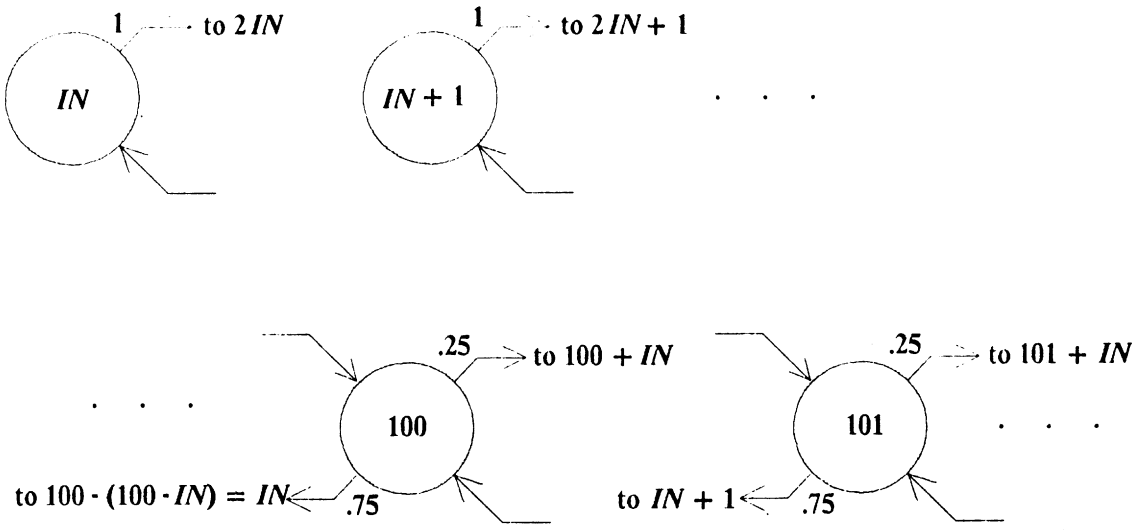
However, as we shall see our analysis of the divided model and simulation of the original model indicate that both models can support input rates close to the maximum rate at which a probabilistic sink can accept packets. Thus, our models suggest that conflict in a given stage of routers does not place a severe constraint on the overall throughput of the network.

*Probability of Buffering in the Divided Model*

In this section, we examine the Markov chain for the state of one of the source-sink pairs of the divided model and use the results to bound the probability in the divided model that at a randomly selected point in time the buffers in one or more of the sinks have at least $(\log_2 N)B$ packets.

The Markov chain $MC_D$ for the state of one of the source-sink pairs of the divided model is shown in Figure 14. The state is equal to the number of subpackets being stored at the source plus 100 times the number of packets in the buffer of the sink since each packet is composed of 100 subpackets. For the purpose of discussion, we use the notation $p_D(i,j)$ to refer to the probability of a transition to state $j$ given that the chain is in state $i$. A stationary distribution, $P_{DS}$, for the chain is a distribution such that $P_{DS}(i)$, the probability that the chain is in state $i$, is given by

Fig. 14. $MC_D$




---

$$P_{DS}(i) = \Sigma_{j \geq IN} \; p_D(j,i) P_{DS}(j). \tag{36}$$

We refer to (36) as the equilibrium equation for $P_{DS}(i)$.

As is shown in the following subsection, any stationary distribution, $P_{DS}$, for $MC_D$ must be such that

$$(1-\alpha^{j-100+IN})(IN/75) \geq \Sigma_{i=100 \text{ to } j} P_{DS}(i) \qquad \text{for } j \geq 100 \tag{37}$$

and

$$\Sigma_{i=100 \text{ to } j+IN-1} P_{DS}(i) \geq (1-\alpha^{j-99})(IN/75) \qquad \text{for } j \geq 100 \tag{38}$$

where

$$0<\alpha<1$$

and $\alpha$ is the root in this range of the equation

$$\alpha^{IN} = .25 + .75\alpha^{100}.$$

For the purpose of discussion, we define some notation. We use the notation $P_{gb}$ to represent the probability in the divided model that at a randomly selected point in time a given buffer has at least $(\log_2 N)B$ packets. We use the notation $P_b$ to represent the probability that at a randomly selected point in time one or more buffers have at least $(\log_2 N)B$ packets.

The relations, (37) and (38), for one buffer can be used to obtain results for the overall divided model.

**Proposition.**

$$1\text{-}e^{(\frac{-a\,IN}{150})} < P_b < 1\text{-}e^{(-(\frac{a\,IN}{150})2^{2/B} - \frac{a^2 2^{4/B}}{N - a\,2^{2/B}})}, \tag{39}$$

where $P_b$ is the probability that at a randomly selected point in time one or more buffers have at least $(\log_2 N)B$ packets, and $a$ is such that

$$\alpha = 2^{(\frac{-1}{100B} + \frac{(\log_2 a)}{100(\log_2 N)B})}. \tag{40}$$

*Proof.* We use (37) and (38) to show the desired relations (39). We express $P_b$, the probability that at a randomly selected point in time one or more buffers have at least $(\log_2 N)B$ packets, in terms of $P_{gb}$, the probability that at a randomly selected point in time a given buffer has at least $(\log_2 N)B$ packets. We bound $P_{gb}$ in terms of a stationary distribution for the Markov chain $MC_D$. We then use (37) and (38) to get the desired bounds, (39), on $P_b$.

Since each connected source and sink of the model are independent of the other sources and sinks of the model and since there are N/2 sinks,

$$P_b = 1\text{-}(1\text{-}P_{gb})^{(N/2)}. \tag{41}$$

$P_{gb}$, the probability that at a randomly selected point in time a given buffer has at least $(\log_2 N)B$ packets, can be expressed in terms of $P_{DS}$ where $P_{DS}$ is some stationary distribution for $MC_D$ such

that the probability that a buffer is in state $i$ at a randomly selected point in time is equal to $P_{DS}(i)$. The choice of the particular stationary distribution may depend on the initial state of the buffer. $P_{gb}$ is equal to $\Sigma_{i \geq 100(\log_2 N)B} P_{DS}(i)$ which is equal to

$$1 - (\Sigma_{i = IN \text{ to } 99} P_{DS}(i)) - (\Sigma_{i = 100 \text{ to } 100(\log_2 N)B-1} P_{DS}(i)). \quad (42)$$

$P_{gb}$ is also equal to

$$(\Sigma_{i = 100 \text{ to } \infty} P_{DS}(i)) - (\Sigma_{i = 100 \text{ to } 100(\log_2 N)B-1} P_{DS}(i)). \quad (43)$$

Using (37), (38), (42), and (43) we can bound $P_{gb}$, the probability that at a randomly selected point in time a given buffer has at least $(\log_2 N)B$ packets. Since the long term average rate of packets out of a buffer of the divided model must equal the rate in,

$$.75(\Sigma_{i \geq 100} P_{DS}(i)) = IN/100 \quad (44)$$

and

$$(\Sigma_{i = IN \text{ to } 99} P_{DS}(i)) = 1 - IN/75. \quad (45)$$

Thus,

$$P_{gb} < IN/75 - (\Sigma_{i = 100 \text{ to } 100(\log_2 N)B-1} P_{DS}(i)). \quad (46)$$

Given (38),

$$P_{gb} < (IN/75)\alpha^{(100(\log_2 N)B - IN - 99)}. \quad (47)$$

(37), (43), and (44), imply that

$$P_{gb} \geq (IN/75) - (IN/75)(1-\alpha^{100(\log_2 N)B - 101 + IN}). \quad (48)$$

Thus, since $\alpha < 1$ and $IN < 101$ we can conclude from (47) and (48) that

$$(IN/75)\alpha^{(100(\log_2 N)B - IN - 99)} > P_{gb} > (IN/75)\alpha^{100(\log_2 N)B}. \quad (49)$$

We introduce some notation that makes the discussion below simpler than it would otherwise be. In particular, we define $a$ to be such that

$$\alpha = 2^{\left(\frac{-1}{100B} + \frac{(\log_2 a)}{100(\log_2 N)B}\right)}. \quad (50)$$

It should be noted that $a$ is a function of $IN$, the source input rate, just as $\alpha$ is a function of $IN$.

Given (49) and (50), $P_{gb}$, the probability that at a randomly selected point in time a given buffer has at least $(\log_2 N)B$ packets, is less than

$$(IN/75)2^{(-(\log_2 N)+\frac{IN+99}{100B}+(\log_2 a)-\frac{(IN+99)(\log_2 a)}{100(\log_2 N)B})} < (IN/75)(a/N)2^{2/B} \tag{51}$$

and

$$P_{gb} > (IN/75)(a/N). \tag{52}$$

From (51) and (52), we deduce bounds on $P_b$, the probability that at a randomly selected point in time one or more buffers have at least $(\log_2 N)B$ packets. Since $0 < P_{gb} < 1$, $(1-P_{gb})^{N/2} < e^{-P_{gb}(N/2)}$. Thus, with $a$ defined as above $P_b$ is greater than

$$1-e^{(\frac{-a\,IN}{150})}. \tag{53}$$

Since $0 < P_{gb} < 1$,

$$(1-P_{gb})^{N/2} = e^{(-P_{gb}-\Sigma_{i=2 \text{ to } \infty}(1/i)(P_{gb})^i)(N/2)} > e^{(-P_{gb}-\frac{P_{gb}^2}{1-P_{gb}})(N/2)}. \tag{54}$$

Thus, with $a$ defined as above $P_b$, the probability that at a randomly selected point in time one or more buffers have at least $(\log_2 N)B$ packets, is less than

$$1-e^{(-(\frac{a\,IN}{150})2^{2/B}-\frac{(IN/75)^2(a/N)^2(N/2)2^{4/B}}{1-(IN/75)(a/N)2^{2/B}})} \tag{55}$$

and

$$P_b < 1-e^{(-(\frac{a\,IN}{150})2^{2/B}-\frac{a^2 2^{4/B}}{N-a\,2^{2/B}})}. \tag{56}$$

This ends the discussion of the proposition. ∎

*Implications of the Models on Network Throughput*

The lower bound, (53), on $P_b$ for the divided model suggests a limit on the maximum input rate of the original model. In the divided model, $P_b$ is the probability that at a randomly selected point in time at least one of the buffers has at least $(\log_2 N)B$ packets. As was discussed earlier, it seems reasonable to assume that input rates that imply a high $P_b$ in the divided model can not be supported in the original model. However, this does not place a strong constraint on the input rate of the original model. From relation (53), we know that if we want $P_b$ in the divided model to be less than some value then we must choose $IN$ such that $1 - e^{(\frac{-a\,IN}{150})}$ is less than that value. As N goes to infinity, if $a$ goes to zero and $IN \geq 1$ then $1 - e^{(\frac{-a\,IN}{150})}$ goes to zero. As N goes to infinity, if $a$ goes to infinity and $IN \geq 1$ then $1 - e^{(\frac{-a\,IN}{150})}$ goes to one. Thus to find an upper bound on $IN$ in the divided model for a particular $P_b$ in the limit as N goes to infinity, we assume that $a$ approaches a constant independent of N. In such a case, equation (50) implies that $\alpha$ approaches $2^{-1/(100B)}$. The corresponding value of $IN$ can be deduced from the equation, $\alpha^{IN} = .25 + .75\alpha^{100}$. For example, if $B$ is equal to five then $\alpha$ must be less than $2^{-1/500}$. $2^{-1/500}$ is approximately equal to .998615. This implies an upper bound on $IN$ of 73. For comparison, the upper bound on $IN$ implied for $B$ equal to one is 67, the bound for $B$ equal to two is 71, and the bound for $B$ equal to ten is 74. Obviously, these upper bounds are close to the upper bound of 75 placed by the fact that, as was discussed in the description of the original model, a probabilistic sink of this section can only accept packets at an average rate of .75 packets per unit time.

In fact, it seems that if the buffers of the original model have at least modest size then the average input rate of the model can be close to the average rate at which packets can be removed from the buffer of a probabilistic sink. We have simulated the original model for several buffer sizes. The model was simulated with both 512 probabilistic sinks and with 1024 probabilistic sinks. A simulation run was made for each combination of model width and buffer size. In each run, the deterministic source was capable of generating a packet in each time unit that the source was not blocked. Thus, the rate at which packets

were generated by the source was determined by blocking. The buffers in the model were full at the beginning of each run. For each run, we measured the number of packets that were generated by the deterministic source during the run. The results are shown in Table I. For each combination of model width and buffer size, the average rate at which packets were generated is listed. As can be seen from the table, the results for 512 sinks are close to the results for 1024 sinks. For both sets of results, the average input rates are not far from .75 packets per unit time for even modest buffer sizes.

Thus, our study of the original model suggests that slow routers of the last stage in an indirect n-cube network do not place a severe constraint on the throughput of the network.

*The Stationary Distribution for the Divided Model*

We now return to the detailed analysis of the divided model in order to show the bounds, (37) and (38), on any stationary distribution for the divided model.

Since direct analysis of $MC_D$, the Markov chain for the divided model, seems difficult, we indirectly analyze it by comparing it to two simpler chains. The first of these chains eliminates the first 100 states of the original chain since we are primarily interested in the later states of the original chain. The second chain is easier to analyze than the first chain and gives information about the first chain and the original chain. We introduce each of these chains and show the the relation between the stationary

---

**Table I. Simulation of the Model of the Effect of the Last Stage**

| $B$ | 1 | 2 | 3 | 5 | 10 |
|---|---|---|---|---|---|
| av. in for 512 Probabilistic Sinks | 50.8 | 64.0 | 67.6 | 70.6 | 72.6 |
| av. in for 1024 Probabilistic Sinks | 50.7 | 63.5 | 67.6 | 70.6 | 72.8 |

distributions of each and the stationary distributions of $MC_D$, the Markov chain for the divided model. We obtain the stationary distribution for the second chain and use it to bound the stationary distribution for $MC_D$.

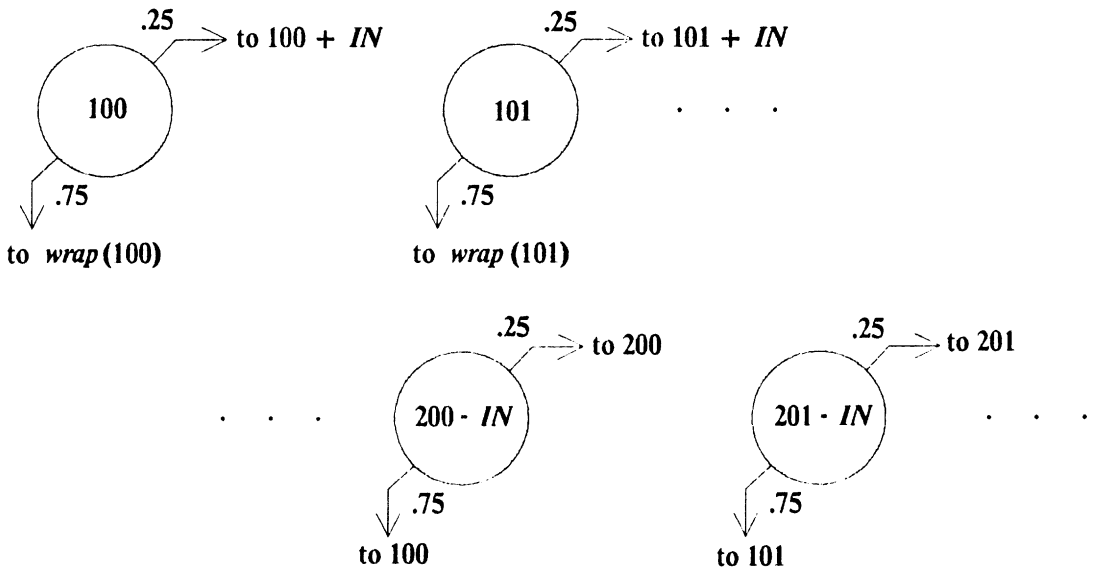
The Markov chain $MC_{DI}$ shown in Figure 15 is very similar to the chain for the divided model with the exception that the states less than 100 have been removed. For $100 \leq i \leq 199\text{-}IN$, $wrap(i)$ is equal to the smallest $j$ such that for some integer $k$, $k\,IN + i - 100 + IN = j$ and $j \geq 100$. $wrap(i)$ is the first state greater than or equal to 100 that would be reached after a transition down from state $i$. We use the notation $p_{DI}(i,j)$ to refer to the probability of a transition to state $j$ given that the chain is in state $i$. A stationary distribution, $P_{DIS}$, for the chain is a distribution such that $P_{DIS}(i)$, the probability that the chain is in state $i$, is given by the equation

$$P_{DIS}(i) = \Sigma_{j \geq 100}\ p_{DI}(j,i)P_{DIS}(j). \tag{57}$$

We refer to (57) as the equilibrium equation for $P_{DIS}(i)$. The equilibrium equations of $MC_{DI}$, are

---

Fig. 15. $MC_{DI}$



very similar to the equilibrium equations of the divided model.

This similarity can be seen more clearly if $P_{DS}(IN)$ through $P_{DS}(99)$, the stationary probabilities for states less than 100 in the divided model, are eliminated from the equilibrium equations of the divided model. We can eliminate $P_{DS}(IN)$ by using the equation for $P_{DS}(IN)$ to substitute for $P_{DS}(IN)$ in the remaining equations. This process can be continued for $P_{DS}(IN+1)$ through $P_{DS}(99)$. The resulting equation for $P_{DS}(i)$ for each $i \geq 100$ corresponds directly to the equilibrium equation for $P_{DIS}(i)$. In particular, if we take the equation for $P_{DS}(i)$ and map $P_{DS}(j)$ to $P_{DIS}(j)$ for each $j \geq 100$, we obtain an equation which is identical to the equilibrium equation for $P_{DIS}(i)$. From this we can conclude that for any stationary distribution $P_{DS}$ for the Markov chain $MC_D$, the chain for the divided model, there exists a stationary distribution $P_{DIS}$ for the Markov chain $MC_{DI}$ such that for some constant $c$ and for all $i \geq 100$, $P_{DIS}(i) = cP_{DS}(i)$. Since $\Sigma_{i \geq 100} P_{DIS}(i) = 1$, we can conclude that $c = \frac{1}{\Sigma_{i \geq 100} P_{DS}(i)}$ and that

$$P_{DIS}(i) = \frac{1}{\Sigma_{j \geq 100} P_{DS}(j)} P_{DS}(i). \tag{58}$$

The chain $MC_{D2}$ shown in Figure 16 is related to $MC_{DI}$ and is therefore also related to $MC_D$, the chain for the divided model. We examine $MC_{D2}$ because it is related to $MC_D$ and because, as we shall see later, the stationary distribution for $MC_{D2}$ is easy to determine since there is a simple geometric relationship among the probabilities of the various states. We use the notation $p_{D2}(i,j)$ to refer to the probability of a transition to state $j$ given that the chain is in state $i$. If $\alpha$ is such that $\alpha^{IN} = .25 + .75\alpha^{100}$ then

$$p_{D2}(i,j) = .75\alpha^{(j-100)}\left(\frac{1-\alpha}{1-\alpha^{IN}}\right) \qquad \text{for } 100 \leq i \leq 199-IN \text{ and } 100 \leq j \leq 99+IN,$$

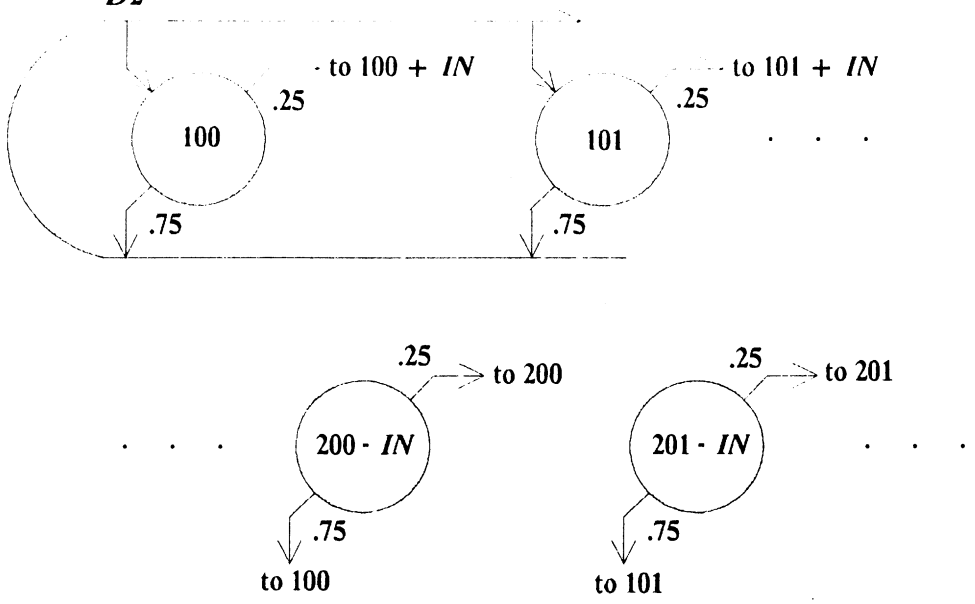
$$p_{D2}(i,i+IN) = .25 \qquad \text{for } i \geq 100,$$

$$p_{D2}(i,i-100+IN) = .75 \qquad \text{for } i \geq 200-IN,$$

and

$$p_{D2}(i,j) = 0 \qquad \text{otherwise.}$$

Fig. 16. $MC_{D2}$




---

(59)

The stationary distribution $P_{D2S}$ for the Markov chain $MC_{D2}$ is the distribution such that $P_{D2S}(i)$, the probability that the chain is in state $i$, is given by the equation

$$P_{D2S}(i) = \Sigma_{j \geq 100}\ p_{D2}(j,i)P_{D2S}(j). \qquad (60)$$

The transitions of the chain $MC_{D2}$ are similar to those of the chain $MC_{D1}$. Many of the possible transitions of $MC_{D2}$ have the same probabilities as the corresponding transitions of $MC_{D1}$. In particular,

$$p_{D2}(i, i+IN) = p_{D1}(i, i+IN) = .25 \qquad \text{for } i \geq 100$$

and

$$p_{D2}(i, i\text{-}100+IN) = p_{D1}(i, i\text{-}100+IN) = .75 \qquad \text{for } i \geq 200\text{-}IN\ .$$

(61)

The transitions of $MC_{D2}$ from $i$ to $j$ where $100 \leq i \leq 199\text{-}IN$ and $100 \leq j \leq 99+IN$ differ from the

corresponding transitions of $MC_{D1}$. But, for any $i$ such that $100 \leq i \leq 199\text{-}IN$,

$$\Sigma_{j=100 \text{ to } 99+IN} \; p_{D2}(i,j) = \Sigma_{j=100 \text{ to } 99+IN} \; p_{D1}(i,j) = (.75). \qquad (62)$$

We show that any stationary distribution $P_{D1S}$ for the chain $MC_{D1}$ is related to the stationary distribution $P_{D2S}$ for the chain $MC_{D2}$.

**Proposition.**

$$\Sigma_{i=100 \text{ to } j+IN\text{-}1} \; P_{D2S}(i) \geq \Sigma_{i=100 \text{ to } j} \; P_{D1S}(i) \qquad (63)$$

and

$$\Sigma_{i=100 \text{ to } j+IN\text{-}1} \; P_{D1S}(i) \geq \Sigma_{i=100 \text{ to } j} \; P_{D2S}(i). \qquad (64)$$

*Proof.* This can be seen by comparing the operation of the two chains during a long period of time. For the purpose of discussion, we define $P_{D1}(t, i)$ to be the probability that chain $MC_{D1}$ is in state $i$ at time $t$. We define $P_{D2}(t, i)$ to be the probability that chain $MC_{D2}$ is in state $i$ at time $t$. We assume that for all $i \geq 100$, $P_{D1}(1, i)$ is equal to $P_{D2}(1, i)$ and is also equal to $P_{D1S}(i)$ where $P_{D1S}$ is the given stationary distribution for the chain $MC_{D1}$.

We show a relationship between $P_{D1}$ and $P_{D2}$ that exists for all $t$. Using this, we show a similar relationship between $P_{D1S}$, the given stationary distribution for the chain $MC_{D1}$, and $P_{D2S}$, the stationary distribution for the chain $MC_{D2}$.

In particular, we show by induction on $t$ that for all $t > 0$ and $j \geq 100$

$$\Sigma_{i=100 \text{ to } j+IN\text{-}1} \; P_{D2}(t, i) \geq \Sigma_{i=100 \text{ to } j} \; P_{D1}(t, i) \qquad (65)$$

and

$$\Sigma_{i=100 \text{ to } j+IN\text{-}1} \; P_{D1}(t, i) \geq \Sigma_{i=100 \text{ to } j} \; P_{D2}(t, i). \qquad (66)$$

Clearly, these relations hold for $t = 1$. We show the case for $t > 1$ by considering the transitions of each chain. From the transitions of the chain $MC_{D1}$, it can be shown that for $t > 1$ and $j \geq 100$ that

$$\Sigma_{i=100 \text{ to } j} P_{DI}(t, i)$$
$$\leq .75\Sigma_{i=100 \text{ to } 199\text{-}IN} P_{DI}(t\text{-}1, i)$$
$$+ .25\Sigma_{i=100 \text{ to } j\text{-}IN} P_{DI}(t\text{-}1, i)$$
$$+ .75\Sigma_{i=200\text{-}IN \text{ to } j+100\text{-}IN} P_{DI}(t\text{-}1, i) \quad (67)$$

or

$$\Sigma_{i=100 \text{ to } j} P_{DI}(t, i)$$
$$\leq .75\Sigma_{i=100 \text{ to } j+100\text{-}IN} P_{DI}(t\text{-}1, i)$$
$$+ .25\Sigma_{i=100 \text{ to } j\text{-}IN} P_{DI}(t\text{-}1, i), \quad (68)$$

and that

$$\Sigma_{i=100 \text{ to } j+IN\text{-}1} P_{DI}(t, i)$$
$$= .75\Sigma_{i=100 \text{ to } 199\text{-}IN} P_{DI}(t\text{-}1, i)$$
$$+ .25\Sigma_{i=100 \text{ to } j\text{-}1} P_{DI}(t\text{-}1, i)$$
$$+ .75\Sigma_{i=200\text{-}IN \text{ to } j+99} P_{DI}(t\text{-}1, i) \quad (69)$$

or

$$\Sigma_{i=100 \text{ to } j+IN\text{-}1} P_{DI}(t, i)$$
$$= .75\Sigma_{i=100 \text{ to } j+99} P_{DI}(t\text{-}1, i)$$
$$+ .25\Sigma_{i=100 \text{ to } j\text{-}1} P_{DI}(t\text{-}1, i). \quad (70)$$

Similarly, from the transitions of the chain $MC_{D2}$ it can be shown for $t>0$ and $j \geq 100$ that

$$\Sigma_{i=100 \text{ to } j} P_{D2}(t, i)$$
$$\leq .75\Sigma_{i=100 \text{ to } j+100\text{-}IN} P_{D2}(t\text{-}1, i)$$
$$+ .25\Sigma_{i=100 \text{ to } j\text{-}IN} P_{D2}(t\text{-}1, i), \quad (71)$$

and that

$$\Sigma_{i=100 \text{ to } j+IN\text{-}1} P_{D2}(t, i)$$
$$= .75\Sigma_{i=100 \text{ to } j+99} P_{D2}(t\text{-}1, i)$$
$$+ .25\Sigma_{i=100 \text{ to } j\text{-}1} P_{D2}(t\text{-}1, i). \quad (72)$$

Thus, given the hypothesis of induction we can conclude that the desired relations, (65) and (66), hold for

$t>1$. As a result, we can conclude that the desired relations hold for all $t>0$. In other words, for $t>0$ and $j \geq 100$

$$\Sigma_{i=100 \text{ to } j+IN-1} P_{D2}(t, i) \geq \Sigma_{i=100 \text{ to } j} P_{D1}(t, i) \tag{73}$$

and

$$\Sigma_{i=100 \text{ to } j+IN-1} P_{D1}(t, i) \geq \Sigma_{i=100 \text{ to } j} P_{D2}(t, i). \tag{74}$$

Since $P_{D1}(t, i)$ equals $P_{D1S}(i)$ for $t>0$ where $P_{D1S}$ is the stationary distribution chosen above for the chain $MC_{D1}$ and since in the limit as $t$ goes to infinity $P_{D2}(t, i)$ goes to $P_{D2S}(i)$ where $P_{D2S}$ is the stationary distribution for the chain $MC_{D2}$, we can conclude that

$$\Sigma_{i=100 \text{ to } j+IN-1} P_{D2S}(i) \geq \Sigma_{i=100 \text{ to } j} P_{D1S}(i) \tag{75}$$

and

$$\Sigma_{i=100 \text{ to } j+IN-1} P_{D1S}(i) \geq \Sigma_{i=100 \text{ to } j} P_{D2S}(i). \tag{76}$$

This ends the discussion of the proposition. ∎

We can relate any stationary distribution $P_{DS}$ for $MC_D$, the chain for the divided model, to the stationary distribution $P_{D2S}$ for the chain $MC_{D2}$.

## Proposition.

$$\Sigma_{i=100 \text{ to } j+IN-1} (IN/75)P_{D2S}(i) \geq \Sigma_{i=100 \text{ to } j} P_{DS}(i) \qquad \text{for } j \geq 100 \tag{77}$$

and

$$\Sigma_{i=100 \text{ to } j+IN-1} P_{DS}(i) \geq \Sigma_{i=100 \text{ to } j} (IN/75)P_{D2S}(i) \qquad \text{for } j \geq 100. \tag{78}$$

*Proof.* From (58), we know that for any stationary distribution $P_{DS}$ for $MC_D$ there exists a stationary distribution $P_{D1S}$ for the chain $MC_{D1}$ such that for $i \geq 100$

$$P_{D1S}(i) = \frac{1}{\Sigma_{j \geq 100} P_{DS}(j)} P_{DS}(i). \tag{79}$$

Since the long term average rate of packets out of a buffer of the divided model must equal the rate in, $.75(\Sigma_{i \geq 100} P_{DS}(i)) = IN/100$. Thus,

$$P_{DS}(i) = (IN/75)P_{DIS}(i) \qquad \text{for } i \geq 100. \tag{80}$$

Given the relationship between any stationary distribution for $MC_{D1}$ and the stationary distribution for $MC_{D2}$, we can conclude that

$$\Sigma_{i=100 \text{ to } j+IN-1}\ (IN/75)P_{D2S}(i) \geq \Sigma_{i=100 \text{ to } j}\ P_{DS}(i) \qquad \text{for } j \geq 100 \tag{81}$$

and

$$\Sigma_{i=100 \text{ to } j+IN-1}\ P_{DS}(i) \geq \Sigma_{i=100 \text{ to } j}\ (IN/75)P_{D2S}(i) \qquad \text{for } j \geq 100. \tag{82}$$

This ends the discussion of the proposition. ∎

The stationary distribution for the chain $MC_{D2}$ can be easily obtained. The equilibrium equations for $MC_{D2}$ are

$$P_{D2S}(i) = .75P_{D2S}(i+100-IN) + .75\alpha^{(i-100)}\left(\frac{1-\alpha}{1-\alpha^{IN}}\right)\left(\Sigma_{j=100 \text{ to } 199-IN}\ P_{D2S}(j)\right)$$
$$\text{for } 100 \leq i \leq 99+IN$$

and

$$P_{D2S}(i) = .75P_{D2S}(i+100-IN) + .25P_{D2S}(i-IN) \qquad \text{for } i \geq 100+IN \tag{83}$$

where

$$0<\alpha<1$$

and $\alpha$ is the root in this range of the equation

$$\alpha^{IN} = .25 + .75\alpha^{100}.$$

From the equilibrium equations, it can be shown that

$$P_{D2S}(i) = \alpha^{i-100}P_{D2S}(100) \qquad \text{for } i>100. \tag{84}$$

Since $\Sigma_{i \geq 100}\ P_{D2S}(i) = 1$, we can conclude that

$$P_{D2S}(100) = (1-\alpha). \tag{85}$$

Thus,

$$P_{D2S}(i) = \alpha^{i-100}(1-\alpha) \quad \text{for } i \geq 100. \tag{86}$$

From (86), the solution for the stationary distribution $P_{D2S}$ for the chain $MC_{D2}$, and (81) and (82), the relations between $P_{D2S}$ and any stationary distribution $P_{DS}$ for the chain of the divided model, we can conclude that

$$(1-\alpha^{j-100+IN})(IN/75) \geq \Sigma_{i=100 \text{ to } j} P_{DS}(i) \quad \text{for } j \geq 100 \tag{87}$$

and

$$\Sigma_{i=100 \text{ to } j+IN-1} P_{DS}(i) \geq (1-\alpha^{j-99})(IN/75) \quad \text{for } j \geq 100 \tag{88}$$

where

$$0<\alpha<1$$

and

$$\alpha^{IN} = .25 + .75\alpha^{100}.$$

### 2.2.3.4 Interaction of Stages

In the following paragraphs, we examine the interaction of routers of different stages. In the previous paragraphs, we examined the effect of conflict at one router when conflict at all other routers was ignored, and we examined the effect of conflict in a given stage of routers when conflict in all other stages was ignored. Now we consider the effect of the interaction of routers in various stages of the network.

The discussion has two parts.

In the first part, we consider all the routers along a given path through the network. The diffusion of packet flow that occurs along such a path due to the interaction of routers of the path seems to be one of the primary factors constraining the overall network throughput. We examine the interaction of routers along a path by ignoring conflict at routers that are not on the path. As we shall see, the interaction of routers along an infinitely long path still allows a nonzero flow into the path. Since this type of interaction between stages represents the strongest constraint on overall network throughput that

we find, our study of such interaction suggests that the normalized throughput of the indirect n-cube network approaches a nonzero asymptote as the size of the network goes to infinity.

In the second part of the discussion, we consider the interaction of routers in a tree of the network. The interaction among such routers can be quite complex. We examine one particular type of interaction. While this interaction does not seem to have an important effect on the overall throughput of the network, it may cause a few of the routers connected to the network inputs to be slow for a long period of time.

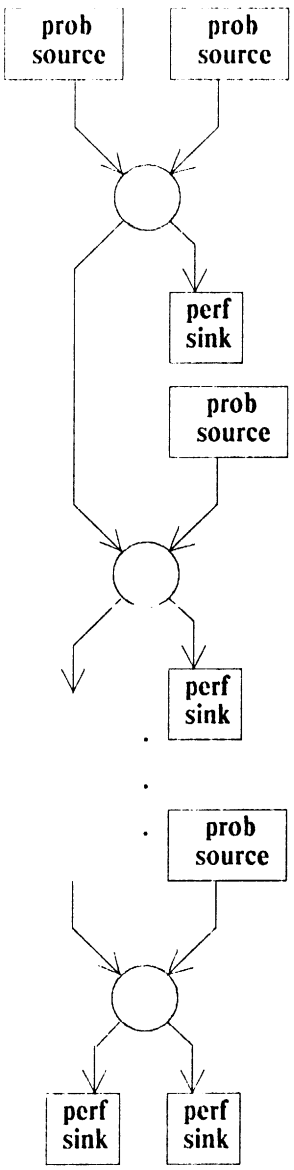
*Interaction of Routers Along a Network Path*

We study the flow of packets along a typical path of the indirect n-cube network using the model shown in Figure 17. The model represents a path through the network. The model allows us to study the effect of conflict at routers along the path in the absence of conflict at other routers. We first describe the model, then we analyze it using simulation.

*Model of a Network Path*

The model reflects the interaction of the routers along a path of the indirect n-cube network. The model ignores the interaction in the network between a router on the path and any router not on the path. The model contains a sequence of 2-input 2-output nodes. The nodes of the model represent the routers along the network path. As shown in Figure 17, the first output of each node of the model, except the last node, is connected to the first input of the next node. The second output of each node is connected to a perfect sink. The second input of each node is connected to a probabilistic source. The first input of the first node is connected to a probabilistic source. The first output of the last node is connected to a perfect sink. The connections of the second input and second output of each node represent the connections of the corresponding router of the network to routers not on the path. The probabilistic source connected to

Fig. 17. Model of a Network Path




the second input of the node provides a steady flow of packets into the node. The rate of flow can be adjusted to equal the average flow into the corresponding input of the corresponding router. The perfect sink connected to the second output can not block. Thus, congestion in the model can be caused only by

conflict in the nodes and represents the congestion along the network path due to conflict in the routers of the path.

Each node of the model has a buffer on each of its inputs. The buffer on the first input has size $B$. The size of the buffer on the second input is several times $B$. The long buffer of the second input ensures that short term variations of the node do not affect the probabilistic source connected to the input.

The operation of each node of the model is similar to the operation of a router in the indirect n-cube network. Each packet entering the node is assigned a one bit tag which determines its route through the node. The tag is randomly selected with zero and one being equally likely. If the tag is zero, the packet must leave on the first output of the node. If the tag is one, the packet must leave on the second output of the node. In each unit of time, the node attempts to transfer a packet from each of its input buffers. For each input buffer, the node attempts to transfer the first packet, the packet that entered the buffer first, to the output that corresponds to its tag. The packet is transferred if the buffer connected to the desired output is not full, and if there is no conflict from the other input buffer or if the packet wins the arbitration of the conflict. In the case of conflict, the node randomly selects a packet to transfer. The two possible choices are equally likely. The node will transfer at most one packet from each of the input buffers in a unit of time.

The probabilistic sources and perfect sinks used in this model are similar to the corresponding devices used in the previous paragraphs. The probabilistic sources produce packets. If the input buffer connected to a probabilistic source is not full at the beginning of a time unit then with some probability the probabilistic source places an additional packet in the buffer. The probabilistic source connected to the first input of the first node generates packets with probability $IN$. The probabilistic sources connected to the second inputs of the nodes generate packets with probability $SI$. The perfect sinks never block and accept packets at whatever rate they are presented.

*Simulation of the Model*

We use simulation of the model to study the effect of conflict in a randomly selected network path on the rate at which packets can enter the first router of the path. In this section, we describe the simulation of the model, discuss the implications of the simulation results, and compare the simulation results for the model to simulation results for the complete indirect n-cube network.

We run the simulation in such a way that we can use the results of the simulation to draw conclusions about the limit that conflict in a randomly selected network path places on the rate at which packets can enter the first router of the path. In the model, we set $IN$ to 1. We examine the rate at which packets enter the first input of the first node as a function of the value of $SI$. We find a value such that when $SI$ is set to the value, the rate at which packets enter the first input of the first node is also equal to the value. We refer to this value as the maximum input rate for the model. The maximum input rate for the model in some sense represents the limit that conflict in a randomly selected network path places on the rate at which packets can enter the first router of the path, if conflict elsewhere in the network is ignored and if it is assumed that each network input receives the same input rate.

We have simulated the model for several values of $B$ and for several path lengths. The maximum input rate for each case is shown in Table II. It should be noted that the values listed are percentages and that only values with whole percentage points were used in the simulation.

The model suggests that the effect of conflict in a randomly selected path increases with the length of the path. The maximum input rate for the model decreases as the length of the model increases. Each node can block all earlier nodes. Nodes at the beginning of a long path can be blocked by any of the later nodes. Thus, the maximum input rate for the model of a long path is less than the maximum input rate for the model of a short path.

**Table II. Simulation of the Model of a Network Path**

| | length | maximum input rate |
|---|---|---|
| $B = 5$ | 128 | 59 |
| | 64 | 60 |
| | 32 | 61 |
| | 16 | 63 |
| | 8 | 64 |
| | 4 | 67 |
| | 2 | 71 |
| $B = 3$ | 128 | 50 |
| | 64 | 51 |
| | 32 | 54 |
| | 16 | 56 |
| | 8 | 59 |

---

However, the model suggests that for very long paths the effect of conflict increases very slowly with path length. The chance in the model of a long path that all of the buffers between a late node and an early node are full is small. The chance that a conflict in the late node blocks the early node is small. While we will not discuss this issue in detail, we expect that the maximum input rate for the model approaches a nonzero asymptote as the length of the model goes to infinity. As is shown in Table II, 128-node models were simulated. We expect that the maximum input rates for the 128-node models are close to the maximum input rates for infinite length models.

The limit of the input rate of a randomly selected network path implies a limit on the overall throughput of a network. Clearly, we do not expect the total throughput of a network to be greater than the width of the network times the maximum input rate of a randomly selected network path. For most networks, this limit is stronger than any of the other limits studied so far. For example, this limit is usually stronger than the limit placed by the slowest router in the last stage when conflict in other stages is ignored. However, it is important to note that this limit does not rule out high throughput for very large

networks. The ratio of this limit to the number of network inputs approaches a nonzero asymptote as the size of the network goes to infinity.

We have not found any factors that place a significantly stronger constraint on network throughput than the interaction of routers along network paths discussed in the previous paragraphs. This suggests that the normalized throughput, throughput divided by the number of network inputs, of the indirect n-cube network approaches a nonzero asymptote as the size of the network goes to infinity.

For comparison, complete indirect n-cube networks were simulated. The results are shown in Table III. While the simulation results do not clearly indicate the normalized throughput of networks of infinite size, the normalized throughputs of the networks simulated are consistent with the results of the model above since the normalized throughput of each complete network is less but not drastically less than the normalized throughput of the corresponding model.

*Interaction of Routers in a Tree*

In the following paragraphs, we consider the interaction of routers in a tree of routers in an indirect n-cube network. The interaction among such routers can be quite complex. We examine one particular type of interaction and its effect on the behavior of the network. While this interaction does not seem to

---

**Table III. Complete Network Simulation**

$B = 5$

| $d$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| N | 2 | 4 | 8 | 16 | 32 | 128 | 64 | 256 | 512 | 1024 | 2048 |
| normalized throughput | .749 | .681 | .643 | .617 | .598 | .583 | .571 | .562 | .553 | .548 | .542 |

have an important effect on the overall throughput of the network, this interaction may cause a few of the routers connected to the network inputs to be slow for a long period of time.

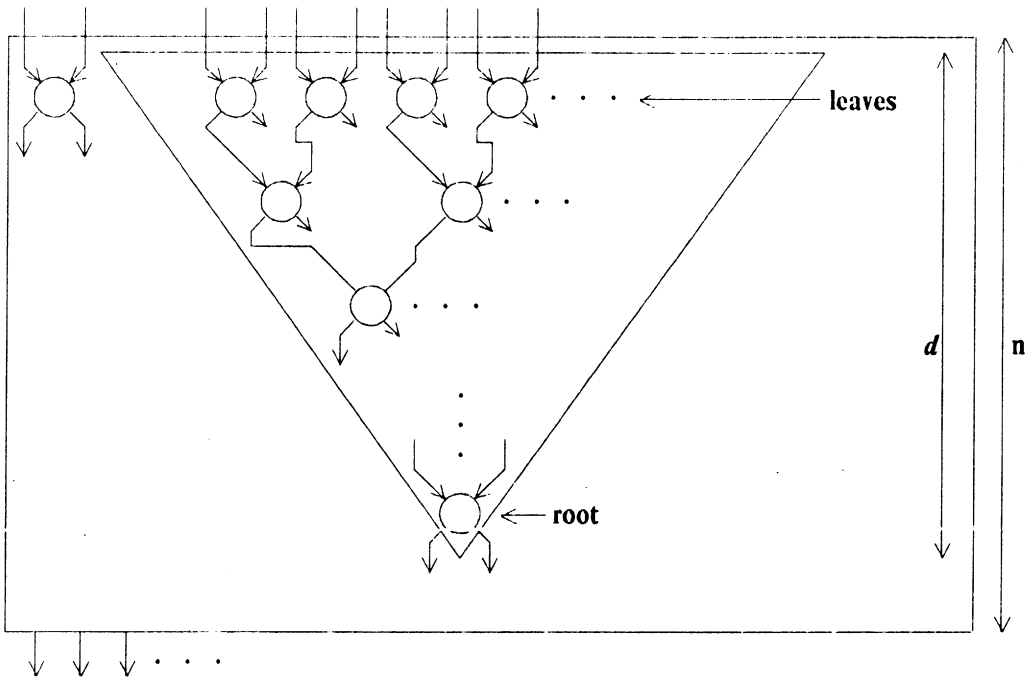
The effect of this interaction is important in networks of modest size, networks with less than 10 stages. As was mentioned above, this interaction may cause a few of the network inputs to be slow for a long period of time. We develop a model for this interaction, use the model to estimate its effect, and check our estimate by simulating the whole network. Our model suggests that if this type of interaction occurred in arbitrarily large trees, its effect would asymptotically grow as the square of the depth of the network. In fact, this interaction only occurs in trees of modest size, but it is strong enough to cause the input rate for the slowest input of a network with eight or nine stages to be less than half the expected input rate for a randomly selected input for a period of forty units of time.

The type of interaction discussed in the previous paragraph is less important in very large networks, networks whose depths are much greater than 10 stages. Since that type of interaction does not occur in very large trees, other factors become more important for very large networks. We briefly consider one of these factors. For very large networks, as we shall see, this factor implies that the slowest input router requires greater than $c_9 n/(\log_2 n)$ time to accept $3B$ packets where $c_9$ is a constant, $B$ is the buffer size, and n is the depth of the network.

*Trees in Networks of Modest Size*

In a network of modest size, we examine a particular interaction of routers in a $d$-stage tree whose leaves are connected to the network inputs. We select a router of the n-$d$ th stage from the final stage of the network where n is the depth of the network. We refer to this router as the root router of the tree. We consider the tree composed of the root router and all of the routers that can direct packets to that router as shown in Figure 18. We refer to the routers of the tree in the $d$-1 st stage from the root router as the leaf routers of the tree. Below, we study the time required by the slowest leaf router to accept $c_1 B^2$

Fig. 18. *d*-Stage Tree in a n-Stage Network




packets where $B$ is the size of the input buffers of the routers and $c_1$ is a constant. We do this using a model of the tree. We introduce the model and analyze it in order to estimate the time required for the slowest leaf router of the model to accept $c_1 B^2$ packets. We use simulation to compare our estimate to the performance of the model and to compare our estimate to the performance of the $d$-stage tree.

We assume that conflict in other parts of the network affects the tree only at the root router. It seems likely that the leaf routers accept packets more quickly with this assumption than without it. Thus, we make this assumption in order to estimate an upper bound on the rate at which the slowest leaf router accepts packets.

We assume that the links connected to the outputs of the root router accept packets very slowly. We assume that the inputs of the network are connected to perfect sources that supply packets as quickly as they can be accepted. We assume that n, the depth of the network, is no more than 10. We assume that the rate at which packets can be accepted from the outputs of the root router is limited by conflict in the later stages of the network and is less than the rate at which the root router can supply packets.

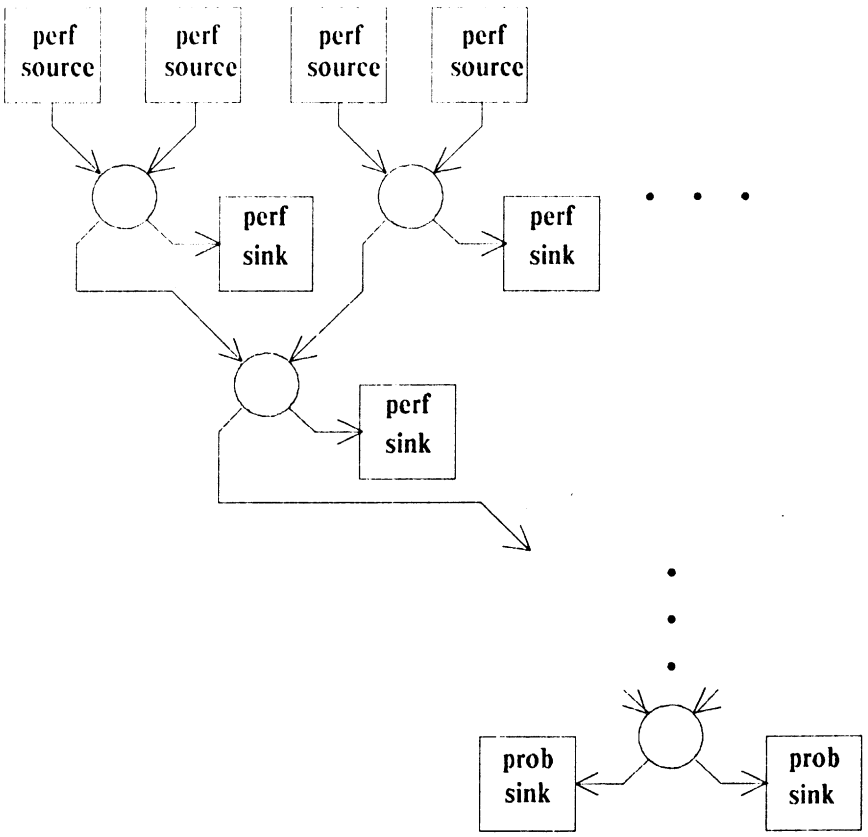
We are interested in the behavior of the tree after the network has been in operation for some period of time. We randomly choose a point in time after the network has been in operation for a long time. We examine the duration of the shortest period beyond that point such that during the period each leaf router of the $d$-stage tree of the network accepts $c_1 B^2$ packets where $B$ is the size of the input buffers of the routers and $c_1$ is a constant.

*Model of a d-stage Tree*

In order to estimate the duration of this period, we study the model of a $d$-stage tree shown in Figure 19. The characteristics of the model correspond for the most part to the assumptions that we made above. The characteristics of the model are such that it seems reasonable to assume that the model will lead to a lower bound on the duration of the period. We refer to the model as the $d$-stage special tree or simply the $d$-stage special.

The $d$-stage special is composed of routers, probabilistic sinks, perfect sinks, and perfect sources as shown in Figure 19. The routers operate in a manner similar to that of the routers of the network. Both of the outputs of the root router are connected to probabilistic sinks. Each probabilistic sink has an input buffer of size $B$. If the input buffer of a probabilistic sink is not empty at the beginning of a time unit then with probability $c_2$ the sink removes a packet from the buffer where $c_2$ is a small constant. Each router of the tree except the root has one output connected to another router of the tree and one output connected to a perfect sink. We refer to the routers of the tree in the $d$-1 st stage from the root router as

Fig. 19. *d*-Stage Special




the leaf routers of the tree. The inputs of the leaf routers are connected to perfect sources that produce packets as quickly as they can be accepted.

*Analysis of the Model*

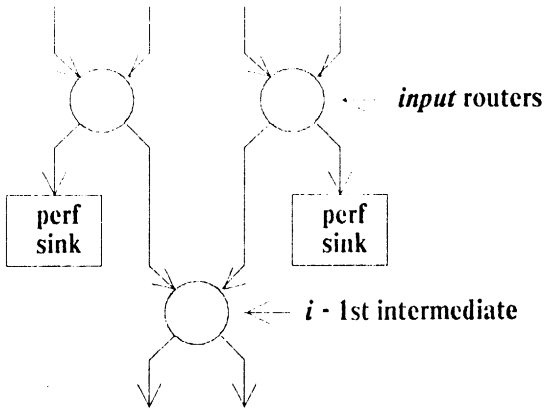
We examine the behavior of the *d*-stage special after it has been in operation for some period of time. We randomly choose a point in time after the *d*-stage special has been in operation for a long period. We examine the behavior of the *d*-stage special after that point.

We are interested in the time required after the selected point for the slowest leaf router to accept a total of $c_1 B^2$ packets. For the purpose of discussion, we refer to this time as the slowest leaf router acceptance time, $T_d$. As before, we use the notation $E[x]$ to refer to the expected value of $x$. Below, we estimate a lower bound on $E[T_d]$. We refer to this estimate as $estT_d$. We develop a recursive definition for $estT_d$. The definition is given in equations 99-101. For large $d$, as we shall see, $estT_d$ grows as the square of $d$. We are primarily interested in $estT_d$ for $d < 10$ since we expect the type of interaction modeled by the $d$-stage special to occur only in trees of depth < 10. As is shown below in Table IV, $estT_d$ grows rapidly even for $d$ in this range.

In order to make the desired estimate of $E[T_d]$, the expected value of the slowest leaf router acceptance time, we consider the behavior of one router from each stage. In each stage, we select the router that accepts the smallest number of packets in the $T_d$ unit period during which the slowest leaf router accepts $c_1 B^2$ packets. For $0 \leq i < d$, we refer to the selected router of the $i$ th stage from the root router as the $i$ th intermediate. For $0 \leq i < d$, we define $numbpackets_d^i$ to be the number of packets accepted by the $i$ th intermediate in the $T_d$ period. In order to estimate $E[numbpackets_d^0]$, the expected value of $numbpackets_d^0$, we consider $E[numbpackets_d^{i-1}] - E[numbpackets_d^i]$ for each value of $i$ such that $0 < i < d$ and we make use of the fact that $E[numbpackets_d^{d-1}]$ is equal by definition to $c_1 B^2$. We use $E[numbpackets_d^0]$, the expected number of packets accepted during the period by the root router, to estimate $E[T_d]$, the expected length of the period. We argue that $E[numbpackets_d^0]$ is big and then assuming that it is big we argue that $E[T_d]$ is big.

For each value of $i$ such that $0 < i < d$, we estimate $E[numbpackets_d^{i-1}] - E[numbpackets_d^i]$, the difference between the expected number of packets accepted by the $i$-1 st intermediate and the expected number of packets accepted by the $i$ th intermediate, by considering the $i$-1 st intermediate and the two routers connected to its inputs as shown in Figure 20. For the purpose of discussion, we refer to the routers connected to the inputs of the $i$-1 st intermediate as the *input* routers. We consider the operation

Fig. 20. $i$-1 st Intermediate and *input* Routers




---

of these components in the $T_d$ unit period after the selected point in time.

The packet tags examined by each of the *input* routers correspond to a Bernoulli process and the packet tags examined by one of the *input* routers are independent of the packet tags examined by the other *input* router, and we use these facts in the following paragraphs to estimate a lower bound on $E[numbpackets_d^{i-1}] - E[numbpackets_d^{i}]$. We argue that during the period one *input* router is likely to receive a larger fraction of packets labeled for the $i$-1 st intermediate than the other *input* router receives. We then argue that during the period one of the *input* routers is likely to accept a smaller total number of packets than the other *input* router accepts. Since the number of packets accepted during the period by the $i$ th intermediate can be no more than the number accepted by the slower *input* router of the $i$-1 st intermediate, we estimate a lower bound on $E[numbpackets_d^{i-1}] - E[numbpackets_d^{i}]$ by estimating the difference between $E[numbpackets_d^{i-1}]$ and the expected number of packets accepted during the period by the slower *input* router of the $i$-1 st intermediate.

For the purpose of discussion, we define some notation. We arbitrarily order the *input* routers. For $k = 1$ and $k = 2$, we refer to the following quantities,

$PA_k$, the number of packets accepted during the period by the $k$ th *input* router,

$TPA_k$, the number of packets accepted during the period by the $k$ th *input* router that are tagged for the $i$-1 st intermediate,

$TPO_k$, the number of packets output during the period by the $k$ th *input* router that are tagged for the $i$-1 st intermediate,

$MPA$, the minimum of $PA_1$ and $PA_2$,

$TPA'_k$, the number of packets that are in the first $MPA$ packets accepted by the $k$ th *input* router and that are tagged for the $i$-1 st intermediate,

$f_k$, $TPA_k / PA_k$,

and

$f'_k$, $TPA'_k / MPA$.

We also define *kmaxtpa* to be equal to one if $TPA'_1$ is greater than or equal to $TPA'_2$ and we define *kmaxtpa* to be equal to two otherwise.

About half of the packets accepted during the period by an *input* router are tagged for the $i$-1 st intermediate. The tag of each packet is independent of the tags on other packets. Thus, the tags on the packets accepted by the router can be considered to correspond to a Bernoulli process. The chance that a packet accepted by the router is tagged for the $i$-1 st intermediate is (.5). Thus for $k = 1$ and $k = 2$, $E[TPA'_k]$, the expected number of packets that are in the first $MPA$ packets accepted by the $k$ th *input* router and that are tagged for the $i$-1 st intermediate, is equal to $(1/2)E[MPA]$.

However, since there are two *input* routers and since the tags on packets received by the two *input* routers correspond to two independent Bernoulli processes,

$$E[TPA'_{kmaxtpa}] > (1/2)E[MPA] + c_3(E[MPA])^{1/2} \tag{89}$$

for some constant $c_3$.

Thus, $E[TPA_{kmaxtpa}]$, the expected number of packets accepted during the period by the *kmaxtpa* th *input* router that are tagged for the $i$-1 st intermediate, is greater than

$$(1/2)E[MPA] + c_3(E[MPA])^{1/2} + (1/2)E[PA_{kmaxtpa} - MPA]. \tag{90}$$

Assuming small deviations from the means, we assume that $E[f_{kmaxtpa}]$, the expected value of $TPA_{kmaxtpa}/PA_{kmaxtpa}$, is greater than

$$1/2 + \frac{c_4(E[MPA])^{1/2}}{E[PA_{kmaxtpa}]} \tag{91}$$

for some constant $c_4$. We assume that $E[PA_{kmaxtpa}] - E[MPA] \ll E[PA_{kmaxtpa}]$ and that as a result $E[f_{kmaxtpa}] > 1/2 + \frac{c_5}{(E[PA_{kmaxtpa}])^{1/2}}$ for some constant $c_5$. Thus, we assume that

$$E[f_{kmaxtpa}] > 1/2 + \frac{c_6}{(E[TPA_{kmaxtpa}])^{1/2}} \tag{92}$$

for some constant $c_6$.

To simplify the discussion, we assume that the input buffers of the $i$-1 st intermediate remain non empty during the period. The motivation for this assumption can be seen by examining the operation of the tree during the period. We assumed earlier that the links connected to the outputs of the root router of the tree accept packets very slowly. Thus, we assume that the root router accepts packets very slowly. We expect the $i$-1 st intermediate to accept packets no more quickly than the root router since the $i$-1 st intermediate is the slowest router of the $i$-1 st stage from the root. Thus, we assume that the *input* routers can supply packets quickly enough that the input buffers of the $i$-1 st intermediate remain non empty.

Since we have assumed that the input buffers of the $i$-1 st intermediate remain non empty during the period, the number of packets removed from each of the input buffers of the $i$-1 st intermediate during the period is independent of the tag bits examined by the *input* routers and is thus independent of *kmaxtpa*, and these facts can be used to bound $E[TPO_{kmaxtpa}]$, the expected number of packets

output during the period by the *kmaxtpa* th *input* router that are tagged for the $i$-1 st intermediate. The expected number of packets removed during the period from the *kmaxtpa* th input buffer of the $i$-1 st intermediate is $(1/2)E[numbpackets_d^{i-1}]$. However, the number of packets in the *kmaxtpa* th input buffer of the $i$-1 st intermediate at the end of the period may depend on the tag bits examined by the *kmaxtpa* th *input* router during the period. Thus, the following relation holds for $E[TPO_{kmaxtpa}]$,

$$(1/2)E[numbpackets_d^{i-1}]-B \leq E[TPO_{kmaxtpa}] \leq (1/2)E[numbpackets_d^{i-1}]+B \ . \qquad (93)$$

Based on the arguments of the previous paragraphs, we estimate a lower bound on $E[numbpackets_d^{i-1}]$ - $E[numbpackets_d^{i}]$, the difference between the expected number of packets accepted during the period by the $i$-1 st intermediate and the expected number of packets accepted during the period by the $i$ th intermediate. Clearly, $E[TPA_{kmaxtpa}]$, the expected number of packets accepted during the period by the *kmaxtpa* th *input* router and tagged for the $i$-1 st intermediate, is less than or equal to $E[TPO_{kmaxtpa}] + 2B$. From (93) and (92), we have $E[TPO_{kmaxtpa}] \leq (1/2)E[numbpackets_d^{i-1}]+B$ and $E[f_{kmaxtpa}] > 1/2 + \frac{c_6}{(E[TPA_{kmaxtpa}])^{1/2}}$. Assuming small deviations from the means, we assume that $E[PA_{kmaxtpa}] = \frac{E[TPA_{kmaxtpa}]}{E[f_{kmaxtpa}]}$. Thus, we conclude that

$$E[PA_{kmaxtpa}] < \frac{E[TPA_{kmaxtpa}]}{1/2+\frac{c_6}{(E[TPA_{kmaxtpa}])^{1/2}}}, \qquad (94)$$

$$E[PA_{kmaxtpa}] < \frac{(1/2)E[numbpackets_d^{i-1}]+3B}{1/2+\frac{c_6}{((1/2)E[numbpackets_d^{i-1}]+3B)^{1/2}}}, \qquad (95)$$

$$E[PA_{kmaxtpa}] < E[numbpackets_d^{i-1}] + \frac{3B - \frac{c_6 E[numbpackets_d^{i-1}]}{((1/2)E[numbpackets_d^{i-1}]+3B)^{1/2}}}{1/2+\frac{c_6}{((1/2)E[numbpackets_d^{i-1}]+3B)^{1/2}}}, \qquad (96)$$

and

$$E[PA_{kmaxtpa}] < E[numbpackets_d^{i-1}] - c_7(E[numbpackets_d^{i-1}])^{1/2} \quad (97)$$

for some positive constant $c_7$. Clearly, $E[numbpackets_d^i]$, the expected number of packets accepted during the period by the $i$ th intermediate, is no greater than the expected number of packets accepted during the period by the slower *input* router (the slower of the two routers connected to the inputs of the $i$-1 st intermediate) and thus is no greater than $E[PA_{kmaxtpa}]$. Thus, $E[numbpackets_d^{i-1}]$ - $E[numbpackets_d^i] > c_7(E[numbpackets_d^{i-1}])^{1/2}$. Since $E[numbpackets_d^{i-1}] > E[numbpackets_d^i]$,

$$E[numbpackets_d^{i-1}] - E[numbpackets_d^i] > c_7(E[numbpackets_d^i])^{1/2}. \quad (98)$$

Based on the arguments of the previous paragraphs, we estimate a lower bound on $E[numbpackets_d^i]$, the expected number of packets accepted during the period by the $i$ th intermediate. We use the notation $estnpkts_d^i$ to refer to the estimate for a lower bound on $E[numbpackets_d^i]$. We define $estnpkts_d^i$ recursively. The basis comes from the fact that, by the definition of the period, $E[numbpackets_d^{d-1}]$ is equal to $c_1B^2$. The recursive step comes from the discussion of the previous paragraphs. Thus, we define

$$estnpkts_d^{d-1} = c_1B^2 \quad (99)$$

and for $0 < i < d$,

$$estnpkts_d^{i-1} = estnpkts_d^i + c_7(estnpkts_d^i)^{1/2}. \quad (100)$$

For very large values of $d$, $estnpkts_d^0$ grows roughly as the square of $d$; we are primarily interested in values of $d$ less than 10, but $estnpkts_d^0$ also grows rapidly for $d$ in this range. Below, we assume example values for $c_1$, $c_7$, and $B$, and compute $estnpkts_d^0$ for values of $d$ less than 10. The results are listed in Table IV.

We use $estnpkts_d^0$ to estimate a lower bound on $E[T_d]$, the expected value of the slowest leaf router acceptance time. Since the zeroth intermediate, the root router, is expected to accept $E[numbpackets_d^0]$ packets during the period, we assume that it is expected to output greater than

$(1/2)(E[numbpackets_d^0] + c_8(E[numbpackets_d^0])^{1/2})$ packets on one of its outputs during the period for some constant $c_8$. We assume that it takes $(1/2)(1/c_2)(E[numbpackets_d^0] + c_8(E[numbpackets_d^0])^{1/2})$ time for the probabilistic sink connected to that output to accept the packets where $c_2$ is the parameter of the probabilistic sink. We use the notation $estT_d$ to refer to the estimate for a lower bound on $E[T_d]$. We define

$$estT_d = (1/2)(1/c_2)(estnpkts_d^0 + c_8(estnpkts_d^0)^{1/2}). \tag{101}$$

As we shall see in the example below, $estT_d$, our estimated lower bound on the time required for the slowest leaf router to accept a total of $c_1 B^2$ packets, grows rapidly with $d$. As we shall see, $estT_d$ for $d$ equal to nine may be several times the size of $estT_d$ for $d$ equal to one.

*Evaluation of the Model*

We use simulation to evaluate how well the behavior of a special tree corresponds to our estimates, and how well the behavior of a tree in an indirect n-cube network of modest size corresponds to the behavior of a special tree. As is discussed below, our simulations suggest that the time required by the slowest leaf router of a $d$-stage special to accept $c_1 B^2$ packets grows at least as fast as $estT_d$ defined above (101), and our simulations suggest that the slow leaf routers of a tree in an indirect n-cube network of modest size are at least as slow as the slow leaf routers of a special tree of corresponding size.

We simulated special trees of various depths and examined the slowest leaf router of each tree in order to compare its behavior to our estimates. In the simulation, the size of the buffers, $B$, was equal to five. $c_2$, the parameter of the probabilistic sinks, was set to (.35). A simulation run was made for each tree depth. Each simulation run was divided into many periods. Each leaf router accepted more than 25 packets during each period. For each period of each simulation run, the time required by the slowest leaf router to accept a total of 25 packets was measured. The average over all the periods of a simulation run was computed. The results of the simulation runs are shown in Table IV. For each simulation run, the

Table IV. Evaluation of $estT_d$

| $d$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| $T_d$ from simulation | 36.3 | 45.8 | 51.1 | 61.1 | 79.5 | 88.5 | 94.1 | 112.2 | | |
| $estT_d$ | 41.1 | 47.0 | 53.2 | 59.8 | 66.8 | 74.3 | 82.1 | 90.3 | 99.0 | |
| $estnpkts_d^i$ | 25.0 | 28.8 | 32.9 | 37.2 | 41.9 | 46.8 | 52.0 | 57.5 | 63.2 | 69.3 |

$B=5,\ c_1B^2=25,\ c_7=.76,\ c_2=.35,\ c_8=.76$

---

average time for the slowest leaf router and the depth of the tree, $d$, are listed.

For comparison with the special tree simulation results, we computed $estT_d$ and $estnpkts_d^0$ for various values of $d$. We assumed that $c_7$ was equal to $c_8$ and that $c_8$ was equal to (.76) and we assumed that $estnpkts_d^{d-1}$ was equal to 25. The computed values are listed in Table IV. The simulation results seem to grow at least as fast as $estT_d$.

We simulated complete indirect n-cube networks of modest size in order to compare their slow input routers to the slow leaf routers of special trees. Networks with buffers of size one, two, and five were simulated. In the simulation, the outputs of the networks were connected to perfect sinks. A simulation run was made for each combination of network depth and buffer size. For each buffer size, $B$, a value of $c_1$ was selected. Each simulation run was divided into many periods such that each input router accepted more that $c_1B^2$ packets in each period. For each period of each simulation run, the time required by the slowest input router to accept a total of $c_1B^2$ packets was measured. The average over all the periods of a simulation run was computed. The results of the simulation runs are shown in Table V.

For comparison with the complete networks, we performed additional simulation of special trees. We simulated special trees with the same buffer sizes as the complete networks. We simulated a special

Table V. Evaluation of the Special Tree Model

| | $n = d + 4 =$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|---|
| | $d =$ | | | | | 1 | 2 | 3 | 4 |
| $B = 1$ | | | | | | | | | |
| $c_1 B^2 = 4$ | | | | | | | | | |
| | $T_{cnslow}$ | 4.6 | 7.1 | 13.1 | 15.1 | 22.6 | 28.0 | 43.1 | 47.2 |
| | $c_1 B^2/(2T_{cnslow})$ | .43 | .28 | .15 | .13 | .09 | .07 | .05 | .04 |
| | $IR_{cn}/2^n$ | .38 | .31 | .26 | .23 | .21 | .20 | .18 | .17 |
| | $T_{spslow}$ ($c_2 = .23$) | | | | | 9.1 | 16.9 | 28.5 | 36.0 |
| $B = 2$ | | | | | | | | | |
| $c_1 B^2 = 16$ | | | | | | | | | |
| | $T_{cnslow}$ | 10.5 | 14.4 | 19.1 | 22.8 | 30.3 | 33.4 | 39.7 | 45.6 |
| | $c_1 B^2/(2T_{cnslow})$ | .76 | .56 | .42 | .35 | .26 | .24 | .20 | .18 |
| | $IR_{cn}/2^n$ | .74 | .62 | .54 | .49 | .46 | .43 | .41 | .40 |
| | $T_{spslow}$ ($c_2 = .23$) | | | | | 16.1 | 21.4 | 24.6 | 29.5 |
| $B = 5$ | | | | | | | | | |
| $c_1 B^2 = 25$ | | | | | | | | | |
| | $T_{cnslow}$ | | | | 28.7 | 31.7 | 35.0 | 42.4 | 50.6 |
| | $c_1 B^2/(2T_{cnslow})$ | | | | .44 | .39 | .36 | .29 | .25 |
| | $IR_{cn}/2^n$ | | | | .62 | .60 | .58 | .57 | .56 |
| | $T_{spslow}$ ($c_2 = .23$) | | | | | 19.1 | 25.3 | 25.9 | 30.3 |

where $T_{cnslow}$ is the time required by the slowest input router of an indirect n-cube network to accept a total of $c_1 B^2$ packets

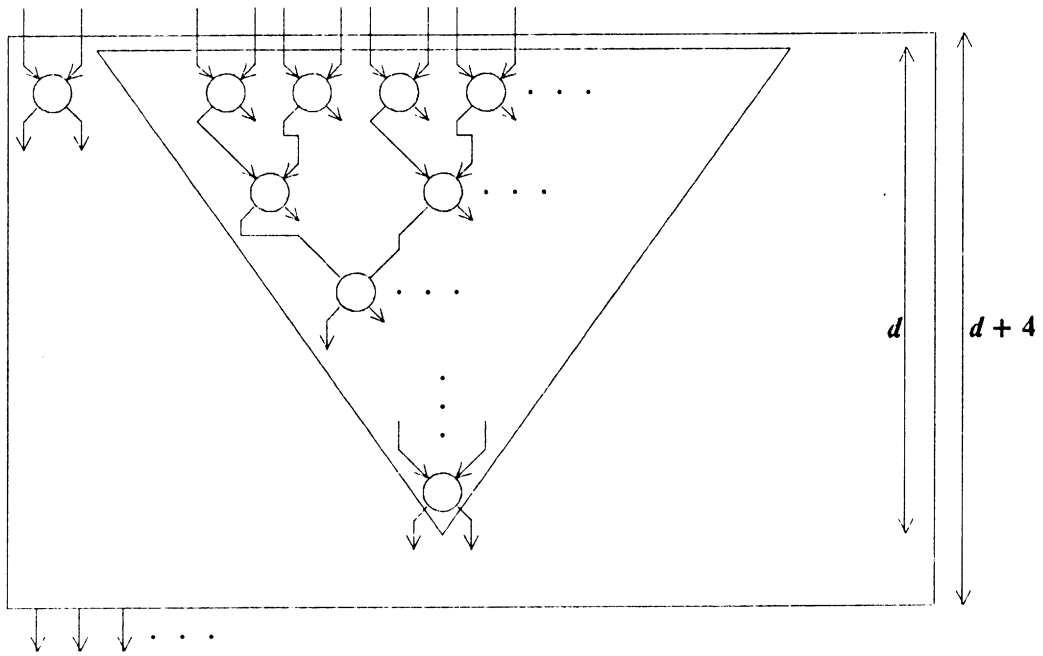
$IR_{cn}$ is the average total input rate of an indirect n-cube network

and $T_{spslow}$ is the time required by the slowest leaf router of a $d$-stage special to accept a total of $c_1 B^2$ packets

tree for each combination of depth and buffer size. For each special tree simulated, if $d$ was the depth of the special then we set the parameters of the special to correspond to the parameters of a $(d+4)$-stage network with the same buffer size and with its outputs connected to perfect sinks. We did this in order to evaluate how well the $d$-stage special represented $d$-stage trees in the first $d$ stages of a $(d+4)$-stage network as shown in Figure 21. As was discussed earlier, the behavior of a special tree is intended to represent the behavior of a tree in the first stages of a complete network. We chose to compare the $d$-stage special to $d$-stage trees of a $(d+4)$-stage complete network. While the exact choice of $d+4$ was not critical, it was important to consider a network that corresponded to the assumptions of the special trees. In particular, it was important to consider a network large enough that the rate at which packets

---

**Fig. 21. $d$-Stage Tree in a $(d+4)$-Stage Network**



were accepted from the roots of the $d$-stage trees in the first $d$ stages of the network was low. The value of $c_1$ used for the special was equal to the value of $c_1$ used for the complete network. The value of $c_2$, the parameter of the probabilistic sinks of the special, was selected to roughly correspond to the rate at which packets could be accepted by an input of a router of the third from the last stage of the complete network. We estimated this rate by considering a four stage complete network with the same buffer size and with its outputs connected to perfect sinks, and by measuring the average rate at which its inputs could accept packets. A simulation run was made for each special tree. Each simulation run was divided into many periods such that if $B$ was the buffer size, each leaf router accepted more than $c_1 B^2$ packets in each period. For each period of each simulation run, the time required by the slowest leaf router to accept a total of $c_1 B^2$ packets was measured. The average over all the periods of a simulation run was computed. The results are listed in Table V. The simulation results for the special trees do not seem to grow any faster than the simulation results for the corresponding complete networks.

Table V also lists simulation results that can be used to compare the input rate of the slowest input router of an indirect n-cube network to the total input rate of the network. For each simulation run, the table lists $c_1 B^2$ divided by twice the average, over all of the periods of the run, of the time required by the slowest input router to accept $c_1 B^2$ packets. This quotient gives an indication of the rate at which the slowest input router accepts packets on each of its inputs. For each simulation run, the table also lists the average total input rate of the indirect n-cube network divided by the number of network inputs. These results suggest that the input rate of the slowest input of an indirect n-cube network can be several times slower than the expected input rate of a randomly selected input.

*Trees in Very Large Networks*

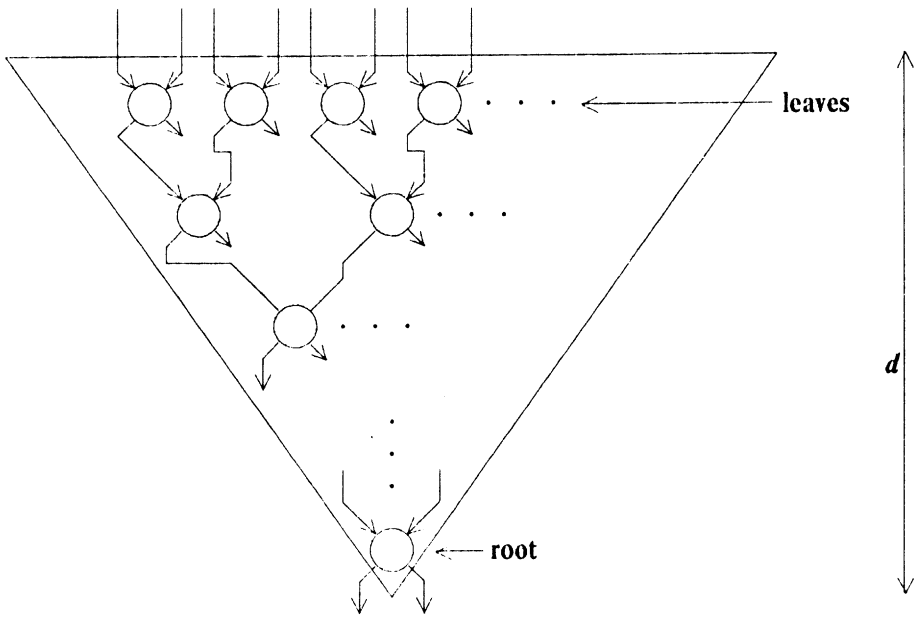
In very large indirect n-cube networks, n much larger than 10, the interaction between a large tree and the rest of the network does not correspond to the assumptions that we made for modest trees in modest networks, n less than 9. In a very large tree, as shown in Figure 22, conflict in the earlier stages of

the tree causes routers in the later stages to receive packets very slowly. The rate at which a router in one of the later stages of the tree receives packets on its inputs is not likely to be larger than the rate at which packets can be accepted from its outputs. The input buffers of such a router may often be empty. As a result, the conclusions that we drew about the slow leaf routers of modest trees in modest networks do not hold for the slow leaf routers of very large trees in very large networks. For $d$ much larger than 10, we do not expect the slowest leaf router of a $d$-stage tree in an indirect n-cube network to require $estT_d$ time to accept $c_1 B^2$ packets. We do not expect such a router to require time proportional to $d^2$ to accept $c_1 B^2$ packets and $estT_d$ is proportional to $d^2$.

However, the time required by the slowest input router of a very large network to accept a constant number of packets is a function of the network's depth. In particular, if the depth of the network is n, we

---

**Fig. 22. $d$-Stage Tree**



expect the slowest input router of the network to require greater than $\frac{c_9 n}{(\log_2 n)}$ time to accept $3B$ packets for some constant $c_9$.

The motivation for this can be seen by considering the network and its operation. For the purpose of discussion, we define

$$K = \frac{c_9 n}{B(\log_2 n)}. \tag{102}$$

We consider the routers in the $(\log_2 K)$-1 st stage from the network inputs. For the purpose of discussion, we refer to the $(\log_2 K)$-1 st stage from the network inputs as the selected stage. The network inputs can be divided into $(N/K)$ groups with $K$ inputs in each group such that the set of routers of the selected stage that can receive packets from one group of inputs is disjoint from the set of routers of the selected stage that can receive packets from any other group of inputs.

We consider the operation of the network after some randomly chosen point in time and argue that with high probability at least one of the inputs accepts less than $c_1 B^2$ packets in a period of $\frac{c_9 n}{(\log_2 n)}$ units of time after the selected point. We define $P$ to be the chance that the first $3BK$ packets to arrive on a group of inputs must pass on the same output of the same router of the selected stage. Thus,

$$P = K(1/K)^{3BK}, \tag{103}$$

$$P > (1/K)^{3BK}, \tag{104}$$

and

$$P = \frac{(\frac{B(\log_2 n)}{c_9})^{3BK}}{n^{3BK}} = \frac{(\frac{B(\log_2 n)}{c_9})^{3BK}}{N^{3c_9}}. \tag{105}$$

We define $P'$ to be equal to the chance that, for at least one of the groups of inputs, the first $3BK$ packets received by that group are tagged for the same output of the same router of the selected stage. Thus,

$$P' = 1-(1-P)^{N/K}. \tag{106}$$

Using the Taylor series for $\log_e(1-P)$,

$$P' = 1-e^{-(N/K)(\Sigma_{i=1 \text{ to } \infty}(1/i)P^i)}, \tag{107}$$

$$P' > 1\text{-}e^{-(N/K)P}, \tag{108}$$

and

$$P' = 1\text{-}e^{-exp} \tag{109}$$

where $exp$ is equal to $(N/K)P$. Thus,

$$exp = (N/K)P = (N^{1-3c_9})(\frac{B(\log_2 n)}{c_9})^{3BK}(1/K). \tag{110}$$

If $c_9$ is much less than 1/3 and N is large then there is a good chance that for at least one group of inputs, all of the first $3BK$ packets received by that group are tagged for the same output of the same router of the selected stage. In such a case, since only $B(2K\text{-}2)$ packets can be buffered between that group of inputs and the output of the router of the selected stage, the operation of the router of the selected stage affects that group of inputs. Since there are $K$ inputs in the group and since only one packet can be transferred per unit time on the output of the router of the selected stage, at least one of the inputs will accept less than $3B$ packets in a period of $BK$ units of time. In other words, at least one of the inputs will accept less than $3B$ packets in a period of $\frac{c_9 n}{(\log_2 n)}$ units of time. Thus, if $c_1 B$ is greater than three then at least one of the inputs will accept less than $c_1 B^2$ packets in a period of $\frac{c_9 n}{(\log_2 n)}$ units of time.

## 2.3 Networks for Systems with Localized Communication

Many localized communication patterns can be supported with networks that are less complex than the uniform communication networks described above. There are, of course, a wide variety of localized communication patterns. While we have not done extensive work on this topic, we describe in this section an obvious family of network structures that seem appropriate for some important localized communication patterns.

In the technologies of this chapter, there is a large class of systems such that each system can be supported by a network that has a cost linear in its number of inputs. Since we are assuming in this

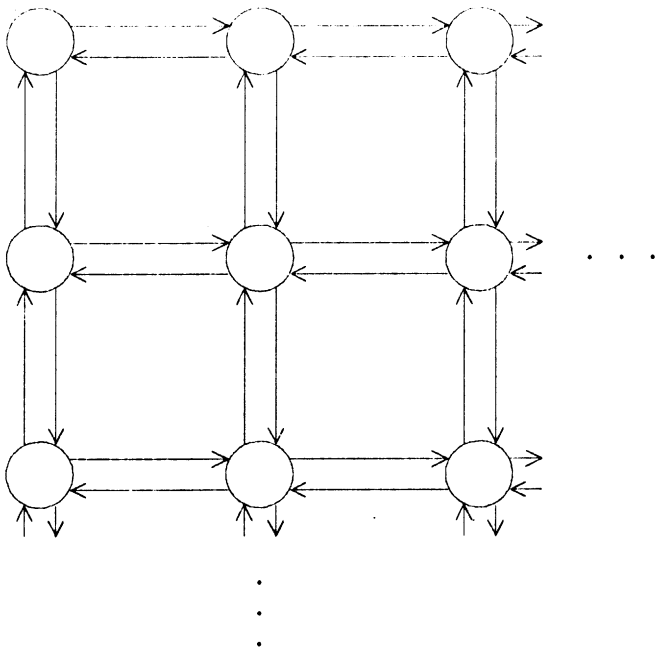
chapter that all wires have the same cost independent of their length, the factors affecting the cost of a network are simply the number and complexity of network nodes, and the number of interconnecting wires. If each source of packets in a given system generates packets for a number of destinations that is less than $c$ where $c$ is some constant independent of the total number of sources then the communication requirements of the system can be supported by a network with a number of nodes and wires proportional to the number of inputs. Such a network can be constructed in the obvious way by associating a node with each input and a node with each output, and connecting each input node to the output nodes that correspond to possible destinations for packets from that input. The total cost of such a network is independent of the identity of the output nodes that must be connected to a given input node. It is important to note that this will not be the case in the technologies of the next chapter.

One linear cost network structure is the grid structure. We consider a grid of two dimensions as shown in Figure 23, but grids of higher dimensions are also useful. Each node of the grid is connected to the nodes adjacent to it. Each node is also connected to a network input and a network output. Clearly, the cost of such a network is linear in the number of network inputs. Such a network can obviously be used in systems that support computations on grid structured data such that the computation on a given grid element involves only the adjacent grid elements.

Another linear cost network structure is the tree structure. In such a structure, the nodes are connected in a tree as shown in Figure 24. The inputs and outputs of the network can be connected in at least two possible ways. One way is to connect each node of the network to a network input and a network output. Another is to connect only the leaf nodes to the network inputs and outputs. In the discussion below, we assume that each node of the network is connected to a network input and a network output.

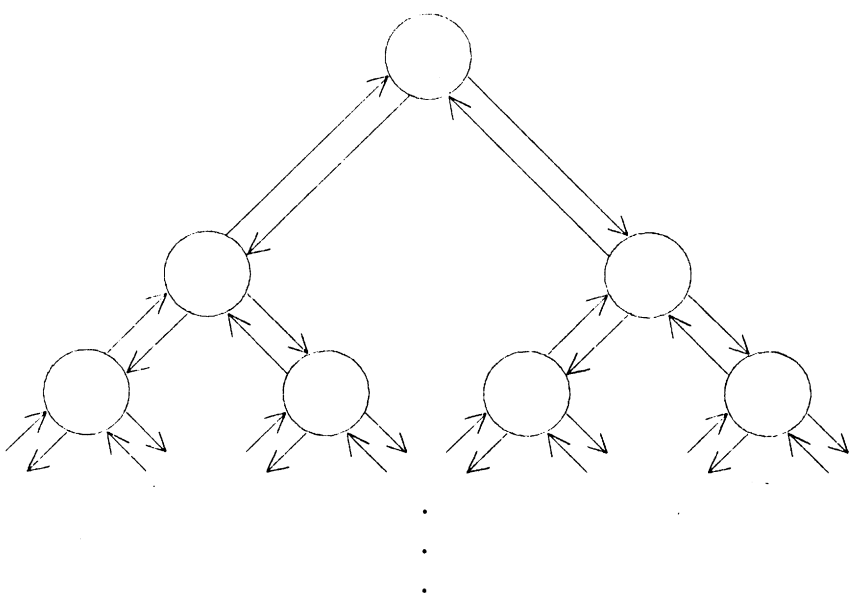
A tree network can be used to support applications, such as divide and conquer algorithms, that require hierarchical communication patterns. A tree network can simultaneously support communication

Fig. 23. Grid Structure



between the PC's in each pair of adjacent leaf PC's. Thus, it can support a high total bandwidth for such communication. But the network has half that bandwidth for supporting communication that must go through the PC's of the second stage from the leaves. In general, if for some $i$ the network has some bandwidth for packets that must go through the $i$th stage then it has half that bandwidth for packets that must go through the $i+1$st stage. Thus, the tree network can be used to support some systems that require hierarchical communication. For example, the network can obviously be used in a tree structured system where each module is the root of a subtree of modules that it controls and where each module requires the same communication bandwidth.

Fig. 24. Tree Structure



# 3. Design of Routing Networks Considering the Cost of Wires

## 3.1 Introduction

In this chapter, we examine the changes that will probably occur in integrated circuit technology in the next five to ten years, and examine the design of routing networks assuming such changes. We refer to the resulting technology as very large scale integration (VLSI).

In the next five to ten years, we expect several improvements in integrated circuit technology. We expect a reduction in the width of wires and the size of transistors. The minimum width of wires may shrink to roughly $1/(2)^{1/2}$ to 1/(2.4) of its present value. The area required at the end of this period to implement a circuit may be 1/2 to 1/6 of the area required at present to implement the circuit. We expect the speed of on-chip circuits to increase to roughly two to four times their present speed. We also expect that by the end of this period the use of multiple layers of metal will be common.

While we expect features on a chip to become smaller and the complexity of on-chip circuits to increase, we do not expect the overall physical size of the chips to increase drastically in this period.

The improvements in integrated circuit technology will allow a large number of network nodes to be placed on a single chip. As a result, the wires interconnecting the nodes will be on-chip wires. Since the chip area required to implement an on-chip wire is proportional to its length, the length of wires in a network subsection will be an important factor in the chip area required to implement the subsection.

However, it appears that for the next five to ten years it will still be possible to drive even very long wires quickly by choosing drivers of the appropriate size. The capacitance to the substrate per square micron of metal will increase as the thickness of the oxide layers decreases. We expect that the width of wires will in general decrease and that the area of a given length of wire will decrease. The combination of increasing capacitance per unit area and decreasing area may cause the capacitance of a given length of

wire to remain roughly constant. A long wire with a large capacitance can be driven quickly if a large driver capable of driving a large amount of current is used. Presently, even a cross chip wire can be driven in time comparable to the delay of about ten logic stages by using a transistor whose area is a few times that of a minimum size transistor. It is not clear exactly how the area required to implement a high current transistor will change as technology changes. The current driving capacity of a transistor is inversely proportional to the resistance of its channel. The transistor channel resistance for a particular ratio of length to width may increase. But since the minimum channel length will decrease, the minimum area for a transistor with a certain current driving capacity may decrease. It seems likely that for the next five to ten years it will still be possible to drive a very long wire in reasonable time with a transistor that is quite small in comparison to the area of the wire.

In this chapter, we first describe a model of VLSI that we will later use to study the area required to implement network structures in VLSI. We expect that the dominant component of the total area required to implement a network in VLSI will be the area required to implement its wires. The features of the wires of the model correspond to what we have assumed above will be the primary characteristics of wires in VLSI. The wires of the model require an area proportional to their length. They have no propagation time. A signal can be asserted on a wire of the model in unit time.

We then examine in this model of technology the design of networks for uniform communication applications. We examine in the VLSI model the fundamental cost of a single chip network to support a certain level of performance for uniform communication applications. We examine a few structures that seem appropriate for implementing a single chip uniform communication network in VLSI. These structures include a crossbar structure, and an indirect n-cube structure. We discuss a technique for interconnecting single chip networks to form larger networks.

We also briefly examine networks for localized communication applications. We examine a few example network structures and describe the communication patterns that they can support.

## 3.2 VLSI Model

The model presented here is for the most part the model suggested by Thompson [25]. The items implemented on the chip can be broken into two primary types, processing centers (PC's) and wires. All processing occurs at the PC's. Transmission of information among the PC's is performed using the wires. All switching functions are performed by the PC's.

It should be noted that the concept of unit time that we use in the VLSI model is different from the concept of unit time that we used in the previous chapter. In the previous chapter, a unit of time was the time required to transfer a single packet on a link. In the VLSI model, a unit of time is the time required to transfer a single bit of information on a wire.

Each wire interconnects some number of PC's. A wire has unit width, and a signal (one bit of information) may be asserted on the entire length of a wire in unit time. The model characterizes a wire as a lumped capacitive and resistive load with a rise time but with no propagation time. The model allows multiple connections to a single wire. It should be noted that the area charged in the model for such connections may be too small. The primary reason for this comes from the fact that the model assumes that in a VLSI implementation the total area required for all of the drivers of a wire is comparable to, or is less than, the area of a wire. In the case of a wire with only one or two drivers, the area required to implement the drivers would be quite small in comparison to the area of the wire. However, in the case of a wire with a number of drivers proportional to its length, it is likely that the total area required to implement the drivers would be at least of the same order as the area of the wire. Thus, this model can be used to obtain lower bounds on the area required to implement a circuit, but the area required for wire drivers must be considered carefully in the actual implementation of any circuit requiring a large number of connections to a single wire.

Each PC (processing center) is connected to some number of wires. In this model, we assume that a PC is square and that the area required to implement a PC is at least as great as $n^2$ where $n$ is the number of wires connected to the center. This area is of the same order as that required to construct a complete cross point switch for switching among the wires. If a center does not need such a powerful switching capability, it should be decomposed into smaller centers. We also assume that information can not pass through a node in less than one unit of time.

We assume that each piece of input data is available on-chip from a specialized input PC and that each piece of output data need only be delivered to an on-chip output PC. We have chosen to separate this model from the problem of getting information into and out of a chip. The input and output capacity of VLSI chips depends critically on the technology used to package the chips. It is not presently clear which technologies will be used as the scale of integration increases. This topic deserves further study as the packaging technology advances.

## 3.3 Networks for Systems with Uniform Communication

### 3.3.1 Wire Cost

In this subsection, we investigate the wire area required by single chip networks capable of high performance in systems with uniform communication. In particular, we obtain for $1 > f > 0$ a lower bound on the area required in the VLSI model by the wires of any single chip N-input N-output routing network capable of supporting an average throughput of $f$ N packets per unit time when its inputs are connected to the uniform communication model sources and its outputs are connected to the non blocking model receivers. For this study, each model source is assumed to produce a new packet within one time unit of the network's acceptance of the source's previous packet. We assume that the label of each packet produced by a model source is independently selected, and that all of the possible destination labels are equally likely.

**Proposition.** For $1 > f > 0$, $\Omega((fN)^2)$ area is required in the VLSI model to implement any single chip N-input N-output routing network capable of supporting an average throughput of $fN$ packets per unit time when its inputs are connected to the uniform communication model sources and its outputs are connected to the non blocking model receivers.

*Proof.* To get the desired lower bound for routing networks, we use an approach similar to that used by Thompson [25] for the discrete Fourier transform and by Abelson [1] for multiplication. This approach uses a concept called the minimum bisection width, which we will define in terms of the graph of a VLSI circuit. For any circuit in our VLSI model, the graph of the circuit is defined to be G = (V,E) where V contains a vertex for each of the PC's in the circuit, and E is the set of all sets {x,y} such that x and y are contained in V and a wire exists between the two PC's corresponding to x and y. The minimum bisection width of the circuit is defined to be the smallest $b$ such that for some partition of V into $H_1$ and $H_2$ with $|H_1| \leq |H_2| \leq |H_1| + 1$, the deletion of $b$ edges from E can disconnect $H_1$ from $H_2$. The minimum bisection width of a subgraph is similarly defined. If U is a subset of V for some graph G = (V,E), then the minimal bisection width of U in G is defined to be the smallest $b$ such that for some partition of U into $H_1$ and $H_2$ with $|H_1| \leq |H_2| \leq |H_1| + 1$, the deletion of $b$ edges from E can disconnect $H_1$ from $H_2$. Thompson has shown that if the minimum bisection width of some subset of the graph of a VLSI circuit is $b$, then the area required in the VLSI model for the circuit's wires and PC's is greater than $b^2/4$. Thus, if we can develop a lower bound on the minimum bisection width of any VLSI circuit capable of performing a particular function in a given period of time, we can deduce a lower bound on the area required by any such circuit.

A lower bound on the minimum bisection width of any single chip VLSI implementation of any N-input N-output routing network with the desired characteristics can be established by examining the communication needs of such a network. Let us consider the graph, G = (V,E), of any such VLSI implementation. We assume that the VLSI implementation has N input PC's and N output PC's. We

assume that each input PC can produce packets as fast as the network can accept them and each output PC can accept packets as fast as the network can present them. We will examine the case for even N. A similar approach can be used for odd N by ignoring one input PC and one output PC. Let O be the subset of V corresponding to the N output PC's, and I be the subset of V corresponding to the N input PC's. If the minimum bisection width of O in G is $b$, then there must be a set of $b$ edges whose removal would cause one half of the vertices in O to be disconnected from the other half. Let $O_1$ and $O_2$ be the two disconnected subsets of O that would result from the bisection where $|O_1| = |O_2|$. Let $I_1$ be the set of all vertices in I that would remain connected to any vertex in $O_1$ after bisection, and $I_2$ be the set of all vertices in I that would remain connected to any vertex in $O_2$. $I_1$ and $I_2$ must be disjoint since by definition the bisection disconnects $O_1$ and $O_2$. It follows from our previous assumption regarding the average throughput of the network, that in some very long period of duration $T$ the network must be able to accept at least $f\,NT$ packets. The characteristics of the model sources imply that the expected number of packets received during such a period that must be routed either from inputs in $I_1$ to outputs in $O_2$ or from inputs in $I_2$ to outputs in $O_1$ is greater than or equal to $f\,NT/2$. Since each wire can transmit only one bit per unit time, there must be at least $f\,N/2$ wires corresponding to the edges in the bisection. Therefore $b$, the number of edges in the minimal bisection of O in G, must be at least $f\,N/2$.

Based on these results and Thompson's theorem it follows that the area required in the VLSI model to implement any N-input N-output routing network with the capacity to support an average throughput of $f\,N$ packets per unit time in the uniform communication model application is $\Omega((f\,N)^2)$. In other words, there exists a constant $c$ such that the area is greater than or equal to $c\,(f\,N)^2$.

This ends the discussion of the proposition. ∎

If we make certain additional assumptions about the network, we can obtain a more detailed lower bound on the area required to implement the network in the VLSI model. In particular, if we assume that there are at least $p$ bits in each packet, and if we continue to assume that the VLSI implementation

has N input PC's and N output PC's, then we can show that $\Omega((p\,f\,N)^2)$ area is required in the VLSI model to implement the network. The argument is similar to the one used above.

Thus under these assumptions, $p$ and N have the same effect on our lower bound.

As we shall see later, some of the networks that we present come close to this bound. In particular, they can support a throughput of $\Omega(N\,\frac{w}{p}\,\frac{1}{\log w})$ packets per unit time and require $O((w\,N)^2)$ area in the VLSI model where $p$ is the number of bits in each packet and $w$ is the number of wires in each link of the network. Thus, the area of the networks differs from the lower bound by a factor of $\frac{1}{\log w}$. This factor is a result of the fact that we assume that a network node requires log $w$ time to receive and acknowledge a group of $w$ bits where one bit comes from each of the $w$ wires of a link.

In considering these results, it should be remembered that the nature of the communication networks required for a particular system depends on the overall design of the system. One important issue in the design of a large system is the decomposition of the system into subsystems such that each subsystem can be implemented on a single chip in VLSI. A number of factors affect the decomposition of the system. The nature of the modules to be interconnected by a network affects the chip area required to implement them, and it affects the feasibility of implementing them on the same chip as the network. The maximum area of a chip and the maximum number of pins on a chip limit the complexity of and the communication bandwidth of a single chip subsystem.

The decomposition of a system affects the characteristics of the communication networks required by the system and thus affects the cost of those networks. To illustrate this, we consider a system with two possible multiple chip implementations. We refer to these two implementations as $implementation_1$ and $implementation_2$. $implementation_1$ is composed of N single chip modules interconnected by $w$ single chip uniform communication networks. In $implementation_1$, we assume that each of the single chip modules has $w$ inputs of one wire each and $w$ outputs of one wire each, and we assume that each of the networks

has N inputs of one wire each and N outputs of one wire each. *implementation*$_2$ is composed of N single chip modules interconnected by a single uniform communication network. In *implementation*$_2$, we assume that each of the single chip modules has one input of $w$ wires and one output of $w$ wires, and we assume that the network has N inputs of $w$ wires each and N outputs of $w$ wires each. The lower bounds above suggest that a N-input N-output uniform communication network with $w$-wire data paths requires more area in the VLSI model than a N-input N-output uniform communication network with single wire data paths. In particular, a network with $w$-wire data paths requires $w^2$ times as much area as a network with single wire data paths. Thus, the total area in the VLSI model required for the networks of *implementation*$_1$ is less than the area required for the network of *implementation*$_2$ by a factor of $w$.

In order to further demonstrate the importance of the decomposition of a system into single chip subsystems, we consider a third implementation of the system of the previous paragraph. We refer to this implementation as *implementation*$_3$. *implementation*$_3$ is similar to *implementation*$_1$ except that all of the components of *implementation*$_3$ are placed on the same chip. In particular, *implementation*$_3$ is a single chip composed of N modules interconnected by $w$ uniform communication networks. In *implementation*$_3$, each of the modules has $w$ single wire inputs and $w$ single wire outputs, and each of the networks has N single wire inputs and N single wire outputs. In the VLSI model, it can be shown that *implementation*$_3$ requires area of the same order as the communication network of *implementation*$_2$. Thus, *implementation*$_3$ requires $w$ times as much chip area as that required by the networks of *implementation*$_1$. The discrepancy is due to the fact that some of the interconnections that are accomplished by on chip wires in *implementation*$_3$ are accomplished by off chip wires in *implementation*$_1$. In particular, the connections in *implementation*$_3$ between the modules and the networks require $\Omega((Nw)^2)$ chip area but the corresponding connections in *implementation*$_1$ are accomplished by off chip wires.

Thus, we have shown that the data paths of a single chip uniform communication network imply a certain lower bound on the area required by the network in the VLSI model. In particular, $\Omega((p\,f\,N)^2)$ area is required in the VLSI model to implement any single chip N-input N-output routing network with an average throughput of $f$N $p$-bit packets per unit time in the uniform communication model application. However, in considering this lower bound it should be remembered that the decomposition of a system into single chip subsystems affects the characteristics of the networks required by the system and thus the cost of those networks.

### 3.3.2 Network Structures

*Introduction*

A number of structures exist for single chip routing networks that are capable of high performance for uniform communication applications. These include a simple structure similar to the standard crossbar switch as well as the indirect n-cube structure. While the overall areas required by these structures in the VLSI model are similar, many of the other characteristics of these structures differ greatly. For example, the N-input crossbar network uses wires with N connections, but the N-input indirect n-cube network uses only wires with two connections. We examine a few network structures and examine the characteristics of each structure that affect its VLSI implementation.

We assume that $w$-bit wide data paths are used for each network structure. For each structure, as we shall see, the $w$ width of the data paths results in a $w^2$ factor in the area required to implement the structure. The area required for each of the network structures is $O((w\,N)^2)$ where $w$ is the number of wires in each link.

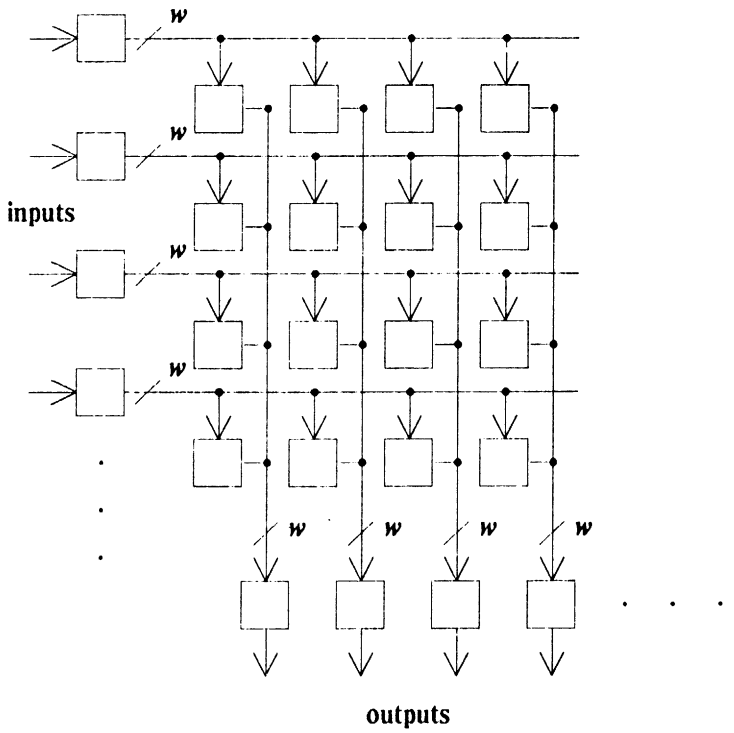
Thus for each of these structures, much less area is required in the VLSI model for $w$ networks with single wire data paths than for a single network with $w$-wire data paths.

*Crossbar Structure*

The first network structure that we examine is similar to the standard crossbar switch. We describe the structure, discuss its complexity, and discuss two of its drawbacks. These drawbacks are the need for many drivers for each long bus and the need for bus arbitration.

A N-input N-output network built according to this structure is composed of $N^2$ PC's arranged in a grid with an additional N input PC's and N output PC's as shown in Figure 25. Each of the PC's in a given row is connected to $w$ wires associated with that row. Similarly, each of the PC's in a given column is connected to $w$ wires associated with that column. Each row of the grid is associated with one of the network inputs. Each column of the grid is associated with one of the network outputs. The input PC's

**Fig. 25. Crossbar Network**



are connected to the network inputs, and the output PC's are connected to the network outputs. A packet entering the network is initially stored in a input PC. The destination label and data for one packet are transmitted by the input node across one row of the grid. Each PC in the grid is capable of determining if its associated output column is idle and if its associated input row is presenting data for that output column. In such a case, the grid PC connects the input row to the output column, and the output PC copies the presented data. Once the output PC has safely stored the packet, the grid PC will terminate its connection, and the input PC will present its next packet. Each output PC passes the packets it has received over the network output associated with it.

Obviously, $(N^2 + 2N)$ PC's are required for a N-input N-output network built according to this structure. Further, if the area required for each PC is independent of the size of the network, then the overall network layout can be done in the grid like fashion we have described using a total area which is proportional to $(Nw)^2$ where $w$ is the number of wires in each data path.

There are some problems associated with the implementation of very large networks that have this structure. The first comes from the fact that each output column wire can be driven by any of the N PC's in the column. For the reasons that we discussed earlier, the total area required for the drivers of a column wire in an actual implementation may be larger than the area of the wire. The area required to implement a N-input N-output crossbar network may grow faster than $N^2$. However in the technology of the next five to ten years, we expect that the growth will be close to $N^2$.

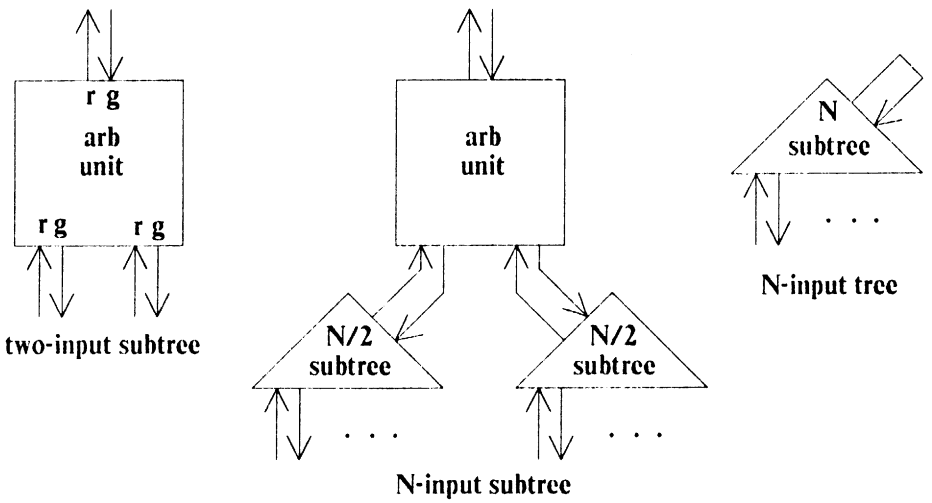
In addition, there are problems associated with the control of the various grid PC's. Only one grid PC in a given column should be allowed to drive the column at a given time. One way to accomplish this is to view the column as a synchronous bus and to use a grant signal that is daisy chained through the PC's of the column. Unfortunately, there are problems with this scheme. This scheme requires that a clock signal be distributed to all the PC's of the column. This scheme can only implement a fixed priority of inputs for the column. Further, the grant signal of this scheme is quite slow since it has to go through

the control circuitry of each of the N+1 PC's of the column. In the VLSI model, $\Omega(N)$ time is required for the grant signal to reach the lowest priority input row. This time would seem to be a serious problem since we would like each input row of the network to be capable of transmitting packets at a high rate. However in a real implementation, the speed of the control circuits in each PC may be much greater than the speed of an input row driver. For example, the grant signal may be propagated through a PC in one tenth of the time required for a signal to be asserted on an input row wire. For networks of the size that we expect in the next five years, networks with perhaps 64 to 128 bit serial inputs, the time required to chain a grant signal through all the PC's of a column may be comparable to the time required to send bit serially the destination address of a packet along an input row. In such a case, it may be feasible to implement a crossbar network with output columns with daisy chained control wires.

Another possible technique for obtaining the mutual exclusion among the grid PC's connected to a column uses a N-input tree of arbitration units (Figure 26) for each column. Each arbitration unit has two incoming request lines from the two arbitration units below it and one incoming grant line from the

---

**Fig. 26. Arbitration Tree**



arbitration unit above it. When the arbitration unit receives a request on either or both of its incoming request lines, it will produce a request on its outgoing request line. When the arbitration unit receives a grant, either one or both of the arbitration units below it must have a pending request. If there is only one pending request, then the unit returns a grant for that request. Otherwise, the unit arbitrarily chooses one of the requests and returns a grant for that request. A grant is removed only after its associated request has been removed. The N-input arbitration tree associated with a given column has an incoming request line from and an outgoing grant line to each of the PC's in that column. The arbitration tree ensures that only one PC in a column can receive a grant at a given time. Unfortunately in the VLSI model, $\Omega(\log N)$ time is required for the request from a PC to receive a grant from the arbitration tree. Since we would like each input row of the network to be capable of transmitting packets at a high rate, the delay of the arbitration tree would seem to be a problem. However in a real implementation, the speed of an arbitration unit may be much greater than the speed of an output column driver, and the total delay of an arbitration tree may be comparable to the delay of an output column driver. It should be noted that if the N PC's of a column are in a straight line, the arbitration tree for the column may require area proportional to $N(\log N)$. The best layout that we know for a crossbar network with arbitration trees requires $\Omega(N^2 w (\log N) + N^2 w^2)$ area in the VLSI model.

Other techniques exist for maintaining mutual exclusion on the output columns. For example, each column wire can be viewed as a broadcast medium, and the mechanisms that have been developed for conflict resolution in broadcast networks can be applied. Mechanisms of this sort have been studied extensively in the literature [18]. Unfortunately, these mechanisms seem to require complex strategies for determining when a message should be retransmitted after a collision. Thus, these mechanisms seem to require a rather complex input PC for each input row.

The performance of a crossbar network depends, among other things, on the technique used for obtaining mutual exclusion on the output links. For uniform patterns of communication the crossbar

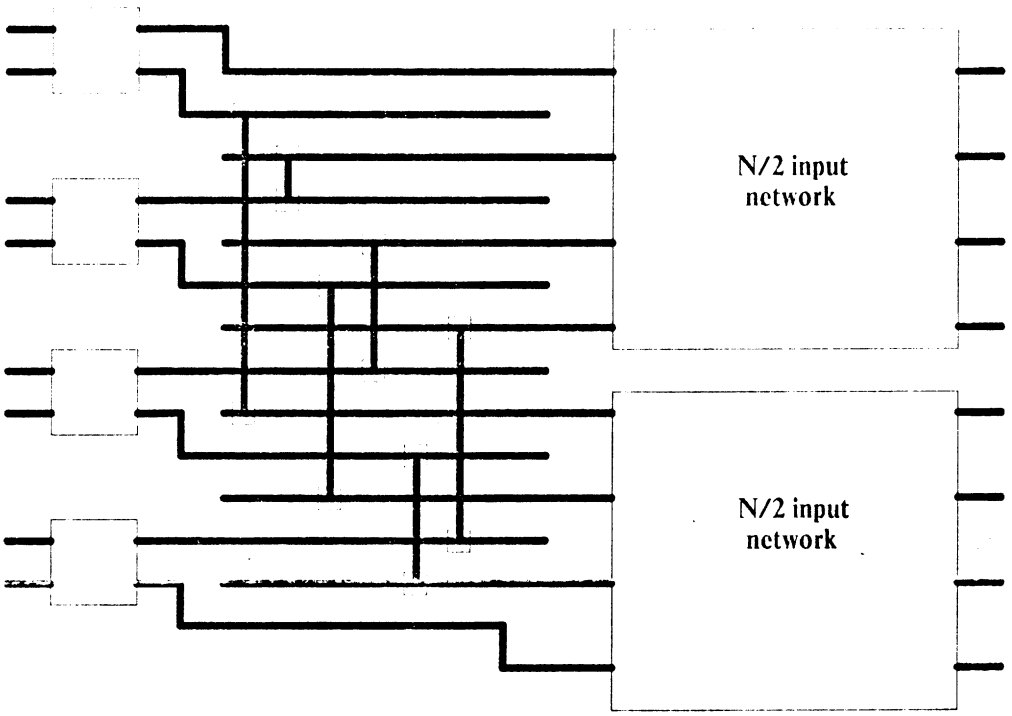
network can support a throughput of $\Omega(\frac{N}{\frac{p}{w}(\log w) + (\text{arbitration time})})$ packets per unit time where $w$ is the number of wires in each link and $p$ is the number of bits in each packet. The denominator corresponds to the time required to handle a single packet, which is the time required to gain control of the necessary output column plus the time required to transfer the packet. We assume that log $w$ time is required to transfer $w$ bits through a PC where one bit comes from each of the $w$ wires of a link.

*Indirect n-Cube Structure*

The indirect n-cube (InC) structure, which we described in the last chapter, can also be used to construct a single chip routing network capable of achieving high throughput in applications with uniform communication. This structure has a number of characteristics that make it interesting for VLSI. The InC network requires only two connections to each wire and thus avoids most of the implementation problems associated with the crossbar network. As we shall see below, the area required in the VLSI model to construct a N-input N-output InC routing network is $\Theta((Nw)^2)$ where $w$ is the number of wires in each data path.

One possible approach for laying out a N-input N-output InC network with $w$ width data paths in $\Theta((Nw)^2)$ area is shown in Figure 27. It should be noted that the figure shows the case for $w$ equal to one. Layouts for $w$ not equal to one can be obtained by replacing each wire in the figure with a group of $w$ wires. A N-input network is constructed from two (N/2)-input networks and N/2 two-input routers. We assume that at least two layers exist and thus crossovers are possible. The first output of the first router is connected to the first input of the first component (N/2)-input network. The last output of the last router is connected to the last input of the second component (N/2)-input network. Other connections between router outputs and inputs to the component routing networks are accomplished using (N-2)$w$ vertical wires. In particular, the $i$th set of $w$ vertical wires connects the second output of the $(i+1)/2$ router to the $(i+1)/2$ input of the second component network if $i$ is odd and connects the

Fig. 27. Layout for an InC Network




---

first output of the $(i+2)/2$ router to the $(i+2)/2$ input of the first component network if $i$ is even. Thus, the area required to implement a N-input network, A(N), is less than $2A(N/2) + c_1(Nw)^2$ for some constant $c_1$. This recurrence implies that A(N) is less than

$$2c_1(Nw)^2 + c_2N \tag{111}$$

for some constant $c_2$. This can be verified by substituting $2c_1(Nw)^2 + c_2N$ for A(N) into the inequality

$$A(N) \leq 2A(N/2) + c_1(Nw)^2 \tag{112}$$

We get

$$2c_1(Nw)^2 + c_2N \;?\; 4c_1(Nw/2)^2 + 2c_2(N/2) + c_1(Nw)^2 \tag{113}$$

or

$$2c_1(Nw)^2 + c_2N = 2c_1(Nw)^2 + c_2N \tag{114}$$

While this layout is certainly not the smallest possible, it does demonstrate that the InC network can be implemented in $\Theta((Nw)^2)$ area.

It should be noted that there is an imbalance between the wires and the PC's of the InC network. In the VLSI model, the InC network requires only $(N/2)(\log_2 N)$ PC's but it requires $\Theta((Nw)^2)$ area for its wires. In addition, the PC's of the InC network are complex. Thus, it seems that most layouts for the InC network will not be homogeneous but will instead contain separate areas for wires and PC's.

The throughput of indirect n-cube networks for uniform patterns of communication was discussed in the previous chapter. The strongest constraint that we studied in the previous chapter still allows a throughput of $\Omega(N)$ with the model of time of the previous chapter where N is the number of inputs. In the VLSI model, this would suggest a throughput of $\Omega(\frac{Nw}{p(\log w)})$ packets per unit time where $p$ is the number of bits in a packet and $w$ is the number of wires in a link. The $\frac{1}{\log w}$ factor comes from the fact that we assume $(\log w)$ time is required for a PC to process $w$ bits where one bit comes from each of the $w$ wires of a link.

While N-input InC networks with $w$-wire data paths and N-input crossbar networks with $w$-wire data paths both require $\Theta((Nw)^2)$ area in the VLSI model, there are some important differences between the two networks. The N-input InC network has only $(N/2)(\log_2 N)$ PC's, but the crossbar has $(N^2+2N)$ PC's. The nodes of the InC network are more complex than the nodes of the crossbar network. Each node of the InC network requires a buffer on each of its inputs and requires a control circuit that is more complex than the control circuit of a node of the crossbar network. As a result, more area is required in the VLSI model to implement a node for the InC network than to implement a node for the crossbar network.

*Other Network Structures*

There may well exist structures that are better suited for single chip high performance uniform communication networks than the InC structure and the crossbar structure. We have discussed the areas required in the VLSI model by the crossbar network and the InC network. There are problems with both the VLSI implementation of the crossbar network and the VLSI implementation of the InC network. As we have discussed earlier, each of the column wires of the crossbar network must be driven by a large number of PC's. The InC network has an imbalance between its wires and its PC's.

We have examined two other network structures. For N-input N-output networks, these structures require $\Theta(N^2)$ PC's and have two PC's connected to each wire. The restriction on the number of connections to a wire ensures that these structures do not have the problems associated with implementation of multiple drivers for a single wire. Further, these structures require roughly the same total area for PC's as they require for wires, and these structures have simple regular layouts.

One network structure that we have examined is the forest network. The N-input forest network is composed of 2N large trees. N of these trees are N-output switch trees, and the other N trees are N-input merge trees. A N-output switch tree is constructed from two (N/2)-output switch trees and a switch as shown in Figure 28. A switch is a device with one input and two outputs, and has the capacity to buffer some number of packets. The switch routes a packet according to the packet's destination tag. A two-output switch tree is simply a switch. Similarly, a N-input merge tree is constructed from two (N/2)-input merge trees and a merge as shown in Figure 29. A merge has two inputs and one output and some amount of internal buffering. The merge funnels all packets on its inputs to its outputs. Packets are output in the order in which they are accepted, and inputs are examined in a round robin fashion. A two-input merge tree is simply a merge.

Fig. 28. Switch Tree

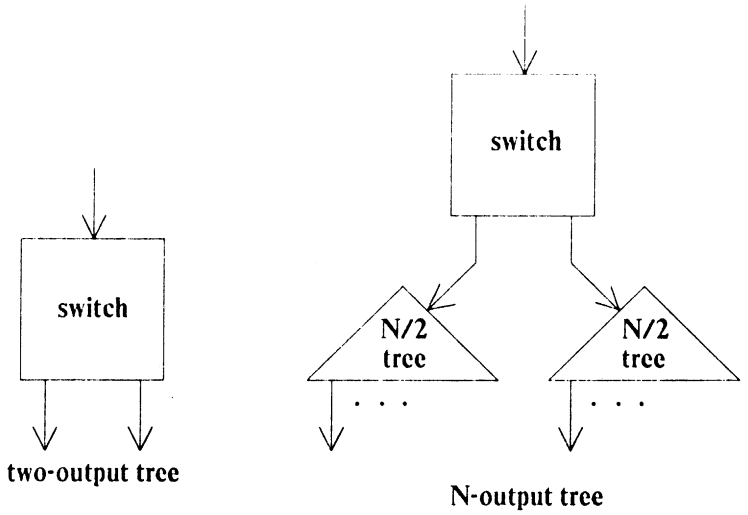
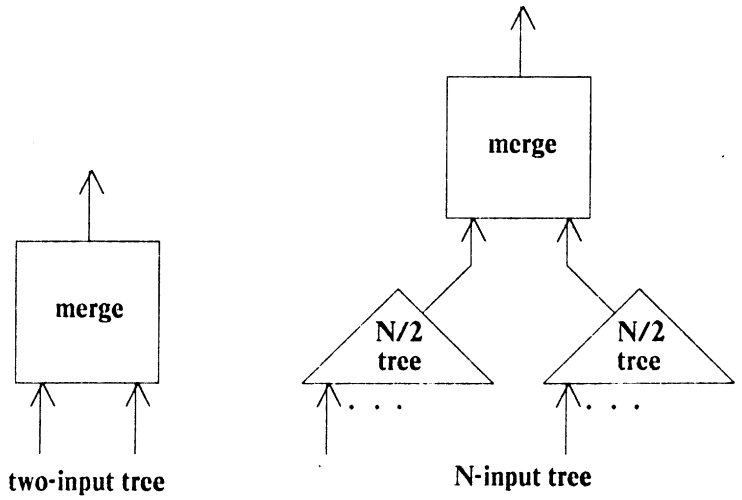



Fig. 29. Merge Tree




Each input of a N-input N-output forest network is connected to the input of a separate N-output switch tree, and each output of the forest network is connected to the output of a N-input merge tree. For

each $i$ and $j$ such that $1 \leq i \leq N$ and $1 \leq j \leq N$, the $j$th output of the switch tree associated with the $i$th network input is connected to the $i$th input of the merge tree associated with the $j$th network output. Thus, each switch tree sorts the packets from a given input according to their destination addresses, and each merge tree collects all packets destined for a given output.

Unfortunately, we have not found a good layout for the forest network. We have found a simple layout, Figures 30-32, that requires $\Theta((Nw \log N)^2)$ area in the VLSI model. We have not shown that this layout is the smallest possible. It should be noted that the figures show the case for $w$ equal to one. Layouts for $w$ not equal to one can be obtained by replacing each wire in the figures with a group of $w$ wires.

We have studied another network that is related to the crossbar network. We call this network the checkerboard network. The checkerboard network, shown in Figure 33, has a grid structure much like that of the crossbar network with a row corresponding to each input and a column corresponding to each

---

**Fig. 30. N Input N Output Forest Network**

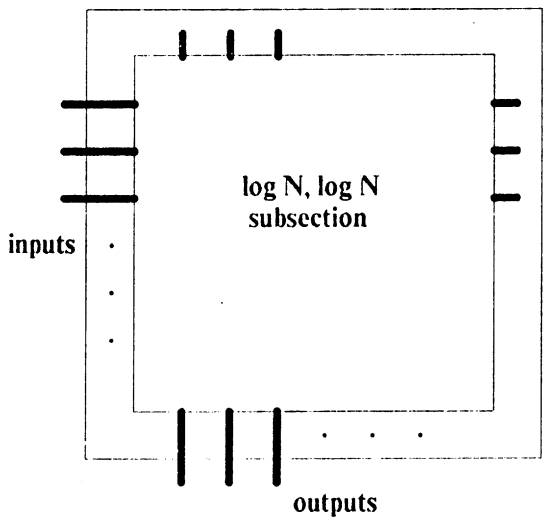
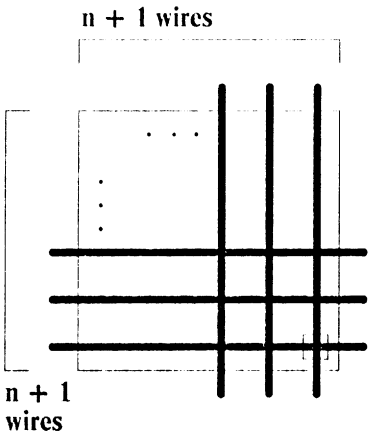


Fig. 31. 0,n Forest Subsection



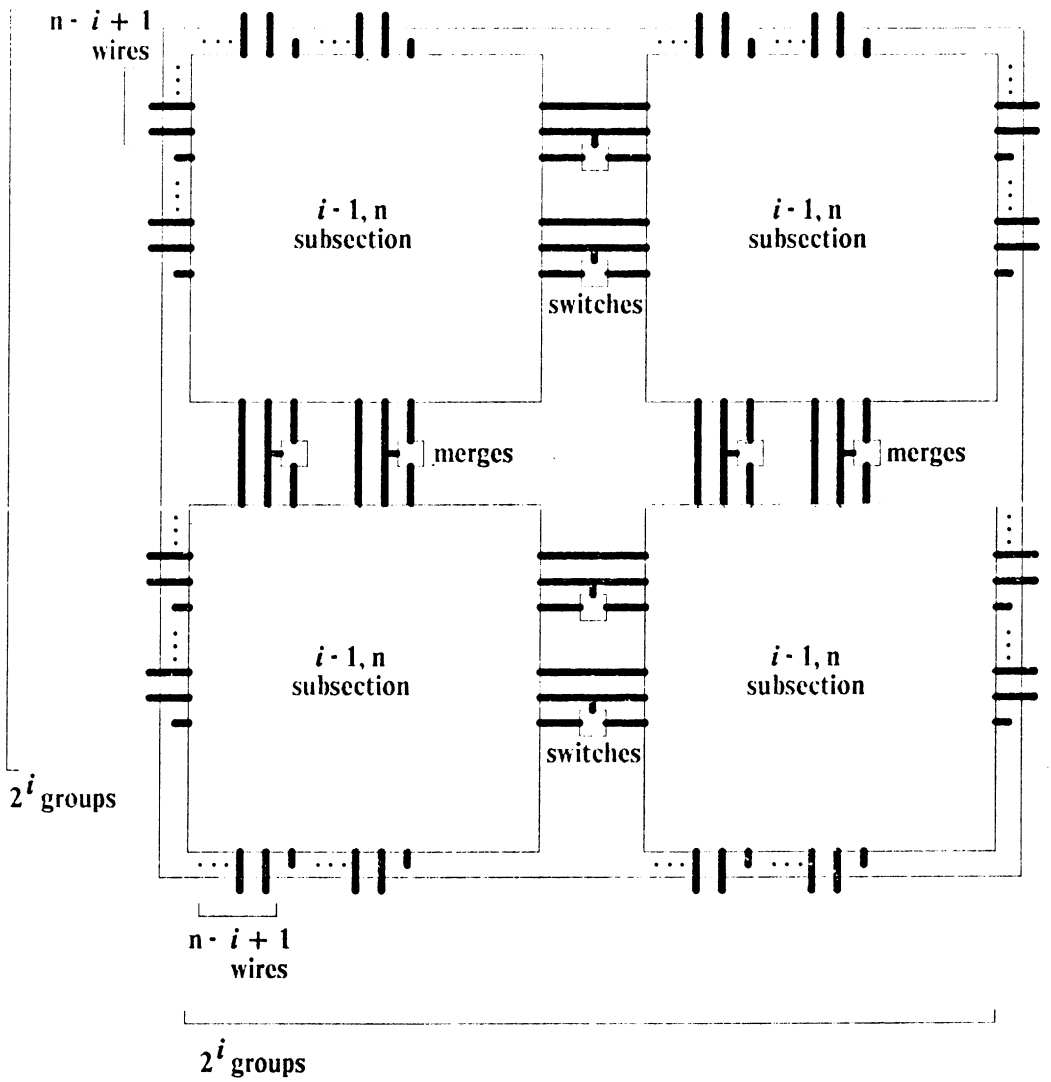

---

output. But unlike the crossbar network that uses one set of $w$ wires to connect the PC's in a given row, the N-input checkerboard network uses N sets of $w$ wires. (N-1) sets of $w$ wires connect adjacent grid PC's in the row, and one set of $w$ wires connects the input PC to the nearest grid PC in the row. Similarly, the N-input checkerboard network uses N sets of $w$ wires to connect the PC's in a given column. (N-1) sets connect adjacent grid PC's in the column, and one set connects the output PC to the nearest grid PC in the column. Thus, each wire in the checkerboard network is connected to only two PC's.

The grid PC's of the checkerboard network, unlike the grid PC's of the crossbar network, have the capacity to buffer some number of packets. A packet is transferred from an input to an output by passing it along a path of connected PC's between the two.

The N-input checkerboard network, like the N-input crossbar network, requires $(N^2 + 2N)$ PC's, and can be laid out using a total area which is proportional to $(Nw)^2$.

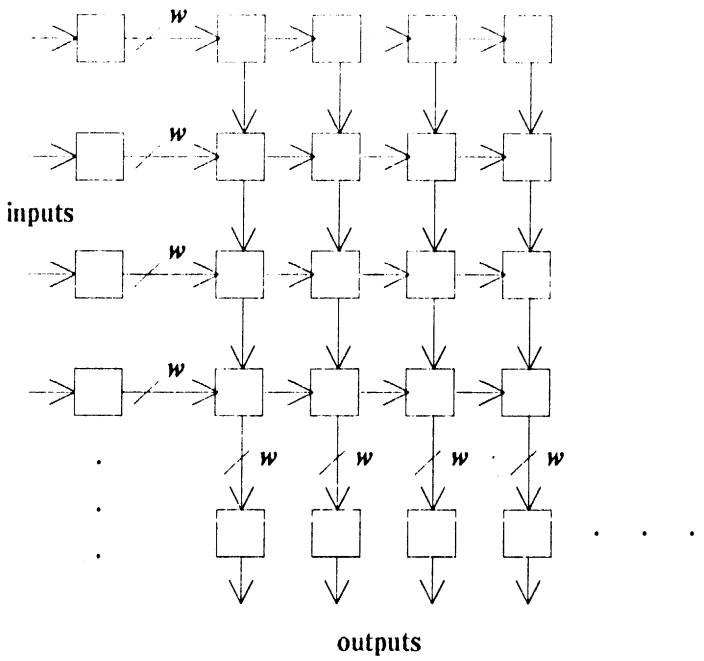
Fig. 32. $i$, n Forest Subsection ($i \leq$ n)




---

Although the throughput of the checkerboard network in uniform communication applications may be very good, the time required for each packet to be transmitted through the network is very long. In particular, if the inputs of a N-input checkerboard network are connected to model uniform

Fig. 33. Checkerboard Network




communication sources, then the expected number of PC's in the path through the network taken by a randomly arriving packet must be $\Omega(N)$. This implies that the average delay for a packet is $\Omega(N (\log w))$ units of time. This long average delay time is the primary weakness of the checkerboard network.

In the technology of the next five to ten years, the checkerboard network seems less interesting than the crossbar network. In this technology, it should be possible to build single chip crossbar networks and single chip checkerboard networks of approximately the same size. The two networks are capable of similar throughput for uniform communication applications, but the expected delay through the checkerboard network is much greater than the expected delay of the crossbar network. However, it is difficult to predict the changes that will occur in technology in the more distant future. Networks such as

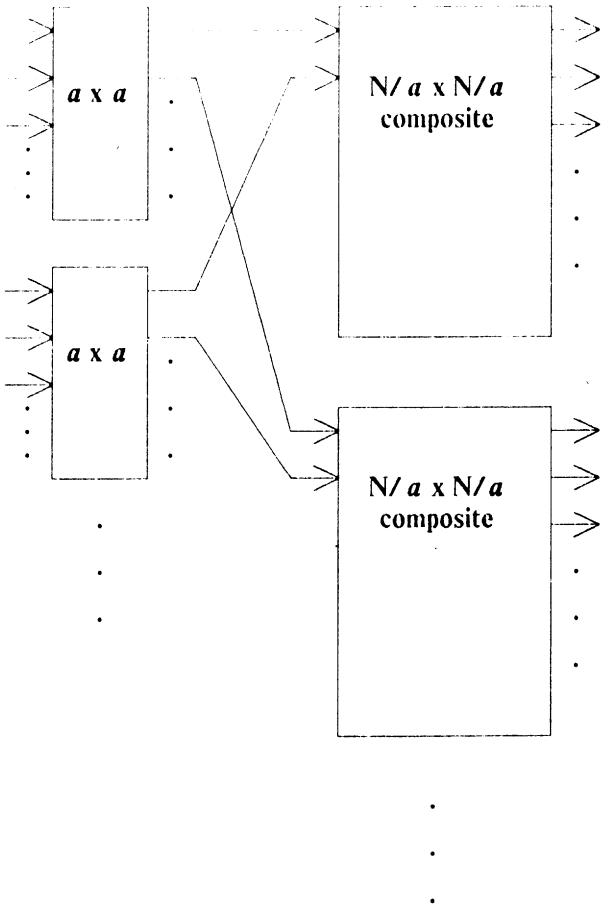
the checkerboard network may become more important as chip complexity increases and the feasibility of the multiply driven wires of the crossbar network decreases.

### 3.3.3 Multiple Chip Networks

If the network required by a particular system can not be implemented on a single chip then, obviously, the network must be implemented as an interconnection of several chips. One technique for constructing a large composite network involves the interconnection of several single chip networks. For the purpose of discussion, we refer to the component single chip networks of such a composite network as the subnetworks of that network. The subnetworks of a composite network are mounted on circuit boards and are interconnected by board wires. Thus, the issues involved in the interconnection of the subnetworks of a composite network are similar to the issues involved in the interconnection of the nodes of a network of the previous chapter. In particular, the length of the wires used to interconnect the subnetworks has less effect on the overall cost of the composite network than the number of such wires and the number of subnetworks. Thus, interconnection patterns similar to those used in the previous chapter may be appropriate for interconnecting the subnetworks of a composite network. In particular, an interconnection similar to the indirect n-cube structure seems interesting. This interconnection has the form shown in Figure 34. If $a$-input $a$-output subnetworks are used then the composite network contains $\log_a$ N stages of subnetworks. For the purpose of discussion, we number the stages from the network inputs to the network outputs with the stage connected to the network inputs being the zeroth stage. The $i$th stage is divided into $a^i$ groups of subnetworks. Each group of the $i$th stage has $a$ associated groups in the $(i+1)$st stage. The $j$th output of each subnetwork of a group of the $i$th stage is connected to the $j$th associated group of the $(i+1)$st stage.

If a composite network is constructed from single chip indirect n-cube networks in the manner described in the previous paragraph then the composite network is an indirect n-cube network. If a composite network is constructed from single chip crossbar networks in the manner described in the

Fig. 34. N x N Composite Network




previous paragraph then we expect the average throughput of the composite network to be at least as large as that of an indirect n-cube network of the same size.

## 3.4 Networks for Systems with Localized Communication

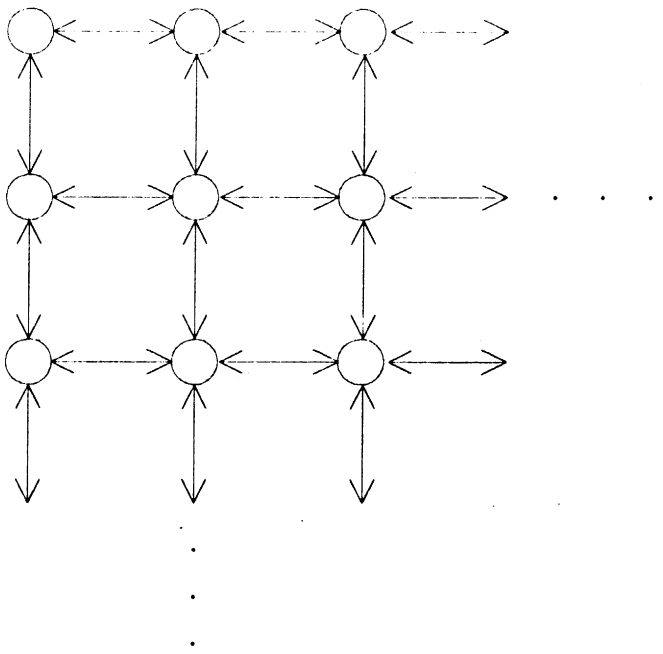
Many localized communication patterns can be supported by networks in VLSI that are substantially cheaper than a uniform communication network. However, the design of networks in VLSI for localized patterns of communication must take careful consideration of the cost of wires. In the technologies of the previous chapter, we determine the cost of a network by considering the number of wires, and the number and size of nodes of the network. In VLSI, the length of each wire must also be considered. If each source module of a system generates packets for only a constant number of destinations then the communication requirements of the system can be supported in the technologies of the previous chapter by a linear cost network. However, many such systems require networks with greater than linear cost in VLSI. For example, the "perfect shuffle" pattern [24] can be implemented using a number of wires proportional to N but it requires $\Theta((N/\log N)^2)$ wire area in the VLSI model [13].

In this section, we describe two obvious but important networks that can be implemented in VLSI in area proportional to their number of inputs. While we do not examine these networks in great detail, we do discuss some of the issues involved in their VLSI implementation.

The first of these networks is the grid network. We consider two dimensional grids as shown in Figure 35. Each grid PC is connected to the grid PC's adjacent to it. If each grid PC is connected to a network input and a network output then the number of PC's is obviously linear in the number of network inputs and outputs. The area required to implement the wires that interconnect a given PC to PC's adjacent to it is proportional to the area of the PC. Thus, the total area required for the wires of a grid network is linear in the number of network inputs.

In VLSI, one of the biggest issues in the implementation of linear cost networks in general and grid structured networks in particular may be the constraint placed by the limited number of input and output

Fig. 35. Grid Network



connections to a chip. Presently, chips with 64 connections are commonly available. In the next five to ten years, packaging technologies capable of handling chips with 100 to 200 connections should be common. However, it will probably be possible to implement a grid network with several thousand PC's on a single chip. This suggests that if a system of modules interconnected by a grid structured network is to be implemented in VLSI, it may be better to place some of the modules and a portion of the network on each chip than to place the modules and the network on separate chips. If some number of the modules and the portion of the grid network required to interconnect them are placed on a chip then the input and output requirements of the chip may be modest. The only signal wires that need to go off of the chip are those wires that connect the grid of the chip to the grids of other chips. These wires connect to the PC's along the perimeter of the grid of the chip.

The second network that we consider is the tree network. In such a network, the PC's of the network are connected in a tree as shown in Figure 36. The inputs and outputs of the network can be connected in at least two possible ways. One way is to connect each PC of the network to a network input and a network output. Another is to connect only the leaf PC's to the network inputs and outputs.

A tree network can be implemented in a small area in VLSI. A tree network can be laid out in $\Theta(N)$ area as shown in Figure 37. But there are some problems caused by the limited number of connections that can be made to a chip. As was the case for the grid network, it may be possible in VLSI to implement on a single chip a tree network with a large number of PC's but such an implementation could not have an off chip connection for each PC. This suggests that each chip in the VLSI implementation of a large tree structured system should contain both a portion of the network and the modules to be connected to that portion of the network. Unlike the case for a grid network where the perimeter PC's represent only a small fraction of the PC's of the network, the leaf PC's of a tree network

**Fig. 36. Tree Network**

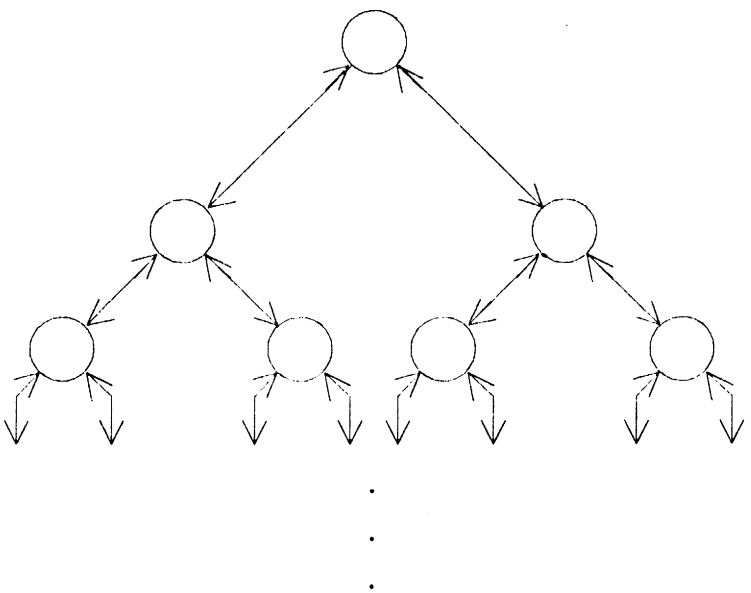
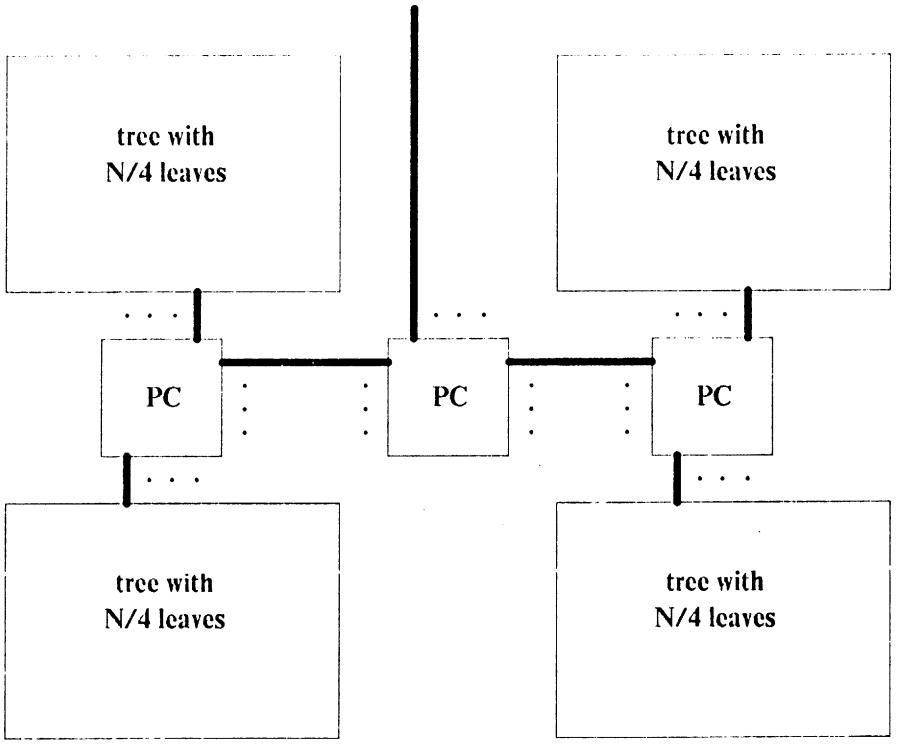


Fig. 37. Layout for a Tree with N Leaves




represent over half of the PC's of that network. Thus, while it may be possible on a single chip to implement a tree structured subsystem with a large number of very small modules, it is probably not possible to make an off chip connection for each leaf module of such a subsystem. As a result, it may be difficult to decompose a large tree structured system of very small modules into subsystems such that each subsystem requires the area of one chip and such that no subsystem requires more connections than the number of connections that can be made to a single chip. However, in the technology of the next five to ten years this may be a problem only for systems with very small modules. Unless the modules of a system are very small, it is likely that only a few dozen of the modules fit on a single chip and it should be possible to provide an off chip connection for each module of each chip.

# 4. Conclusion

## 4.1 Summary

In this thesis we examined the design of routing networks under two different sets of assumptions about the implementation technology. One set corresponds to present (1984) technology where only a small number of nodes can be implemented on a single integrated circuit. The other set corresponds to a VLSI technology where a large number of network nodes can be implemented on a single integrated circuit.

In present technology, we examined the design of routing networks for systems with uniform communication and we briefly examined the design of routing networks for systems with a few particular patterns of localized communication.

We showed that $\Omega$(N log N) nodes are required by any N-input N-output network capable of supporting an average throughput of $\Omega$(N) packets per unit time for our model of uniform communication.

We studied in detail one particular routing network, the indirect n-cube routing network, which seems well suited for uniform communication and requires O(N log N) nodes. We examined certain important characteristics of the operation of very large indirect n-cube networks and the effect of these characteristics on network performance.

We examined the buffering of packets in front of a slow router. Such buffering involves a tree of routers in front of the slow router. Our model suggests that expected number of packets buffered in front of the slow router is greater than $2^{(\frac{IN}{2(OUT-IN)(B+1)} - 2)} - 1$ where $IN$ is the rate at which packets are generated on each network input, $B$ is the size of the buffer on each input of each network node, and $OUT$ is the rate at which the slow router can accept packets.

We examined the effect that congestion in routers of a given stage of an indirect n-cube network has on the network's performance. We chose to study the effect of congestion in routers of the last stage since analysis of the last stage is somewhat easier than analysis of other stages. We examined the buffering caused by such congestion and the effect of such buffering on network throughput. Our study suggests that congestion in a single stage of routers does not place a severe constraint on the throughput of the network. Our study suggests that this type of congestion still allows the normalized throughput of the network (the total throughput of the network divided by the number of network inputs) to approach a non zero constant as the size of the network goes to infinity, and that even for modest buffer size this constant is not significantly less than the normalized throughput of a two-input two-output network.

We examined the effect of the interaction of routers of different stages on network performance. We studied the interaction of routers along a network path and we studied the interaction of routers in a tree of the network.

We studied the interaction of routers along a network path primarily by simulating a model of a network path. Our model reflects the interaction of routers along a randomly selected network path while ignoring the interaction between a router on the path and any router not on the path. Our study suggests a limit on the input rate of a randomly selected network path and thus implies a limit on the overall throughput of the network. This limit on network throughput is stronger than any of the other limits studied. However, this limit still allows normalized throughput to approach a non zero constant as network size approaches infinity.

We examined the interaction of routers in a tree of the network. We examined one particular type of interaction that occurs in trees of modest size, trees of less than 10 stages. Our study indicates that this type of interaction does not have an important effect on the overall throughput of the network but it does cause a few of the routers connected to the network inputs to be slow for a long period of time. For example, our study suggests that this type of interaction can cause the input rate for the slowest input of a

network with eight or nine stages to be less than half the expected input rate for a randomly selected input for a period of forty units of time.

We also briefly considered a factor that has an influence on the speed of the slow inputs of very large networks. This factor causes the slowest input router to require $\Omega(\frac{n}{\log n})$ time to accept $3B$ packets where $B$ is the buffer size and n is the depth of the network.

In summary, our work indicates that for present technology the indirect n-cube network is a good network for handling uniform communication. The strongest constraint on throughput that we studied still allows throughput to grow linearly with network size. However, our study also indicates that even in indirect n-cube networks of modest size some of the network inputs can be slow for a long period of time.

We briefly examined one obvious family of networks that are appropriate in present technology for some important localized communication patterns. This family includes grid structured networks and tree structured networks.

We also briefly examined the design of routing networks in VLSI. We described a model of VLSI based on assumptions about the characteristics of VLSI. The model reflects the fact that in VLSI the cost of a wire is proportional to its length.

We examined the design of uniform communication networks in VLSI. We showed that $\Omega((fN)^2)$ area is required in the VLSI model to implement any single chip N-input N-output routing network capable of supporting an average throughput of $fN$ packets per unit time for our model of uniform communication. We examined a few structures that are appropriate for implementing a single chip uniform communication network. These included a crossbar structure and an indirect n-cube structure. The crossbar network is probably the most attractive since it has a simple regular layout. However, the crossbar network requires long buses with a large number of drivers that must be arbitrated. The indirect

n-cube network does not require multiply driven buses, but it does require more complex network nodes and may require a more complex layout. We discussed a technique for interconnecting such single chip networks to form larger uniform communication networks.

We briefly examined the design in VLSI of networks for localized patterns of communication. We discussed the fact that some networks that can be implemented in present technology with a number of nodes proportional to their number of inputs require greater than linear wire cost in the VLSI model. We discussed two obvious but important networks that can be implemented in VLSI in area proportional to their number of inputs, the grid network and the tree network. We discussed the pin out problems of both networks for very dense VLSI. We concluded that in very dense VLSI the processing modules of a system using either network should be placed on the same chips as the modules of the network.

## 4.2 Suggestions for Further Work

There are many areas where further work could be done. Some of these are discussed below.

There are interesting open questions concerning the performance of large indirect n-cube networks for uniform communication. The strongest constraint that we studied still allows the throughput of an indirect n-cube network for our model of uniform communication to grow linearly with the size of the network. However, we did not prove such a linear growth. Such a proof appears to be difficult. It may be possible to obtain a proof if additional constraints are placed on the operation of the network. As was discussed in 2.2.3.1, an approach similar to Pippenger's may be effective.

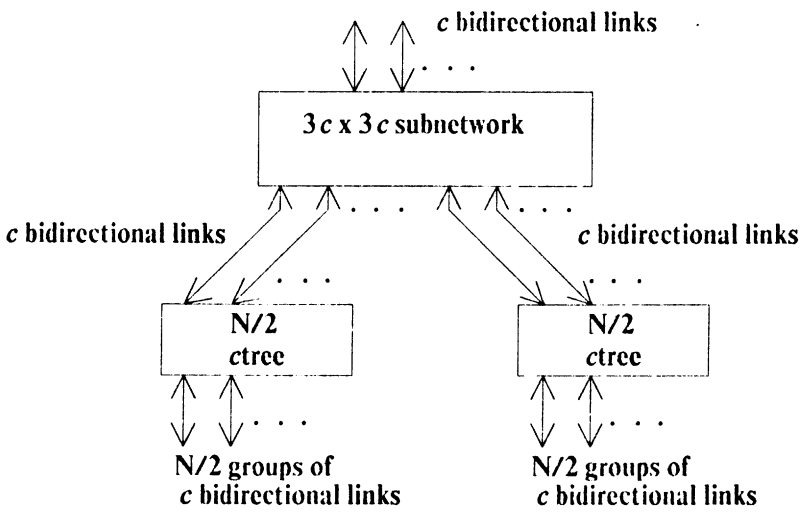
Clearly, more work can be done on the design of routing networks in present technology for localized patterns of communication. There are of course a wide variety of localized patterns of communication and it is probably not useful to try to examine all possible patterns. However, there may be interesting families of communication patterns that can be efficiently supported by families of

networks. For example, there may be an interesting family of communication patterns that can be supported by the family of networks described below. Each network of the family corresponds to a tree network of the same size. As is shown in Figure 38, the network has $c$ links for each link of the tree network and a $3c$ x $3c$ subnetwork for each node of the tree network where $c$ is a constant. The subnetwork is some type of $3c$-input $3c$-output routing network. Different members of the family have different values of $c$. A network of this family may be able to handle $c$ times as much traffic between distant nodes as the corresponding tree network. Other families of networks related to the tree network may be able to support interesting families of communication patterns. For example, Leiserson [17] has studied a more sophisticated family of networks called fat trees that seems to be able to support a wide class of communication patterns.

Similarly, more work can be done on the design of single chip routing networks in VLSI. It would seem that low level implementation issues will continue for some time to be important in the design of

---

**Fig. 38. N *c*tree**



single chip routing networks. Thus, a more detailed look at the implementation of crossbar networks, indirect n-cube networks, and other potential networks is needed. In order to fairly evaluate data path sizes for the crossbar network and the indirect n-cube network, and arbitration schemes for the crossbar network, it may be useful to examine tentative chip layouts.

One important issue that has not been considered in this thesis is the issue of real time fault detection and fault masking in routing networks. Some related work has been done elsewhere [2, 19], but more is needed. Detection of some faults can be accomplished by schemes that use check fields in each packet. Some types of fault masking can be accomplished if multiple paths exist between each source and each destination. Such paths can easily be introduced in a network such as the indirect n-cube network by adding one or more additional stages.

## References


1. Abelson, H., Andrea, P., "Information Transfer and Area-Time Tradeoffs for VLSI Multiplication", Communications of the ACM, Vol. 23, No. 1, Jan. 1980.

2. Adams, G.B., Siegel, H.J., "The Extra Stage Cube: a Fault-Tolerant Interconnection Network for Supersystems", IEEE Trans. on Computers, Vol. C-31, No. 5, May 1982.

3. Ajtai, M., Komlos, J., Szemeredi, E., "An O(n log n) Sorting Network", Proc. of the 15th Annual ACM Symposium on Theory of Computing, April 1983.

4. Batcher, K.E., "Sorting Networks and their Applications", Proc. of the 1968 Spring Joint Computer Conference, 1968.

5. Boughton, G.A., Routing Networks in Packet Communication Architecture, Dept. of Electrical Engineering and Computer Science, M.I.T., S.M. Thesis, June 1978.

6. Dennis, J.B., "Packet Communication Architecture", Sagamore Computer Conference on Parallel Processing, Aug. 1975.

7. Dennis, J.B., Leung, C.K.C., Misunas, D.P., A Highly Parallel Processor Using a Data Flow Machine Language, Laboratory for Computer Science, M.I.T., CSG Memo 134-1, June 1979.

8. Dias, D.M., Jump, J.R., "Analysis and Simulation of Buffered Delta Networks", IEEE Trans. on Computers, Vol. C-30, No. 4, April 1981.

9. Dias, D.M., personal communication, March 1984.

10. Goke, R., Lipovski, G.J., "Banyan Networks for Partitioning on Multiprocessor Systems", Proc. of the First Annual Symposium on Computer Architecture, 1973.

11. Gold, B., Rader, C.M., Digital Processing of Signals, McGraw-Hill, 1969.

12. Kleinrock, L., Queueing Systems, Volume 1, Wiley, 1975.

13. Kleitman, D., Leighton, F.T., Lepley, M., Miller, G., "New Layouts for the Shuffle-Exchange Graph", Proc. of the 13th Annual ACM Symposium on Theory of Computing, May 1981.

14. Lang, T., Stone, H.S., "A Shuffle-Exchange Network with Simplified Control", IEEE Trans. on Computers, Vol. C-25, No. 1, Jan. 1976.

15. Lang, T., "Interconnections Between Processors and Memory Modules Using the Shuffle-Exchange Network", IEEE Trans. on Computers, Vol. C-26, No. 5, May 1977.

16. Lawrie, D.H., "Access and Alignment of Data in an Array Processor", IEEE Trans. on Computers, Vol. C-24, No. 12, Dec. 1975.

17. Leiserson, C., personal communication, Aug. 1984.

18. Mosely, J., An Efficient Contention Regulation Algorithm for Multiple Access Channels, Dept. of Electrical Engineering and Computer Science, M.I.T., S.M. Thesis, May 1979.

19. Padmanabhan, K., Lawrie, D.H., "A Class of Redundant Path Mulitstage Interconnection Networks", IEEE Trans. on Computers, Vol. C-32, No. 12, Dec. 1983.

20. Patel, J.H, "Processor-Memory Interconnections for Multiprocessors", Proc. of the 6th Annual Symposium on Computer Architecture, 1979.

21. Pease, M.C., "The Indirect Binary n-Cube Microprocessor Array", IEEE Trans. on Computers, Vol. C-26, No. 5, May 1977.

22. Pippenger, N., personal communication, June 1984.

23. Preparata, F.P., Vuillemin, J., "The Cube-Connected Cycles: A Versatile Network for Parallel Computation", Com. of the ACM, Vol. 24, No. 5, May 1981.

24. Stone, H.S., "Parallel Processing with the Perfect Shuffle", IEEE Trans. on Computers, Vol. C-20, No. 2, Feb. 1971.

25. Thompson, C.D., "Area-Time Complexity for VLSI", Proc. of the 11th Annual ACM Symposium on Theory of Computing, May 1979.

26. Tripathi, A.R., Lipovski, G.J., "Packet Switching in Banyan Networks", Proc. of the 6th Annual Symposium on Computer Architecture, 1979.

27. Upfal, E., "Efficient Schemes for Parallel Communication", Proc. of the Symposium on Principles of Distributed Computing, Aug. 1982.

28. Valiant, L.G., Brebner, G.J., "Universal Schemes for Parallel Communication", Proc. of the 13th Annual ACM Symposium on Theory of Computing, May 1981.

# Scanning Agent Identification Target

Scanning of this document was supported in part by the **Corporation for National Research Initiatives,** using funds from the **Advanced Research Projects Agency** of the **United states Government** under Grant: **MDA972-92-J1029.**

The scanning agent for this project was the **Document Services** department of the **M.I.T Libraries.** Technical support for this project was also provided by the **M.I.T. Laboratory for Computer Sciences.**

**Scanned**

**Date:** 4/27/1995

**M.I.T. Libraries**
**Document Services**

**CS-TR Scanning Project**
**Document Control Form**

Date : 4/21/95

**Report #** LCS-TR-341

Each of the following should be identified by a checkmark:
Originating Department:

☐ Artificial Intellegence Laboratory (AI)
☒ Laboratory for Computer Science (LCS)

Document Type:

☒ Technical Report (TR) ☐ Technical Memo (TM)
☐ Other:_____

## Document Information

Number of pages: 123 (130-IMAGES)
Not to include DOD forms, printer intstructions, etc... original pages only.

Originals are:
☒ Single-sided or
☐ Double-sided

Intended to be printed as :
☐ Single-sided or
☒ Double-sided

Print type:
☐ Typewriter ☐ Offset Press ☐ Laser Print
☐ InkJet Printer ☒ Unknown ☐ Other:_____

Check each if included with document:

☐ DOD Form ☐ Funding Agent Form ☒ Cover Page
☒ Spine ☒ Printers Notes ☐ Photo negatives
☐ Other: _____

Page Data:

Blank Pages(by page number):_____

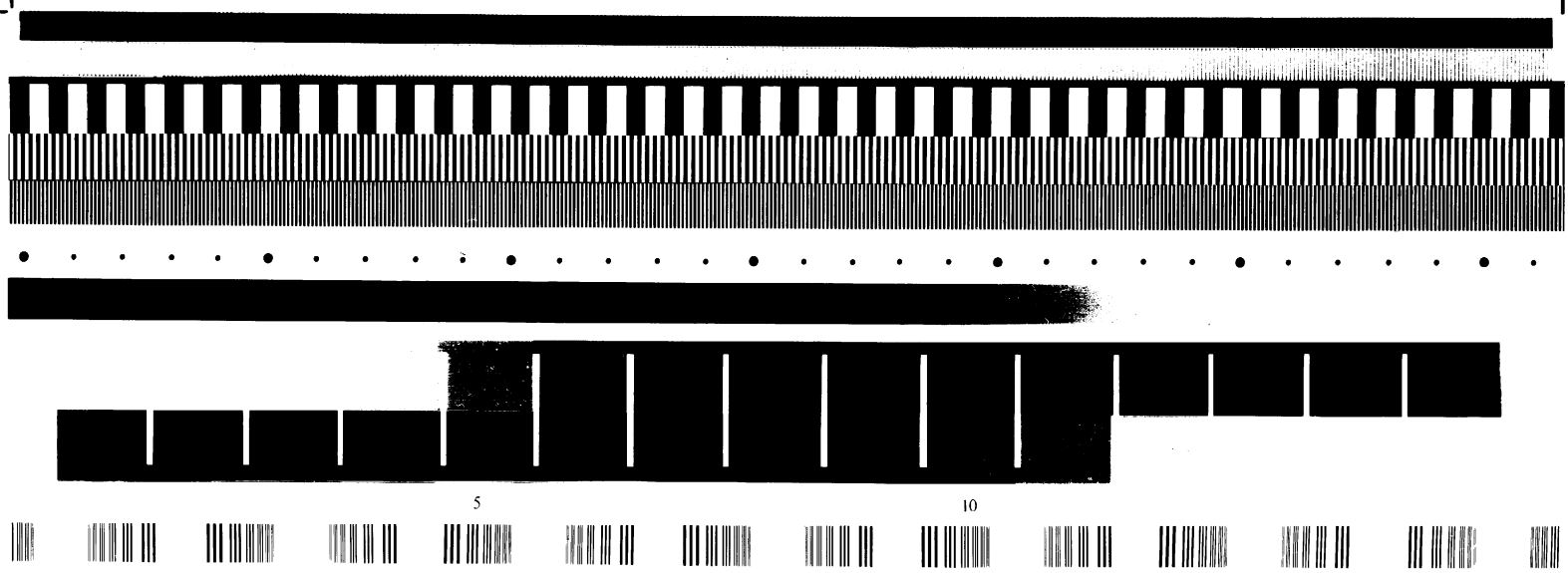
Photographs/Tonal Material (by page number):_____

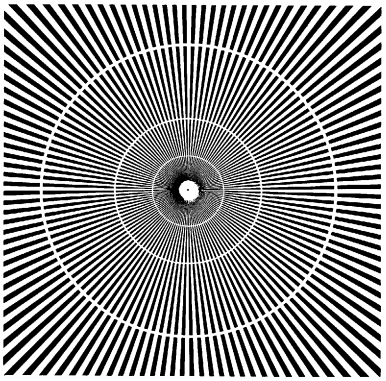
Other (note description/page number):

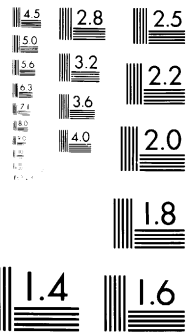
| Description : | Page Number: |
|---|---|
| IMAGE MAP : (1) UNNUMBERED TITLE PAGE | |
| (2-123) PAGES #'ED 2-123 | |
| (124-127) SCANCONTROL, COVER, SPINE, PRINTER'S NOTES | |
| (128-130) TRGTS (3) | |

Scanning Agent Signoff:
Date Received: 4/21/95 Date Scanned: 4/27/95 Date Returned: 5/4/95

Scanning Agent Signature: Michael W. Cook



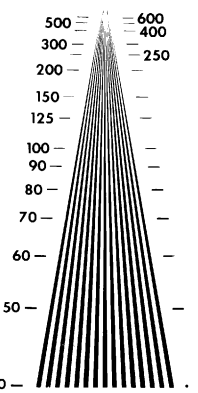



NMA MICROFONT QJKLPYZ
6BS12GH5D4X7U3W8V9E
PQR45DE9UV670FG8STHIJNOWXABYZ
3KLM12C

ABCDEFGHIJKLMNOPQRSTUVW
XYZabcdefghijklmnopqrst
uvwxyz0123456789 OCR-B

ABCDEFGHIJKLMNOPQRSTUV
WXYZabcdefghijklmnopqr
stuvwxyz1234567890PICA

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
1234567890 Elite



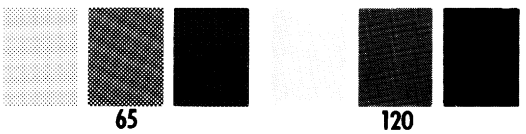
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
1234567890 Spartan Medium 6 pt

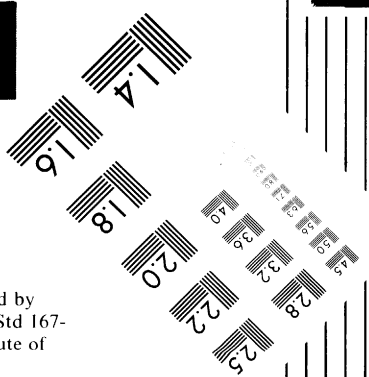
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
1234567890 Spartan Medium 4 pt

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
1234567890 Spartan Medium 8 pt

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
1234567890 Spartan Medium 10 pt

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
1234567890 Spartan Medium 12 pt




# IEEE Std 167A-1987

## FACSIMILE TEST CHART

Prepared by the IEEE Facsimile Subcommittee and printed by Eastman Kodak Company. Use in accordance with IEEE Std 167-1966, Test Procedure for Facsimile. Copyright 1987, Institute of Electrical and Electronics Engineers.

0 1 2 3 4 5 6
123456

0 1 2 3 4 5 6

0 1 2 3 4 5 6
654321

A4 Page 6543210

# AIIM SCANNER TEST CHART # 2

## Spectra

4 PT ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz;:",./?$0123456789
6 PT ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz;:",./?$0123456789
8 PT ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz;:",./?$0123456789
10 PT ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz;:",./?$0123456789

## Times Roman

4 PT ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz;:",./?$0123456789
6 PT ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz;:",./?$0123456789
8 PT ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz;:",./?$0123456789
10 PT ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz;:",./?$0123456789

## Century Schoolbook Bold

4 PT ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz;:",./?$0123456789
6 PT ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz;:",./?$0123456789
8 PT ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz;:",./?$0123456789
10 PT ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz;:",./?$0123456789

## News Gothic Bold Reversed

4 PT ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz;:",./?$0123456789
6 PT ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz;:",./?$0123456789
8 PT ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz;:",./?$0123456789
10 PT ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz;:",./?$0123456789

## *Bodoni Italic*

*4 PT ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz;:",./?$0123456789*
*6 PT ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz;:",./?$0123456789*
*8 PT ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz;:",./?$0123456789*
*10 PT ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz;:",./?$0123456789*

## Greek and Math Symbols

4 PT ΑΒΓΔΕΞΘΗΙΚΛΜΝΟΠΦΡΣΤΥΩΧΨΖαβγδεξθηικλμνοπφρστυωχψζ≧∓",./≦±=≠°><⋗⋖⋗⋖≡
6 PT ΑΒΓΔΕΞΘΗΙΚΛΜΝΟΠΦΡΣΤΥΩΧΨΖαβγδεξθηικλμνοπφρστυωχψζ≧∓",./≦±=≠°><⋗⋖⋗⋖≡
8 PT ΑΒΓΔΕΞΘΗΙΚΛΜΝΟΠΦΡΣΤΥΩΧΨΖαβγδεξθηικλμνοπφρστυωχψζ≧∓",./≦±=≠°><⋗⋖⋗⋖≡
10 PT ΑΒΓΔΕΞΘΗΙΚΛΜΝΟΠΦΡΣΤΥΩΧΨΖαβγδεξθηικλμνοπφρστυωχψζ≧∓",./≦±=≠°><⋗⋖⋗⋖≡

White

Black

Isolated Characters

| | | | | | |
|---|---|---|---|---|---|
| e | m | 1 | 2 | 3 | a |
| 4 | 5 | 6 | 7 | o | ° |
| 8 | 9 | 0 | h | I | B |

123456

654321

A4 Page 6543210

MESH HALFTONE WEDGES

65

85

100

110

133

150

A4 Page 6543210

0123456

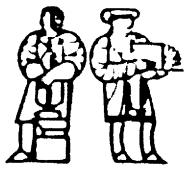
6 5 4 3 2 1 0

6 5 4 3 2 1 0

654321

6 5 4 3 2 1 0

MASSACHUSETTS INSTITUTE OF TECHNOLOGY



Laboratory for Computer Science

# TECHNICAL MEMORANDA — TECHNICAL REPORT — PUBLICATION SHEET

NO. ASSIGNED: TM- TR-341 NUMBER OF PAGES: 123

REPORT TITLE: Routing Networks for Packet Communication Systems

AUTHOR(s): George Andrew Boughton

LCS/GROUP: CSG

THESIS: B.S. M.S. E.E. Ph.D.

---

KEY WORDS REQUIRED ON REPORT

*SUPERVISOR: Do you wish extra copies? No. Yes Amt. 10

The enclosed report has been proofread by the author and approved for publication in this form.

SIGNATURE OF THE AUTHOR: George Andrew Boughton

Date: June 7 1985

This report has been read and approved for publication as a Technical Report or Technical Memoranda.

SIGNATURE OF THE GROUP LEADER: [signature]

Date: 7 June 85

DO YOU RECOMMEND THIS FOR AN M.I.T. PRESS MONOGRAPH: No Yes

Signature of the Director