PACKET SWITCHING IN FUTURE FIBER-OPTIC WIDE-AREA NETWORKS

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL ENGINEERING
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

By
Zygmunt Haas
May 1988

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

David R. Cheriton

Prof. David R. Cheriton
(Computer Science Dept.)
(Principal Adviser)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

Joseph W. Goodman

Prof. Joseph W. Goodman
(Electrical Engineering Dept.)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

R M Gray

Prof. Robert M. Gray
(Electrical Engineering Dept.)

Approved for the University Committee on Graduate Studies:

Dean of Graduate Studies

Preface

Abstract

The potential of computer communication is, at present, severely handicapped by the poor performance of wide-area networks. The geographically dispersed clusters of machines operated by military, commercial, government, and research organizations are information and resource "islands" that limit the efficiency, capability, and responsiveness of these organizations. Moreover, distributed environments and more performance-demanding applications will characterize future wide-area communication. Consequently, wide-area networks that are matched in delay and bandwidth to the performance of local area and metropolitan area networks are required to solve today's and tomorrow's communication needs.

Optical fiber provides a long distance channel technology that makes this goal feasible. As a consequence, communication networks are evolving towards a new physical layer. Fibers replace twisted pairs and coaxial cables, and bring with them the benefit of very high bandwidth, two or three orders of magnitude higher than that of the existing networks. However, this benefit can easily be wasted if an inappropriate switching technique is used for these high-bandwidth networks.

Packet-switching and circuit-switching are two possible techniques to be used in high-performance communication. Circuit-switching is characterized by static allocation of communication resources. Packet-switching, on the other hand, exercises dynamic allocation of the communication channel.

At a first glance it may seem that, because of the very large bandwidth available in fiber-optic networks, circuit-switching would be the preferred switching technique.

I claim, however, that circuit-switching is a poor choice for high-speed, distributed-environment computer traffic, and I make the case for packet-switching. The challenge is, consequently, to provide packet-switching node that handle high data rates with minimal delay. In particular, the switching node must minimize the packet-routing decision delay. Also, it must be able to make switching decisions at the packet rate, which is determined by the traffic on the incoming channels.

In a photonic network the signal propagates through the network as light, is amplified, possibly regenerated, and switched as light, without being converted to an electrical signal at any stage of its path through the network. One step further, which is considered in this work, consists of implementation of the switching nodes based on photonic devices only (i.e., without any electronic devices). Such an implementation possesses some salient advantages over the conventional electronic implementation such as immunity to electro-magnetic interference, increased speed and bandwidth, higher security, lower design complexity, and increased design flexibility. However, because of the prediction that optical RAM will, at least in the near future, remain expensive, the conventional electronic architecture of a switch needs to be replaced by an architecture suitable for photonic implementation. *Blazenet*, introduced in this work, is a packet-switching network based on optical fiber and high-performance switching nodes. *Blazenet* provides an appropriate solution for photonically implementable, wide-area switching node architecture. Using the *Blazenet* design, the data path of the switching node can be fully photonically implemented with today's state-of-the-art technology.

However, the importance of *Blazenet* extends beyond providing another communication network design. The *Blazenet* concept demonstrates the feasibility of packet-switching in high speed networks. In other words, the *Blazenet* design shows that it is not necessary to resort to circuit-switching to handle the data rates made possible by optical fiber. The network is presented here as a wide-area network. I see it as a backbone network whose nodes are gateways to other networks. However, the concept of *Blazenet* is easily applied to smaller networks and, with some constraints on maximum packet size, even to local-area networks.

Acknowledgements

I would like to thank my doctoral committee consisting of Prof. David R. Cheriton, Prof. Joseph W. Goodman, Prof. Robert M. Gray, and Prof. Donald L. Igelhart. I am particularly grateful to my thesis advisor, Prof. David Cheriton, for his continual support and encouragement throughout this work. I also owe gratitude to the members of the Distributed Systems Group at Stanford: Marc Abrams, Per Bothner, Pat Boyle, Steve Deering, Ross Finlayson, Steve Goldberg, Hendrik Goosen, Carey Gray, Kieran Harty, Bruce Hitson, Hemant Kanakia, Tony Mason, Joseph Pallas, Ed Sznyter, Michael Stumm, Marvin Theimer, Cary Williams, and Michael Wolf for providing me with a stimulating atmosphere for this research.

Special thanks go to Dr. Itzhak Birk for his enlightened suggestions and criticism.

This work was sponsored in part by the Defense Advanced Research Projects Agency under contract N00039-86-K-0431, by Digital Equipment Corporation, by AT&T Information Systems, and by Bell Northern Research.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

The main motivation behind this research is to design a wide-area network (WAN) whose level of performance is comparative to that of a local-area network (LAN), or in other words:

To provide LAN-grade of performance in WAN.

This envisioned high-performance wide-area network is a backbone network that connects many local-area networks. Jobs with appropriate qualifiers are submitted by the users of these local-area networks. These qualifiers consist, for example, of the required machine speed, load, or available software. A job can be executed on a machine residing on other local-area network than the one the user resides on. The user, however, is not supposed to notice whether the job is executed on his machine, on a machine in his local environment, or on a machine physically located a few thousand miles away. In this way, all the resources connected by this high-performance wide-area network turn into one big (distributed) computing environment, an environment that can be spread over hundreds or thousands of miles. Nevertheless, the user gets the impression that all the resources are located in his immediate vicinity. Thus the idea is to adopt the approach of local distributed environments and extend it to wide-area distributed environment.

The wide-area networks of today cannot support such performance; the delays associated with transmission of a message over the existing networks and the throughput of these networks make them totally unsuitable for future communication requirements. Consequently, the need for high-performance wide-area networks has been expressed ([4, 5, 6, 7, 8]).

This chapter introduces some terms that are extensively used throughout this work: high-performance communication, switching techniques, fiber-optic communication, and photonic switching and processing.

1.2 High-performance communication

High-performance (H-P) communication is essential for development of future computing environments, which will be characterized by distributed processing and distributed information. By high-performance communication I mean low-delay, multipoint, *on-demand* delivery of large amounts of data (on the order of Mbits). I will discuss these concepts in this chapter.

The low-delay requirement is necessary to achieve the goal of high-performance communication: the local-area-like response over a wide-area network. However, the even more basic reason for the need for low delay is that distributed systems performance is vulnerable if long latency is associated with a transaction. The classical example is atomic transactions. Consider, for example, an atomic operation performed on different, physically remote, machines. In particular consider a distributed database. An update in the database needs, in general, to lock access to some information residing on more than one machine. If the locking operation of data in distant machines is slow because of the network speed, the concurrency of other transactions will be affected, and the performance of the whole system degraded. This is especially significant in a distributed environment, in which a transaction may involve many machines. (Moreover, if the network is slow, some of the timers associated with the transaction may expire, and, as a result, additional overhead and delay may be required to determine the status of the transaction.) Thus, the low-delay requirement is crucial if high-performance communication is to be realized. Low-delay is also required for real-time traffic such as voice and interactive video (for example, delay on the order of 50 msec is required for voice communication), and for remote control operation (for example, remote robotics or interactive games).

It should be stressed that high-capacity (large bandwidth) systems are not sufficient to provide high-performance communication, because such systems are not necessarily low-delay. For example, high-capacity satellite communication, in spite of the fact that it has large bandwidth, suffers from long delays.

Multi-point communication is the ability of the network to set up a connection between more than two entities in such a way that a single transmission initiated by any member of the group is automatically received by all the members of the group. This requirement, though not essential, is very helpful in some environments. Multi-user conferencing, multicast queries (sending the same query to multiple recipients by transmitting it only once), and timely update of network status are a few examples where multi-point communication is beneficial. Also, broadcast, which is a special instant of multicast, is essential in some situation such as an environment with mobile hosts. Multi-point communication is not a basic feature of network design, and can be implemented by higher layers (like the transport layer, for example). It is, however, in my opinion an extremely useful feature, and should be provided as a service by the network itself. The reason is that multi-point communication is also used for managing the network (coping with emergency situations, regulating load, distributing the network status, etc.), thus providing improved performance on the network level.

The on-demand characteristic refers to the unpredictable (random) manner in which information is required to be transmitted over the network. The unpredictability of the information transmission pattern in high-performance networks arises mainly from the types of applications expected to use these networks (workstations requiring a rare access to remote storage and using the transactional communication model, for example), as well as from the fact that these networks are supposed to support many users with different requirements, users residing on different and heterogeneous local networks. Another reason for the randomness of the traffic in high-performance networks is the use of various data compression algorithms whose purpose is to reduce the bandwidth requirements. For example, a digitized video signal that undergoes data compression may turn into traffic with constantly changing bandwidth requirements.

Somewhat connected to the on-demand characteristic, is the increased *burstiness* of the traffic on high-performance networks. The source of this characteristic is the high-speed transmission associated with such networks. The increase in the communication bandwidth "shrinks" the transmission time of information, resulting in increased traffic burstiness ([9]). Furthermore, some of the traffic that appears to be of the *stream* type on low-speed links, becomes more of the bursty type on high-speed links. For example, packetized voice and video (possibly compressed) looks like short bursts when transmitted over a multi-gigabit-per-second link. Because of the aggregation of many such traffic patterns on a single high-speed channel, the resulting traffic appears to be a highly random process.

The characteristic of large data size is determined by the nature of the applications that are expected to use high-performance networking. For example, it is necessary to transmit 3 Mbytes to fill a single color monitor screen of 1000 × 1000 points using the 24 bit/pixel RGB representation (as in interactive video and data application, for example) . (The corresponding transmission rate is, in this case, 1440 Mbit/sec.) As many as 200 Mbytes are needed to store a single-page, high-resolution color poster; 500 Mbytes are needed for a few page color brochure ([10]). HDTV (High-Definition TV), if uncompressed, may require as much as 100 – 200 Mbyte/sec transmission speed ([11]). A single X-ray takes about 30 Mbytes. Advanced medical imaging techniques (CT, MRI, etc.) produce about a dozen of images during a single test ([10]), which need to be catalogued and latter retrieved, possibly many times, for reference and comparison. Similar examples illustrating the large required data volumes can be found in the fields of geology, astronomy, mechanical engineering, etc. Also, in spite of the fact that most of the above applications can use lossy data compression techniques, possibly reducing the required bandwidth by factors of 10 to 1000 ([12, 4]), the resulting data volumes remain large. (Moreover, medical imaging data can be compressed with techniques that introduce only limited amount of noise and distortion, and hence have a much lower data compression ratio.)

Another source of the large data size in high-performance communication is the trend, as communication becomes cheaper, to send a piece of information each time it is needed and not to permanently keep it stored. Some examples are diskless workstations, font files attached to every document, and context loading for AI environments. This trend to rely on communication, rather than on stored information, has several motivations. In some cases the justification is the difference between hardware and software implementations of a different equipment. Thus, for instance, instead of assuming some standard fonts implementation, a document can be accompanied by the required font definition and not rely on the font definition in the destination printer. In other cases, as in diskless workstations for example, lowering the price of the hardware is the driving force. Yet another reason might be the fact that as networks become a common, cheap, and fast resource it might be quicker to copy a piece of software from the main memory of another machine, than to get it from a magnetic tape or an optical disc. Whatever the reason is, the fact is that the amount of data involved in future transactions is, on average, expected to increase.

Somewhat connected to the trend of transmitting the data rather the storing it, is the trend of remote program execution. In this case, instead of using the local computing power, a program with its data is transmitted over the network to a remote facility to be executed there. (Also, programs can migrate throughout a network in the middle of their execution.) The reasons for such an action are the specific attributes needed for the program execution or simply better load condition on the remote machine. In any case, such a transfer will, in general, be associated with large data volumes, since the code (source or object) and possibly the data will be of large size.

Also, as communication becomes cheaper and high-performance communication becomes available to private users, new commercial applications will emerge that will utilize the capability provided by such communication. Examples are HDTV, high-quality voice transmission suitable for voice identification and speech recognition, hypermedia ([13]), distributed electronic games, etc.

It is important to note that the on-demand characteristic of high-performance communication is crucial in designing the communication system. The following example illustrates this point.

Site A needs to process medical image data (to perform FFTs, for example) on a powerful processor located at site B. The processing is done on pictures composed of 1000 × 1000 pixels each. Assuming 8 bit/pixel representation, each picture requires 1 Mbyte. Assume that the processing requires that the data of a whole picture be present before the processing on this picture starts. Suppose that the time to process a single picture is 100 msec. A satisfactory solution is to provide a low-speed, low-throughput, channel (1 Mbps, for example) that constantly sends the data from A to B. B buffers the data and passes it to the processor at processor pace. However, if the next picture to be processed depends on the results of the computation done on the previous one, the effective time within which the whole process is performed is the sum of the times for the two serially done processes: transmission over the network and processing at the processor. Because the link is a low-speed one, each 100 msec of the processor operation will be interleaved with 8 sec of waiting for the next picture to arrive. Note that I have assumed that the network introduces no delay. In reality, if the channel is low-speed, additional delay is introduced. Another solution is to provide a high-speed, high-throughput channel (of 1 Gbps, for instance), used only when B issues a request for the next picture of data. Now the rate at which the whole process is performed is approximately the processing rate only, since the transmission is done at very high speed. Thus the processor receives the next picture only 8 msec after the request is initiated (neglecting the propagation delay introduced by the channel), and the 100 msec processing periods are interleaved with only 8 msec transmission overhead. Moreover, retransmissions, required because of errors, have a minor effect on the whole process rate in this case. The two solutions are sketched in Figure 1.1.

The remainder of this chapter presents some examples of applications that illustrate the need for high-performance communication.

Large-scale distributed data base systems in general, and file systems in particular, are applications that require high-performance communication. Caching the data in a file system is usually an unpredictable process. Thus, the data in a file system is

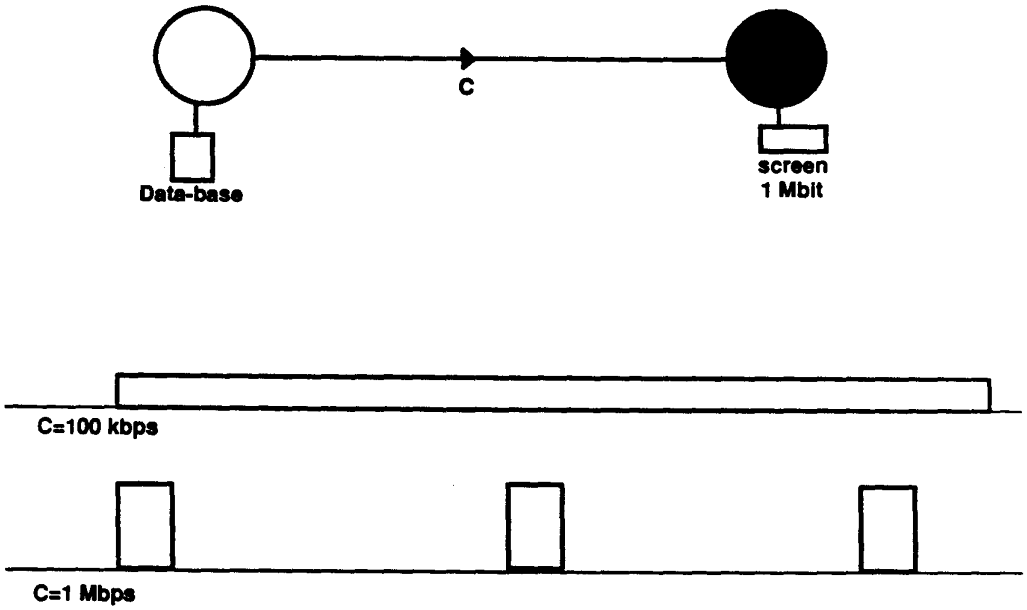


Figure 1.1: Low- vs. high-throughput channel for traffic with the on-demand characteristics

accessed on an on-demand basis. Moreover, large caches, which tend to have improved performance, need to communicate large amounts of data. Thus high-throughput is required. Furthermore, in order to keep the data base consistent, fast communication is needed. Finally, because the data base is distributed and replicated, multi-point communication is important.

Another example of the need for high-performance communication is multi-media conferencing. Such systems obviously require multi-point communication. Various mixtures of transmission are involved: image, high-resolution graphics, voice, etc. Some of the data types require communication of large amounts of data. Most of the communication is done on an on-demand basis by the current speaker, who upon receiving the "floor," may send a large chunk of data that includes a mixture of multiple types of data.

Some examples involve systems that do not possess all the four characteristics of high-performance communication (low-delay, multi-point, on-demand, and large amount of data). One such example is a data base of high-resolution images. In this particular case, there is no requirement for multi-point delivery, but the other characteristics of high-performance communication are required. The low-delay is needed for displaying the results of the browse operation through the database. Since browsing is selective, an unpredictable data access is required. Also, a large amount of data must be transmitted because of the high-resolution character of the images. As the example shows, even those applications that may not require all the characteristics of high-performance communication can greatly benefit from the high-performance communication. Moreover, integration of different traffic having different requirements, can be more easily performed in systems that offer a broader range of services.

I have presented a few examples of applications where the need for high-performance communication exists. These are only representative examples; I expect many other applications to arise when high-performance wide-area communication becomes truly available.

1.3 Overview of switching techniques

The switching techniques used in communication networks in general, and in computer networks in particular are: *circuit-switching*, *message-switching*, and *packet-switching*. *Circuit-switching* is characterized by static allocation of sub-channels created by partitioning the total channel capacity. These sub-channels are assigned to a particular pair of source/destination hosts for the whole period of the conversation between these two entities. In its conventional implementation, *circuit-switching* requires a setup procedure that sets a path (composed of a sequence of sub-channels) from source to destination. This setup procedure, which precedes the actual exchange of information between the two entities can be a lengthy process, especially for a connection involving a large number of hops. However, once a path is set, there is no delay in the exchange of the information between the two entities; the path is always available, since it is dedicated to the conversation. Figure 1.2 shows an example of the *circuit-switching* scheme.

Packet- and *message- switching* differ from *circuit-switching* in the dynamic nature of their bandwidth allocation. In these methods a channel is not dedicated to any traffic, but is shared on a time-division basis between all the users requesting the

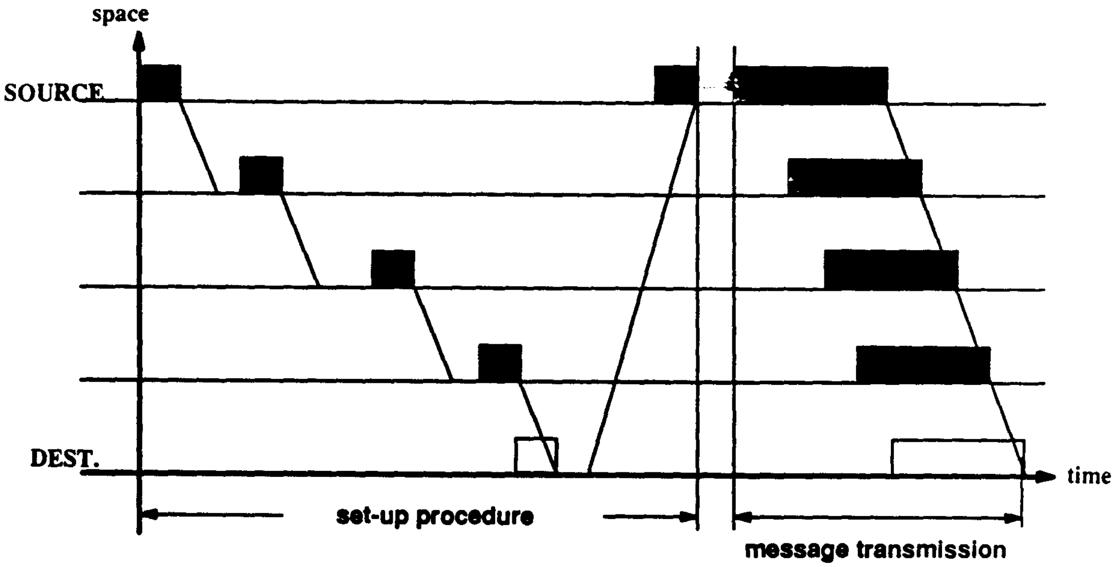


Figure 1.2: A *circuit-switching* example

channel. No setup procedure is involved in a conversation, and the information is immediately forwarded on the appropriate channel as the channel becomes available. In the *message-switching* scheme the whole message travels through the network as one unit, whereas in the *packet-switching* scheme a message is divided into smaller quanta, called packets. Packets are (usually) of fixed length, and much shorter than the average message length. Packets of the same message travel independently through the network from the source to the destination and, in general, may follow different paths. Thus, packets may be received at the destination in an out-of-order sequence. In both *message-* and *packet- switching* schemes, queues form at the entrances to links, because of the statistical variations in the arrival of different traffic streams at switching nodes. Consequently, messages and packets are delayed in each one of the switching nodes on their path.

A message in the *message-switching* scheme (and a packet in the *packet-switching* scheme) has to be completely received in a switching node before it is forwarded on an output link. Thus, a packet in a *packet-switching* network encounters a smaller delay than a message in the *message-switching* network. However, while packets of the same message travel through a network, the inter-packet gaps between packets of

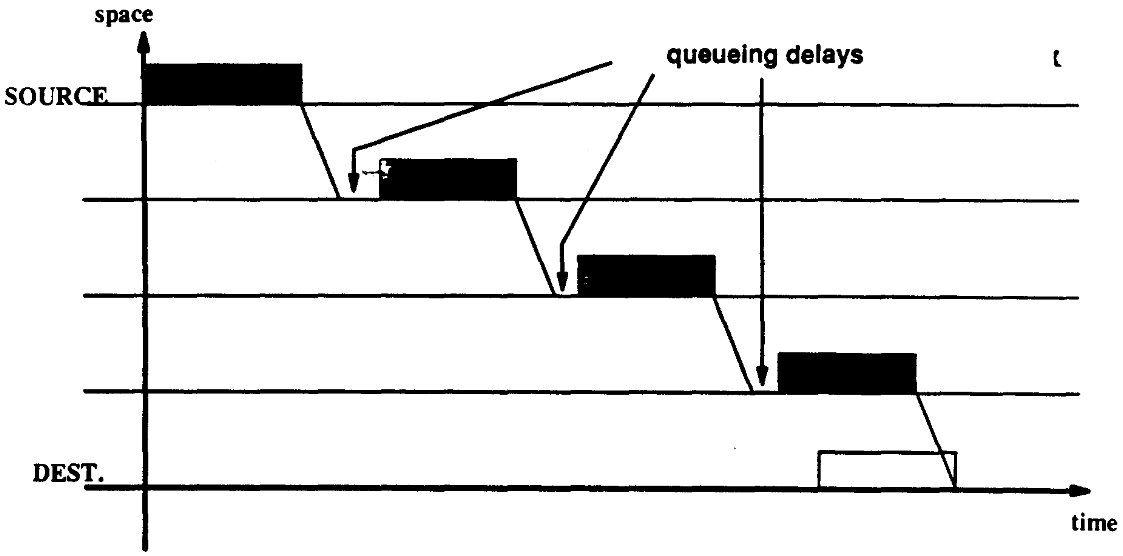


Figure 1.3: A *message-switching* example

the message increase because of the interruption caused by packets of other messages. Consequently, in order to compare the two schemes, one needs to evaluated the delay for a particular set of network parameters. An example of the *message-switching* scheme is shown in Figure 1.3 and of the *packet-switching* in Figure 1.4.

An improvement to both *message-* and *packet-switching* introduced by Kermani and Kleinrock ([14, 15]) called *virtual cut-through switching*, takes advantage of the fact that a message (packet) arriving at an available output link does not need to be completely received. In fact, only the header of a message (packet) must be received in order for the message (packet) to be forwarded. Further improvement to the *virtual cut-through* was introduced by Ilyas ([16]). In this case, a message (packet) arriving at an empty queue does not need to be completely received and will wait till the link output becomes available. Finally, Abo-Taleb and Mouftah ([17]) introduced the *general cut-through* permitting an arriving message (packet) to be partially received even if the output queue is occupied. In this scheme, a message (packet) will be completely received only if the waiting time for the output link exceeds the time needed to receive the message (packet). An example of the *general cut-through* is presented in Figure 1.5.

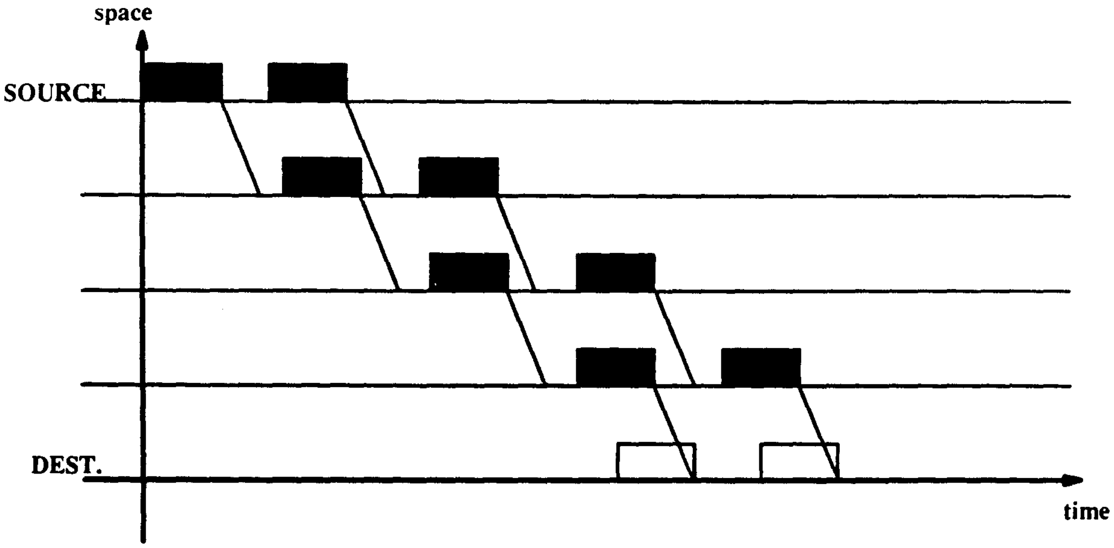


Figure 1.4: A *packet-switching* example

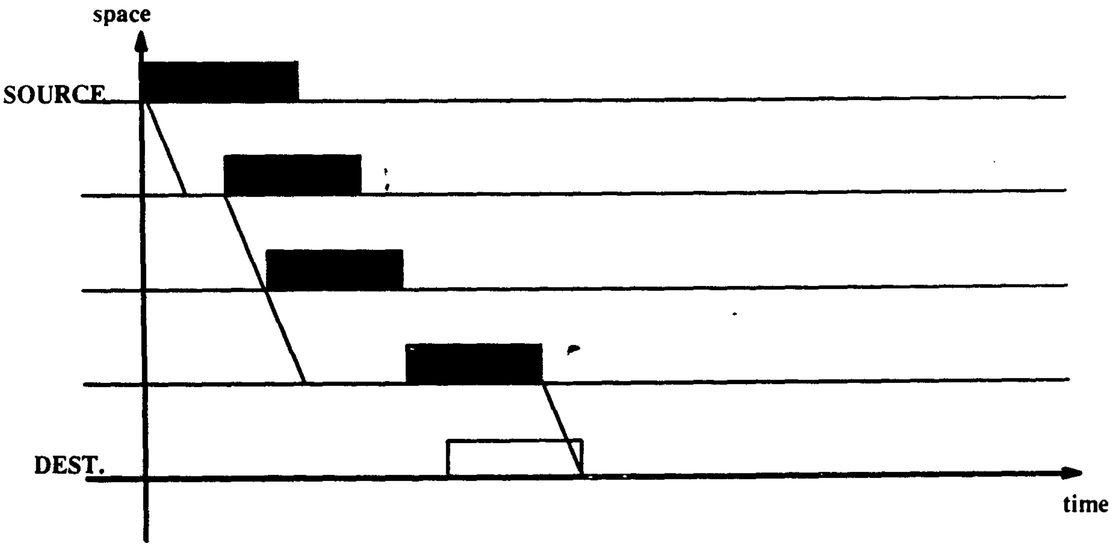


Figure 1.5: A *general cut-through switching* example

Circuit-switching has been traditionally used for voice applications, where, on average, the duration of a conversation is long compared to the setup time. Computer traffic, by contrast, is known to be bursty ([18, 19, 20]), requiring usually large bandwidth for a short time duration. In general, because of the static allocation of the bandwidth, *circuit-switching* performs poorly in computer networks compared to the *packet-switching* technique ([20]). This fact was the justification behind wide-area packet-switching networks like Arpanet, Transpack, Tymnet, Telenet, etc.

These wide-area networks cannot, however, support high-performance communication because of their long delay and low-throughput characteristics. Moreover, the introduction of new technologies such as fiber-optic communication and photonic switching and processing changes most of these networks' parameters, reviving the question of what is the best switching technique for such networks. Intuitively, it is clear that as the bandwidth of a network link increases, the packets "shrink" in time, thus creating more bursty traffic, which, as mentioned, makes the *packet-switching* technique preferable. On the other hand, the introduction of the *transactional* communication model ([21], see Figure 1.6), the expected increase in the message size, and the possibility of canceling the setup stage, suggest that *circuit-switching* may be preferable. Moreover, if the bandwidth is increased dramatically, it might be possible to permanently dedicate a sub-channel to every pair of communicating entities in the network, creating the fully connected topology. This work shows, however, that for a reasonable range of network parameters *packet-switching* remains the more favorable method for high-performance communication.

1.4 Advances in technology

The technological achievements in the fields of fiber-optic communication and of photonic switching can provide the means for the implementation of high-performance networks. In particular, optical fiber provides a long distance channel technology with transmission rates of gigabits per second and bit error rate on the order of 10^{-9} over tens of kilometers ([23, 24, 25, 26, 27]). Because of the low interference between optical signals and themselves and between optical signals and electronic signals, the

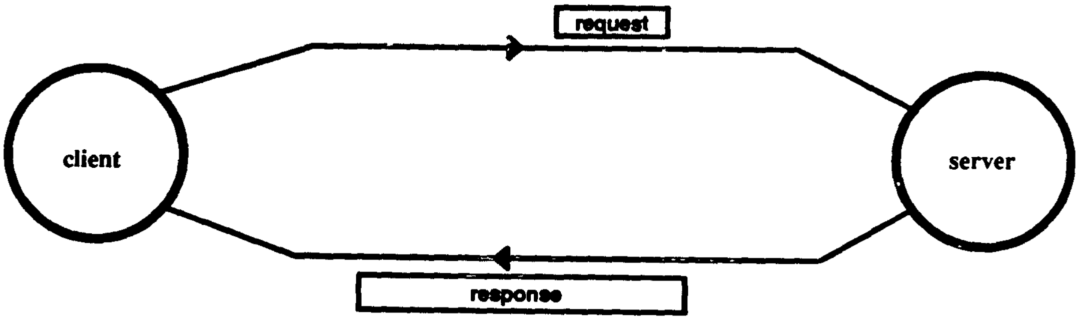


Figure 1.6: The *transactional* communication model

major source of the errors is receiver noise. Lower data rates can be achieved by coding. Because of the large bandwidth of optical fibers some researchers propose to use forward error correction codes.

There are two major classes of fibers: single-mode fibers and multiple-mode fibers. Single mode fibers have lower propagation dispersion than the multiple-mode fibers. Consequently, single-mode fibers have larger bandwidth and can, thus, conduct higher bit-rate signals. On the other hand, multi-mode fiber are cheaper and easier to install. The modulation used presently in fiber-optic communication is the direct modulation of light emitted from a semiconductor light-emitting diode or from an injection laser diode. With the introduction of coherent optical transmission in the future, increased data rates and longer inter-repeater distances are expected ([28, 29, 30]). The modulation is usually done on digital signals because of the difficulties in ensuring analog signal integrity in fiber-optic communication. The detection is done by an avalanche photodiode or a p-i-n diode.

Because of the advances in fiber-optic communication and technology, optical fibers are being installed extensively [31], replacing twisted pairs and coaxial cables, and bringing with them the benefit of very high bandwidth, two or three orders of magnitude higher than that of the existing networks. However, as mentioned above, the choice of the switching technique is crucial in order to reap all the benefits of this high-bandwidth medium.

Photonic switching and processing of the optical transmission opens new dimensions in future networking. Photonic implementation, as opposed to conventional electronic implementation, offers increased switching speeds [32, 33, 27]. In addition, a network built totally out of optical components is less vulnerable to electro-magnetic interference and electro-magnetic pulse, and, hence, provides more secure transmission. Since the technology of optical devices is still in its infancy, a network design based on a simple node design is highly desirable.

In order to prove the feasibility of high-performance, photonically implementable wide-area packet-switched network, I propose a network design christened *Blazenet*[1]. The design is characterized by its simplicity, and thus provides a basis for photonic implementation, and by its capability of fast packet-switching. It can be extended to support real-time traffic, multicast, and priority delivery.

1.5 Thesis outline

The work described in this thesis concentrates on wide-area networking mainly up to, and including, the network layer. In particular, its goal is to research the switching methods for high-performance communication, to determine the preferable one, and to propose a network architecture that uses this switching method.

This dissertation is organized as follows: Chapter 2 discusses other work related to this thesis and summarizes other approaches to high-performance networks. Chapter 3 presents *Blazenet*, one possible novel approach to high-performance networks. In particular, the chapter describes *Blazenet*'s design and operation, addressing the topics of packet switching and packet blockage, switching node design, expected performance, and some extended features that can be incorporated into *Blazenet*'s design, such as priority, multicast, and the loop reservation scheme. Chapter 4 compares the candidates for the switching techniques in high-performance networks, justifying the use of packet-switching in such networks. Chapter 5 presents the arguments for the all-photonic design of a communication networks, together with some implementation

[1] The name *Blazenet* refers to the use of lasers with fiber optics as well as the boomerang aspect of the returning packets that cannot proceed onwards, i.e., Boomerang Laser network. It also refers to the notion of a packet "blazing" a route through the network, and the speed at which it does so.

issues in such networks. Finally, Chapter 6 summarizes the thesis and presents its conclusions. Some additional mathematical derivations are presented in the Appendices.

Chapter 2

Related Work

2.1 Switching techniques

The three basic switching techniques: *circuit-switching*, *packet-switching*, and *message-switching*, as well as variations of these techniques, have been extensively compared in the literature ([34, 35, 36, 14, 37, 16, 38, 39, 17]). However, past research has been done based on the one-way communication model (rather than the *transactional* model ([40, 21, 41]) that is assumed in this work). Moreover, the previous work does not show the effect of the increase in the channel bandwidth, which is became a significant effect with the introduction of fiber-optic communication. (Note that the transactional communication model increases the delay of the packet-switching and leaves the circuit-switching delay essentially unchanged compared to the corresponding delays of the one-way communication model. Thus, circuit-switching benefits more than packet-switching from the introduction of the transactional model.)

The most extensive comparison between the switching techniques can be found in [15]. The work concluded that even through the cut-through technique is superior to circuit-switching for broad range of values for the different system parameters (number of sub-channels, message length, utilization factor, and number of hops), nevertheless, in some cases circuit-switching outperforms cut-through techniques. The cut-through switching technique used in that paper is the *full-cut-through* technique. It is shown in [16, 38, 17], however, that the *general-cut-through* technique has a significant advantage over the *full-cut-through* and *quasi-cut-through* techniques, especially for the low to intermediate range of the utilization factor. (A packet is said to make a *full-cut* when, upon its arrival, the output link is free and the packet is forwarded on the output link without having been previously stored. A packet is said to make a *partial-cut* if the output queue is free, the output link is busy, and the packet is not completely stored before being forwarded. The method allowing *full* and *partial cuts* is referred to as the *quasi-cut-through* technique [16, 38]. The *general-cut-through* technique, proposed in [17], goes one step further by partially storing a packet even when the output queue is busy. [39] refers to this technique as *quasi-cut-through.* The analytical treatments for the *quasi-* and the *general-cut-through* are different. In particular, the *general-cut-through* has a simpler solution.) As shown in [39], the *general-cut-through* also performs better than the usual packet-switching in a noisy environment, provided the error-rate is relatively low. Consequently, the *general cut-through* has been chosen as the packet- and message-switching technique for this work.

The comparisons in previous works do not attempt to optimize the number of sub-channels in the circuit-switching scheme. [15, 37] present the optimum number of sub-channels for different cases. However, the comparison itself is done for a discrete and fixed number of sub-channels.

In this work, I compare circuit-switching methods and cut-through switching methods, in their *general-cut-through* form. I made the comparison for the *transactional* model of communication, using a range of values of parameters appropriate for fiber-optic communication. (These values differ by orders of magnitude from the ones used in previous work). In particular, I assume very large bandwidth (of the order of Gbps), a large amount of data to be delivered in each transaction, low error rate, and low average channel utilization. The number of sub-channels for the CS delay calculation is optimized for the particular load in question in each case. Also, the set-up procedure of the CS is assumed to be nonexistent. This assumption leads to an improved CS scheme.

In order to perform the comparison, an analytical model for the CS delay is proposed and solved. (The model presented in [15] relies on simulation for its solution.)

This circuit-switching model takes into account the reservation of sub-channels and the reservation scheme accounts for the propagation delay over the path within the network.

As pointed out in [37], most of the other works assume that the sub-channel holding time in the CS model is equal to the message transmission time, an assumption that is far from realistic especially in the wide-area environment. However, also [37] fails to correctly model this holding time, assuming, for the sake of simplicity, a negative exponential distribution. The circuit-switching model of the present work corrects this defect.

As the first step, the comparison between packet- and message-switching, both operating in the *general-cut-through* mode, is presented in the next section. The reassembly delay of a packetized message plays an important role in the comparison. The formula for reassembly delay was developed in [36] and [42]. This delay is neglected in many previous works, although it is the factor that is responsible for the superiority of message-switching over packet-switching for high channel utilizations (as shown in [36]). This behavior, as shown here, is different in the *general-cut-through* mode.

2.2 High-speed switching

Several different approaches to high-speed networking have been described in literature. Most of the approaches belong to one of the two categories: an improved architecture of a switch or novel topological configuration of a network. Examples from the first group are: STARLITE and the Knockout switch, and from the second group: Hubnet and Manhattan Street Network. These examples will be described briefly.

STARLITE [43, 44]) is a representative of a family of switch designs that uses some configuration of a sorting network to facilitate fast switching. STARLITE uses a self-routing, non-blocking, constant latency packet-switch based on Batcher sorting network. The switch supports switching of various traffic mixtures, as well as some special features such as broadcast and multicast. There has been a lot of work done on various improvements to the STARLITE idea and on similar approaches ([45, 46, 47, 48]).

The Knockout switch ([49]) is another example of a self-routing, non-blocking, and low-latency switch architecture. The main idea behind the Knockout switch is the use of broadcast bus per input line and shared buffer pool per output line. The switch receives traffic on its input line in time-slotted fashion. A received packet is broadcasted on a bus that belong to the line the packet was received on. The broadcasted packet is recognized by the output line and is stored in a buffer that belongs to the output line. A lossy concentration is provided for each output line, because of the finite number of output buffers. Among other advantages of the Knockout switch are modularity and maintainability.

Hubnet ([50, 51]) is a local-area network based on optical fibers and tree topology. The network is a contention-based network, where the central hub performs the function of arbitration. Successful packets are broadcasted to all the network stations, thus providing self acknowledgement. If a packet is blocked, it is not broadcasted by the central hub, and thus not seen back by the packet's source. After some timeout, the source of the blocked packet realizes that its packet has been blocked and will retransmit the packet. The high-speed LANs generally provide excellent performance in terms of low delay and high throughput. Unfortunately, in nearly all of the cases the performance of a network sharply degrades as the span or the number of stations on the network increases. Consequently, this class cannot serve as WAN.

Manhattan Street Network ([52, 53]) is a metropolitan network with torus topology. Since each node receives traffic from exactly two nodes and forwards traffic to exactly two other nodes, there is no need for buffering of through traffic, routing algorithm can be significantly simplified, and packet-switching can be done "on the fly." If two packets contend on the same link, one packet makes it through the switch and the blocked packet is routed to the other link (which is always available in this situation). Manhattan Street Network provides good solution for fast packet-switching in metropolitan size, regular mesh topology networks. Unfortunately, this design cannot be extended to larger and unconstrained topologies, topologies that characterize wide-area networking.

There is also a lot of work done on high-speed circuit-switching. In some of the work improvements are achieved by faster memory devices and by incorporating changes in the circuit switch architecture ([54]). The more interesting work is the one that employs photonic devices to provide fast circuit-switching ([55]).

Besides the attempts to achieve high-speed switching by improving the existing switch and network architectures, there are proposals for novel approaches to the switching process. One such example is the self-routing photonic switching demonstration ([56, 57, 58]) that uses optical spread-spectrum coding to perform the switching operation.

The purpose of the *Blazenet* design presented in this work is to propose a possible approach to high-performance wide-area networking. *Blazenet* belongs to the group of novel topological configurations that are suitable for wide-area networking and are photonically implementable.

Chapter 3

One approach to high-performance WAN

3.1 Introduction

In this chapter, I describe the desigb of a packet-switched network architecture, named *Blazenet*, based on optical fiber and high-performance switching nodes ([1, 2, 3]). Three key ideas behind the *Blazenet* design are: source routing, packet loop-back on blockage, and photonic implementation, which is made possible by the first two principles. Fiber loops that constitute *Blazenet* links provide the temporary storage for blocked packets in transit, thus using the storage inherently present in the links. (*Blazenet* is presented here as a wide-area backbone network, whose nodes are gateways to other networks. Nevertheless, the concept of *Blazenet* is easily applied to smaller networks and, with some constraints on transmission rate and on maximum packet size, even to local area networks.)

It should be pointed out here that the network in this work is assumed to be composed of relatively slow sources that can perform various operations in software, such as protocol implementation. The network, on the other hand, is very high speed such that any processing should to be kept to a minimum in order to be feasible. Another reason for keeping the network design as simple as possible is to facilitate its photonic implementation.

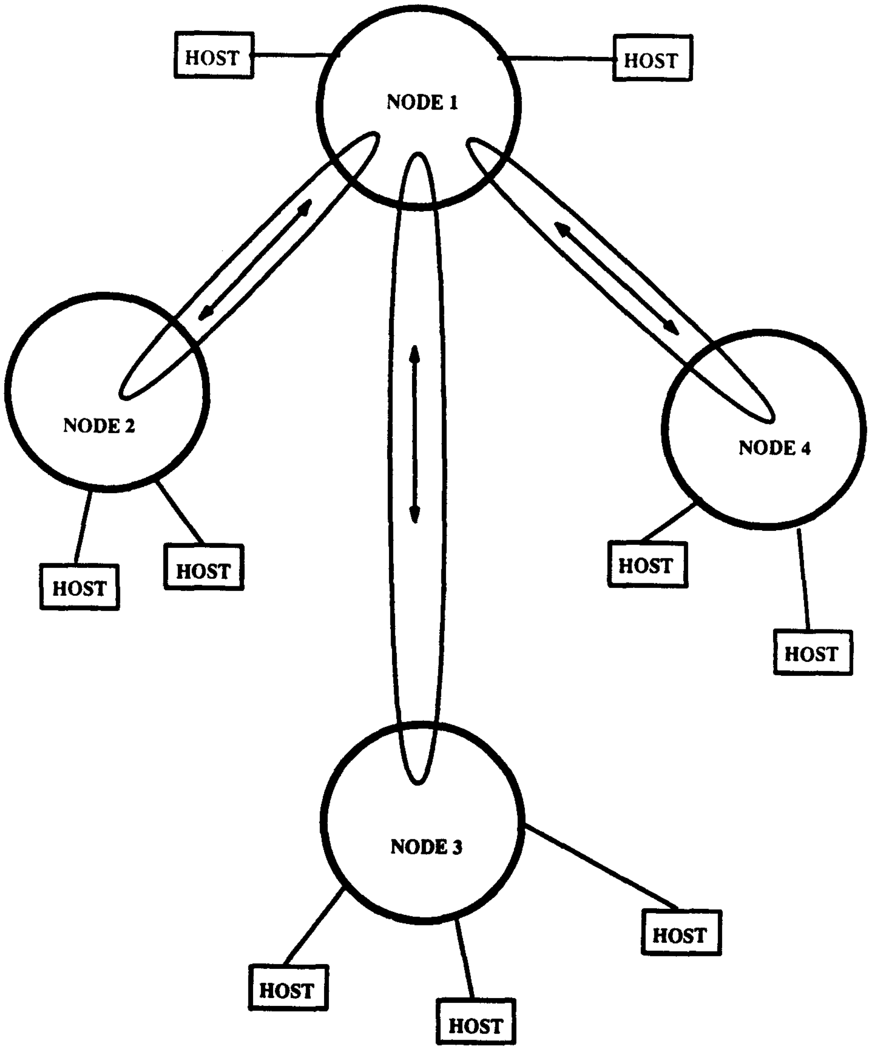


Figure 3.1: A four node *Blazenet* example

Section 3.2 describes the *Blazenet* design, addressing the issues of packet-switching and traffic congestion. Section 3.3 presents a detailed switching node design. Section 3.4 shows *Blazenet*'s performance determined by simulation and analytical solution. Section 3.5 discusses some extended features that can be incorporated into *Blazenet*'s design, such as priority, multicast, and the loop reservation scheme. Section 3.6 presents some higher layers issues that have direct implication on *Blazenet*'s operation. Finally, Section 3.7 summarizes the conclusions about the design and the implications.

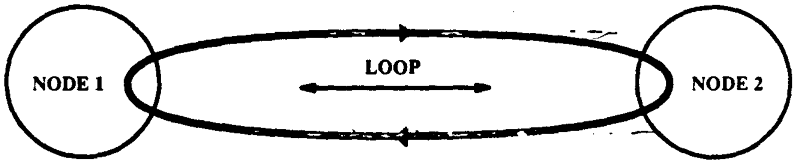


Figure 3.2: A *Blazenet* loop

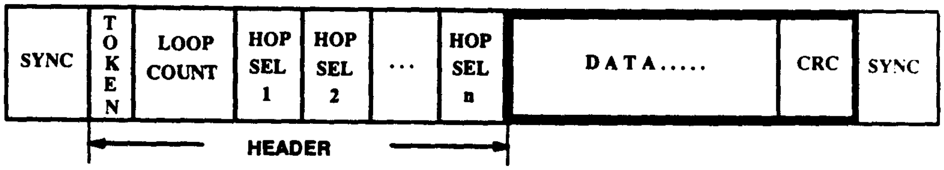


Figure 3.3: The *Blazenet* packet format

3.2 *Blazenet* design

A *Blazenet* is composed of a set of switching nodes interconnected by point-to-point logical links formed by the fiber loops. The hosts and gateways on the periphery of the network act as sources and sinks for the network traffic. Packets, generated by hosts, are passed to the switching nodes to which the hosts are connected. The packets are then forwarded from node to node until they arrive at the switching node connected to the destination host. At this point, the packets are removed by the switching node and passed to the destination host. An example of a four node *Blazenet* is shown in Figure 3.1.

A *Blazenet*'s loop, shown in Figure 3.2, consists of two point-to-point logical links. In such a configuration each bidirectional link connecting two adjacent nodes is replaced by a single-loop. A number of loops can be multiplexed on a single fiber by the *Wavelength Division Multiplexing* technique, for example.

The *Blazenet* packet format, shown in Figure 3.3, is composed of two delimiting synchronization fields (*syncs*), a header, and a data portion. Within the header, the *token* identifies whether the packet is a blocked packet (indicated by the *token* being set) or not (reset *token*), the *loopcount* limits the packet lifetime within the network (as described later), and the *hop-selects* dictate the hop-by-hop route for the packet to reach its destination. The data portion contains higher level protocol data and can, optionally, be protected by a checksum.

3.2.1 Source route packet switching

Blazenet uses source routing ([63, 64]). Each packet contains a sequence of *hop-selects*, specified by the source host. The *hop-selects* represent the switching operations to be taken in the sequence of nodes along the packet path through the network from its source to its destination. Each *hop-select* field indicates the output link on which the packet is to be forwarded for that hop. When a packet arrives at a switching node with its *token* reset, the first *hop-select* field in the packet is examined by the switching node to determine the next output link for the packet. If that output link is available for transmission of a new packet, the first *hop-select* field is zeroed and the packet is immediately routed to the available output link. The zeroing of the first *hop-select* field during the forwarding process means that the first non-zero *hop-select* field in the packet always represents the current hop selection. A packet, arriving at a switching node with its *token* set, is simply left on the loop to be returned to the blocking node after the *token* field is reset.

This design has several advantages. First, because of the simple logic required to make the hop selection, it is possible to perform the switching function at gigabit per second data rates. In particular, no table lookup is required for the switching decision. Second, the delay for switching in a node is limited to the time required to interpret the packet header, check the availability of the output link, and perform the actual switching operation (if the output link is available). This extra delay introduced by a switching node is estimated to be only a small fraction of the propagation delay of a link in a wide-area network. Finally, the simplicity of the node logic suggests that a photonics implementation is feasible.

3.2.2 Handling packet blockage

A packet is called blocked if it arrives at a switching node when the next output link is unavailable. A blocked packet is routed back to the previous switching node on the reverse link of the loop that the packet arrived on. Upon its arrival at the previous switching node, the returned packet is sent out again, to arrive at the blocking switching node one round-trip time after its first arrival at this node. Thus, the loop effectively provides short-term storage of the packet, causing the packet to reappear at the blocking switching node a short time later. The *loopcount* field is decremented and examined each time a packet is returned. When *loopcount* reaches zero, the packet is removed from the network. This removal prevents a packet from indefinitely looping within the network under failure or very heavy load conditions.

This approach to handling blockage has several advantages. First, compared to a design in which a packet blocked at the outgoing link is simply dropped, a design referred to here as a *Lossy* network, *Blazenet* dramatically reduces the average packet delay through a loaded network and increases the network capacity. When a packet is dropped in a *Lossy* network, it has to be retransmitted by the source after some timeout, at least one round-trip time long. Since the probability of a packet being blocked increases with path length, as does the network investment in the blocked packet, dropping the packet seriously degrades the network performance under load for wide-area networks with a realistic diameter.

Second, the design does not require memory in the switching node of the size and speed required to store all blocked packets, such as would be needed for a conventional *Store-and-Forward* design. Several megabytes of memory operating at 1 Gbps would increase the cost of the switching nodes and make their realization in optics less attractive (at least in the near future). The combination of the high data rates, the wide-area span of the links, and the predicted low bandwidth utilization makes this form of storage attractive. For example, a 100 km link (= 200 km loop) operating at 1 Gbps can store 1 Mbit or 125 packets of 1 kbyte each.

Finally, the loop-back technique exerts back pressure on the link over which the packet was received, because the loop is then less available for new packets to be forwarded on it. In the extreme, this back pressure extends back from the point of contention to one or more packet sources. Besides alerting the packet source of congestion, the back pressure provides fast feedback to the source routing mechanism, allowing it to react quickly to network load and topological changes.

A potential disadvantage arises when the link between switching nodes is very

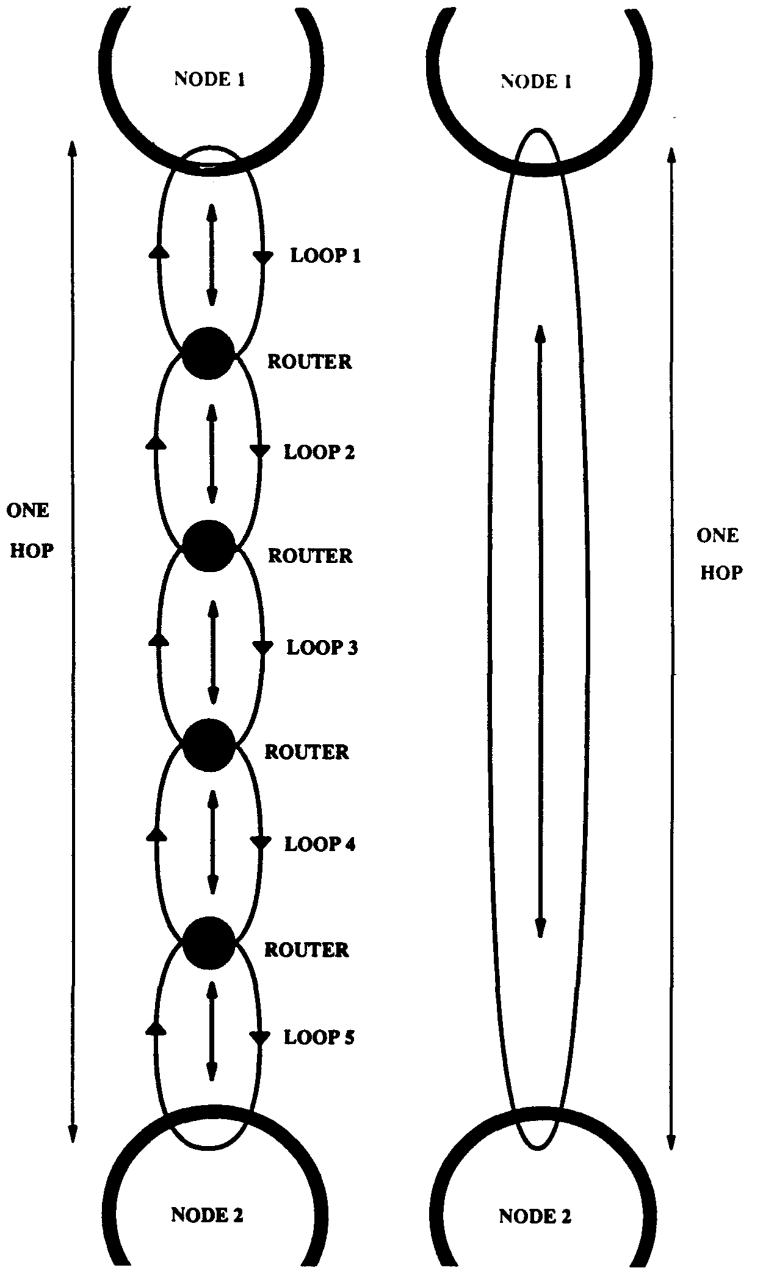


Figure 3.4: Partitioning a single loop into five sub-loops

long, since the round-trip delay on the loop may be excessive. This problem can be avoided by including loop-back support in the optical repeaters, which are required anyway every few tens of kilometers on a fiber optic link. An example of a loop divided into five smaller sub-loops is shown in Figure 3.4. Thus, a packet that is blocked at a switching node is looped back either to the previous switching node or to the previous repeater, whichever is closer. If, for example, the distance between adjacent switching nodes is 100 km, the round-trip delay is approximately 1 msec. A *Blazenet* switching node can be used as a repeater, because it regenerates the optical signal, and thereby, automatically supports the loop-back function. (As a repeater, only two input and two output loops are used.) The network is then built from one type of interconnection component rather than two. In such a design, a packet can loop at intermediate loops on a long link; hence its delay through the network is reduced (assuming the congestion clears before the packet is dropped). Consequently, the packet is delayed in time units corresponding to the round trip time of the intermediate loop rather than that of the entire link. Another improvement consists of designing the last sub-loop (which is the sub-loop connected to the next node) shorter than other sub-loops on the link. Consequently, blockage at low-load operation has a smaller effect on packet delay.

3.3 Switching node design

A *Blazenet* switching node can be implemented as a simple interconnection of a number of photonic components. I assume the availability of fast-switching devices capable of switching within a small fraction of the duration of a header bit once the switching command has been initiated. Devices having an operation speed of at least 3 GHz exist. Slower devices can be employed for lower cost by maintaining an adequate inter-packet gap. It should be pointed out here that a full photonic implementation of *Blazenet* is still not possible due to the fact that photonic processing is yet not a mature technology. However, the data path of a *Blazenet*'s switching node, as shown in Figure 3.5, can be implemented using photonic switching devices with the today's state-of-the-art technology.

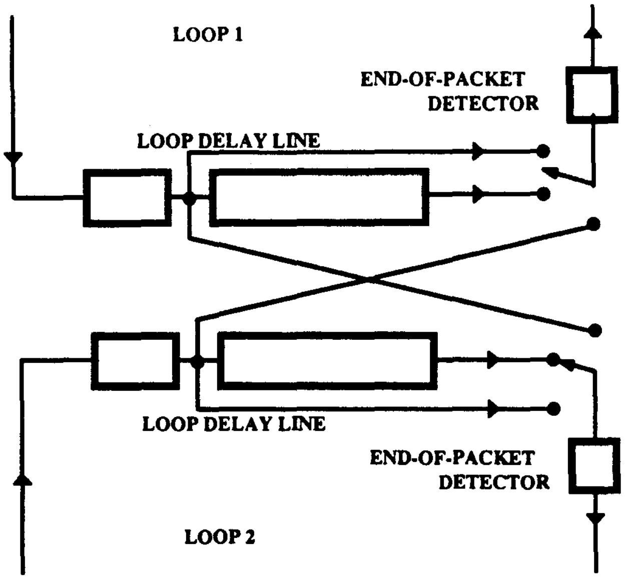
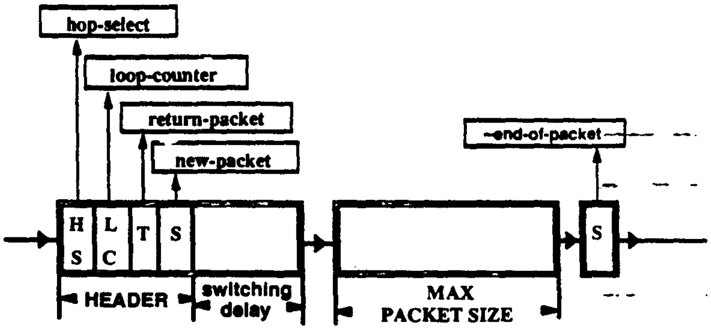


Figure 3.5: Switching node design

A *Blazenet* switching node connects several *Loops.* Each *Loop* has a *Delay Line,* consisting of a piece of fiber. The *Delay Line* is long enough to contain a packet header, a maximum-sized packet, and the number of bits corresponding to the time the control logic requires to do the actual switching. Figure 3.6 shows the signals extracted from the transmission entering the *Delay Line* and the corresponding timing. The signal extraction is initiated by the *new-packet* signal generated by a pattern detection circuit, which searches for the *sync* pattern. Upon *sync* detection, the circuit raises the *new-packet* line, indicating to the *Control* the arrival of a new packet. At this time the *Control* looks for the values of the *token*, of the *loop-count*, and of the *hop-select* signals. The indication that a packet leaves the *Delay Line* is provided to the *Control* by the *end-of-packet* line of the input or the output *Loop,* depending whether the packet is forwarded or blocked. The switching decision and the actual switching operation are performed during the period of time named "switching delay," at the end of which a switching command is issued.

A *Delay Line* is considered to be free if it does not contain a packet or any part of a packet. By using the two signals *new-packet* and *end-of-packet*, the *Control* can uniquely determine the status of a *Delay Line.*

When a packet is to be forwarded to a loop, the availability of this loop (i.e, the condition that no connection of any other loop to this loop already exists) and the



S=SYNC, T=TOKEN, LC=LOOP-COUNTER, HS=HOP-SELECT

Figure 3.6: Timing of *Delay Line* signals

availability of its *Delay Line* are checked. If both are available, the packet from the input loop is clocked onto the output loop. If, on the other hand, the output loop and/or its *Delay Line* are busy, the packet is blocked, its *token* is set, and the packet is returned by being clocked out on the loop it came on. In case more than one packet tries to enter a specific loop, only one packet wins (the one with the higher priority, or one chosen randomly in the case the of equal priorities), and the other(s) are clocked out on their loops. Upon its arrival at the other end of the loop, a blocked packet is clocked into the corresponding *Delay Line*, blocking access to this loop for any new arrival. When the packet reaches the end of the *Delay Line* it is clocked out onto the loop it came on, after the *token* is reset. (Another possible solution is to check the loop availability upon reception of a returned packet, and to enter the returned packet into the *Delay Line* only if the loop is busy forwarding another packet. If, however, the loop is found free upon arrival of the returned packet, the packet is clocked out immediately. This improvement has the advantage of including an additional *Delay Line* delay only in the case in which the loop is busy. On the other hand, this solution has the disadvantage of complicating the switching process and therefore, the *Control* itself).

The *Switching Element*, shown in Figure 3.5 as a set of switches, can be designed in several ways ([65, 66]). One such a way is to use a switching matrix, as shown on Figure 3.7. In this case maximal connectivity can be achieved.

The *Control* performs the actual routing decisions based on the signals that indicate the status of the loops. The signals entering the *Control* are shown in Figure

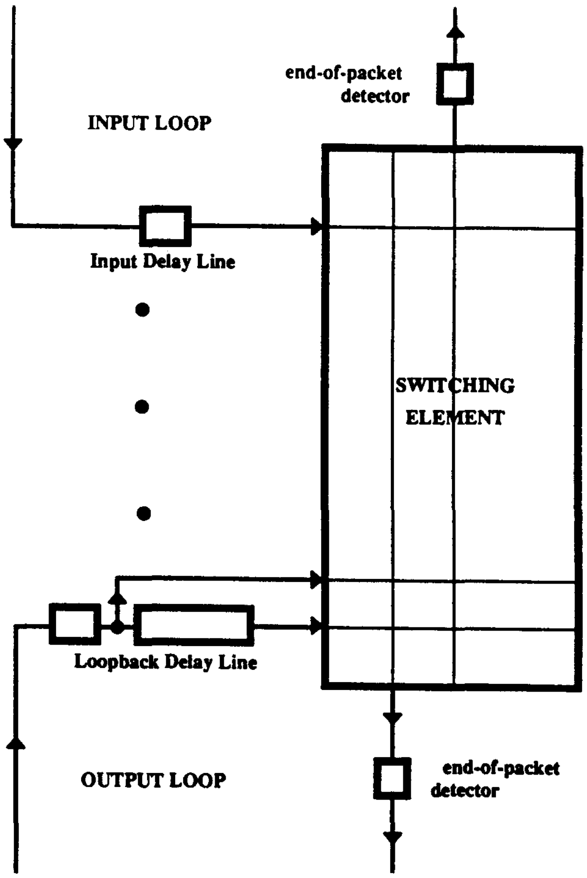


Figure 3.7: Switching node design using a switching matrix

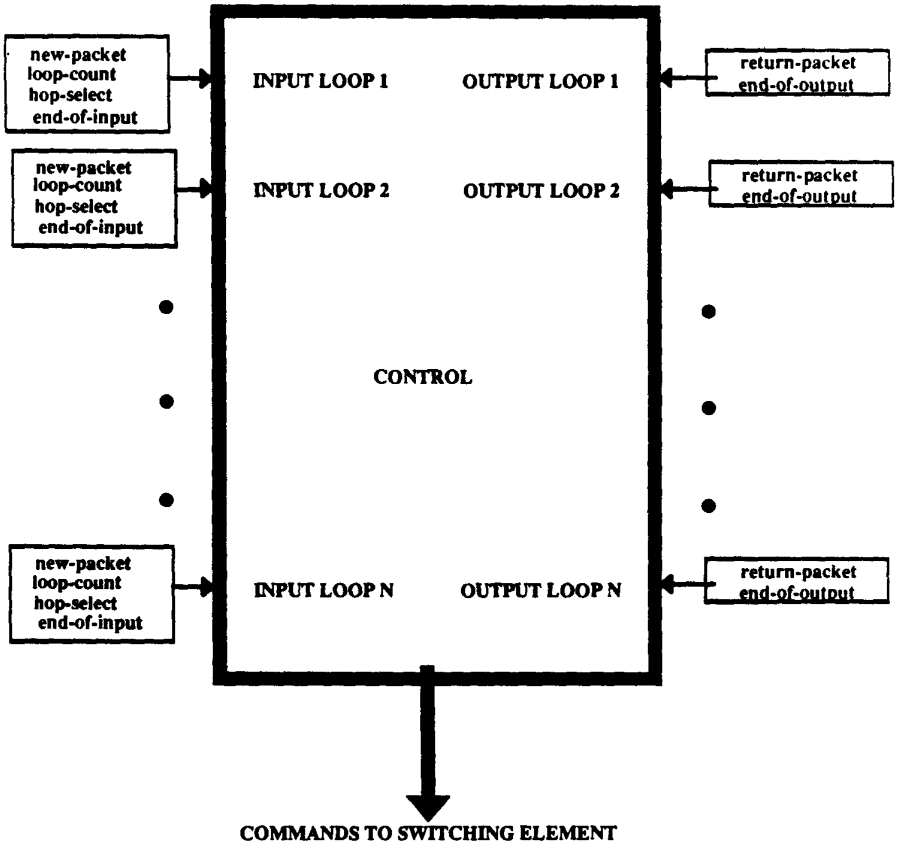


Figure 3.8: Switching node control signals

3.8. The routing algorithm takes into account the following parameters:

1. input packet destination (number of the output loop),
2. availability of the switching element (if switch with less than maximal connectivity is used),
3. availability of the output loop and its *Delay line*
4. priority of the packet (as explained in Section 3.5.1).

In general, a packet is either completely forwarded or returned. However, in some cases of extraordinary priority, it may be necessary to abort transmission of a packet currently being forwarded, to clear the line for such high priority transmission. A simple mechanism can be incorporated into the design such that upon reception of a packet with the extraordinary priority, that packet is immediately forwarded on the appropriate loop.

The input traffic from a host connected to the switching node is switched in a way similar to that in which the traffic from any loop is switched. The main difference is in the indication of an available packet. An indication line from a host to the *Control* continues to show the presence of a packet until it is forwarded. No incoming packet is ever returned.

The output traffic destined to a host connected to the switching node is received on one of the outputs of the Switching Element and passed to the appropriate host. A host is assumed to be always ready to accept its traffic. If the host is unavailable, the packets are simply discarded. Therefore, the major difference between the through traffic and the exiting traffic is that the latter is never returned. The reason for not allowing the return of exiting blocked traffic is to avoid situations in which the network could possibly be paralyzed because of a host malfunction.

If the speed of the lines increases to a value where bit recognition of the header becomes a problem, representation of a header bit by several actual bits will allow more time for decoding. Thus, by keeping the header bit "duration" constant, the problem of header bits recognition is essentially made independent of the actual line speed.

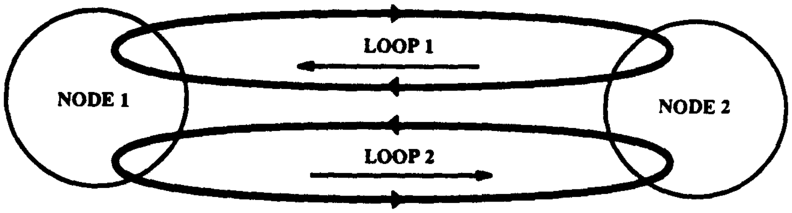


Figure 3.9: The double-loop configuration

3.3.1 Alternative design choices

One alternative to the presented *Blazenet* design is to use slotted loops. Another variation is to use double-loop configuration for the bi-directional transmission. I will consider the slotted version first.

In the *Blazenet* slotted version, the loops are divided into slots of the packet size (in the case of variable packet size, the slots are of the maximum packet size, referred to as "max packet size"). Packets can be inserted only into empty slots, indicated by some bit within the packet format.

The arrival of packets in the *Blazenet* slotted version can be synchronized or not. The synchronization is done by including an appropriate delay in each one of the input loops, so that all the packets arrive at the same time. Packets can then be interchanged between the slots of the various loops. For the slotted version with no packet synchronization, the presented design can still be used. However, the max packet length will now be equal to the slot time. For the slotted version with synchronization between the arriving packets, no delaying of the packets is necessary. Consequently, the *Input Delay Line* does not need to include the max packet length portion, and the switching decision can be made as soon as a packet's *main-header* has arrived. The slotted version has some advantage in performance over the non-slotted approach. Nevertheless, the required slot synchronization is a serious disadvantage of the slotted version (see also Section 5.3.1). Also, in a network with a variable packet size the use of the maximum packet length as the slot size may be of some disadvantage.

In the double-loop version of the network, two loops replace a bi-directional link of a conventional network. Such a configuration is presented in Figure 3.9. The lower portion of loop 1 serves the transmission from node 2 to node 1, while the

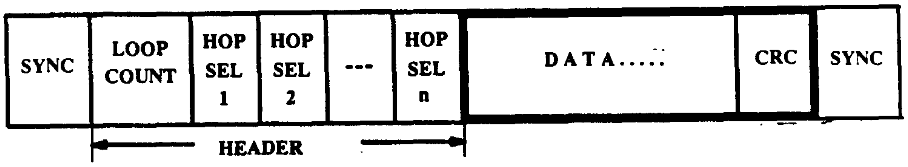


Figure 3.10: Packet format for the double-loop *Blazenet*

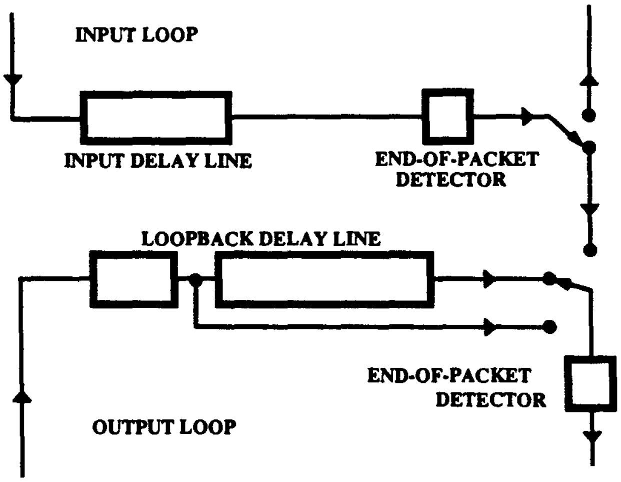


Figure 3.11: The double-loop switching node design

lower portion of loop 2 serves the transmission from node 1 to node 2. A blocked transmission is returned on the upper portion of the loop it came on. No indication of a packet being a returned packet is necessary in the double-loop case, since the usage of the upper portion of a loop indicates that the packet is a blocked one. The modified packet format is shown in Figure 3.10.

Figure 3.11 shows the modified block design of a *Blazenet* switching node to accommodate the double-loop configuration. Each *Input Loop* has a Delay Line (*Input Delay Line*), consisting of a piece of fiber. The *Input Delay Line* is long enough to contain the leading *sync*, the *loopcount*, the *hop-selects*, and the number of bits corresponding to the time for the control logic to do the actual switching. Figure 3.12 shows the signals extracted from the transmission entering the *Input Delay Line* and the corresponding timing. (The pattern detection circuit is an optical correlator, consisting possibly of a fiber delay line [33]). Upon *sync* detection, a pattern detection

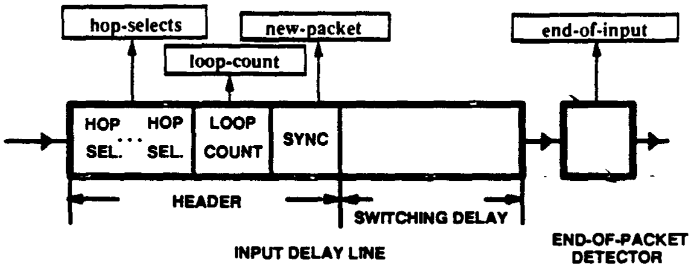


Figure 3.12: *Input Delay Line* signals and their timing

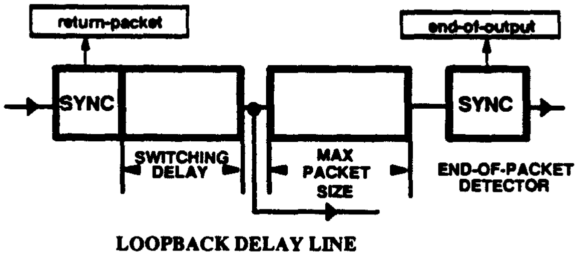


Figure 3.13: *Loopback Delay Line* signals and their timing

circuit raises the *new-packet* line, indicating to the *Control* a new packet arrival. At this time the *Control* looks for the values of the *loop-count* and the *hop-select* signals. The indication that a packet leaves the *Input Delay Line* is provided to the *Control* by the *end-of-input* line.

Each *Output Loop* also has its own Delay Line, the *Loopback Delay Line.* The *Loopback Delay Line* must be the length of the maximum packet size plus the switching delay. The signals extracted from the information within the *Loopback Delay Line* and their timing are presented in Figure 3.13. Upon detection of the leading *sync* pattern of a returned packet, the *return-packet* signal is raised. This indicates the occupation of the *Loopback Delay Line.* A similar circuit, positioned at the end of the *Loopback Delay Line*, scans for the trailing *sync.* Detection of the trailing *sync* by this circuit initiates the *end-of-output* signal, which indicates when a packet leaves the *Loopback Delay Line.* Using the two signals, the *Control* can uniquely decide on the *Loopback Delay Line* state.

The process of forwarding a packet in the double-loop configuration is similar to that of the single-loop case, the main difference being the functions of the *Loopback*

Delay Line of the single-loop configuration and the *Input Delay Line* of the double-loop configuration.

The main advantage of the single-loop over the double-loop version is in reduction of hardware: fibers, transmitters, receivers, etc. The double-loop version is simpler to implement, possesses some reliability advantages, and has lower delay and stable throughput under heavy-load.

Blazenet variations can be combined. Thus, single-loop and double-loop *Blazenet* can operate on slotted or unslotted loops. In this work I concentrate on the non-slotted single-loop version.

3.4 *Blazenet* performance

In order to evaluate *Blazenet* performance,I used two tools: network simulation and analytical solution. The simulation program receives as an input the network topology, the routing matrix, and the traffic matrix, and produces the graph of the average packet delay vs. the total network throughput. Average packet delay is the period of time from when the packet is passed to the network until it is delivered to its destination averaged over all packets entering the network, and includes the queuing time at the network entrances. The total network throughput is the aggregate rate of packets entering the network through all the network entrances. The simulation is general in the sense that there are no constraints on the network topology, the traffic matrix, or the routing matrix.

The analytical solution in its general form consists of solving a set of $2L$ (generally nonlinear) equations, where L is the total number of loops in the network (including the sub-loops). The solution provides the network capacity and the delay as a function of the network throughput.

For simple case of topologies and traffic patterns, the network solution can be achieved by applying fundamental rules of the probability theory. As an example of such development, the double-loop *Blazenet* configuration in its slotted and unslotted versions are solved for capacity and delay in Appendix F.

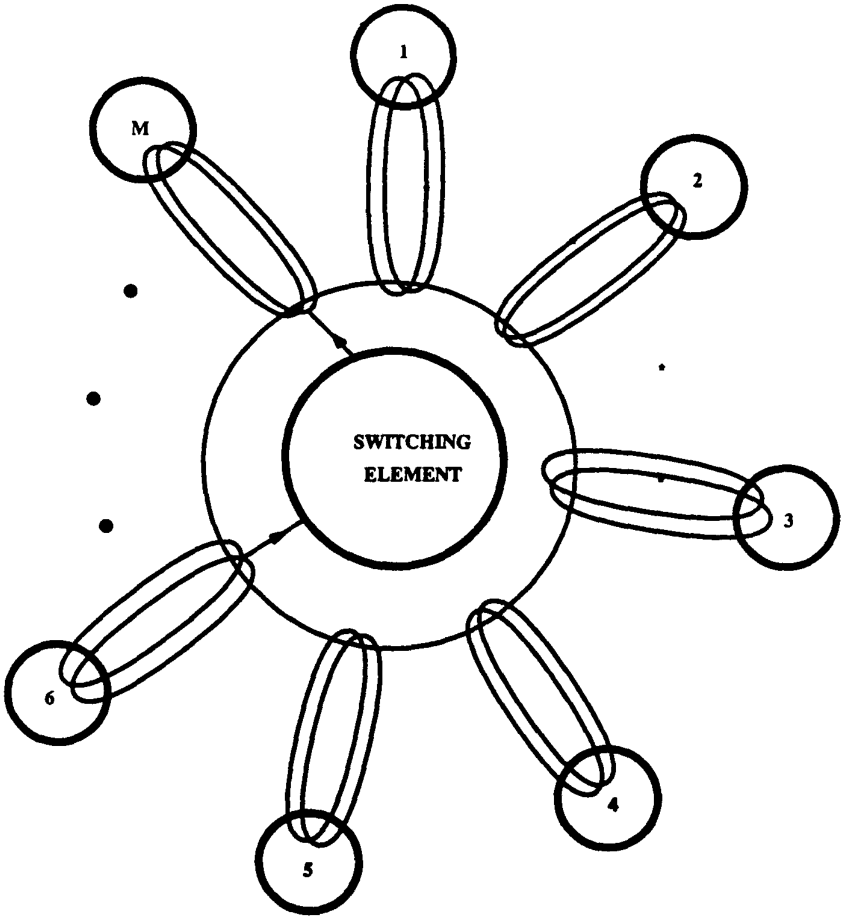


Figure 3.14: *Star* topology

3.4.1 *Blazenet* simulation

The simulation that was developed to evaluate *Blazenet*'s performance is a general, event-driven program written in Pascal. Its results, besides evaluating the absolute *Blazenet* performance, provide a way to compare the network's performance with that of the ideal case of a nonblocking network (whose delay is the propagation delay only) and with that of the *Lossy* network. The following graphs show *Blazenet*'s performance for several different network architectures and different packet sizes. In all of the examples, I assume that the traffic matrix is symmetric, the link capacity is 1 Gbps, and the links are all equal and approximately 100 km long. I also assume infinite *loop-counter* value and equal priority of all packets.

The performance was evaluated for packet sizes of 5 kbit and 10 Kbit. These values represent a reasonable trade-off between the long *Delay Line* for large packet

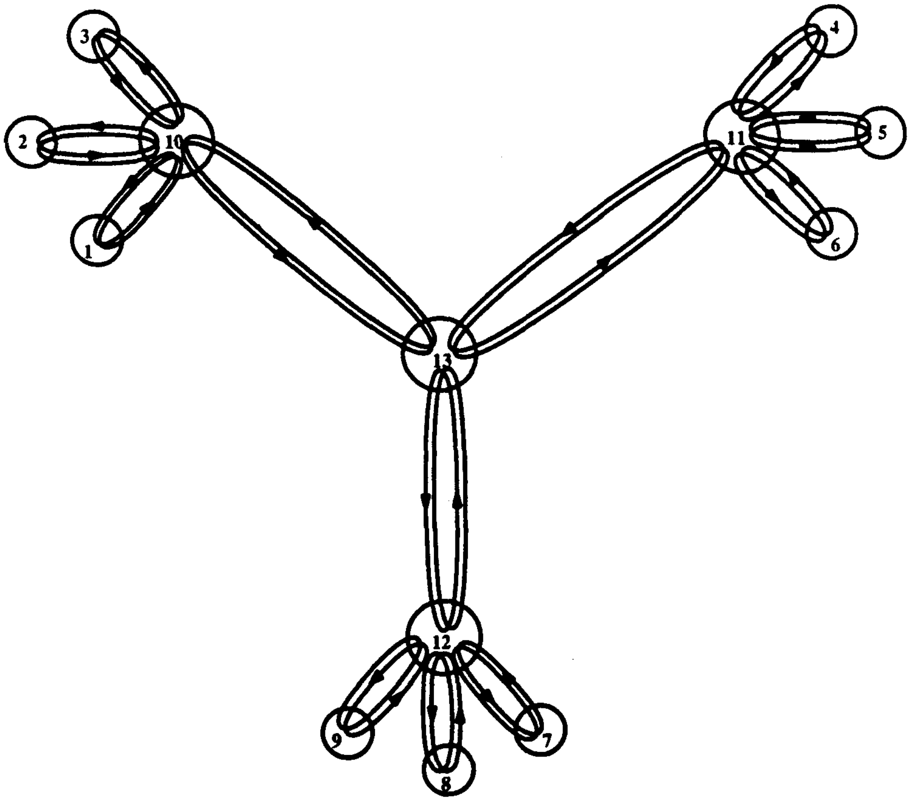


Figure 3.15: *Triangle-of-Star* topology

size and the excessive header overhead (*Blazenet*'s and high-level protocols') of small packets. For example, the combination of *Blazenet*, VMTP ([21, 41]), and IP headers could total 100 bytes. Thus, a 10 kbit packet put the headers' overhead under 10%. Moreover, 5 kbit and 10 kbit packets correspond on a 1 Gbps link to a 1 km and a 2 km *Delay Line*, or to transimssion times of 5μsec and 10μsec), respectively. Consequently, the design is a practical one.

The first simulation example is the *Star* topology with 5 inputs. A general *Star* topology with M inputs is shown in Figure 3.14. The delays as a function of network throughput, evaluated for single- and double-loop configurations, are presented in Figure 3.17. The propagation delay through the network, that is the lower limit on performance of any network, is also shown for comparison.

The second case is the *Triangle-of-Stars* topology, shown in Figure 3.15, with corresponding results in Figure 3.18.

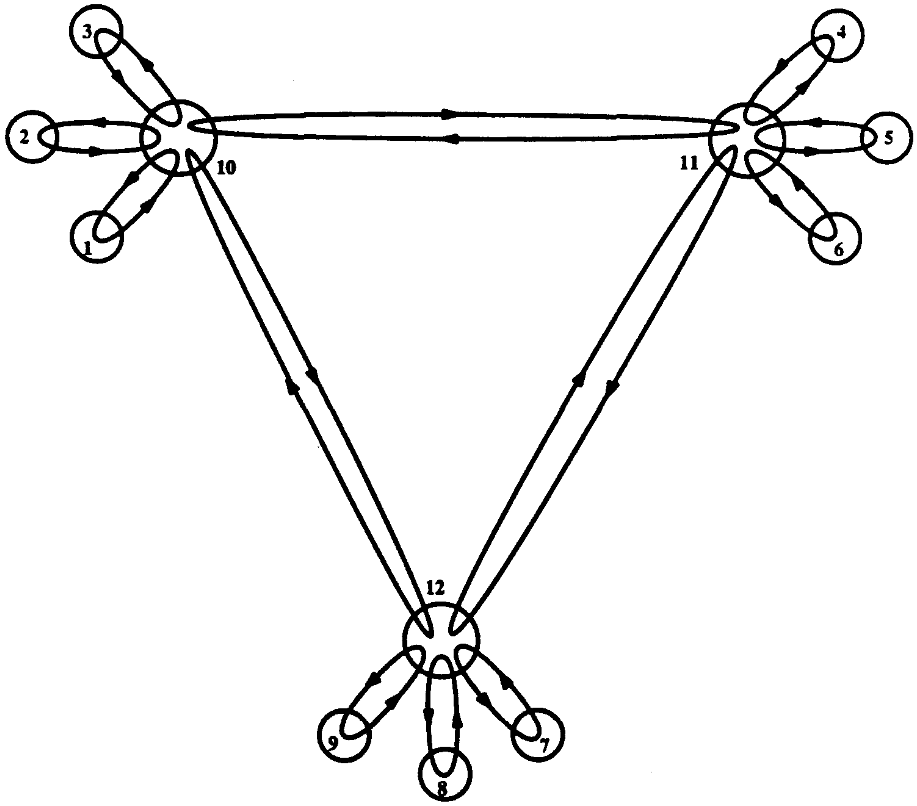
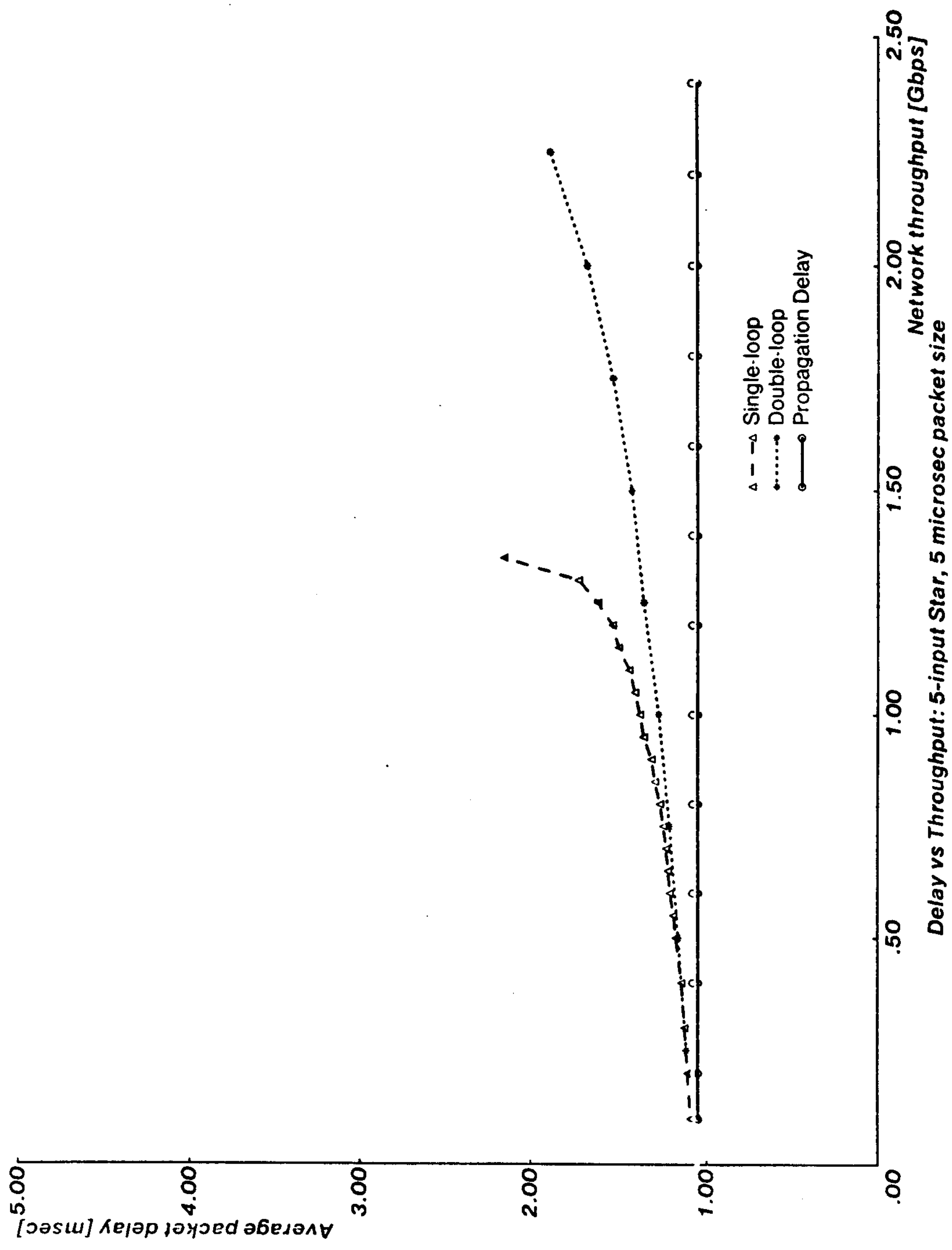


Figure 3.16: *Star-of-Stars* topology



Delay vs Throughput: 5-input Star, 5 microsec packet size

Figure 3.17: Delay of *Star Blazenet*

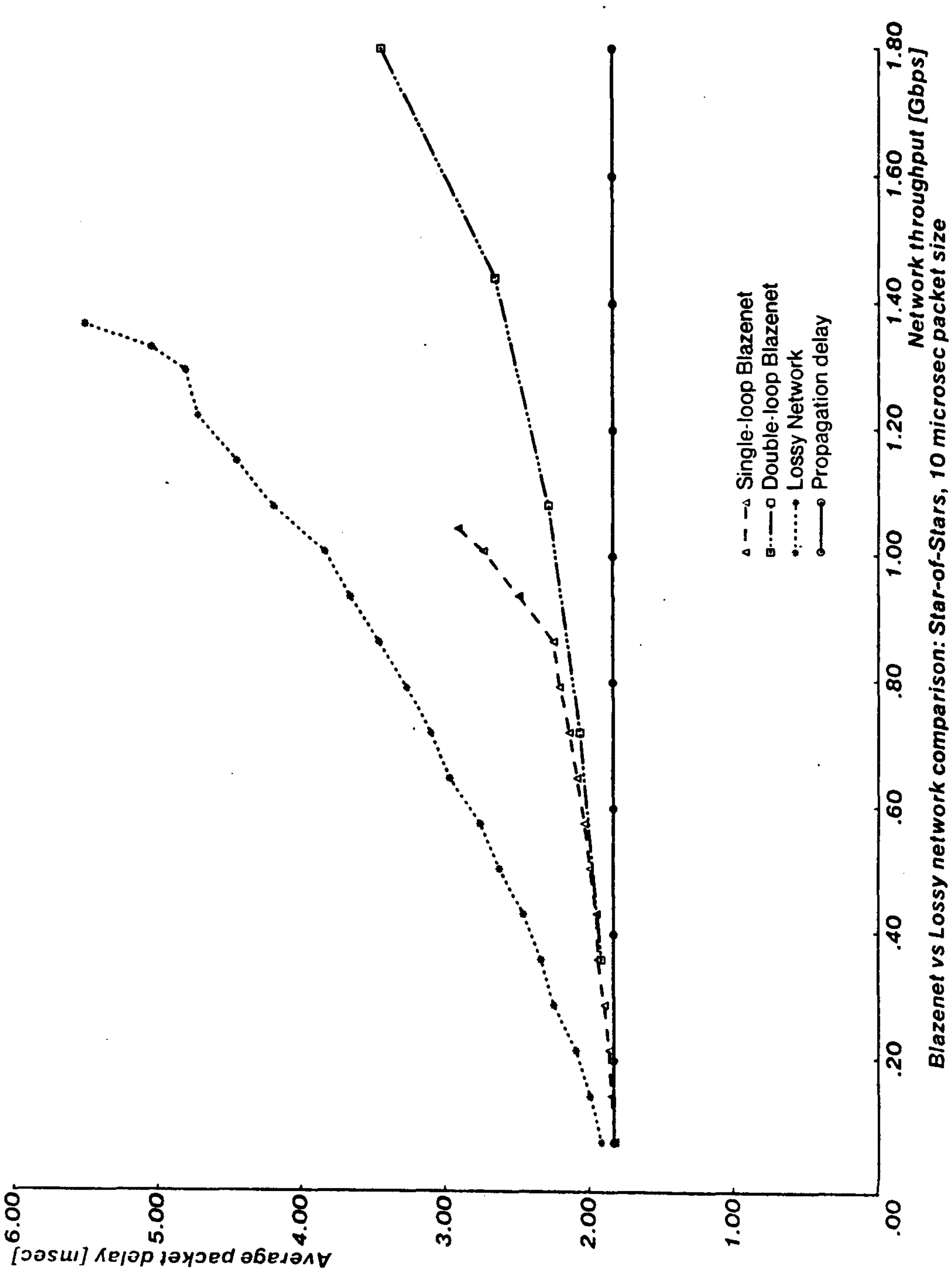


Figure 3.18: Delay of *Triangle-of-Star Blazenet*

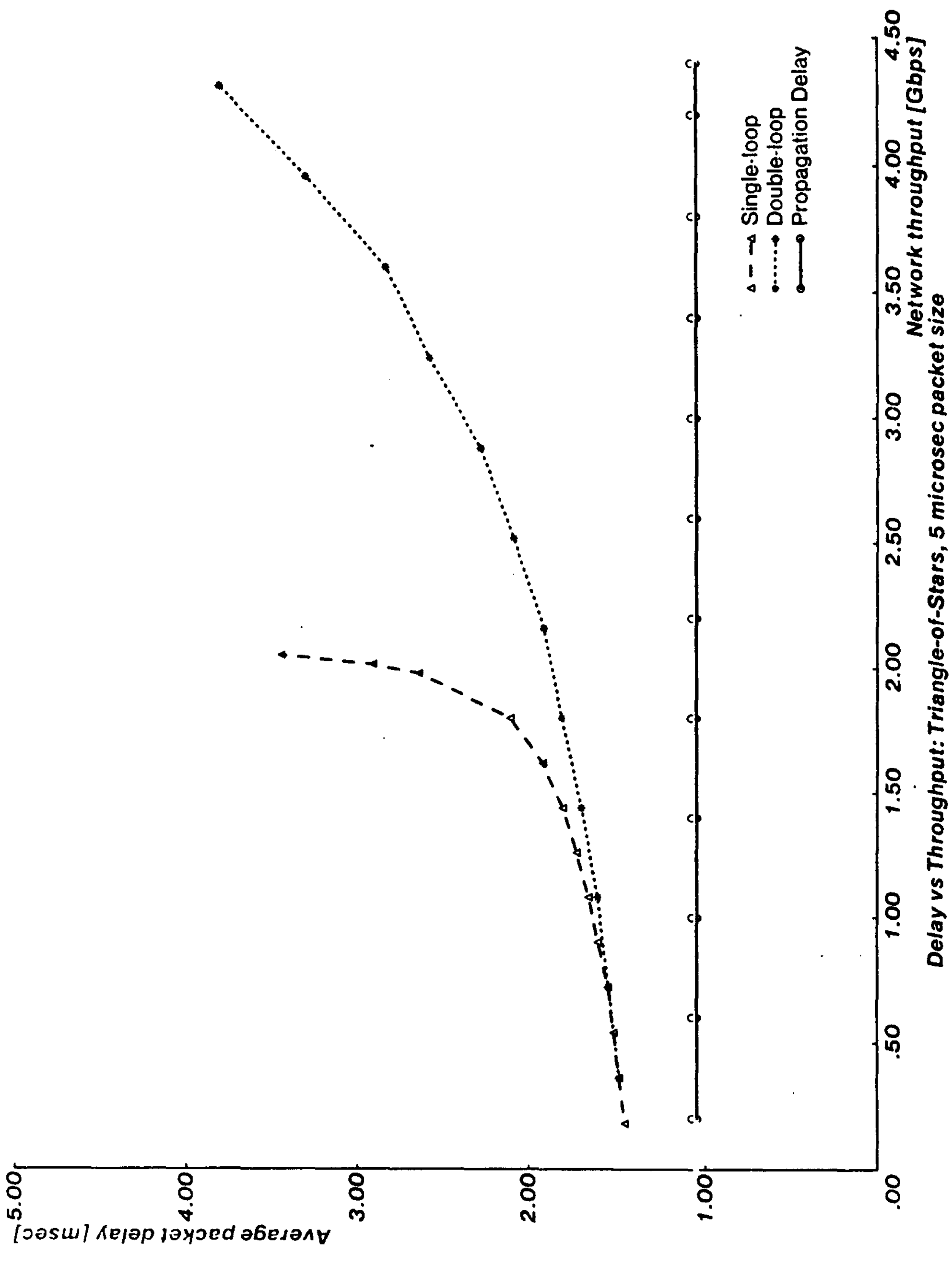


Figure 3.19: Delay of *Star-of-Stars Blazenet*

The final example is the *Star-of-Stars* topology, shown in Figure 3.16. The simulation results are presented in Figure 3.19. The comparison of *Blazenet*'s performance with that of the *Lossy* network for the *Star-of-Stars* topology is also shown in Figure 3.19. In the *Lossy* network case it is assumed that a blocked packet is retransmitted immediately after a single round trip delay between the source and the destination without any processing overhead. Thus this assumption favors the *Lossy* case. Also, the small network span somewhat favors the *Lossy* approach in this example, since *Blazenet*'s advantages are emphasized in networks with large average path length.

From these and other simulation results I conclude that the penalty in delay paid by *Blazenet* for not having conventional memory, as opposed to the ideal case (i.e., the nonblocking network), is in the range of a few tens of percents for low-load operation, which is the load for which the network is assumed to be designed. In addition *Blazenet* experiences considerably shorter delay than the *Lossy* network. Double-loop *Blazenet* does not have excessive delay or decrease in the network throughput for heavy-load operation ("Aloha-like" behavior), a behavior that single-loop configuration experience. (I believe that the future networks will offer very high bandwidth, letting the network operate in the low-load condition. For example, consider a *Blazenet* connecting a collection of 10 Mbps Ethernets operating at 10% utilization. Assume further that 25% of an Ethernet traffic is to be transferred on the backbone *Blazenet* consisting of links of ten 1 Gbps fiber each link. Thus, as many as 4000 Ethernets can coexist on *Blazenet*, utilizing only 10% of the network capacity. Such utilization is considered here as low-load operation condition.) Consequently, I consider the single- and double-loop *Blazenet* as an attractive future high-performance network design.

3.4.2 *Blazenet* analytical solution

A loop can be modeled as a feedback system, as shown in Figure 3.20. The amount of traffic entering and exiting a loop is designated by ρ. However, only part of the entering traffic makes it through a loop at the first attempt. The blocked part is returned on the reverse portion of the loop, joining the new incoming traffic. Thus the traffic on a loop, labeled δ and referred to here as the offered traffic, is larger than

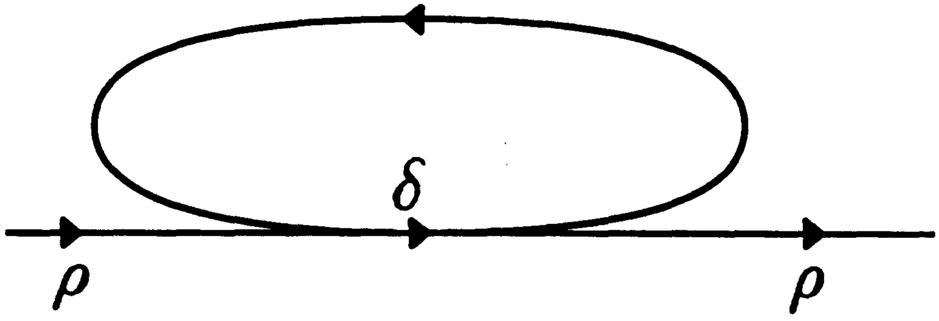


Figure 3.20: Loop as a feedback system

the actual loop throughput ρ. The behavior is similar to the behavior of Aloha, in which colliding (blocked) traffic is discarded and retransmitted.

The value of the offered traffic, δ, is a function of the throughput of a loop. It is an increasing, alas nonlinear, function of ρ. The capacity of a loop (defined as the maximum throughput of a loop) is determined by the maximal value of δ that still yields a stable solution.

The average delay of a loop is evaluated by calculating the average number of blockages a packet undergoes in a single attempt to cross a switching node. This average number of blockages is easily computed from the probability that a packet will make it through a switching node on a single attempt, assuming independence between consecutive attempts of a packet to cross a switch. In order to calculate this probability, the value of δ is needed.

Thus the solution of a loop consists of solving for the function $\delta(\rho)$. Once this function is known, the capacity and the delay of the loop can be calculated. The value of δ for a particular loop depends (aside from the major dependence on ρ) on the values of δ's for all adjacent loops that forward traffic to the loop in question. (Adjacent loops are loops having a common switching node or repeater/router between them.) Therefore, in general, a solution of a network (finding the network capacity and the average packet delay) consists of solving a set of $2L$ non-linear equations, where L is the total number of loops in the network. In some special cases, however, because of symmetry or independence in the network traffic and topology, a network solution can be obtained by solving a subset of these $2L$ equations.

Figure 3.21 shows an example of a single hop within a network. It consists of five sub-loops with two switching nodes on both sides of the hop. Each one of the switching nodes is assumed to connect M loops. It is also assumed that the traffic matrix of each switching node is fully symmetrical, and that the next and previous switching nodes have the same topology and traffic pattern. The solution in this case can be obtained separately for the single hop, without referring to the rest of the network. In the steady state, the traffic passing through each one of the sub-loops is equal, and, as before, labeled ρ. The actual traffic on each one of the sub-loops is, however, different. This internal traffic is labeled $\delta_k^{U/D}$, where k represents the number of the sub-loop and U/D indicates the direction of the traffic, U for "up" and D for "down." Note that the δ's represent the traffic associated with some direction on a portion of a loop; it is not the total traffic on that portion of the loop (which, in general, consists also of the blocked traffic in the opposite direction). In each switching node and in each repeater/router, some of the traffic that tries to make it through the switch is blocked, returned, and combines with the traffic in the opposite direction. This traffic in the opposite direction consists of traffic blocked at a switch at the other end of the loop plus the new incoming traffic ρ. Solution of the set of $2L$ equations in the variables $\{\delta_k^U, \delta_k^D;\ k = 1, ..., L\}$ is the first step to obtain the network solution. Once the variables are known, the probability of a packet blockage and the average number of blockages in each switching node can be calculated. Thus the average packet delay can be evaluated. The following are the 10 equations in the variables $\{\delta_k^U, \delta_k^D;\ k = 1, ..., 5\}$:

$$\delta_5^D \cdot (1-(\delta_1^U-\rho)-(\delta_1^D-\rho)) \cdot (1+(M-2)\cdot(1-\frac{\delta_5^D}{(M-1)})^{M-2}) = \rho \cdot (M-1)\,, \quad (3.1)$$

$$\delta_1^U \cdot (1-(\delta_5^D-\rho)-(\delta_5^U-\rho)) \cdot (1+(M-2)\cdot(1-\frac{\delta_1^U}{(M-1)})^{M-2}) = \rho \cdot (M-1)\,, \quad (3.2)$$

$$\delta_1^D \cdot (\delta_2^U - \rho + \delta_2^D - \rho) = \rho\,, \quad (3.3)$$

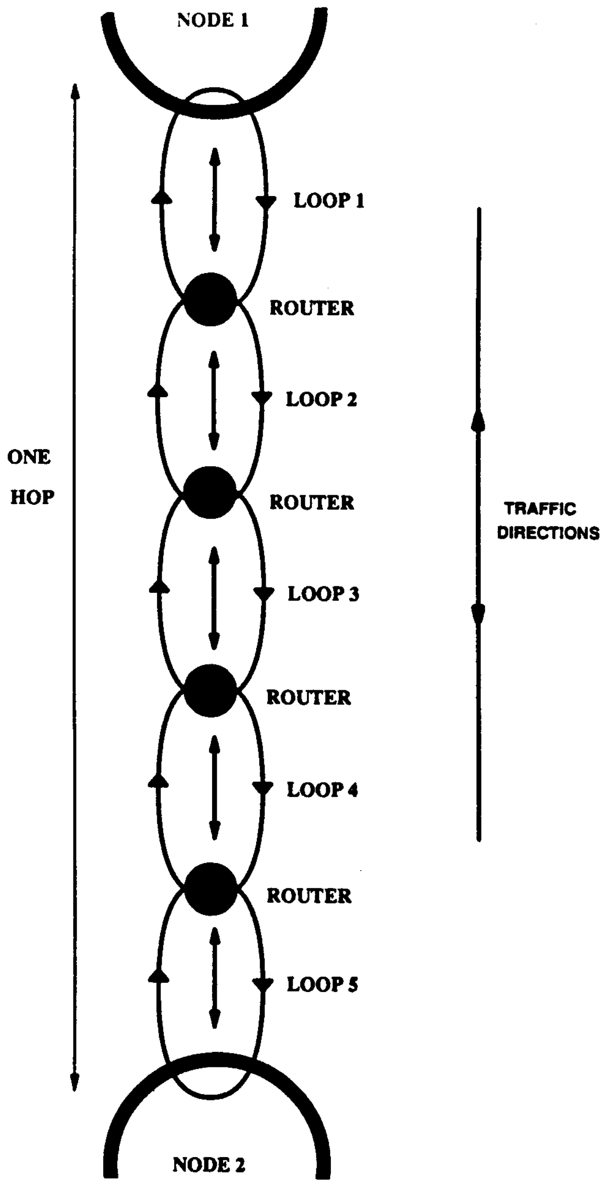


Figure 3.21: Example of a hop composed of five single-sub-loops

$$\delta_2^D \cdot (\delta_3^U - \rho + \delta_3^D - \rho) = \rho\,, \tag{3.4}$$

$$\delta_3^D \cdot (\delta_4^U - \rho + \delta_4^D - \rho) = \rho\,, \tag{3.5}$$

$$\delta_4^D \cdot (\delta_5^U - \rho + \delta_5^D - \rho) = \rho\,, \tag{3.6}$$

$$\delta_2^U \cdot (\delta_1^D - \rho + \delta_1^U - \rho) = \rho\,, \tag{3.7}$$

$$\delta_3^U \cdot (\delta_2^D - \rho + \delta_2^U - \rho) = \rho\,, \tag{3.8}$$

$$\delta_4^U \cdot (\delta_3^D - \rho + \delta_3^U - \rho) = \rho\,, \tag{3.9}$$

$$\delta_5^U \cdot (\delta_4^D - \rho + \delta_4^U - \rho) = \rho. \tag{3.10}$$

The term $(1 - \frac{\delta_5^D}{(M-1)})^{M-2})/(M-1)$ in equations (3.1) and (3.2) expresses the probability that a packet makes it through a switching node with M input/output loops on a single attempt, and is developed in Appendix E. (The above ten equations can be compressed into five equations because of the symmetry on both sides of the path. That is, $\delta_1^U = \delta_5^D$, $\delta_1^D = \delta_5^U$, $\delta_2^U = \delta_4^D$, $\delta_2^D = \delta_4^U$, $\delta_3^U = \delta_3^D$.)

The set of equations was solved by MACSYMA ([67]) for the whole range of ρ. The values of the variables were used to compute the average number of blockages in every repeater. This was done by first solving for the probability of crossing a switch. For example, the average number of blockages in repeater no.1 is calculated by the following: First obtain the probability of crossing the repeater no.1 by the following formula:

$$\text{Probability of crossing repeater no.1} = \frac{1}{(\delta_1^D - \rho + \delta_1^U)}. \tag{3.11}$$

The average number of blockages is now obtained by using the fact that is is equal to the reciprocal of the probability of crossing the switch. Thus:

$$\text{Average number of blockages in repeater no.1} = (\delta_1^D - \rho + \delta_1^U). \tag{3.12}$$

Average number of blockages in the repeaters 2, 3, and 4 is calculated in a similar way. The average number of blockages in repeater no.5 is computed by the following:

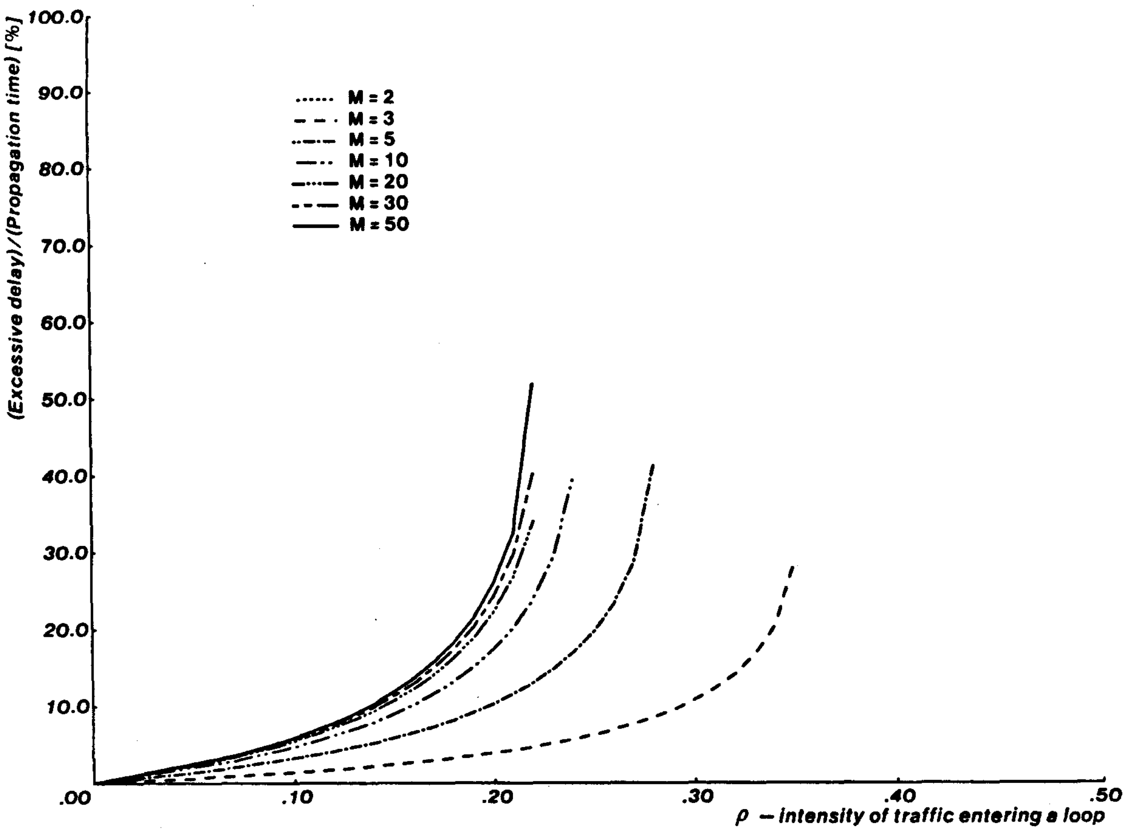


Figure 3.22: Excessive packet delay of a switch with M input/output single-loops

Average number of blockages in repeater no.5 =

$$\frac{1}{\text{Probability of crossing repeater no.5}} = \frac{(1-\frac{\delta_5^D}{2\cdot(M-1)})^{M-2}}{(M-1)}. \tag{3.13}$$

Figure 3.22 shows the excessive packet delay as a function of ρ. The excessive packet delay is the delay a packet experiences in addition to the propagation delay alone, and is computed for each sub-loop directly from the average number of blockages in each repeater.

A similar treatment can be performed for the double-loop *Blazenet* configuration. The solution of one hop of the double-loop *Blazenet* shown in Figure 3.23 is shown in Figure 3.24.

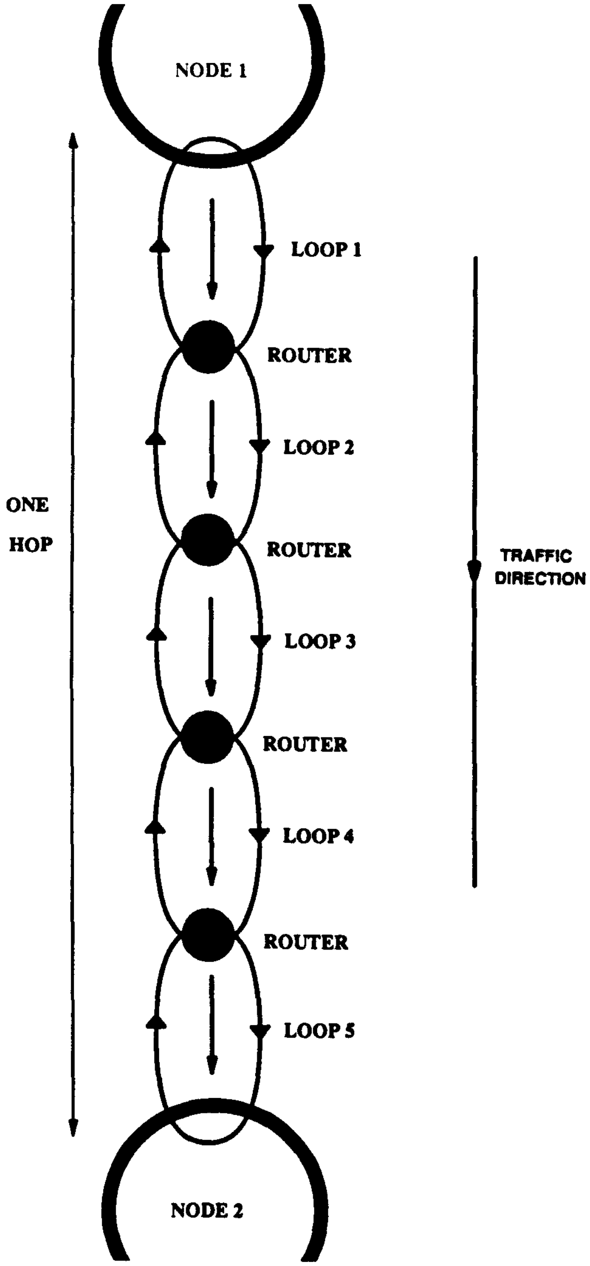


Figure 3.23: Example of a hop composed of five double-sub-loops

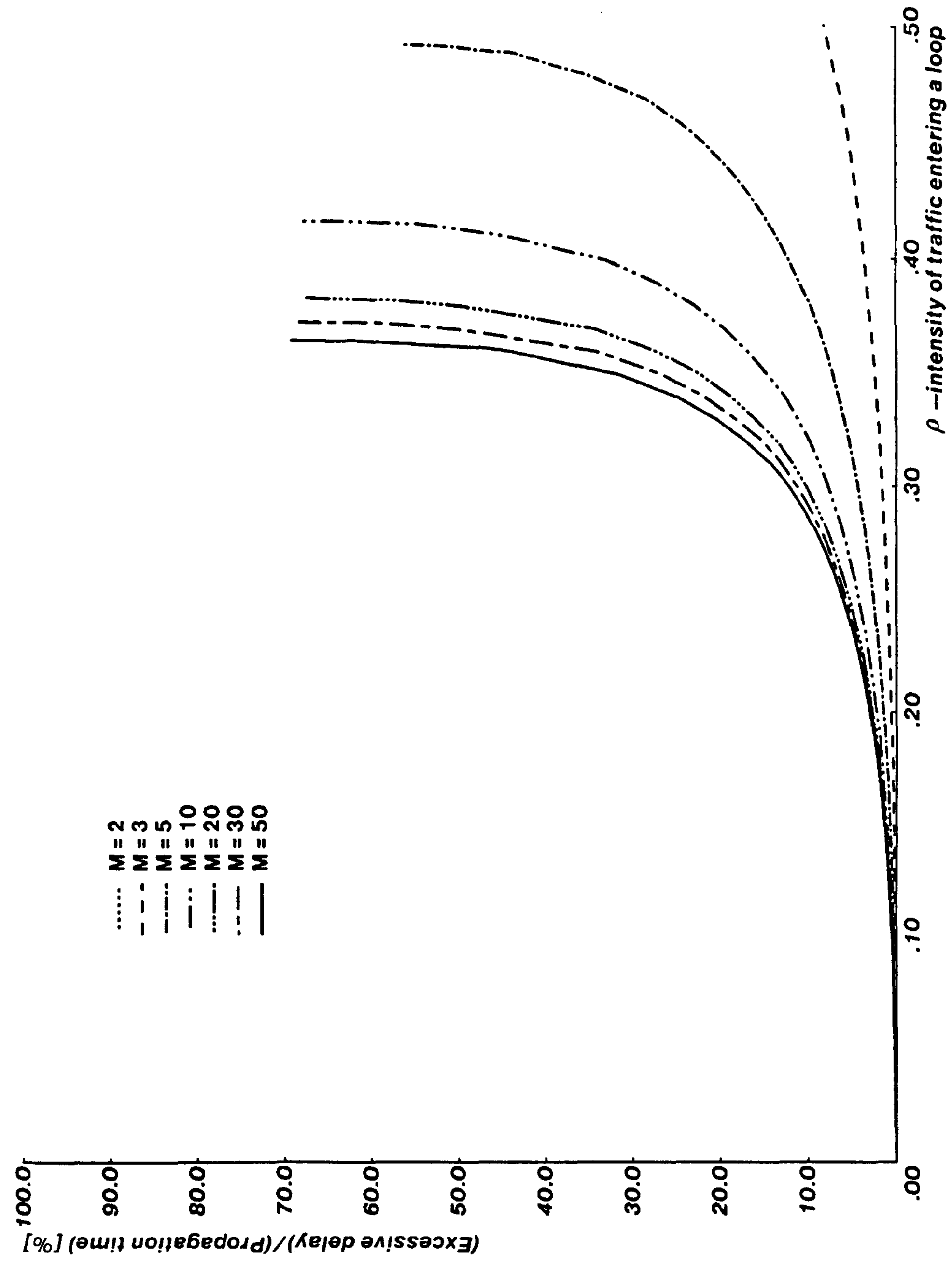


Figure 3.24: Excessive packet delay of a switch with M input/output double-loops

As will be seen, the switching node (rather than the repeaters/routers) is the "bottleneck" in the flow of the packets on a path composed of a number of consecutive sub-loops. This means that the loops close to the switching node are more populated (i.e., in the previous example $\delta_5^D > \delta_4^D > \delta_3^D > \delta_2^D > \delta_1^D$). Consequently, the capacity of a path in a network is limited by the capacity of the switching nodes. The capacity of a switching node depends on the number of input/output loops connected to the switch, and equals the value of ρ that corresponds to the maxima of the function $\delta(\rho)$. The function is determined by the following equation:

$$\delta_5^D \cdot (1 - (\delta_1^U - \rho) - (\delta_1^D - \rho)) \cdot (1 + (M-2) \cdot (1 - \frac{\delta_5^D}{(M-1)})^{M-2}) =$$

$$\rho \cdot (M-1)\,, \tag{3.14}$$

where δ_1^U, δ_1^D, δ_5^U, and δ_5^D for single-loop *Blazenet* are defined in Figure 3.25. For the double-loop case all δ's in the non-active direction equal zero. For example, if the loop traffic is "Down," then $\delta_1^U = 0$, $\delta_2^U = 0$, $\delta_3^U = 0$, etc. The function $\delta(\rho)$ for the example in the double-loop configuration is determined by the following equation:

$$\delta_5^D \cdot (1 - (\delta_1^D - \rho)) \cdot (1 + (M-2) \cdot (1 - \frac{\delta_5^D}{(M-1)})^{M-2}) =$$

$$\rho \cdot (M-1). \tag{3.15}$$

The graphs in Figures 3.26 and 3.27 show the function $\delta(\rho)$ for the single- and double-loop *Blazenet* configuration, respectively. In both cases it is assumed that the hop consists of a single loop. Also, for the single-loop configuration, it is assumed that the traffic on all loops is equal and that the traffic from each loop is symmetrically destined to all the other loops. Likewise, for the double-loop configuration, it is assumed that all the traffic on the input-loops is equal and that it is symmetrically destined to all the output loops. As can be seen the capacity of the single-loop *Blazenet* is about 17% as $M \to \infty$. The corresponding capacity for the double-loop configuration is about 23%.

The graphs on Figures 3.28 and 3.29 show the function $\delta(\rho)$ for the same configurations. However, now the number of sub-loops on each hop is five. A significant

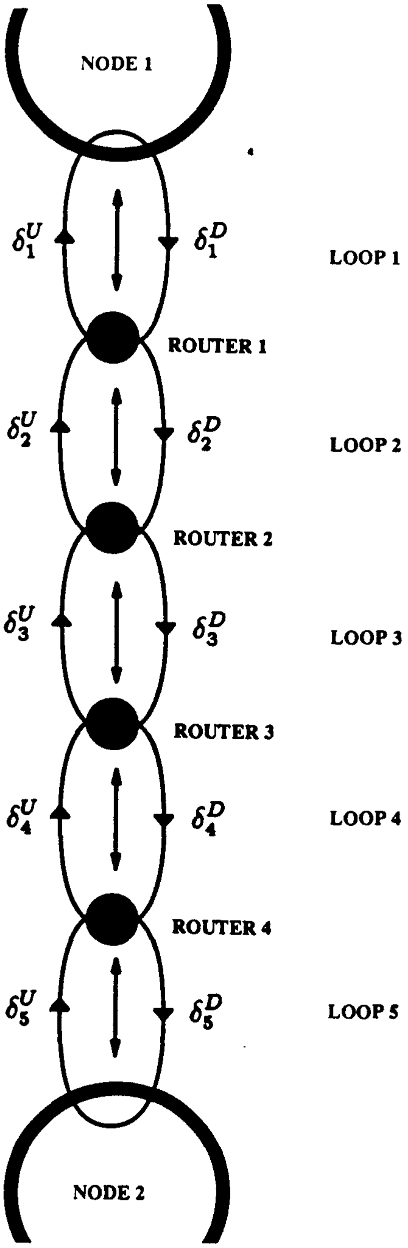


Figure 3.25: Definition of variables for a switch capacity calculation

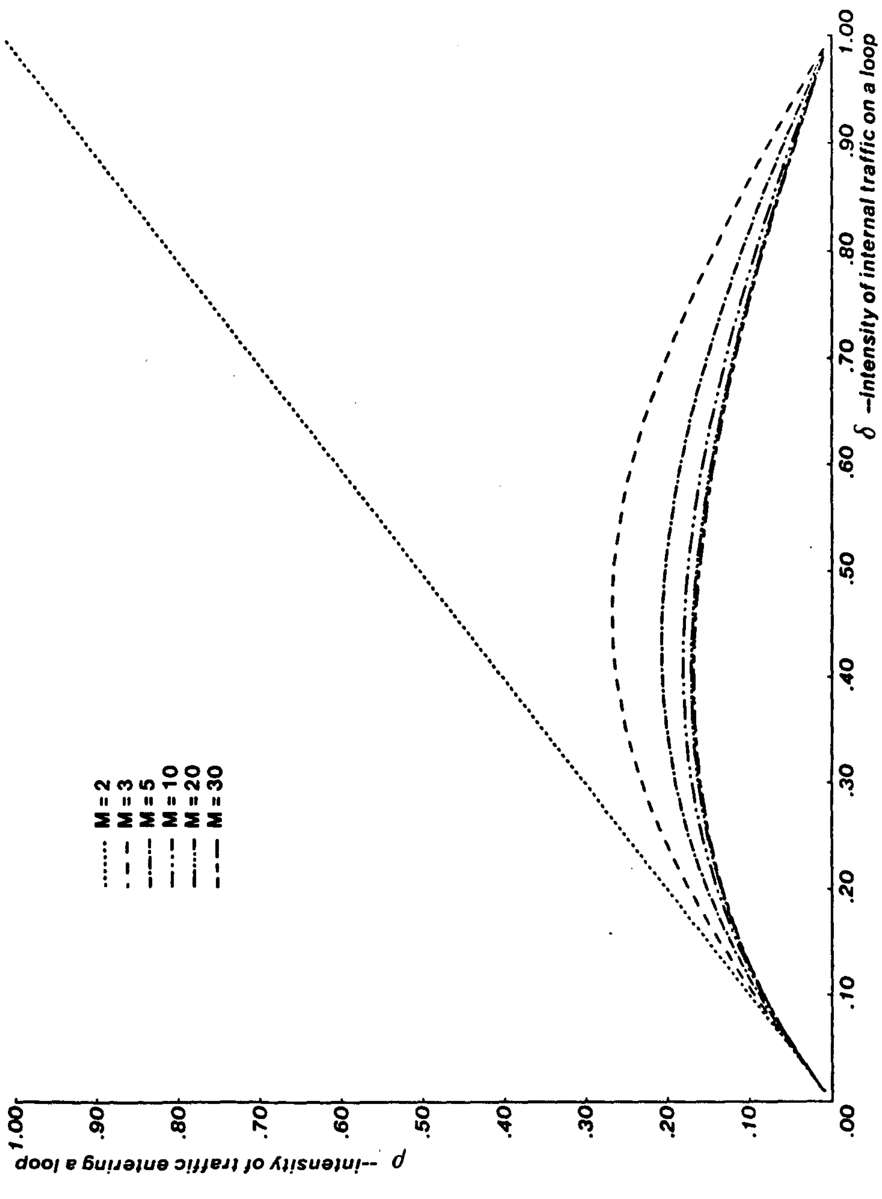


Figure 3.26: The function $\delta(\rho)$ for a one-loop hop in the single-loop configuration

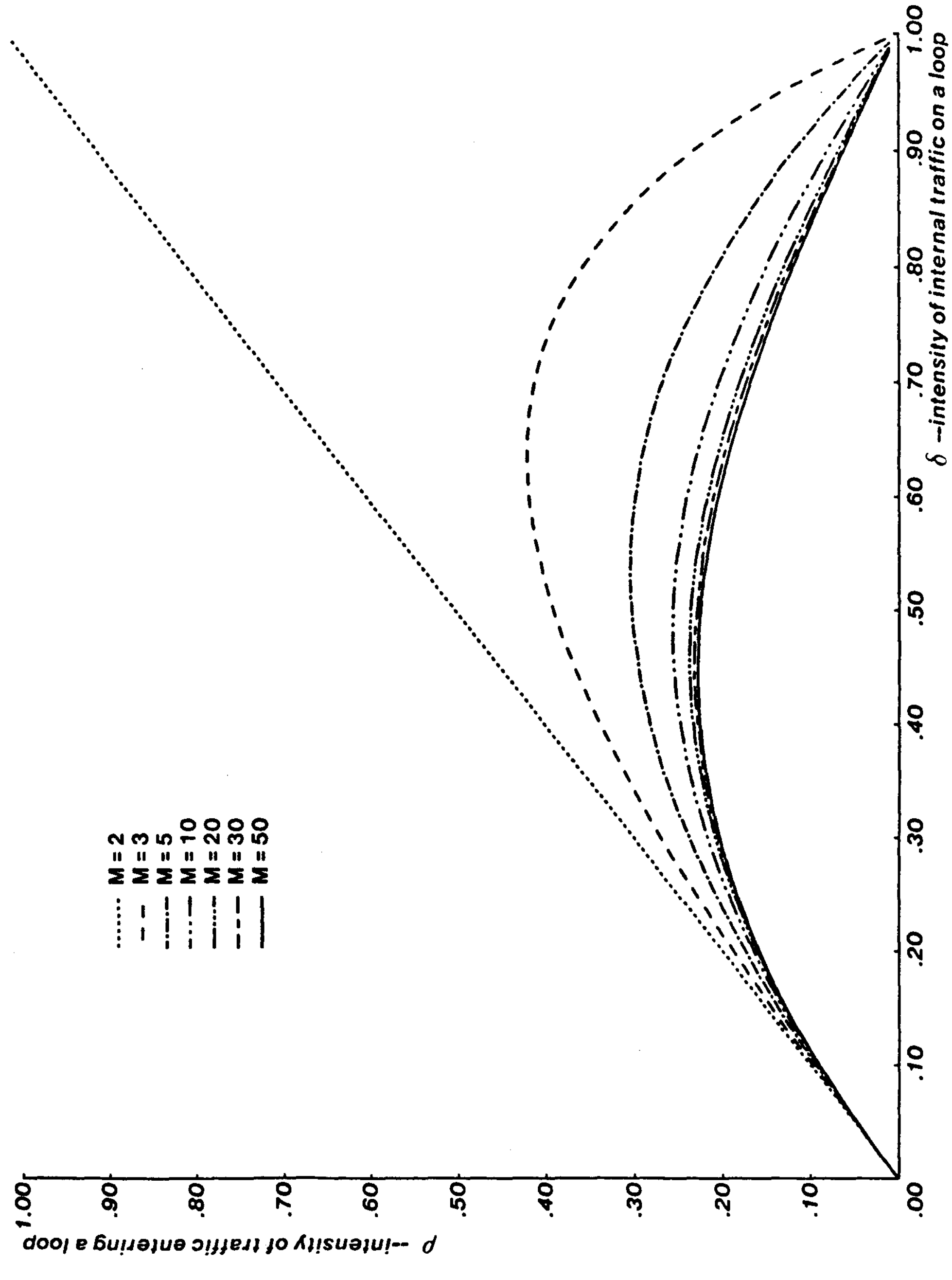


Figure 3.27: The function $\delta(\rho)$ for a one-loop hop in the double-loop configuration

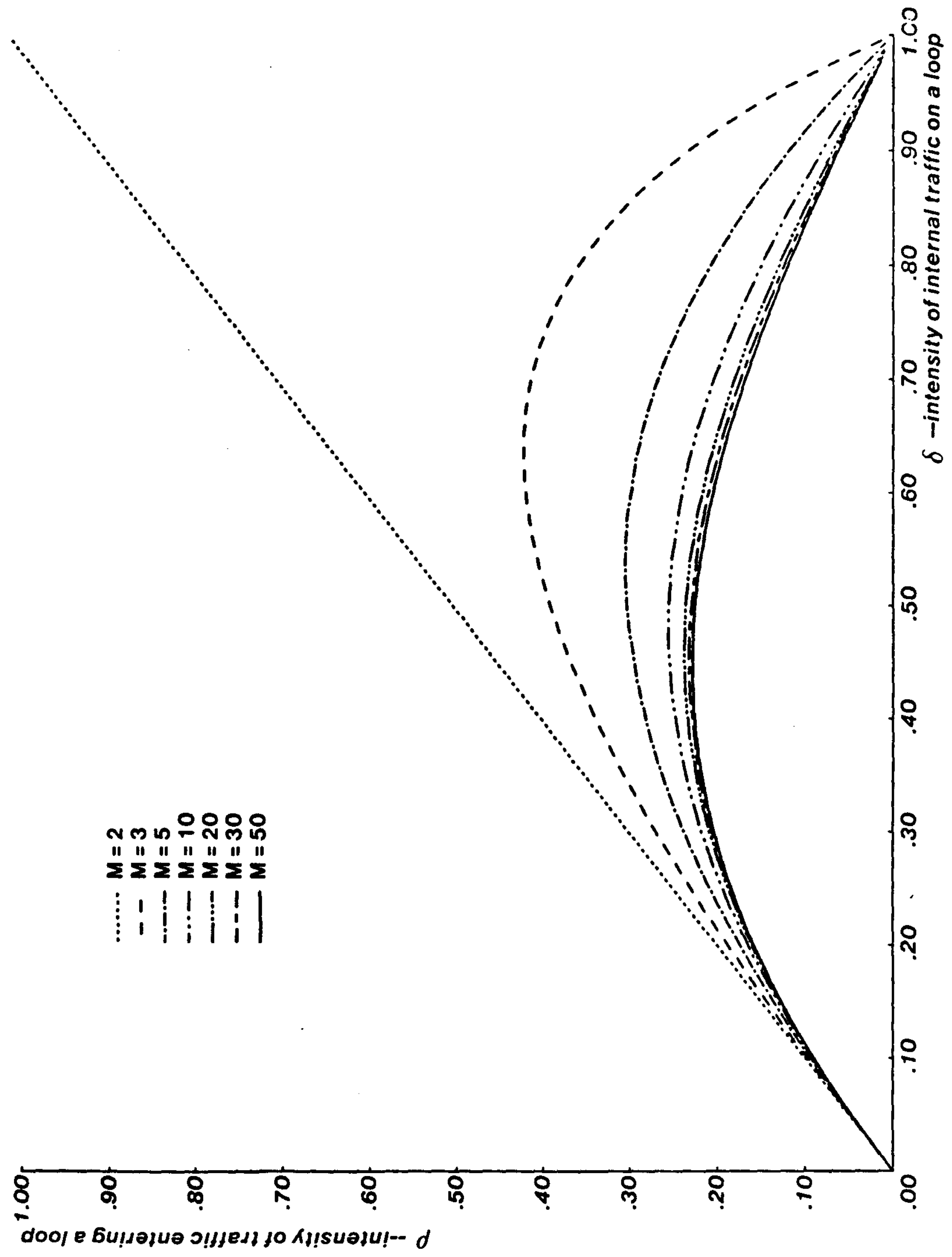


Figure 3.28: The function $\delta(\rho)$ for a hop with many sub-loops in the single-loop configuration

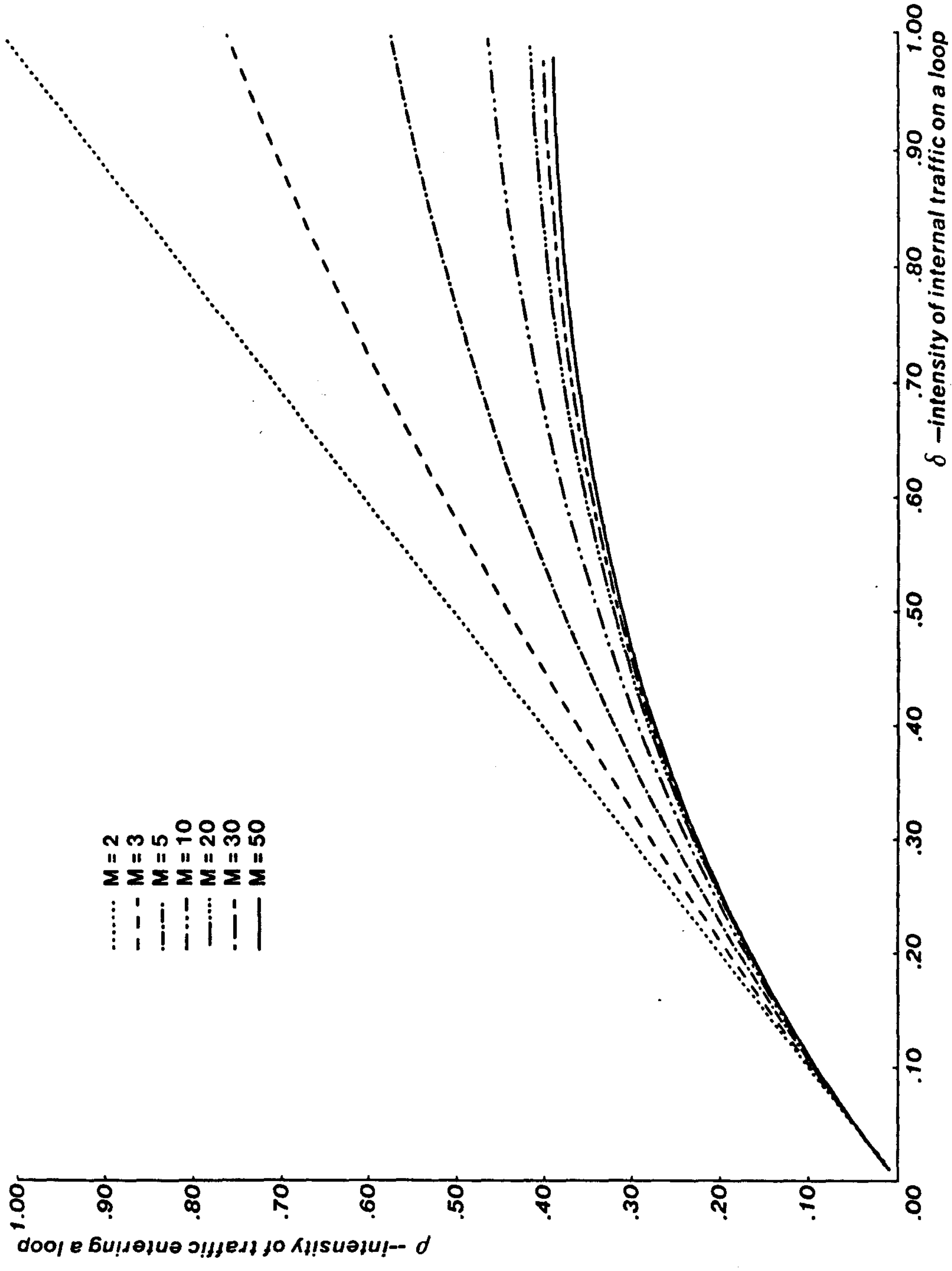


Figure 3.29: The function $\delta(\rho)$ for a hop with many sub-loops in the double-loop configuration

improvement can be observed, leading to the capacity of 23% for the single-loop configuration and the capacity of 38% in the double-loop case.

Several conclusions can be drawn from these examples. By increasing the number of sub-loops on the network hops, significant increase in the network capacity can be achieved. Increasing the number of sub-loops also decreases the average packet delay. (Also, the length of the last sub-loop—the one connected to the switch—is more important than the length of the others; by decreasing the last sub-loop length, the delay can be decreased.) The required number of sub-loops in the network hops has to be calculated for every specific case. This number depends on the demanded performance and on the required utilization of the loops, ρ.

From the graphs in Figures 3.26, 3.27, 3.28, and 3.29 it follows that increasing the number of input/output loops to a switch decreases the switch capacity. Since a repeater/router connects only two input/output loops, it follows that the capacity of a repeater/router is larger than the capacity of any switching node connecting at least three input/output loops. Consequently, the switching nodes are the "bottlenecks" for traffic flow.

3.5 Extended features

Section 3.3 presented the basic *Blazenet* and its node design. This Section expands on the basic node design to include some of the more sophisticated features: priority traffic, limiting of the life-time of a packet, broadcast and multicast, and network monitoring. Besides providing very important services to the network users, these features increase the ability of the network to cope with abnormal situations, increasing, therefore, the network's reliability.

3.5.1 Priority traffic

In *Blazenet*, traffic priority can be implemented in two ways: by including a priority field in the packet format, or by giving preference to some traffic during the forwarding process. The former approach is considered first.

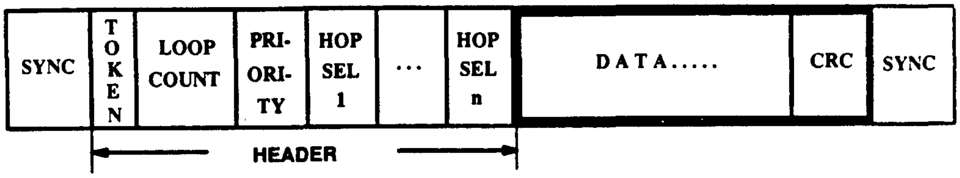


Figure 3.30: Modified packet format

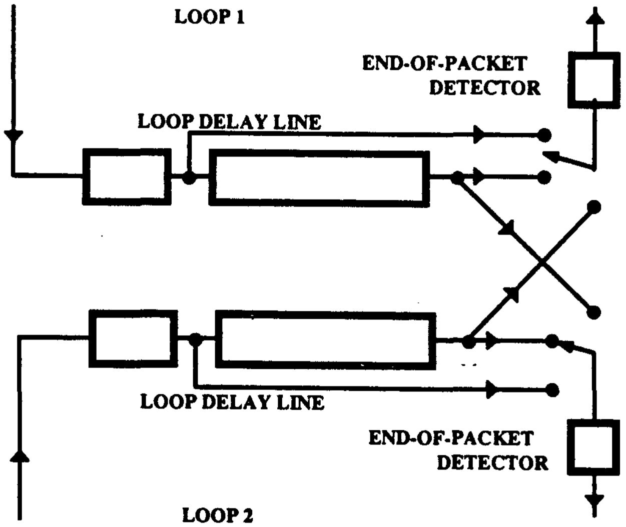


Figure 3.31: Modified node design

The packet format with the *priority* field is presented in Figure 3.30. In this method priority is implemented by delaying the forwarding of a packet by a period equal to the transmission time of a maximum packet length. At the end of this period, the packet with the highest priority is forwarded, while other packets (if any) are looped back. The hardware of the basic *Blazenet* node design has to be modified in order to accommodate this additional feature. The main adjustment is to include within the *Delay Line* a packet detector circuit that initiates the *packet-ready* signal. The modified node design is shown in Figure 3.31 and the modified *Delay Line* in Figure 3.32.

A packet that is clocked into a *Delay Line* and has not reached the *packet-ready* point is called an active packet. The set of active packets at any given time is the set of packets competing on the loops.

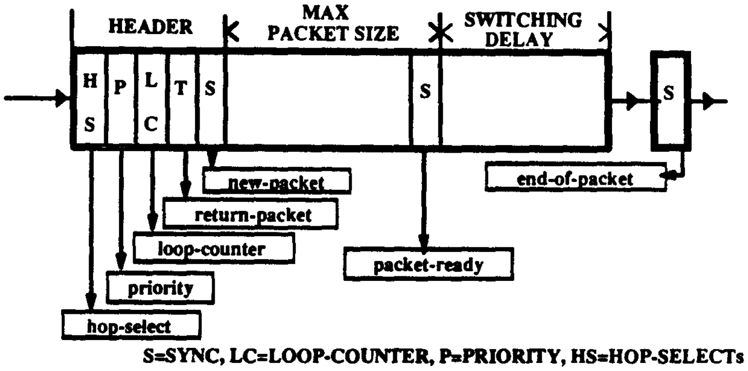


Figure 3.32: Modified *Delay Line* structure

The new packet arriving on a *Loop* is clocked into the *Loop*'s *Delay Line.* After its *main-header* (composed of the fields: *sync*, *token*, *loopcounter*, *priority*, and the *hop-selects*) are received, the *Control* is notified of the packet's arrival and the packet *main-header* information is passed on to the *Control.* The *Control* gathers this information from all the *Delay Lines.* When a packet is shifted to the *packet-ready* point in the *Delay Line*, the decision is made whether the packet will be forwarded or looped back. The decision is made according to the following algorithm:

```
IF ( (priority $\geq$ priority of all active packets
        with the same hop-select)
    AND (no transmission in progress)
    AND (destination Delay Line is free) )
THEN forward the packet
ELSE loop the packet back;
```

The forwarding or blocking (namely the switching) operations are performed as before.

The essence of the above procedure is that, by delaying all packets by one packet length (i.e., by having a one packet look-ahead), the priorities of all the relevant packets can be gathered and the correct decision about which packet to forward can be made. Therefore, the only difference in this modified version of *Blazenet* is the instant in time when the *Control* decision is made.

Using the scheme mentioned above, the delay of a forwarded packet is increased by the transmission time of a packet of the maximum size. However, this time is negligible compared to the propagation delay encountered by a packet on a link in a wide-area network. (For example, for 10 kbit packets on 1 Gbps *Blazenet* the additional delay is only 10 μsec, a delay that is small compared to the 500 μsec propagation time of a 100 km link.) The total delay is, therefore, essentially unaffected by this hardware modification.

Another way to implement priority traffic in *Blazenet* is to give preference to some traffic during the forwarding process. One possibility is to prefer always the traffic coming from the hosts connected to the node over all the other traffic. Such a mechanism is useful for coping with temporary traffic surges from the node's hosts. However, although this approach lowers the delay of the preferred traffic, it increases mean packet delay in the whole network. Therefore, in order to ensure fairness, usage of such a mechanism should be restricted.

The preference given to some traffic can be based on other criteria. Traffic arriving at some loops (for example, traffic coming from congested areas) may be given higher priority in the forwarding process. The preference criteria can be based on various network parameters and can be adjustable in time, as the network load and topology change.

3.5.2 Limiting packet's lifetime

The network needs to limit packet lifetime for three reasons: to eliminate erroneous traffic that exists in the network and interferes with valid traffic, to discard real-time traffic that could not be delivered on time and became obsolete, and to avoid wrap-around of packet sequence numbers in high-level protocols.

In *Blazenet*, the *loop-counter* provides the mechanism for limiting the lifetime of packets within the network. The *loop-counter* is decreased each time a packet is blocked and returned. When the *loop-counter* reaches zero, the packet is discarded. The *loop-counter* represents, therefore, the maximum number of times a packet can loop back. The value of the *loop-counter* is set by the source host, according to packet type and time limitations for packet delivery.

If the loops are of equal length, the *loop-counter* mechanism provides an accurate means for limiting a packet's lifetime within the network. If the loops are of unequal length, using the minimum loop length of the packet path for the calculation of the *loop-counter* can be an adequate approach. Let n represent the refractive index of the fiber, w_{min} [km] the minimum loop length of the packet path (= twice the distance between the adjacent switching nodes), w_i [km] the length of the i^{th} loop, h number of hops on the packet path (= number of switching nodes on the path $-$ 1), t_{min} [sec] the minimum lifetime of a packet in the network, and c [km/sec] the speed of light in vacuum. Then, the value of the *loop-counter* can be calculated using the equation:

$$\textit{loop-counter} \geq \frac{c \cdot t_{min}}{n \cdot w_{min}} - \frac{1}{2 \cdot w_{min}} \cdot \sum_{i=1}^{h} w_i. \tag{3.16}$$

Another approach would be to use some weighted average of the loop lengths of the packet path, w_{avg} [km]. In this case the *loop-counter* is calculated by the formula given above, after substituting w_{avg} for w_{min}.

If the general repeater/switching node design is used, all network loops are of equal length, w [km] (possibly with the exception of the last loop encountering the node), and the calculation of the required value of the *loop-counter* becomes:

$$\textit{loop-counter} \geq \frac{c \cdot t_{min}}{n \cdot w} - \frac{h}{2}. \tag{3.17}$$

If some minimum value, t_{min} [sec], is imposed on the packet lifetime, the packet is not discarded for at least this period of time, unless for reasons other than lifetime expiration. This is advantageous in situations where the network designer is more concerned with the possibility of discarding a still valid packet, than with the possibility of an obsolete packet living in the network or even being passed on to the destination. In the opposite case, namely the case when the network designer is more concerned with the excessive load created by obsolete traffic than with the possibility of discarding valid traffic, he should use some maximum permissible value for the packet lifetime, t_{max}, instead. (The above formulas continue to be valid in this case,

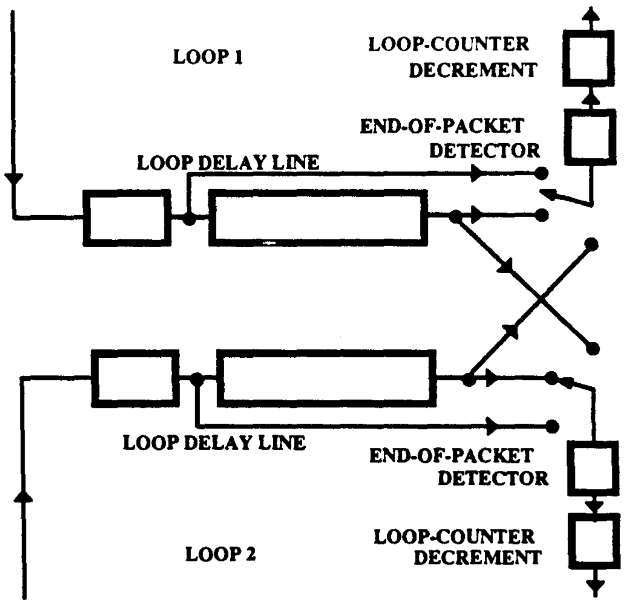
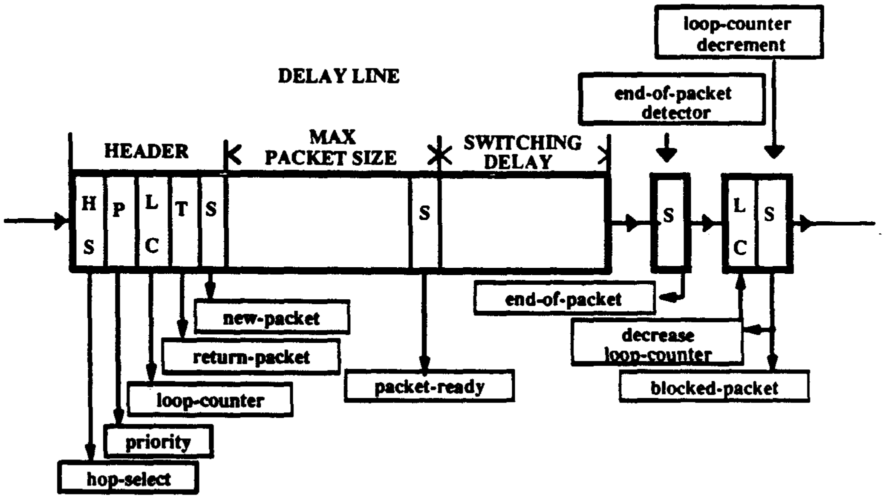


Figure 3.33: Node design with *loop-counter* implementation

with the substitution of maximum loop length w_{max} for the w_{min} and reversal of the unequality sign.) Note, that by manipulating the current value of the packet lifetime, the network can regulate its load. However, such a manipulation is justified only in some special circumstances and for traffic that does not require reliable transport through the network.

The implementation of the *loop-counter* mechanism includes a decrement circuit. This circuit, as well as the circuit that tests the value of the *loop-counter*, operates on returned packets only. No action is necessary when a packet is forwarded. The modified node design that includes the implementation of the *loop-counter* is presented in Figure 3.33. The structure of a *Delay Line* is shown in Figure 3.34. While the packet enters a *Delay Line* the *loop-counter* is checked by the *Control.* If the value of the *loop-counter* is zero, the packet is discarded by connecting the output of the *Delay Line* to ground. The other possibility is to pass the packet to a special host (called Monitor), which monitors the the switching node operation.

As a blocked packet is clocked out of the *Delay Line*, the *blocked-packet* circuit detects the *sync* and the *token* of the packet, which together initiate a *decrease-loop-counter* signal if the *token* is set. The delay between the *blocked-packet* and the *decrease-loop-counter* circuits is exactly such that when the *blocked-packet* signal is raised, the *loop-counter* is received by the *decrease-loop-counter* circuit. The operation



S=SYNC,T=TOKEN,LC=LOOP-COUNTER,P=PRIORITY,HS=HOP-SELECTs

Figure 3.34: *Delay Line* with *loop-counter* implementation

is, therefore, fully autonomous, not requiring any intervention of the *Control*.

When the *loop-counter* is represented by a binary number, the hardware needed for the *loop-counter* decrement may be difficult to implement. A somewhat easier solution may be to use a bit pattern as a *loop-counter*. In this scheme the *loop-counter* is composed of a string of 1's. The number of 1's is equal in number to the required value of the *loop-counter*. Each decrement of the *loop-counter* consists now of resetting one such bit. An all zero pattern indicates the value of zero of the *loop-counter*. This scheme has the disadvantage of providing an unnecessarily long *loop-counter* field. Fortunately, the maximum value of the *loop-counter* is expected to be small. Consequently, the ease of implementation justifies the bit wastage.

Yet another approach to the *loop-counter* usage is to provide a special *loop-counter* for each hop. In this case, instead of the *hop-selects* fields, the packet header contains fields composed of *hop-selects* and *loop-counters*. The advantage of this scheme is the possibility of an exact calculation of the packet's lifetime, as well as the possibility of selectively limiting the delay of each of the loops on the packet's path.

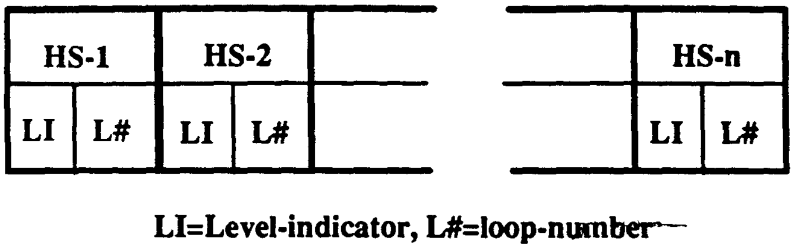


Figure 3.35: Modified *hop-select* structure for multicast delivery

3.5.3 Broadcast and multicast

Multicast refers to sending a packet to multiple destinations by a single transmission from the packet source. The motivation for *Blazenet*'s multicast is to provide *host groups*, as described in [68].

Routing of multicast packets on *Blazenet* is achieved by a tree-like forwarding path, where the source is the root and the destinations are the leaves. A multicast packet is forwarded as a single packet up to the point where it is split into two or more packets forwarded on different links. The split packets can also be multicast packets, in which case each one is split again at some subsequent node.

A multicast packet address is, in fact, a mapping of this tree graph to a linear notation. The linear notation consists of a list of *hop-selects* obtained by searching the tree in the following way: visit the leftmost unvisited son of the current node, if any, whose subtree contains at least one destination. Each *hop-select* consists now of two subfields: the *level-indicator* and the *output-number*. The *level-indicator* indicates the level of the current node in the whole tree, while the *output-number* is the number of the loop the packet has to be forwarded on (in the current node). The *level-indicator* is actually the hop distance of the current node from the source. Figure 3.35 shows the *hop-select* structure incorporating the above changes.

Upon an arrival of a packet at a switching node, the requested loops are checked for availability and the packet is split and forwarded to all these requested output loops that are available, if any. The packet is also returned carrying the addressing information of all the blocked outputs, if at least one output loop is unavailable.

While a multicast packet is split within a switching node, the newly generated packets carry the addressing representation of the relevant subtree only. The address field is, therefore, divided among the newly generated packets, whereas the *syncs*, the

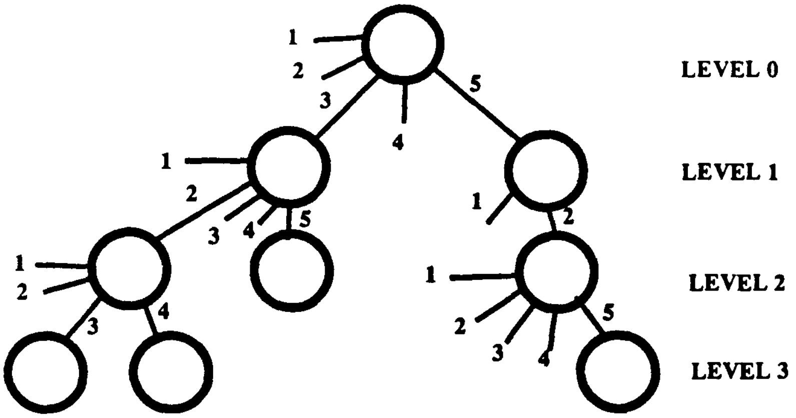


Figure 3.36: Tree graph of the multicast example

$\boxed{0}$ 3	$\boxed{1}$ 2	$\boxed{2}$ 3	$\boxed{3}$ 0	$\boxed{2}$ 4	$\boxed{3}$ 0	$\boxed{1}$ 5	$\boxed{2}$ 0	$\boxed{0}$ 5	$\boxed{1}$ 2	$\boxed{2}$ 5	$\boxed{3}$ 0

$\boxed{X}$ Y X=LEVEL INDICATOR, Y=LOOP-NUMBER

Figure 3.37: Initial address field for the multicast example

token, the *loop-counter*, the *priority*, and the *data* portion of the packet are replicated within each one of the new packets. The replication is performed by connecting the input loop to more than one output loops. The division of the address field is performed by replicating the whole address field in each one of the new packets and erasing the irrelevant portion of the address field in any one of the new packets.

The following example clarifies the multicast addressing structure. Assume a single packet is to be multicasted to four destinations. The corresponding tree graph is shown in Figure 3.36. The initial address field is presented in Figure 3.37. The first number of each *hop-select* represents the level indicator and the second one the output loop number. The first path is composed of the following sequence of *hop-selects*: 3, 2, 3, 0; the second of 3, 2, 4, 0; the third of 3, 5, 0, and the fourth of 5, 2, 5, 0. The *hop-select* of the last forwarding node on the packet path (the destination node)

0 00101	1 01001	2 00110	3 00000	3 00000	2 00000	1 01000	2 00001	3 00000

X YYYYY : X=LEVEL-INDICATOR,
Y=1 : FORWARD ON CORRESPONDING LOOP

Figure 3.38: Address field for the multicast example using bit representation.

is by definition 0. Therefore, all the paths end with *hop-select* equal to 0.

When the packet in the example arrives at the first node, it is split into two packets; the first to be multicast to destinations 1, 2, 3, and the second to be unicast to destination 4. The second packet is forwarded to its destination along the route 2, 5, 0, whereas the first packet, after arriving at the second node on its path, is split once more. One of the new packets goes on output line number 2, the other is forwarded directly to its destination on output line number 5.

The address adjustment for multicast packets is more complicated than the adjustment for unicast packets, because of the necessity of splitting the address field. In the multicast case, the *Control* first looks for the *level-indicator* of the first *hop-select.* Then, the address field is divided into pieces. The division is performed by breaking the address field on the boundary of *hop-selects* with values of *level-indicator* equal to the value of the *level-indicator* of the first *hop-select.* Each new packet carries one of the resulting pieces and is then forwarded according to the first *hop-select.* During the forwarding process the first *hop-select* is erased. The address field of the new packet is, therefore, composed of only the relevant sub-tree.

Another possible addressing scheme for multicast in *Blazenet* uses a single *hop-select* field to indicate multiple output connections. In this scheme M bits are used for each route, each bit for one of the M possible output loops. A bit is set if the packet has to be forwarded on the corresponding output loop. In this scheme, as in the previous one, the nested structure of the various paths realizes the multicast delivery. Figure 3.38 shows such a representation for the above multicast example. This scheme is more efficient in the case of multicast to many destinations. Consequently, the preferred solution depends on the particular network parameters.

It should be noted that in both addressing schemes the packets created by splitting

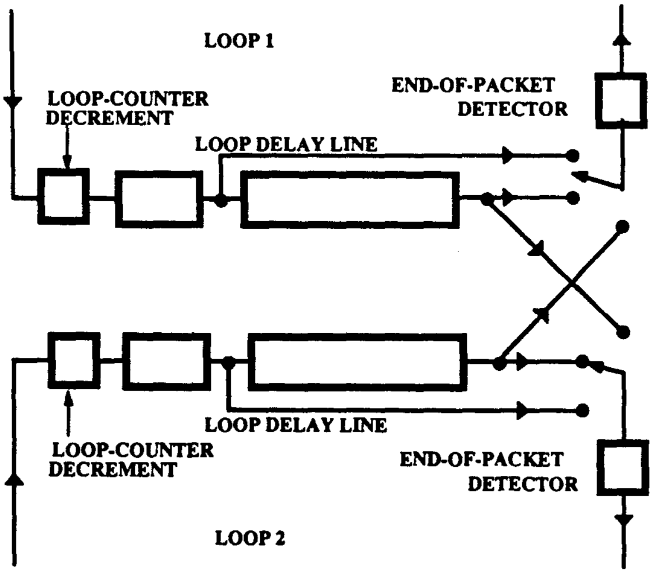


Figure 3.39: The *Blazenet* node design, as modified to include flooding.

the original packet have unused gaps in the address field. Moreover, even in the unicast case, erasing the used *hop-selects* creates gaps. Although it is possible to eliminate these gaps, the cost of the gaps is insignificant, since typically the header is only a small portion of the whole packet.

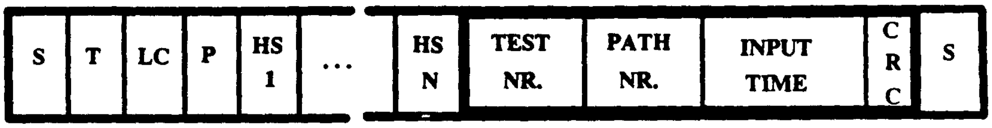
A special case of multicast is broadcast, where a packet is to be transmitted to all the possible network destinations. Broadcast can be implemented by different algorithms ([69, 70]), which include: transmission of separately addressed packets, multidestination addressing, hot-potato forwarding, spanning-tree forwarding, source-based forwarding, and reverse-path forwarding. A broadcasting algorithm can be used in *Blazenet* if the processing associated with the algorithm is minimal and there is virtually no memory requirement for the algorithm. Consequently, the spanning-tree forwarding, the source-based forwarding, and the reverse-path forwarding algorithms are too complex to be implemented in *Blazenet.*

Transmission of separately addressed packets and multidestination addressing are easily implemented in *Blazenet* by the unicast and multicast addressing schemes, respectively. I refer to the hot-potato forwarding algorithm as flooding. Flooding can be implemented by different schemes. In one such a scheme, a specific *hop-select* value is reserved for broadcast packets and instructs the forwarding nodes to forward the packet on all its loops (possibly with the exception of the loop directed to the node the packet comes from). The first *hop-select* does not need to be erased in the forwarding process. To guarantee damping of the flooding process, the processing of the *loop-counter* in the switching nodes must be slightly modified: the value of the *loop-counter* must be decreased each time a packet is received in a switching node, whenever the packet is blocked or successfully forwarded. This hardware modification requires placing the *loop-counter* decrement unit before the switch of the *Delay Line*, as shown in Figure 3.39. In order to make packet reception by all the network nodes possible, the value of the *loop-counter* should be specified to be the maximum path length from the packet source to any network destination with some reasonable addition for packet loop backing. With this flooding mechanism, a broadcasted packet can be received more than once. Consequently, higher layers protocols must discard the duplicated packets. Flooding can be used to cope with abnormal network behavior and to increase network reliability.

3.5.4 Monitoring the network

Topological and load condition changes in the network require constant adjustments of routing tables and forwarding policies within hosts. The source routing used in *Blazenet* makes it easy to acquire information about network changes in a distributed manner. This information is gathered by hosts named Monitors that perform the network data collection operation. Each Monitor initiates tests for link availability and link load condition. These tests are performed by the Monitor, which sends source-routed packets back to itself over specific paths in the network. Packets sent through an unoperative link are not delivered back to the Monitor. By analyzing information from many network paths, the Monitor can detect incremental changes in the network load and network topology (e.g., availability of a specific link).

A test packet structure is shown in Figure 3.40. In order to avoid confusion, the *test-nr* field differentiates between various tests (that can be performed concurrently)



S=SYNC, T=TOKEN, LC=LOOP-COUNTER,
P=PRIORITY, HS=HOP-SELECTs

Figure 3.40: Test packet structure

and the *path-nr* field uniquely identifies the specific path tested. The *input-time* records the time a packet was entered into the network and serves for calculation of the packet delay through the specific path.

In the following discussion I assume that the network changes are incremental, that is that the probability of a failure of more than one link or node between any two tests is negligible. Therefore, it is assumed that at any time the Monitor's ignorance of the network's status is at most a state of one variable.

The tests are performed in the following manner: Each Monitor sents packets over the network to cover all the network links. If a packet does not return, the Monitor initiates more tests in order to determine which link on the missing packet path is down. The intersection of all the missing packets' paths gives the unoperative link (there is only one unoperative link, if any). However, in some cases a failure cannot be uniquely identified. For example, consider a part of a path consisting of two consecutive links connected by a node that have no other input/output links, or that the other input/output links are unaccessible to a Monitor. In this case the Monitor cannot determine which one of the two links actually failed.

If a Monitor decides that a link is unoperative, it passes this information to all the other hosts connected to the Monitor's switching node and causes changes in their routing tables. Later, from time to time, the Monitor might reissue some tests to check if the status of an unoperative link has changed.

The same approach can be used in order to locate the areas of congestion in the network. However, more sophisticated algorithms must be used in order to analyze the packets' delays and to evaluate the state of the congestion of a specific link or group of links. A useful assumption is that a link's load does not change rapidly. This assumption can be justified by the fact that the network has high-throughput characteristic. In a low-throughput network, the influence of a single event might have a dramatic effect on a link's load; in a high-throughput networks the high-capacity makes the effect far less significant. Also, the fact that the networks are of a mesh topology, contributes to the smoothing effect.

Conclusions from these tests help in determining forwarding policies and alternate routing schemes. It should be pointed out that these tests should be implemented in such a way that they do not significantly contribute to the network load. This is accomplished by designing the tests so as to decrease their number, by performing them with proper frequency, and by using a test packet of small size.

The Monitor, besides continuously determining the state of the network, can be assigned other tasks. One such a task may be to serve as a collector of discarded packets. A packet discarded by a switching node is handed to the Monitor connected to the switching node. An additional field in *Blazenet*'s packet format might instruct the Monitor on the necessity of announcing the discarding operation to the source of the packet. The Monitor examines the discarded packet and may initiate a special negative acknowledge (NAK) packet, sent back to the packet's source. The NAK packet is composed of the discarded packet header and the reason why the packet was discarded.

The Monitor, as presented in this section, serves as a network tool that copes with malfunctions and with abnormal behavior of a network. By performing such a function, the Monitor turns *Blazenet* into an immune communication channel, unloading some processing burden from the network interfaces, and increasing the reliability of the whole communication process.

3.5.5 Support for stream traffic

As shown in Chapter 4, the *general cut-through* switching method is the preferred switching technique in the *transactional* environment. *Blazenet* is not an exact implementation of any conventional *cut-through* technique. The reason is that even though *Blazenet* does not store an unblocked packet, it stores a blocked packet for a longer time than necessary due to the fact that storage of the loops can be "accessed" only at discrete instances. Simulation results presented in Section 3.4 show, however, that the resulting delay degradation is not significant (especially at low-load). If the last loop on each link is of the length of a single packet, the implementation tends more to resemble the *full virtual cut-through* technique. Although, the storage is still "accessed" at discrete instances in this case, the access is now more frequent. Consequently, the gap between the end of the current transmitted packet and the "stored" packet is of the maximum length of one packet size.

As pointed out in [3] and in Chapter 4, for wide-area networks operating under low-load conditions, there is only marginal gain in having a single very high-speed channel of capacity larger then some threshold value (which is in the range of Gbps). It is more advantageous to have multiple parallel high-speed channels (loops in *Blazenet*'s case) operating at the capacity of the threshold. This arrangement also has the advantage of providing increased network reliability.

In order to improve *Blazenet*'s performance for transmissions that are more of the *stream* type, I present a different switching scheme that can be incorporated into *Blazenet*'s design and that can integrate *stream* traffic without the excessive overhead of conventional circuit-switching, and yet is capable of dedicating a path through the network. I call this scheme *Loop-Switching.* The basic idea (which is similar to that of the *permanent virtual circuit* in [35]) is to reserve a loop for the duration of the message transmission. This is done in the following manner: A host generating traffic injects its packets into a loop. When the stream of these packets arrives at the next switching node, it tries to reserve the next loop on the path. If the loop is unavailable, some of the first packets may be returned. Immediately after the loop becomes available, the next packet in the packets' stream will reserve the loop, allowing it and the subsequent packets in the stream to be forwarded on the now-reserved loop. A reserved loop is dedicated for the transmission until there is no arriving packet to be forwarded on the loop for a period of at least one round trip time of the loop. At this point in time, the *Control* decides that the message is complete and the loop is freed for another connection. Thus, the packets of a message diffuse through the network reserving the path for the message. *Loop-Switching* assumes the existence of many loops operating in parallel and forming a single link. The use of the scheme is justified only for *stream* transmissions whose channel occupation time is at least on the order of the propagation delay of a single hop. Also, a special instruction must be included in the packets header to indicate that the *Loop-Switching* service is required. When the *Loop-Switching* technique is optionally provided the integration of *stream* and *bursty* traffic is easy to perform.

3.6 Issues resolved by the transport and higher layers

Blazenet does not provide error detection on the *header* portion of a packet. Hence, packets can arrive at a wrong destination. However, if the information on the packet destination is embedded into the data portion of the packet, and the data portion is protected by error-detection (or error-correction) code, the transport layer discovers packet misdelivery and discards the erroneous packet. Packets discarded by the destination because of a transmission error, as well as packets discarded by the network and packets lost while in transit, are retransmitted by the source of the packet after a NAK is received or after some time-out has expired.

Packets are also not guaranteed to arrive in the order in which they are entered into the network, since an earlier-sent packet may be blocked and may arrive after a later-sent packet that was not blocked. In this case too, the transport layer has to take care of the packet reordering, creating a transparent service for the end-to-end communication.

Blazenet provides some limited flow control on the physical layer. When the load increases, the loops become more populated and less traffic can be inserted into them. Thus, the flow control is performed by the back-pressure that propagates from the point of congestion to the entrances of the network. This flow control is basically at the *hop level* and in a limited sense also at the *entry-to-exit level. Blazenet* does not support higher level flow control.

The *loop-counter* mechanism in *Blazenet*, which implements packet time stamping, has two major roles: to support real-time traffic delivery and to avoid erroneous infinite traffic circulation. In more sophisticated applications, the priority of the packet can be varied according to the value of the time stamp. The priority of packets with smaller residue lifetime will be increased. The transport layer, the session layer, and possibly even the application layer may play a role in the determination of the value of the *loop counter.*

3.7 Summary and conclusions

In this chapter, I have presented *Blazenet*, a wide-area high-speed packet-switched network suitable for fiber optics implementation. I have discussed *Blazenet*'s architecture, operation, switching node design, performance, and extended features.

A closer look at *Blazenet* reveals some of the network's salient properties: high-speed switching, the lack of conventional memory, good behavior under traffic load, flow control performed by the back-pressure mechanism, and the possibility of implementation of priority traffic, of multicast delivery, and of photonic design. Specifically, *Blazenet* provides switching of multi-gigabit per second data rates, low delay, and good behavior under load.

The use of source routing allows each switching node to make switching decisions on the fly, minimizing the switching delay. The use of a loop-back channel, which effectively stores packets that are blocked at the switch, minimizes packet loss under load without requiring additional memory within the switch. Simulation results indicate that *Blazenet*'s performance is comparable for low-load operation to that of the ideal case of a nonblocking network, and that the *Blazenet*'s performance is siginficantly better than that of the *Lossy* network.

Finally, the simplicity of the switching node, which results from the use of source routing, and the absence of switching buffer memory makes it feasible to realize the switching node through the use of photonics. Photonics makes the switching node more immune than electronics to electromagnetic monitoring or interference. It also provides greater performance and reliability, especially as photonic technology matures ([56]).

Chapter 4

Switching methods for H-P communication

4.1 Introduction

The large capacity of optical fibers suggests that circuit-switching (CS) may become a more attractive switching method than packet-switching in future communication networks. However, it is shown in this chapter that under some reasonable assumptions the delays associated with circuit-switching make the technique inferior to cut-through packet-switching in a high-performance, distributed environment, an environment that is characterized by the *transactional* model of communication. *Blazenet*, introduced in Chapter 3, provides the proof for the claim that fast packet-switching is, in fact, feasible.

4.2 Message-switching vs. packet-switching

In this section, I present the comparison between message- and packet-switching. Both switching schemes operate in the *general-cut-through* mode. It is assumed that the message length has some general distribution. (Instead of *message length,* I use in the following analytical development *message transmission time* in seconds, which is equal to message length in bits divided by channel capacity C [bit/sec]. Thus all links are assumed to be of equal capacity. Consequently, when I refer here to message and packet size, it should be remembered that these are actually transmission times expressed in seconds.) The random variable (r.v.) $\tilde{x}$ represents message transmission time in seconds, and $\lambda(\tilde{x})$ is the average arrival rate of messages per second with transmission time equal to $\tilde{x}$ [sec]. (Tilde sign above a variable indicates a random variable. Expected value of a random variable is indicated by a bar over the variable. Queuing theory abbreviations are the same as in [88].) Also, it is assumed that the length of a path with in the network is l hops.

The message arrival process is assumed to be Poisson. This is not strictly true when cut-through switching is used. However, as shown in Appendix D, for small (and large) utilization factor ρ the distribution of interdeparture times from an M/M/1 queue closely resembles the exponential distribution. Also, for intermediate values of ρ the difference between the exponential and the actual interdeparture time distributions is small. This fact, together with Kleinrock's independence assumption ([20, 89]) justifies the assumption of Poisson arrival process.

In the message-switching scheme (MS) the message delay is composed of the queuing time at each one of the switching nodes on the message path plus the message transmission time. Thus, the MS delay, $\widetilde{D_m}$, is

$$\widetilde{D_m} = l \cdot \widetilde{W}_{M/G/1} + \tilde{x} \text{ [sec]}, \tag{4.1}$$

where $\widetilde{W}_{M/G/1}$ is the waiting time in seconds at each node, which is obtained from the *Pollaczek-Khinchin* formula ([88]). Thus

$$\overline{\widetilde{D_m}} = l \cdot \frac{\lambda_m \cdot \overline{x^2}}{2(1-\rho)} + \bar{x} \text{ [sec]}, \tag{4.2}$$

where ρ is link utilization factor. The value of λ_m, the total message arrival rate, is calculated using

$$\lambda_m = \int_0^\infty \lambda(x)dx \text{ [messages/sec]} . \tag{4.3}$$

In the packet-switching case (PS) it is assumed that all packets are of equal size and their transmission time is represented by p (this requires, in general, padding of the last packet of a packetized message). The packet arrival rate, λ_p, is larger

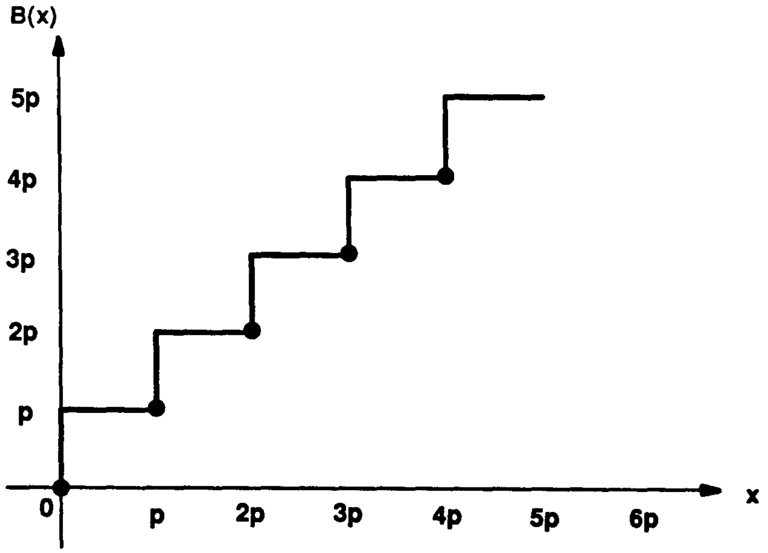


Figure 4.1: The $B(x)$ function

than the message arrival rate since, in general, messages are divided into a number of packets. The calculation of λ_p is given by

$$\lambda_p = \int_0^\infty \lambda(x) \cdot B(x) dx = \sum_{k=1}^{\infty} \int_{p(k-1)}^{pk} k \cdot \lambda(x) dx \text{ [message/sec]}, \tag{4.4}$$

where $B(x)$ is a weight function, as presented in Figure 4.1.

A packet in the PS case encounters the queuing delay at each one of the switching nodes on its path and the transmission delay, p [sec]. Packets of a message are, however, interleaved by packets belonging to other messages, thus creating inter-packet gaps. These gaps prolong the message delay, since a message is declared received only when its last packet is received. The r.v. $\tilde{v}$ represents the length of the inter-packet gap in units of a packet's transmission time. Thus, the PS delay, $\widetilde{D_p}$, is

$$\widetilde{D_p} = l \cdot \widetilde{W}_{M/G/1} + \lceil \frac{\tilde{x}}{p} - 1 \rceil \cdot \tilde{v} \cdot p + \overline{x} \text{ [sec]} . \tag{4.5}$$

In Appendix A, it is shown that if the number of packets, each message is divided into, is much larger than 1 then $\overline{v} \approx l \cdot \rho$. Thus,

$$\overline{D_p} = l \cdot \frac{\lambda_p \cdot p^2}{2(1-\rho)} + \overline{x} \cdot (1 + l \cdot \rho) \text{ [sec]}, \tag{4.6}$$

where it is assumed that:

- the messages are divided into many packets, i.e., $\frac{\bar{x}}{p} \gg 1$,
- $\tilde{v}$ and $\tilde{x}$ are independent,
- ρ remains the same for both MS and PS.

The last assumption is not strictly correct because of the possible padding of the last packet of a message. Thus in reality $\rho_{MS} < \rho_{PS}$. The assumption (favoring the PS scheme) is, however, a good approximation for small p.

It follows directly from the last assumption that

$$\lambda_p = \lambda_m \cdot \frac{\overline{x}}{p} \text{ [messages/sec]} \ . \tag{4.7}$$

Thus

$$\overline{D_m} = \frac{\lambda_m \cdot \overline{x^2}}{2(1-\rho)} \cdot l + \overline{x} \text{ [sec]} \tag{4.8}$$

and

$$\overline{D_p} = \frac{\lambda_m \cdot \overline{x} \cdot p}{2(1-\rho)} \cdot l + \overline{x} \cdot (1 + l \cdot \rho) \text{ [sec]} \ . \tag{4.9}$$

It is instructive to note that the corresponding formulas for MS and PS working in the conventional store-and-forward (not cut-through) mode are

$$\overline{D_m} = \frac{\lambda_m \cdot \overline{x^2}}{2(1-\rho)} \cdot l + \overline{x} \cdot (1 + l) \text{ [sec]} \tag{4.10}$$

and

$$\overline{D_p} = \frac{\lambda_m \cdot \overline{x} \cdot p}{2(1-\rho)} \cdot l + p \cdot l + \overline{x} \cdot (1 + l \cdot \rho) \text{ [sec]} \ . \tag{4.11}$$

The new term $\overline{x} \cdot l$ in the equation for $\overline{D_m}$ is the time required to completely receive the message in each switching node along the message path. The corresponding term in the equation for $\overline{D_p}$ is $p \cdot l$.

As $Var(x) \geq 0$ and $\overline{x} \geq p$, it follows that the first term in $\overline{D_m}$ is always greater than the corresponding term in $\overline{D_p}$. Consequently, by sufficiently decreasing the packet size, p, the delay of packet-switching can always be made smaller than the delay of message-switching in the store-and-forward mode. This is not true, however, in the cut-through mode. Suppose, for example, that the messages are of fixed size. Thus, $\overline{x^2} = \overline{x}^2$. Then for $\rho < 1/2$, and even for p=0, one still obtains $\overline{D_p} \geq \overline{D_m}$.

Assuming small packet size, $p \rightarrow 0$ (note that this assumption is supported by the large link capacity in fiber optic networks), the condition for having a message-switching delay that is lower than the packet-switching delay is

$$\frac{\overline{x^2}}{\overline{x}^2} < 2(1-\rho) \tag{4.12}$$

or,

$$C_b^2 < 1 - 2\rho\,, \tag{4.13}$$

where C_b, the coefficient of variation, is defined as

$$C_b = \frac{\sigma_x}{\overline{x}}\,. \tag{4.14}$$

Obviously, a large mean and a small variance of the message size tend to favor message-switching. Also, small ρ has the same effect. For example, the distribution of the message length measured in our local environment at Stanford (the V-system) is shown in Figure 4.2. After substitution of the first and the second moments of the message length calculated from the above graph, one concludes that for $\rho < 0.69$ the message-switching outperforms packet-switching. Since it is reasonable to assume that the system will operate at loads lower than 0.69, the system should employ message-switching as its switching method.

The two effects that interplay here are: the queuing delay is shortened by shortening the transmission time of the information and the inter-packet gaps are increased when messages are divided into packets. It should be stressed that the equation for inter-packet gap assumes independent packets. In reality this independence assumption is not valid, especially near the network entries. The actual range over which message-switching is advantageous is larger than that predicted by the above analysis. For example, Appendix B shows that for the case of fixed message length, message-switching always has lower delay than packet-switching (due to the correlation in arrival times between the packets of two colliding messages), when observed at the network entries.

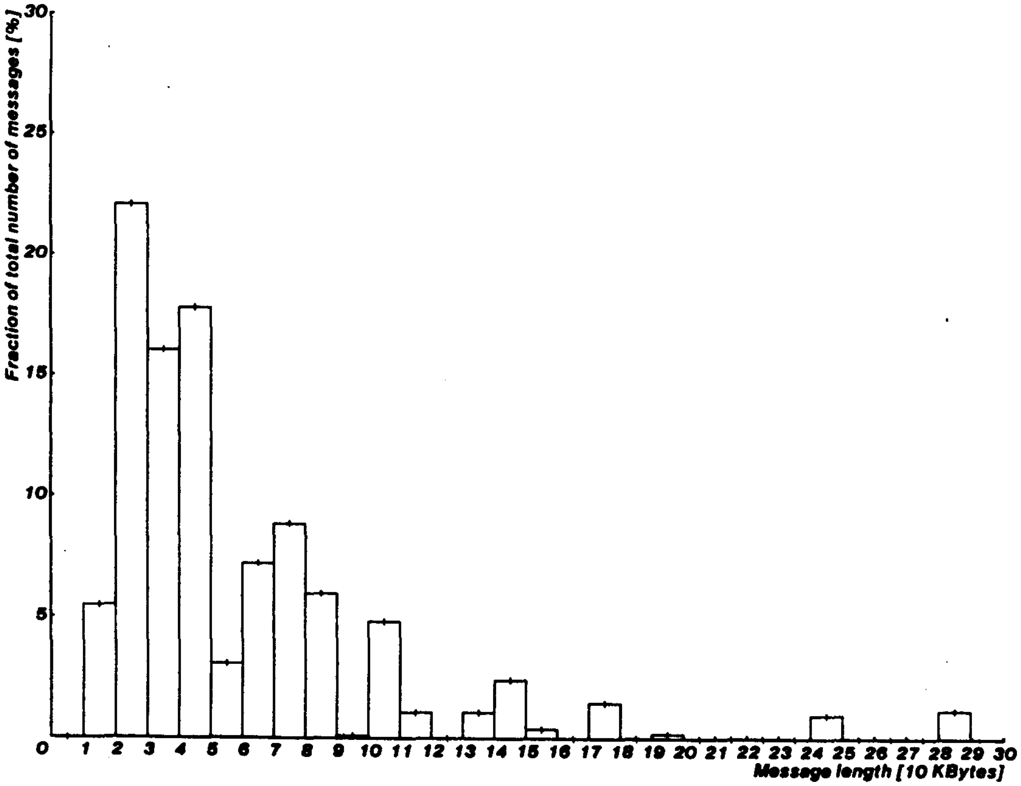


Figure 4.2: Sample message length distribution in V-system

However, packet-switching possesses advantages as well. Error detection is usually provided on a per packet basis. Long messages, for which the probability of an error in the message is increased, cause the whole message to be retransmitted, and result in large overhead. This problem can easily be avoided by providing error protection on blocks within the message. Since error checking will probably be done only at the destination (and the *cut-through* technique somewhat limits the error checking in the intermediate nodes) this solution provides the same level of overhead as packet-switching.

Yet another advantage of packet-switching is that different packets of the same message may be routed on different routes through the network. Messages, in the message-switching technique, are usually routed along a single path. However, in future high-speed networks the overhead of the computation of dynamic routing may be too high, and hence, some simplified routing mechanism like a constant routing scheme will probably be employed. Moreover, since the prediction is that future high-speed networks will operate under low utilization, the routing will be done most of the time using a single (best choice) decision. Consequently, I do not see this disadvantage of message-switching as a very crucial one.

In this work I chose to compare the message-switching (and not packet-switching) technique to the circuit-switching scheme, mainly for two reasons. The first is the fact that low utilization ($\rho \rightarrow 0$) tends to favor message-switching. The second is the fact that with increased link capacity the message transmission time shrinks and thus further fragmentation considerably increases the processing overhead associated with many very small packets. In the proposed message-switching scheme, the messages can be divided into logical blocks that usually travel back-to-back as a unit through the network. Blocks might be separated under some special conditions like interruption by an extraordinary priority traffic. Note, also that a *request*, being a short message, can be designed to interrupt back-to-back packets of a long *response*, partially alleviating the increased delay of message-switching for traffic with large variance of messages lengths.

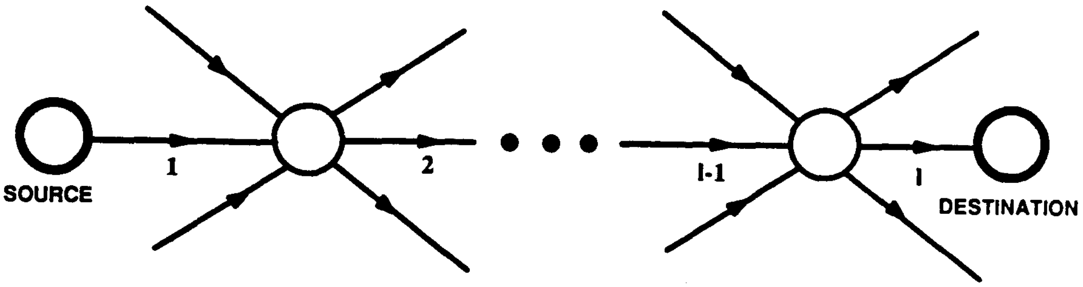


Figure 4.3: Model of a single path in the network

4.3 The model

The model I chose to compare the performance of the circuit-switching technique with that of the message-switching technique is presented here. The model includes a single path of length l hops, as presented in Figure 4.3. I assumed that the network in question is large: many links enter and exit each one of the switching nodes. The links in the network are of equal capacity C [bits/sec] each. Moreover, I assumed that the network is totally symmetric and that traffic in the network is totally balanced. Consequently, all the network links are equally utilized. In the circuit-switching case the link capacity is divided into N sub-channels, each sub-channel of capacity C/N [bits/sec].

The communication model assumed is the *transactional* model. A request, which is assumed to be a small amount of information (usually of the order of a small fraction of the average response size), is sent from a client to a server. The server, after processing the request (for α [sec]) answers by sending a response of d [bits] back to the client. This response is assumed to be a large amount of data. Measurements done on Stanford V-system (see also [40]) tend to support these relative sizes of requests and responses. Different cases of the relative sizes of request and response are considered in Section 4.7.

The discussion focuses on wide-area networks of the order of hundreds of kilometers, with a coast-to-coast span being the maximum size. The one-way propagation delay over the path is called T_{prop} [sec]. (On a 1000 km path, $T_{prop} \approx 5$ msec.) To simplify the calculations, I assumed that all links are of equal length, 100 km, and that the speed of light in fiber is 200,000 km/sec.

I would like to point out that since a preliminary analysis has shown that the *general-cut-through* message-switching technique outperforms the circuit-switching technique in this environment (i.e., in the case of high-speed network, high-performance communication, and response-request protocol of communication), some of the assumptions that are made in this work are *conservative* in the sense that they decrease the delay of the circuit-switching scheme more than the delay of the message-switching scheme.

For the message-switching implementation, I assume the *general-cut-through* mode of operation on a store-and-forward network configuration with infinite storage.

The circuit-switched network is assumed to work in the following manner: while the request makes its way through the path from a client to a server, it reserves a reverse sub-channel for the response. This means that at each switching node the request competes for one of the N sub-channels on the link in the reverse direction. The reservation is made by leaving a dummy copy of the request in each one of the switching nodes on the request path, while the real request continues its way to the next switching node on its path. Each one of the switching nodes waits for completion of the reservation of a sub-channel and for the arrival of the dummy request from the previous node. Only then it sends its dummy request to the next node on the path. (The first node does not need to wait for a dummy request.) When a dummy request arrives at the server, the reverse channel is set, and as the server finishes the processing, the transmission of a response is initiated. Note that using the request to set up the circuit has an obvious advantage in terms of delay over the conventional round-trip circuit set-up procedure that occurs before any communication takes place.

For both schemes, I assume that because of the small size of the requests, they encounter no queuing delays. (The requests' queuing delays can be virtually eliminated if a separate channel for the requests is designed.) However, in the circuit-switching version a dummy request is delayed at each switching node until a reverse free sub-channel is found. In the message-switching version a request is assumed to speed through the network without any delay whatsoever. Because of the transmission of a dummy request from each one of the switching nodes, the transmission time of the request at each switching node should be added to the total delay of the transaction in both switching schemes. However, because of the small transmission time of the request compared to the propagation delay of each hop, this time is neglected. (Note that this is a conservative assumption since in circuit-switching the channels are smaller, thus the transmission time of a request is increased.)

In the following analysis, I examine the influence of the following four parameters on the comparison of circuit-switching with message-switching delays:

- Channel utilization, ρ ,
- Data transmission time, d/C [sec],
- Path length, l [hops],
- Request processing time, α [sec].

4.4 Message-switching delay

The queuing model for this case is M/D/1. Since *general-cut-through* is assumed, a message encounters the following waiting time at each node on its path:

$$W_{M/D/1} = \frac{\rho \cdot d/C}{2(1-\rho)} \text{ [sec]} . \qquad (4.15)$$

The transmission time is, however, encountered only once during the message transmission. Thus, the delay of a *transaction* is:

$$Delay\{MS\} = 2 \cdot T_{prop} + \frac{d}{C} + \alpha + \frac{\rho \cdot d/C}{2(1-\rho)} \text{ [sec]} , \qquad (4.16)$$

where d/C is the transmission time in seconds of the data and T_{prop} is a one way propagation delay in seconds of the path. The one way propagation delay is equal to the path length divided by the speed of signal propagation in the media. Thus, assuming hops of 100 km,

$$T_{prop} = 0.5 \cdot l \ \text{ [msec]} \ . \qquad (4.17)$$

From equations (4.16) and (4.17), the delay as a function of the four parameters of the message-switching technique can be easily evaluated. In the case of packet-switching, one needs to include the reassembly time of a packetized message. The reassembly time results from the fact that packets of the same message will be interleaved with packets of other messages, while the message is in transit. Thus, the actual time of the reception of the last packet of a given message is delayed by these inter-packets gaps. The formula for this extra delay, derived in Appendix B, shows that the inter-packets gaps increase the message transmission time by a factor $l \cdot \rho$. Consequently, the packet-switching delay totals:

$$Delay\{PS\} = 2 \cdot T_{prop} + \frac{d}{C} \cdot (1 + l \cdot \rho) + \alpha + \frac{\rho \cdot d/C}{2(1-\rho)} \cdot l \text{ [sec]} . \tag{4.18}$$

4.5 Circuit-switching delay

The queuing model for the circuit-switching case is M/G/N. However, because of the complexity involved in solving this model, I used an approximate solution; i.e., the Nozaki-Ross approximation for M/G/N ([90, 91]).

First, let us concentrate on the evaluation of the service time. During the process of a circuit set-up the sub-channel of the first link is held for the whole time of the transaction, while the sub-channel of the last link is held only for a fraction of the total transaction time: the round-trip propagation time over a single link, the request processing time, and the transmission time of the response. Thus, the service times of the consecutive nodes on the path are decreasing. I assumed uniform distribution of the service time with maximum equal to the whole transaction time and minimum equal to the holding time of the last link. Furthermore, I assumed that the path of all messages is of equal length. (More general cases can be treated in the same way, with the average path length weighed by the factor of $\overline{l^2}/\overline{l}^2$. For more details see Appendix C.)

The Nozaki-Ross approximation gives the waiting time of an M/G/N queue as a function of the second moment of the service time, $E\{h^2\}$:

$$W_{M/G/N} = \frac{\lambda \cdot E\{h^2\} \cdot \rho^{N-1}}{2(N-1)! \cdot (N-\rho)^2 \cdot [\sum_{i=0}^{N-1} \frac{\rho^i}{i!} + \frac{\rho^N}{(N-1)! \cdot (N-\rho)}]} =$$

$$\frac{\lambda \cdot E\{h^2\}}{2(N-\rho)^2 \cdot [\sum_{i=0}^{N-1} \frac{(N-1)!}{\rho^i \cdot (N-1-i)!} + \frac{\rho}{(N-\rho)}]} \text{ [sec]} . \tag{4.19}$$

The approximate M/G/N delay can now be calculated. The calculation is, however, complicated by the fact that the evaluation of the first moment of the composite service time is, by itself, dependent on the total delay. Therefore, recursive approximation was used to solve for the delay.

The total circuit-switching delay is obtained by

$$Delay\{CS\} = 2 \cdot T_{prop} + \frac{d}{C/N} + MAX(\alpha, W_{M/G/N}) \text{ [sec]} . \tag{4.20}$$

4.6 Comparison between the switching techniques

In order to compare between message- and circuit-switching, the delay of a transaction was calculated for both the scheme using 4.16, and 4.20 with 4.19, respectively. The delay was evaluated for $0 < \rho < 1$ and for different values of the parameters d/C, α, and l. In particular, d/C was assigned 10^{-3} sec, 10^{-4} sec, 10^{-5} sec, and 10^{-6} sec; α values of 10 msec, 100 msec, 200 msec; and l values of 5 hops, 10 hops, and 20 hops.

The number of sub-channels, N, is a crucial parameter in the performance of the circuit-switching scheme. Increasing the value of N, on one hand, decreases the contention over sub-channels, but on the other hand, increases the transmission time of the response, since now the sub-channels are of smaller capacity. Thus, for a given set of other parameters (ρ, α, d/C, and l) there is an optimal value for N that yields minimum delay. Since the goal of the work is to prove message-switching superiority, optimization on the value of N was done in each case. This means that the number of sub-channels was changed each time the value of any other parameter changed. In some sense this is an unrealistically *conservative* assumption, since in practice the number of sub-channels is rarely altered.

The relative increase of the circuit-switching delay over the message-switching delay is calculated by the following formula:

$$\frac{Delay\{CS\} - Delay\{PS\}}{Delay\{PS\}} \text{ [\%]} . \tag{4.21}$$

However, more interesting is the net increase of the circuit-switching delay over the message-switching delay. The net communication delay is the delay encountered by a

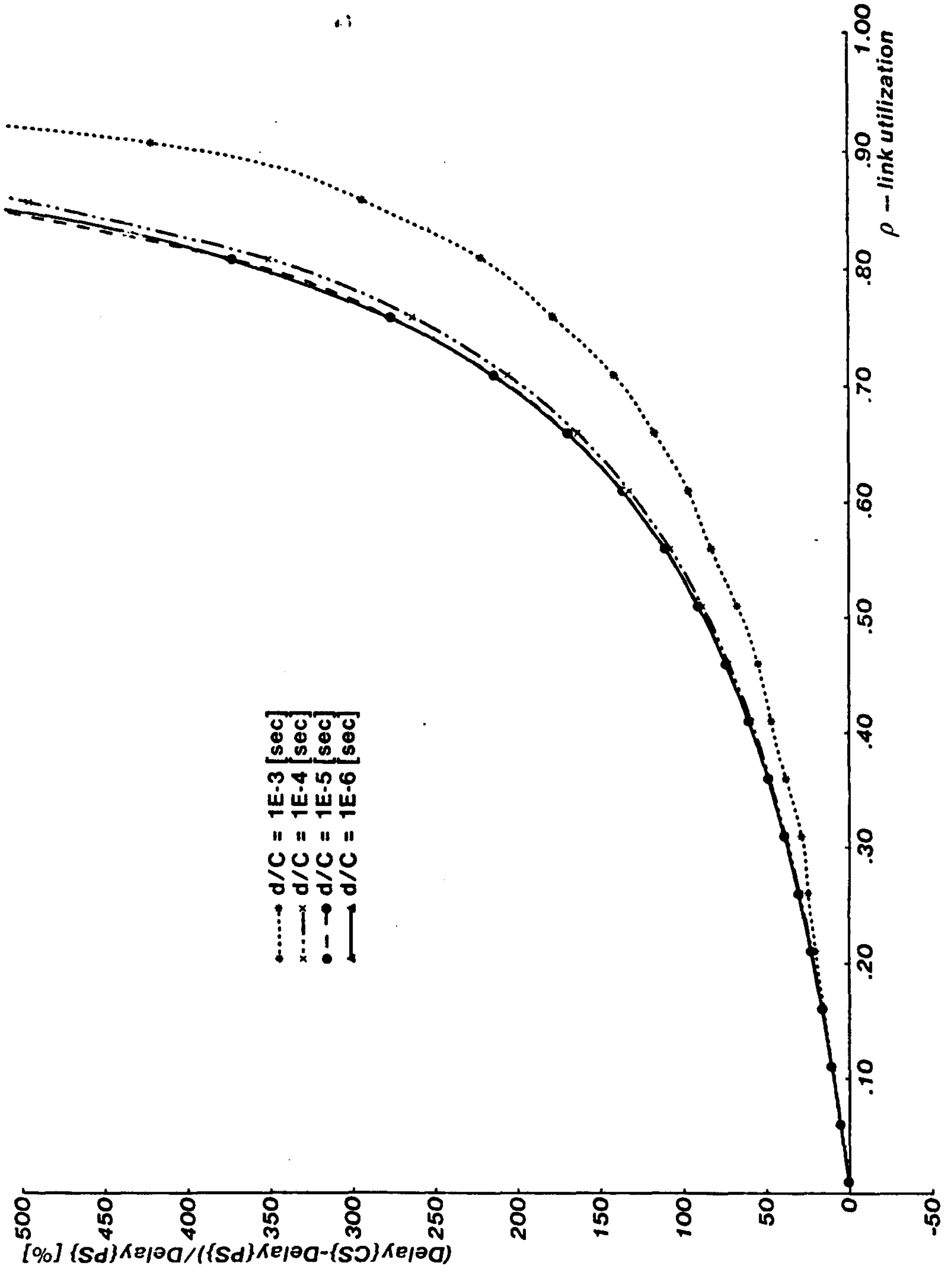


Figure 4.4: Relative increase of total delay in CS over PS; l=5 [hops], α=0.010 [sec], C=variable

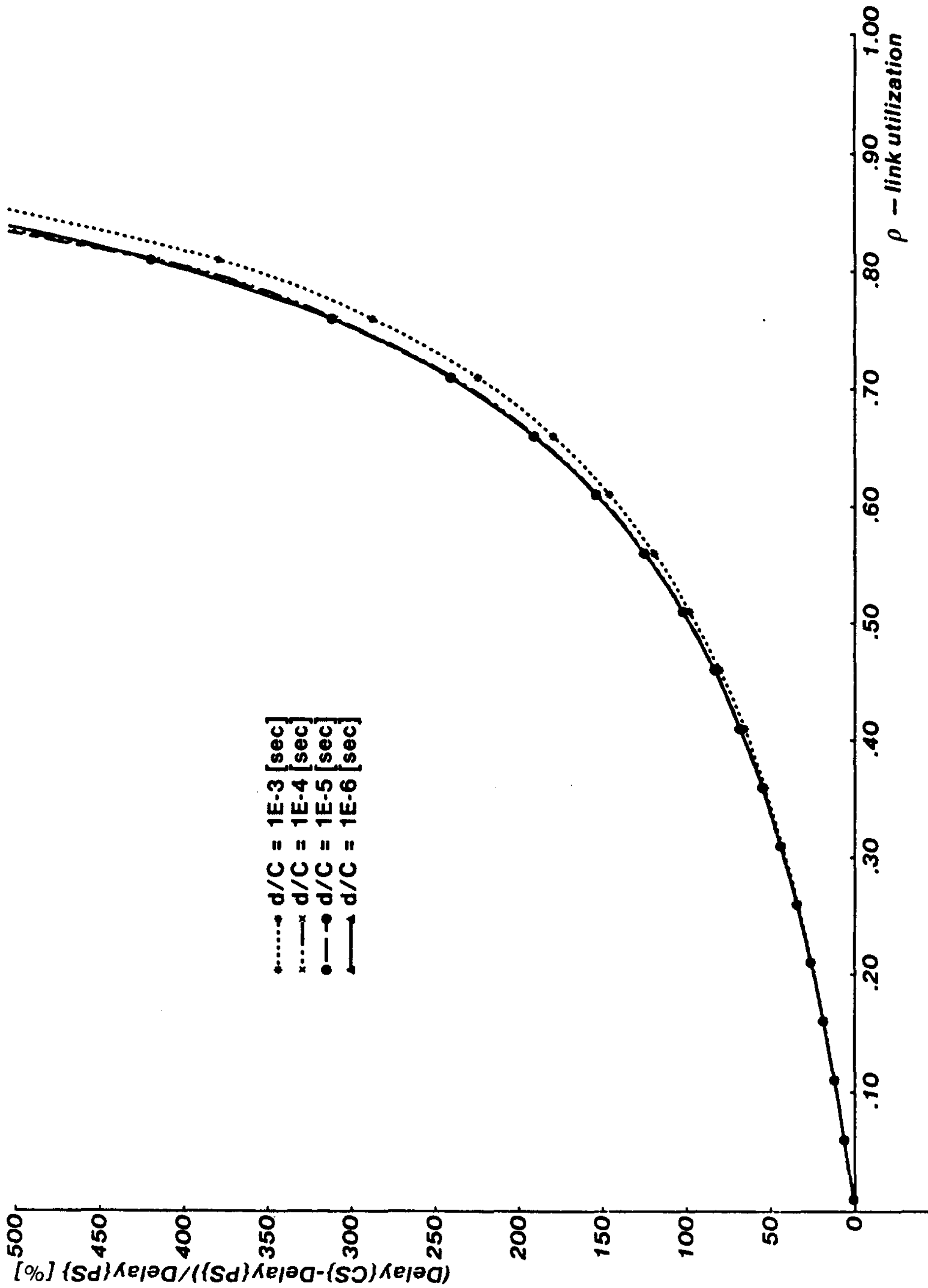


Figure 4.5: Relative increase of total delay in CS over PS; l=5 [hops], α=0.100 [sec], C=variable

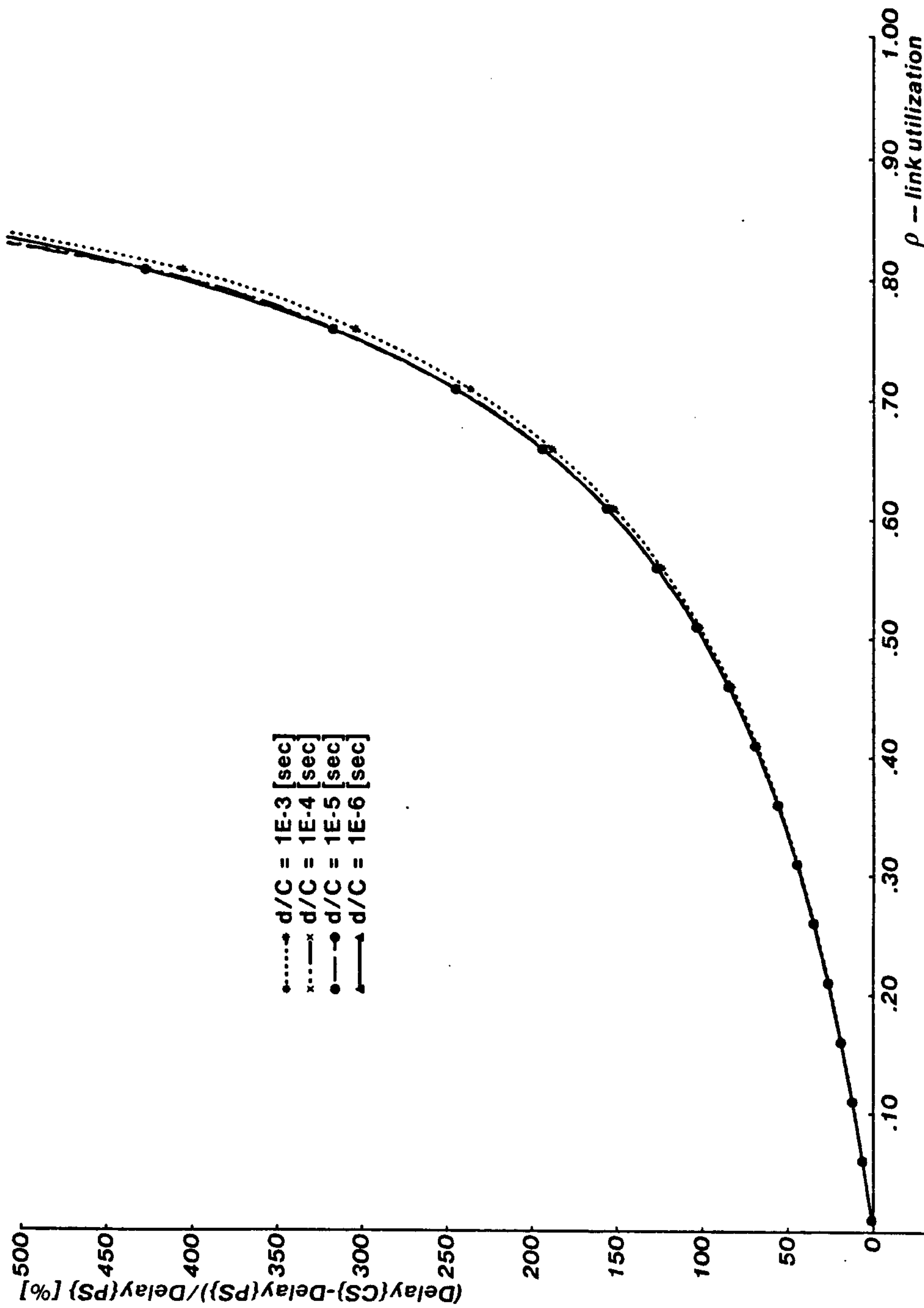


Figure 4.6: Relative increase of total delay in CS over PS; l=5 [hops], α=0.200 [sec], C=variable

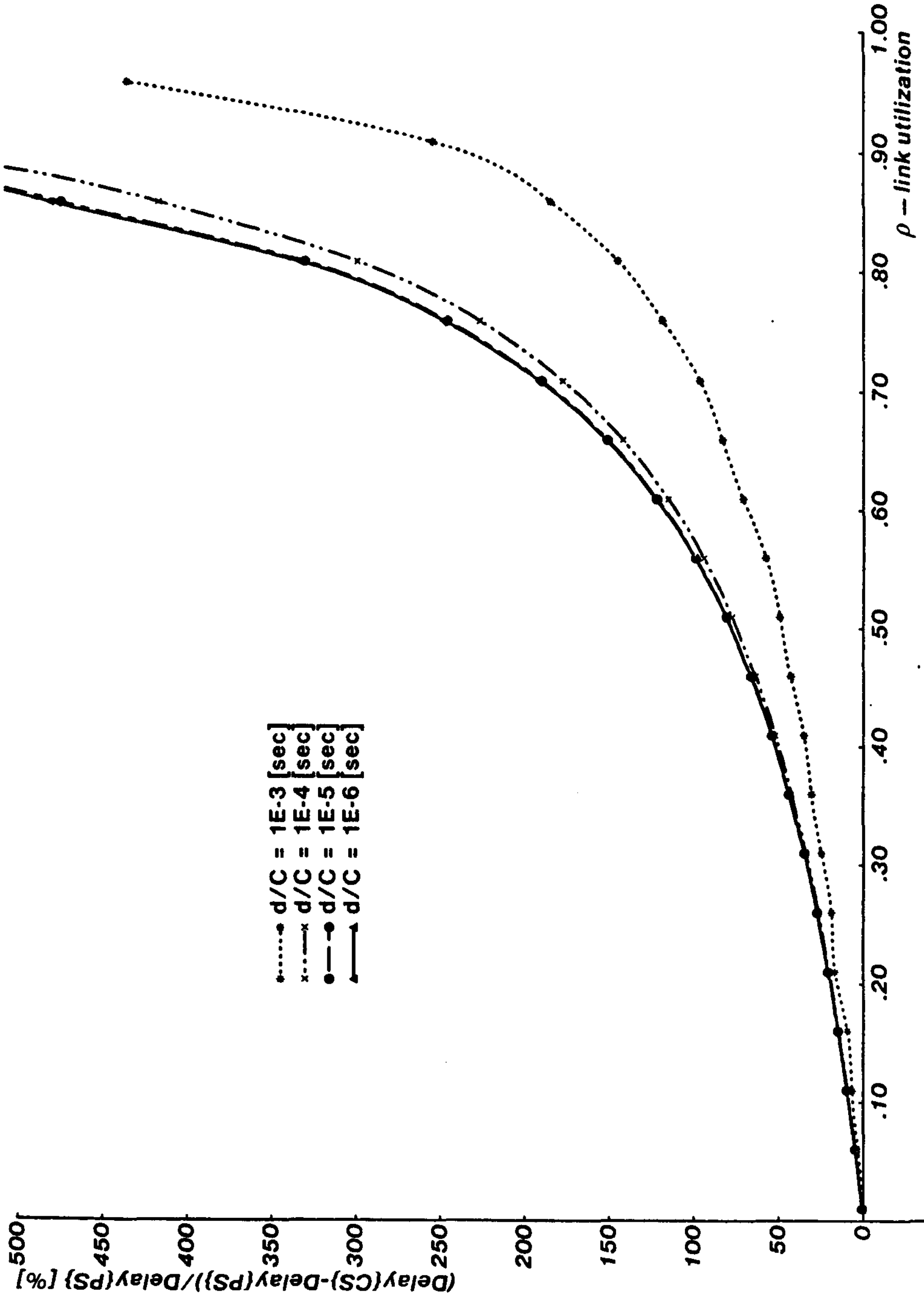


Figure 4.7: Relative increase of total delay in CS over PS; l=10 [hops], α=0.010 [sec], C=variable

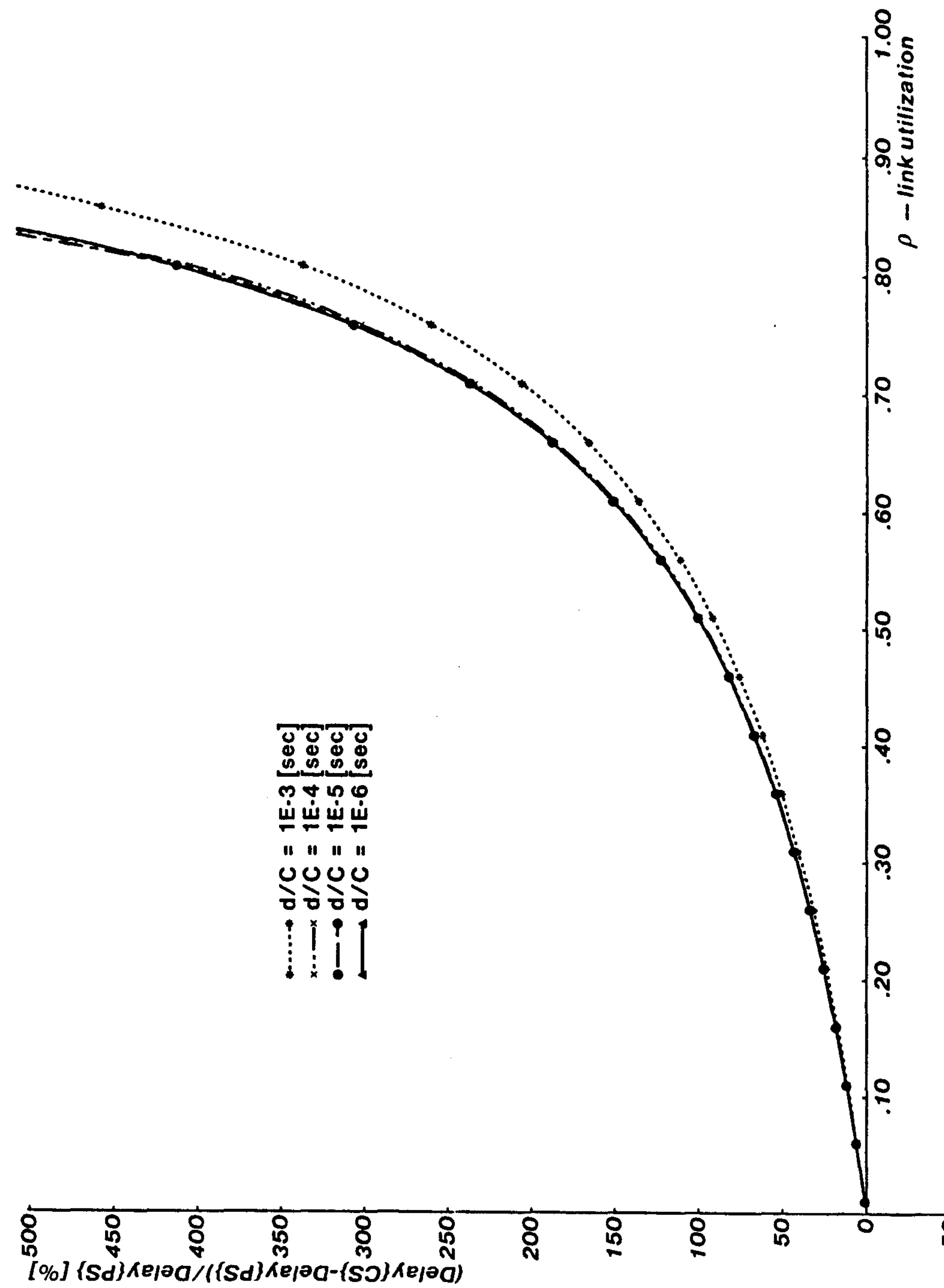


Figure 4.8: Relative increase of total delay in CS over PS; l=10 [hops], α=0.100 [sec], C=variable

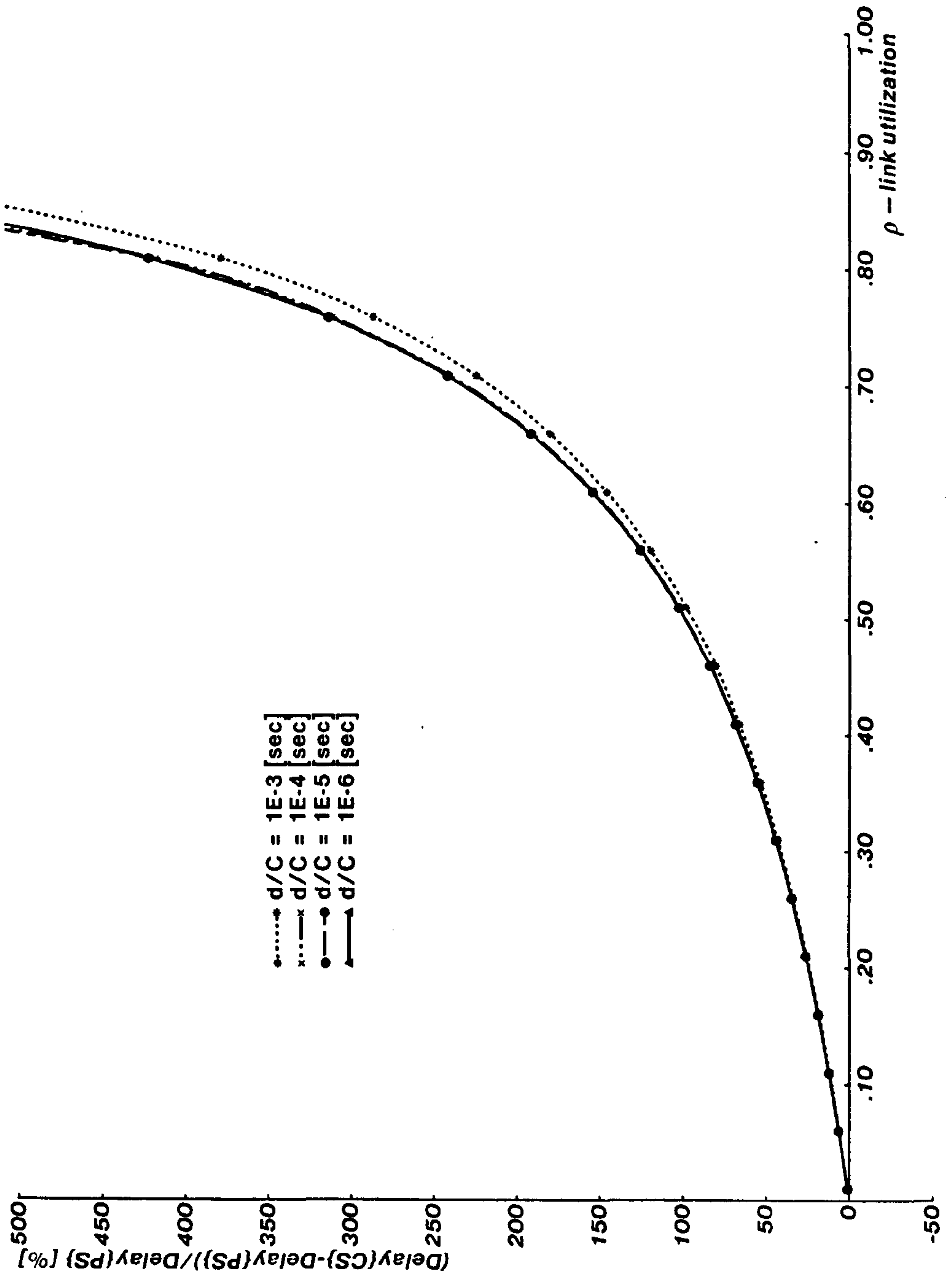


Figure 4.9: Relative increase of total delay in CS over PS; l=10 [hops], α=0.200 [sec], C=variable

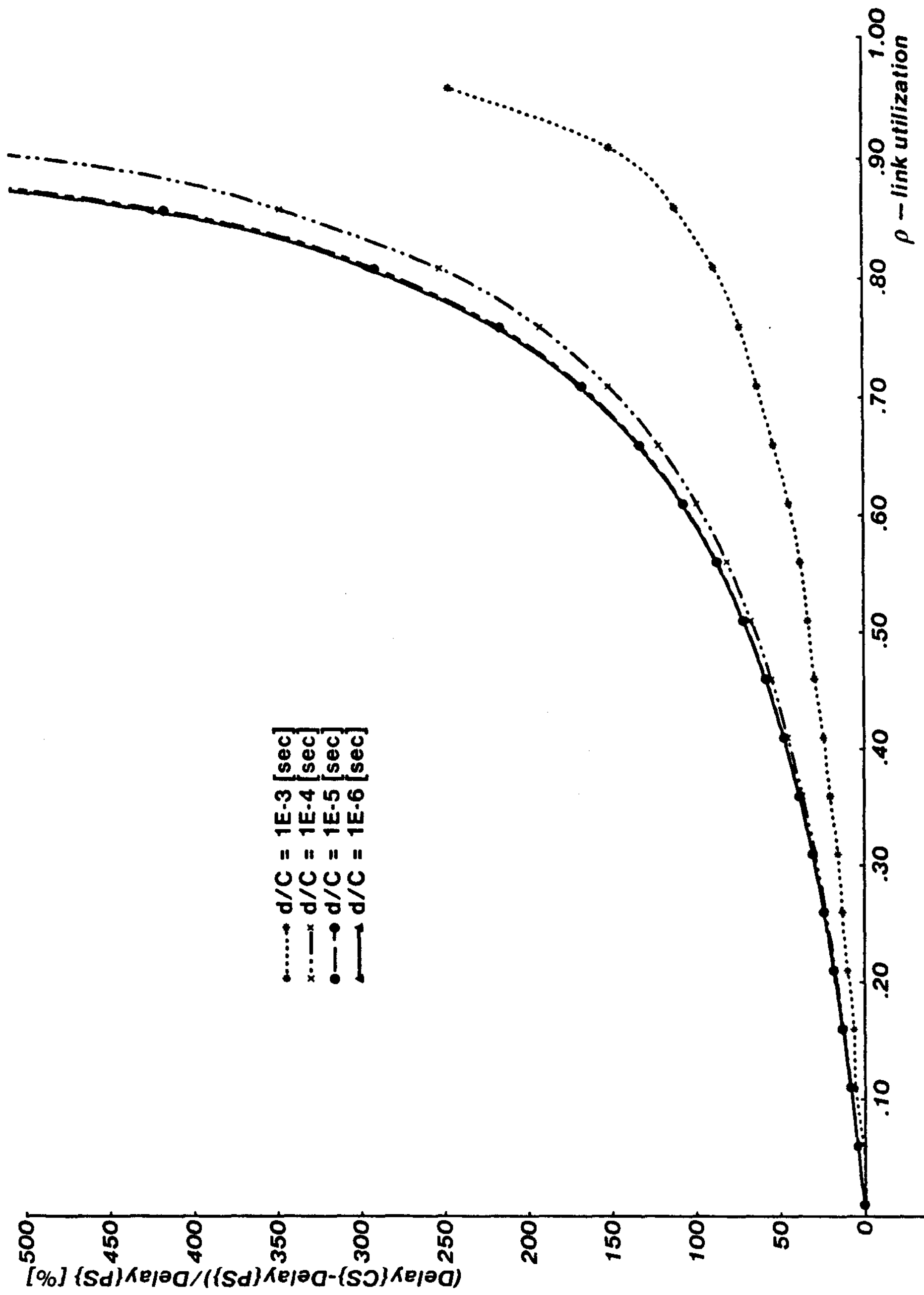


Figure 4.10: Relative increase of total delay in CS over PS; l=20 [hops], α=0.010 [sec], C=variable

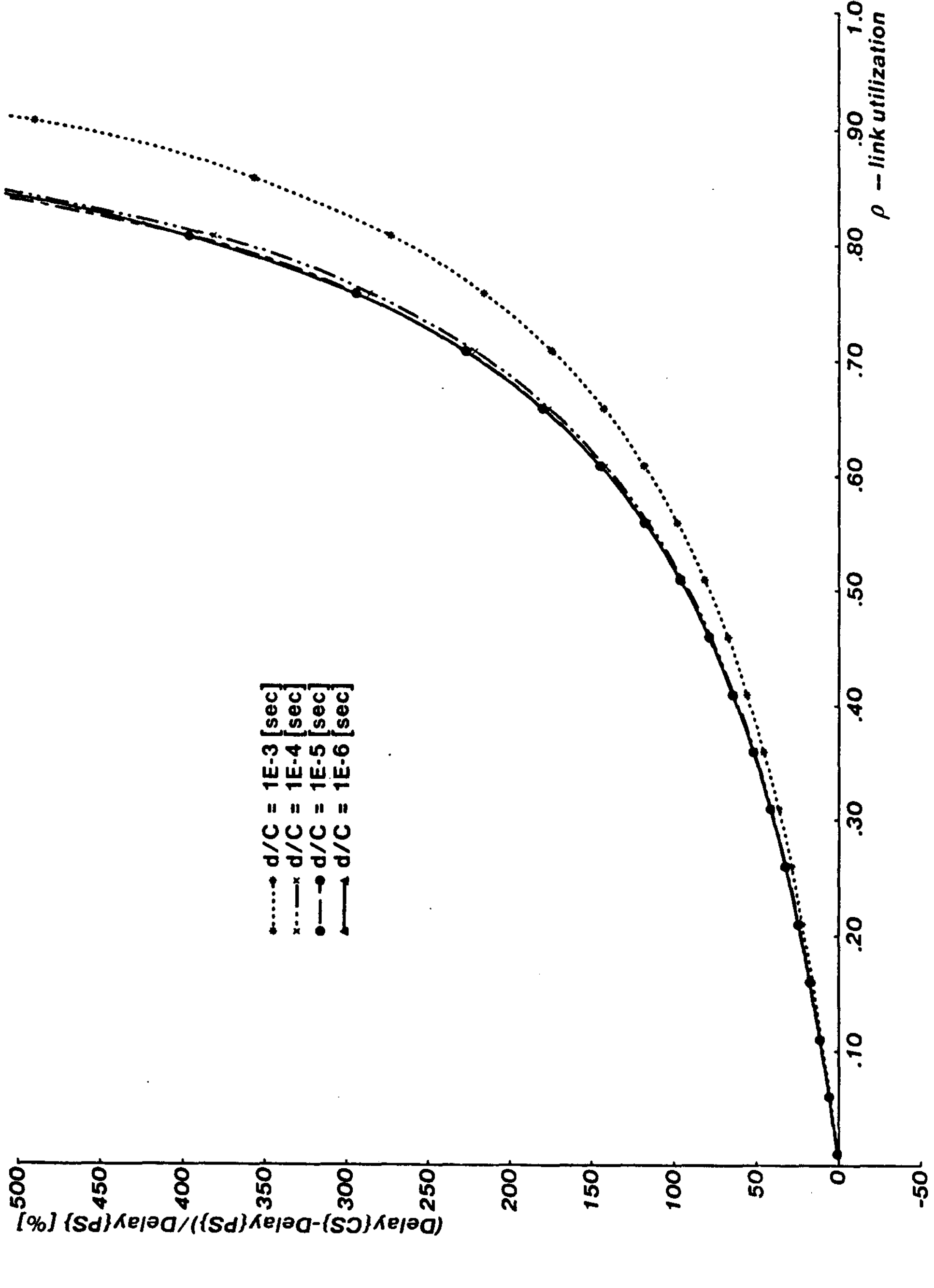


Figure 4.11: Relative increase of total delay in CS over PS; l=20 [hops], α=0.100 [sec], C=variable

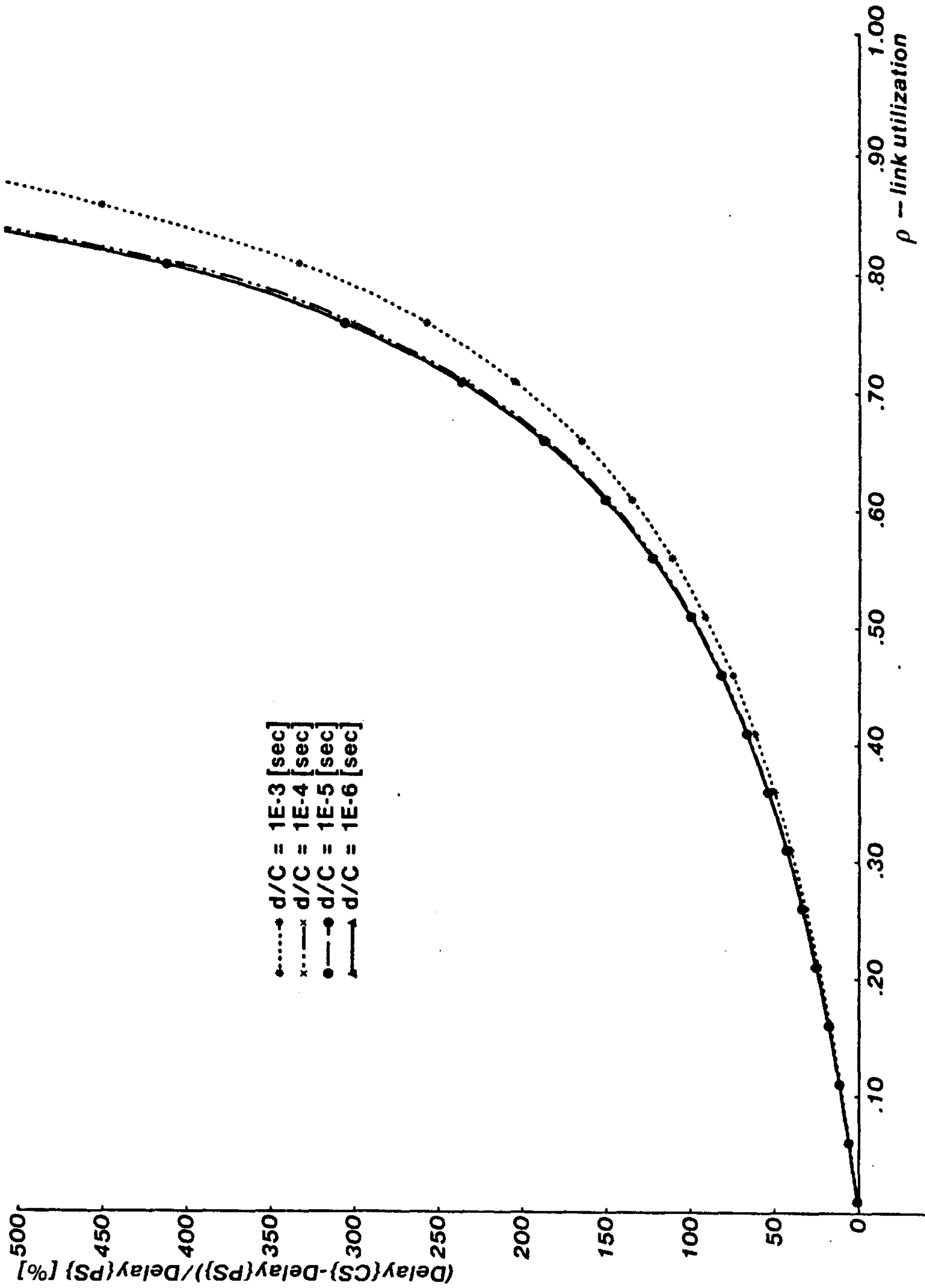


Figure 4.12: Relative increase of total delay in CS over PS; l=20 [hops], α=0.200 [sec], C=variable

message that needs no processing time, but that is communicated in an environment of messages that have some processing time requirements. This might be the case, for example, with high-priority traffic requiring immediate and short response. The relative increase in the net communication delay is calculated using the following formula:

$$\frac{(Delay\{CS\} - \alpha) - (Delay\{PS\} - \alpha)}{(Delay\{PS\} - \alpha)} \; [\%] \, . \tag{4.22}$$

These comparisons were made for the whole range of ρ and for d/C as a parameter. The results for all the nine combinations of α and l are presented in two sets of nine separate graphs each: Figures 4.4 - 4.12 for the total delay and Figures 4.13 - 4.21 for the net communication delay.

Graphs 4.22 and 4.23 show respectively the effect of variations in l and in α.

In order to examine the effect of keeping the number of sub-channels constant (no optimization on N), the comparison was made for six combinations of parameters. The results are shown in Figures 4.24 - 4.29. In each case three values were chosen for N. The chosen values are those that optimize the circuit-switching delay for ρ: 0.26, 51, and 0.76.

The last graph in Figure 4.30 shows the boundaries between the two techniques in the $l \times (d/C)$ plane with α as parameter. In the graph the area above any of the curves specifies the region where message-switching has lower delay than circuit-switching for any value of ρ. However, the boundaries in the graph should be considered more as limits than as exact boundaries because of the many conservative assumptions used in the process of deriving the graph.

4.7 Discussion

The comparison of the two switching techniques, as presented in Figures 4.4 to 4.12, does not reveal the main disadvantage of the CS scheme, since the increase in the CS delay is dominated by the large values of propagation delay and of α. Figures 4.13 to 4.21 show the comparison for net communication delay, i.e., excluding α. The dramatic overhead of the CS scheme can now be noticed, especially for short path

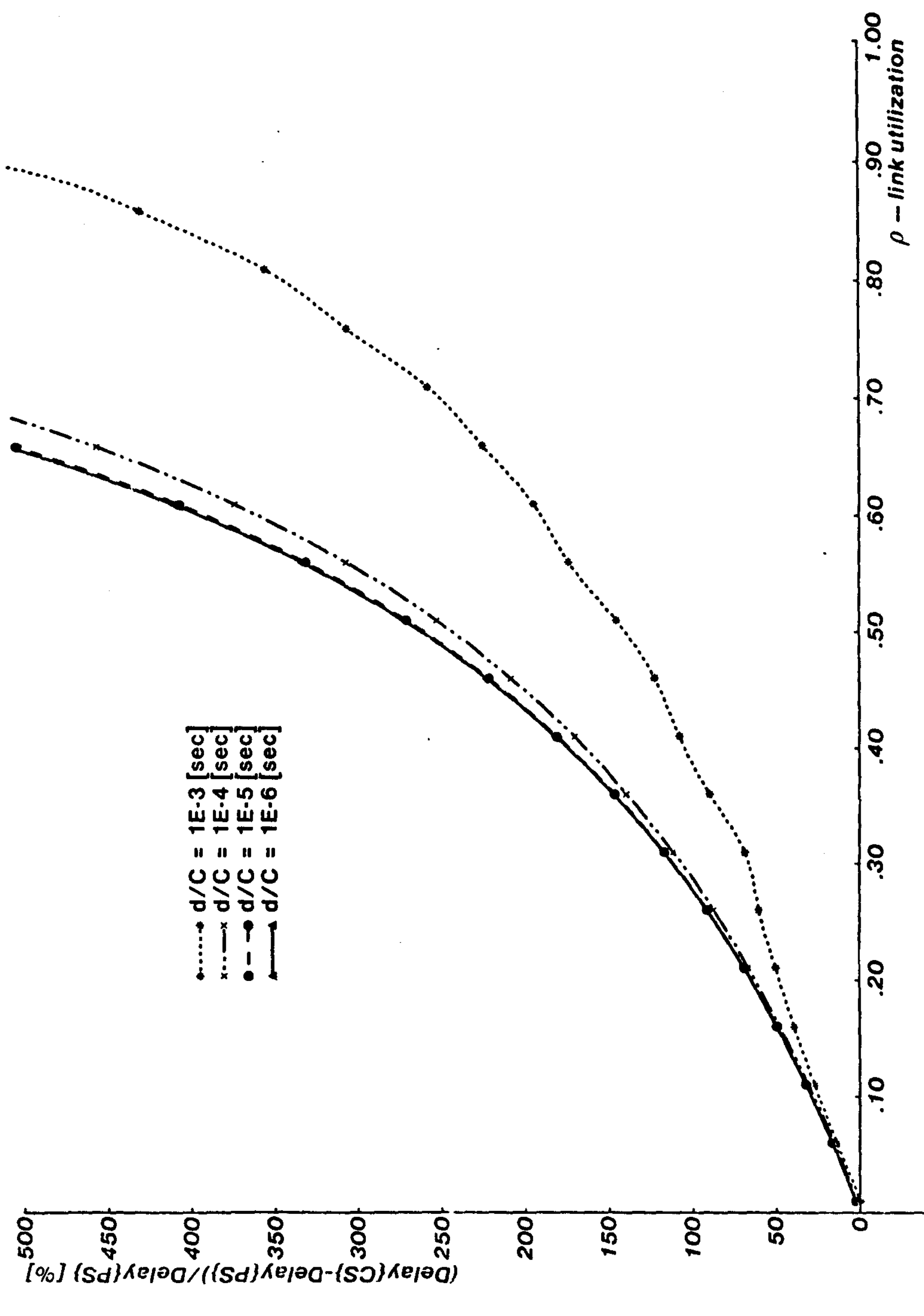


Figure 4.13: Relative increase of net transmission delay in CS over PS; l=5 [hops], α=0.010 [sec], C=variable

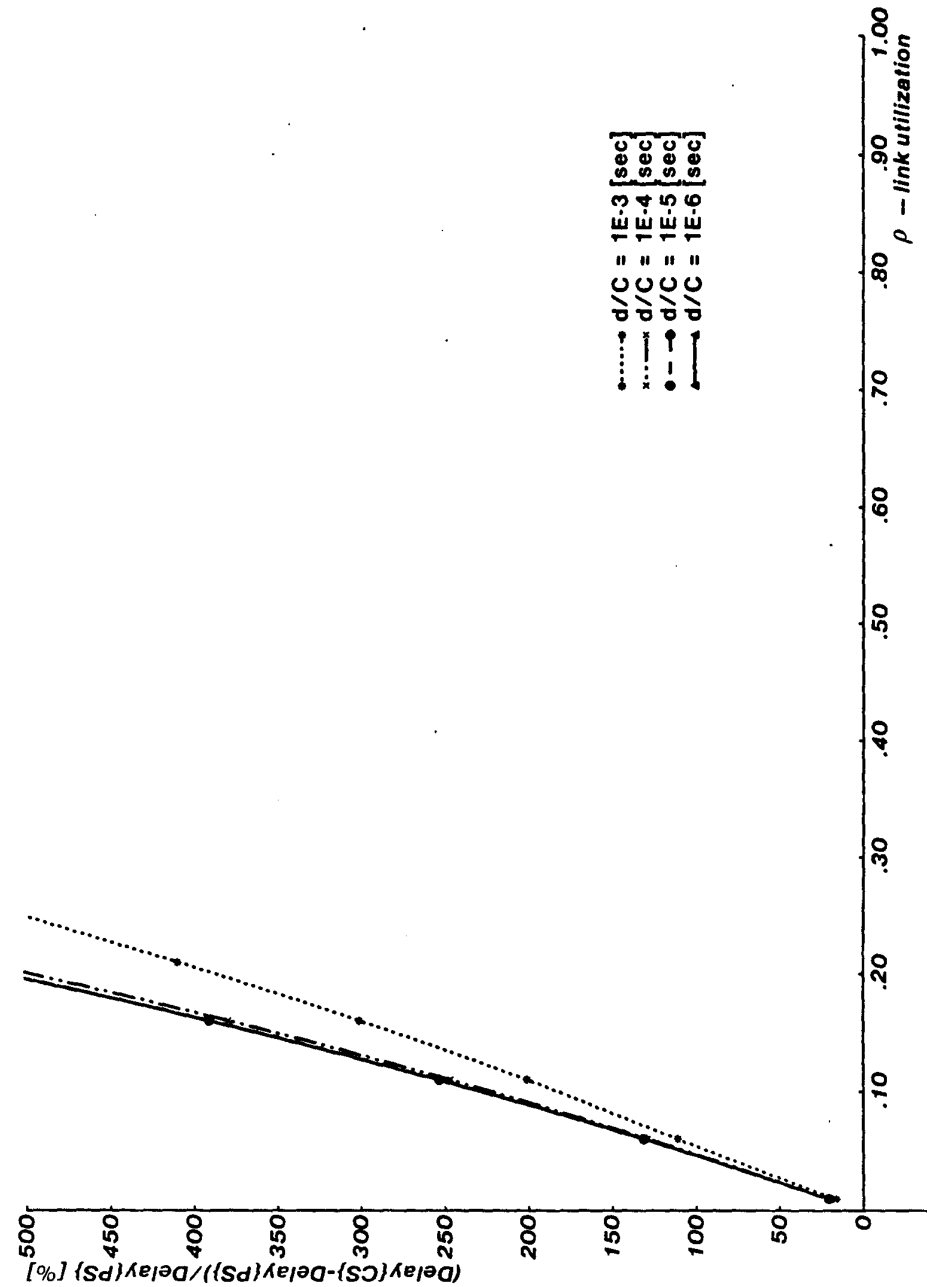


Figure 4.14: Relative increase of net transmission delay in CS over PS; l=5 [hops], α=0.100 [sec], C=variable

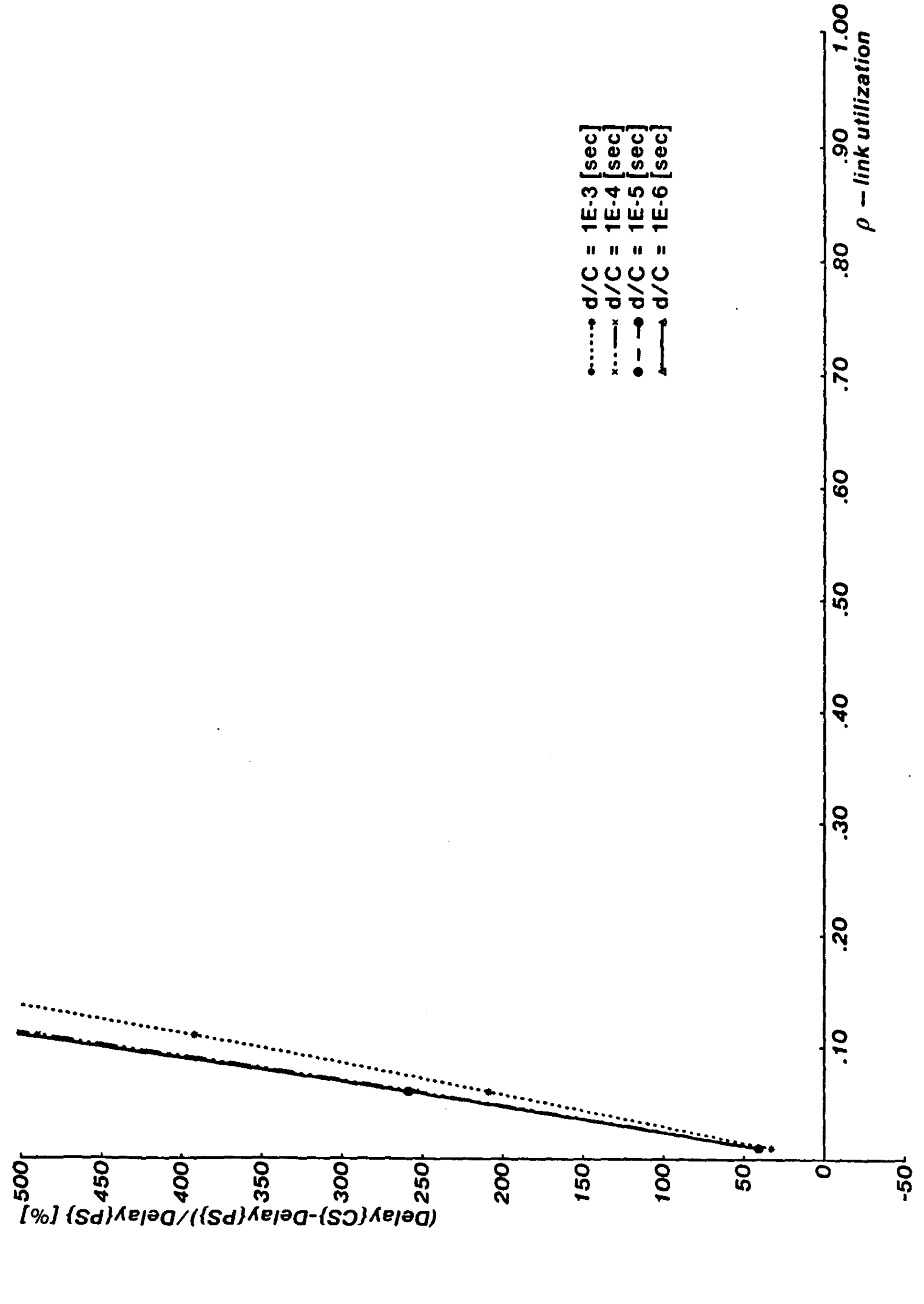


Figure 4.15: Relative increase of net transmission delay in CS over PS; l=5 [hops], α=0.200 [sec], C=variable

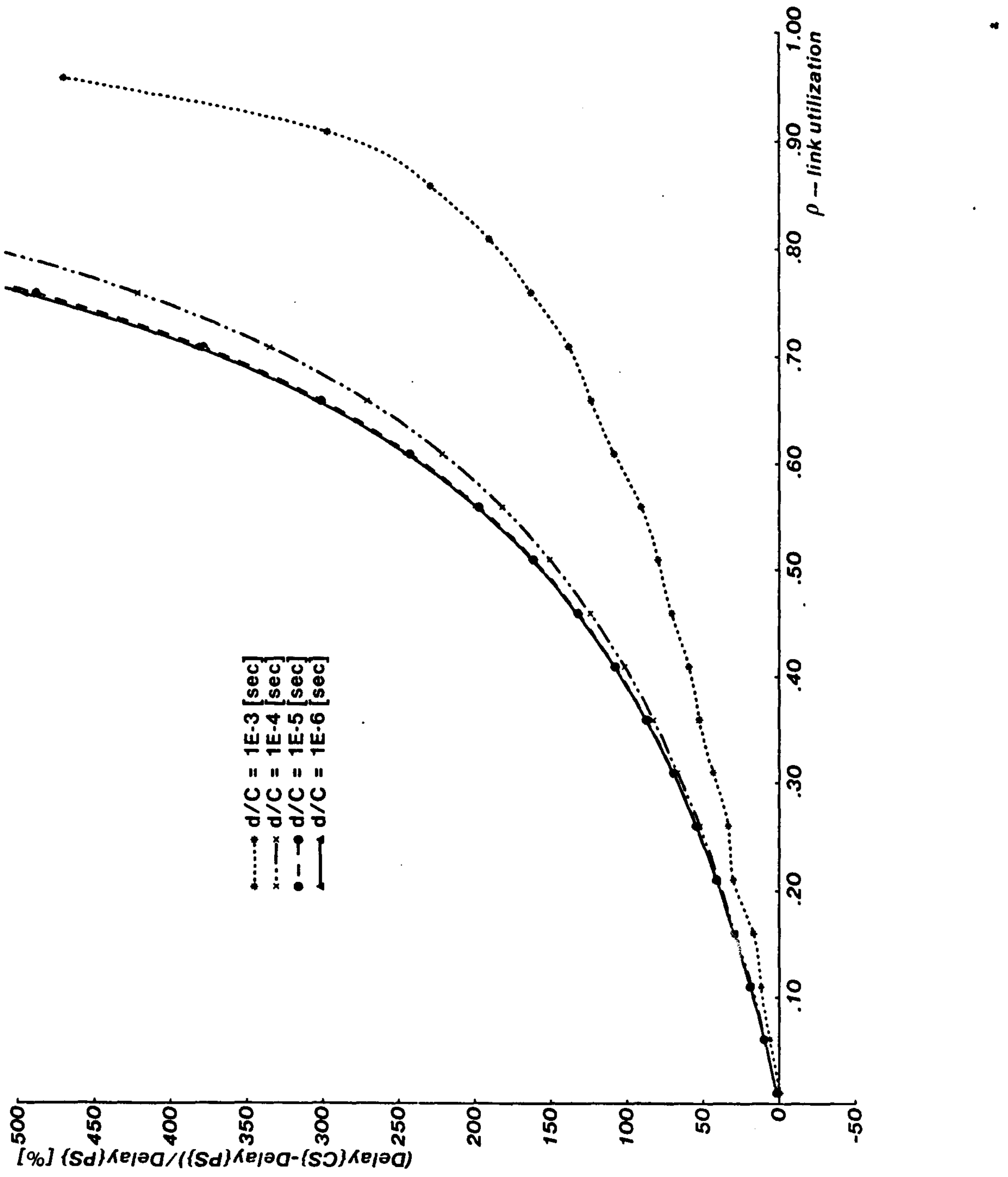


Figure 4.16: Relative increase of net transmission delay in CS over PS; l=10 [hops], α=0.010 [sec], C=variable

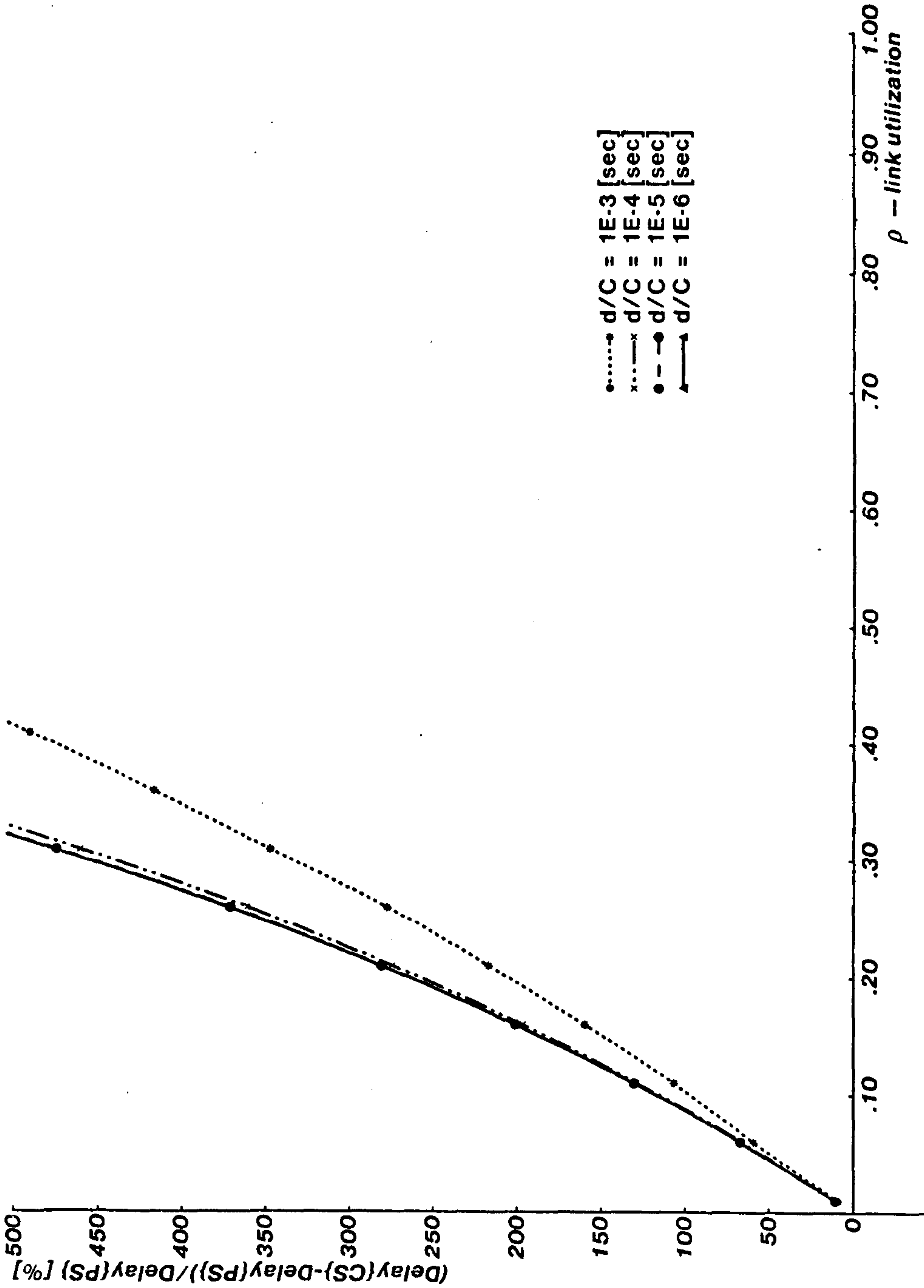


Figure 4.17: Relative increase of net transmission delay in CS over PS; l=10 [hops], α=0.100 [sec], C=variable

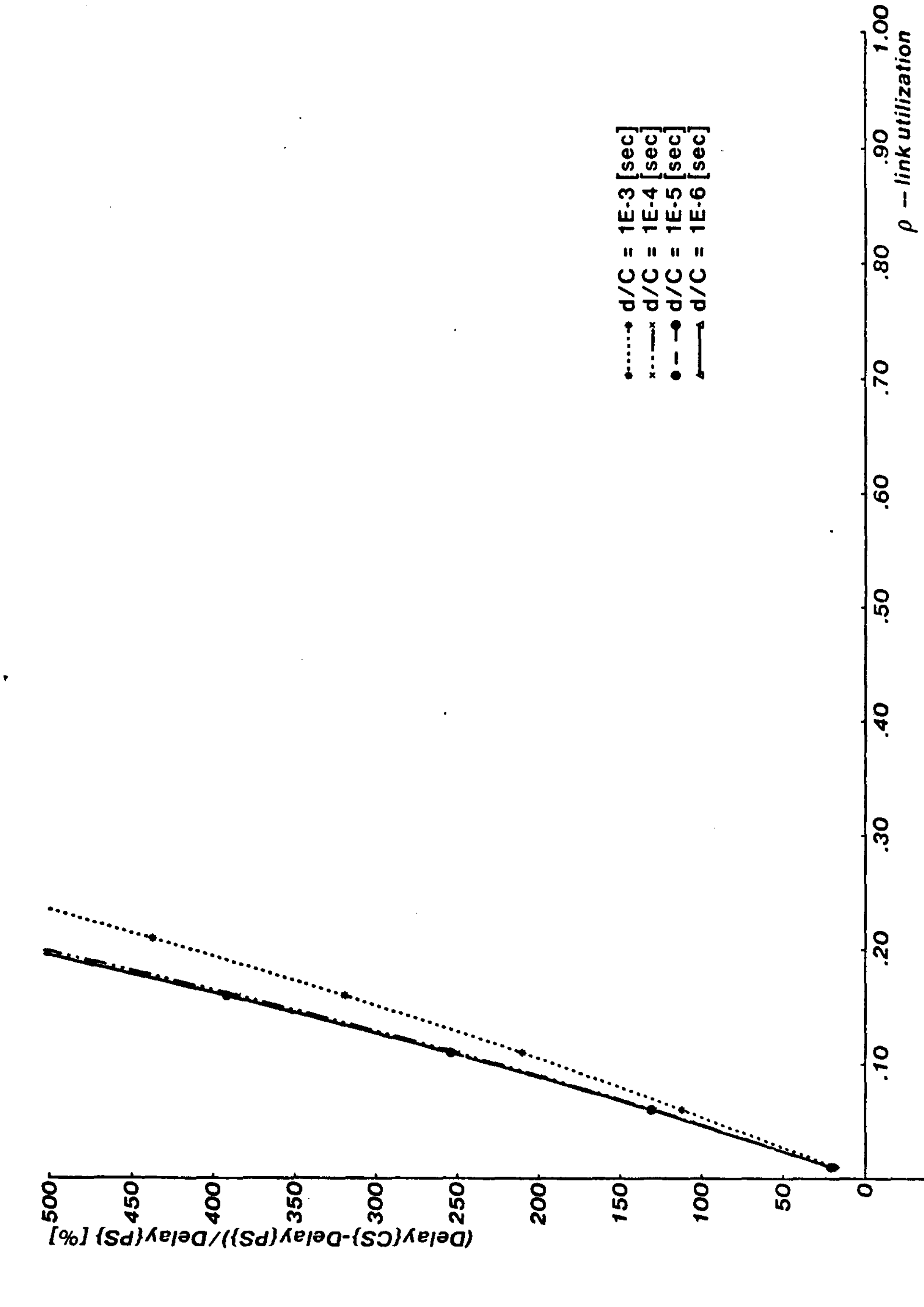


Figure 4.18: Relative increase of net transmission delay in CS over PS; l=10 [hops], α=0.200 [sec], C=variable

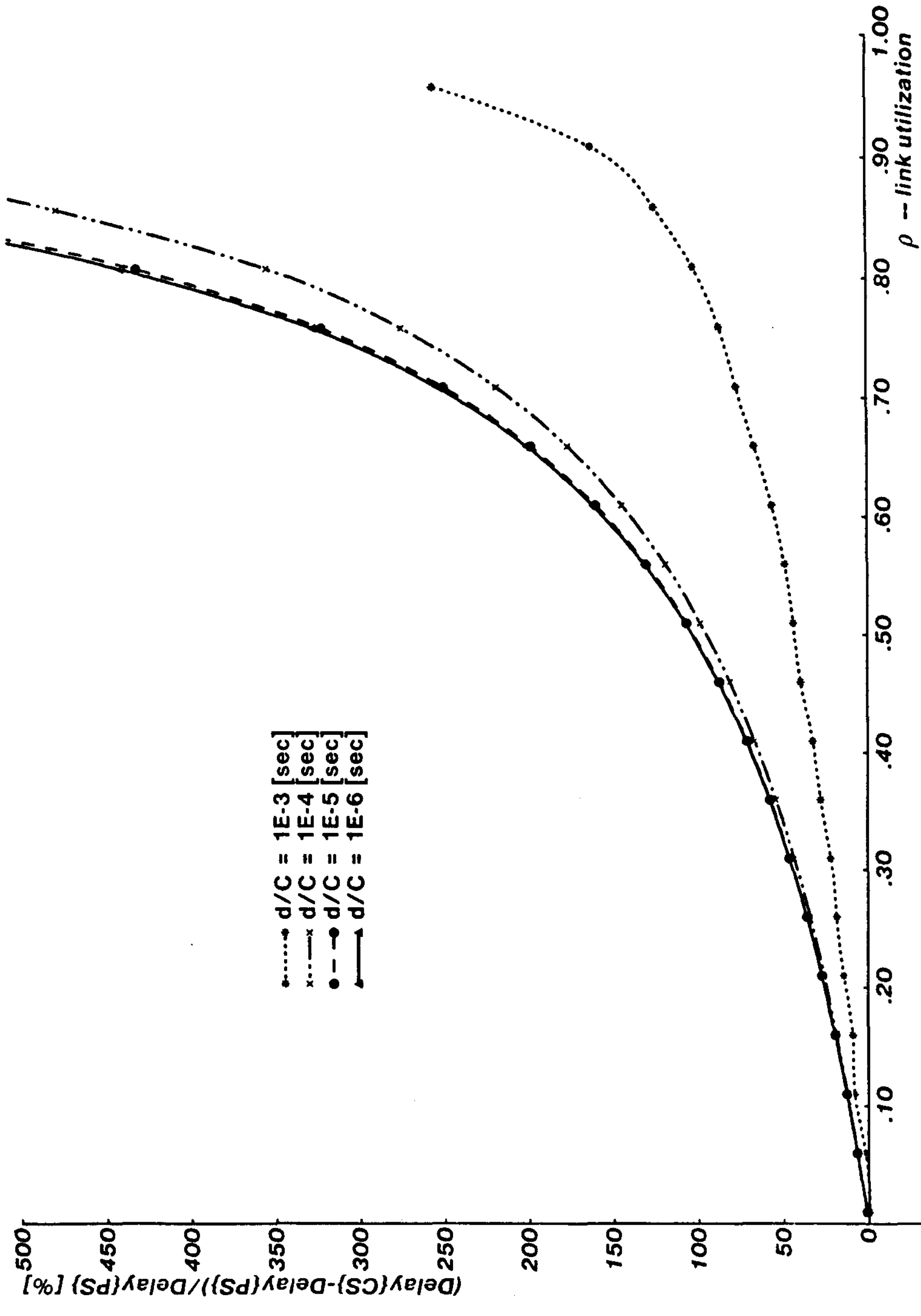


Figure 4.19: Relative increase of net transmission delay in CS over PS; l=20 [hops], α=0.010 [sec], C=variable

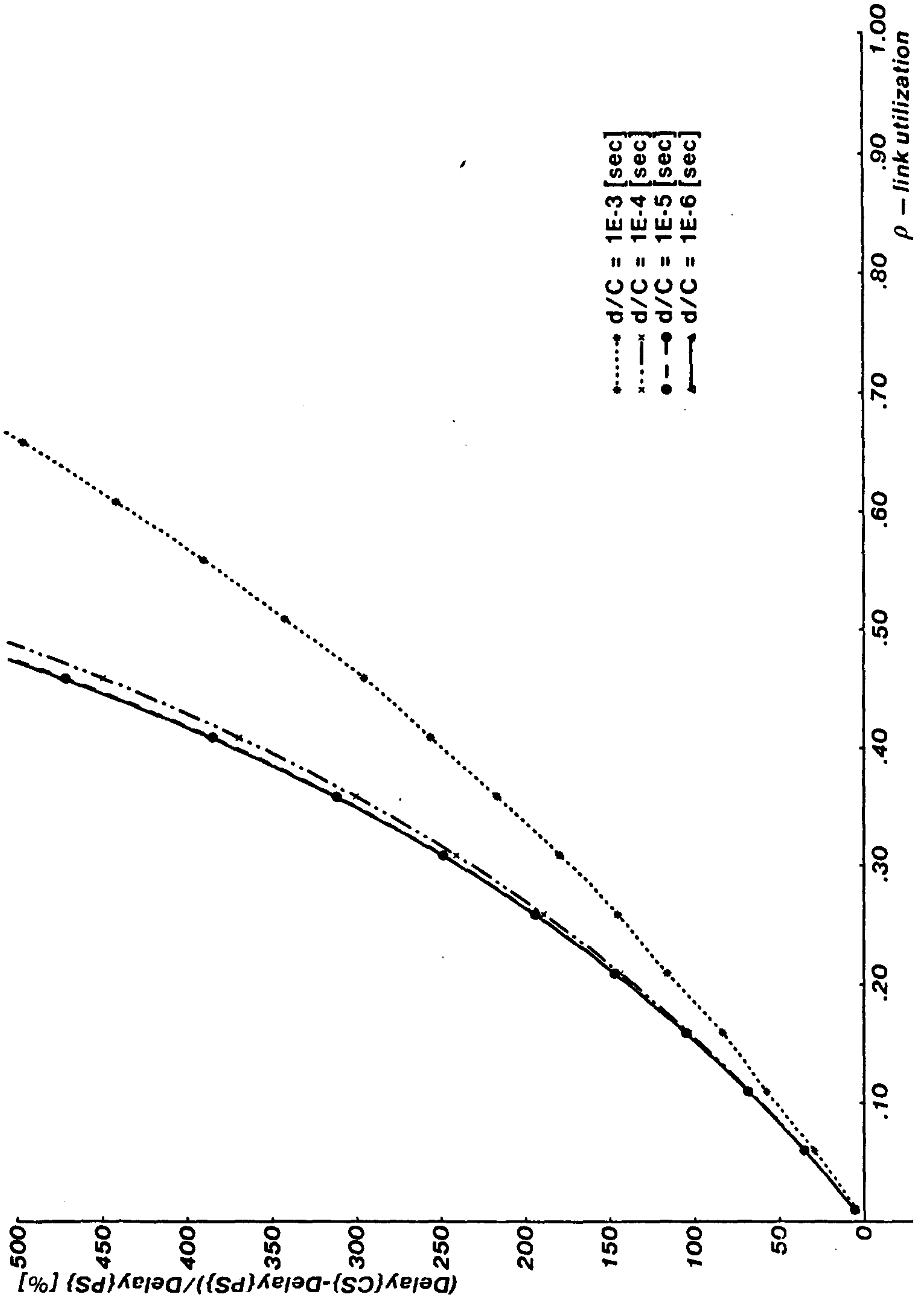


Figure 4.20: Relative increase of net transmission delay in CS over PS; l=20 [hops], α=0.100 [sec], C=variable

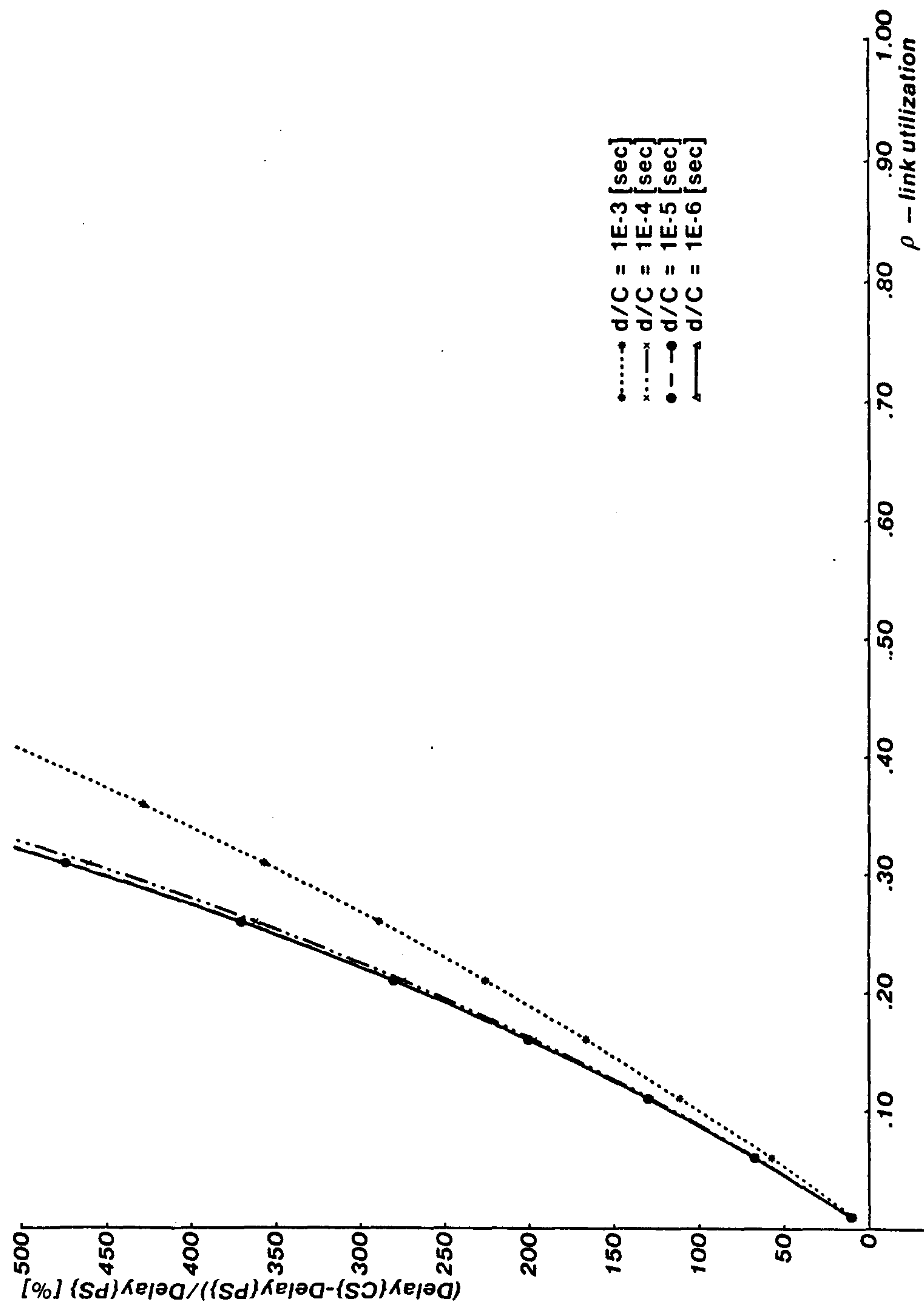


Figure 4.21: Relative increase of net transmission delay in CS over PS; l=20 [hops], α=0.200 [sec], C=variable

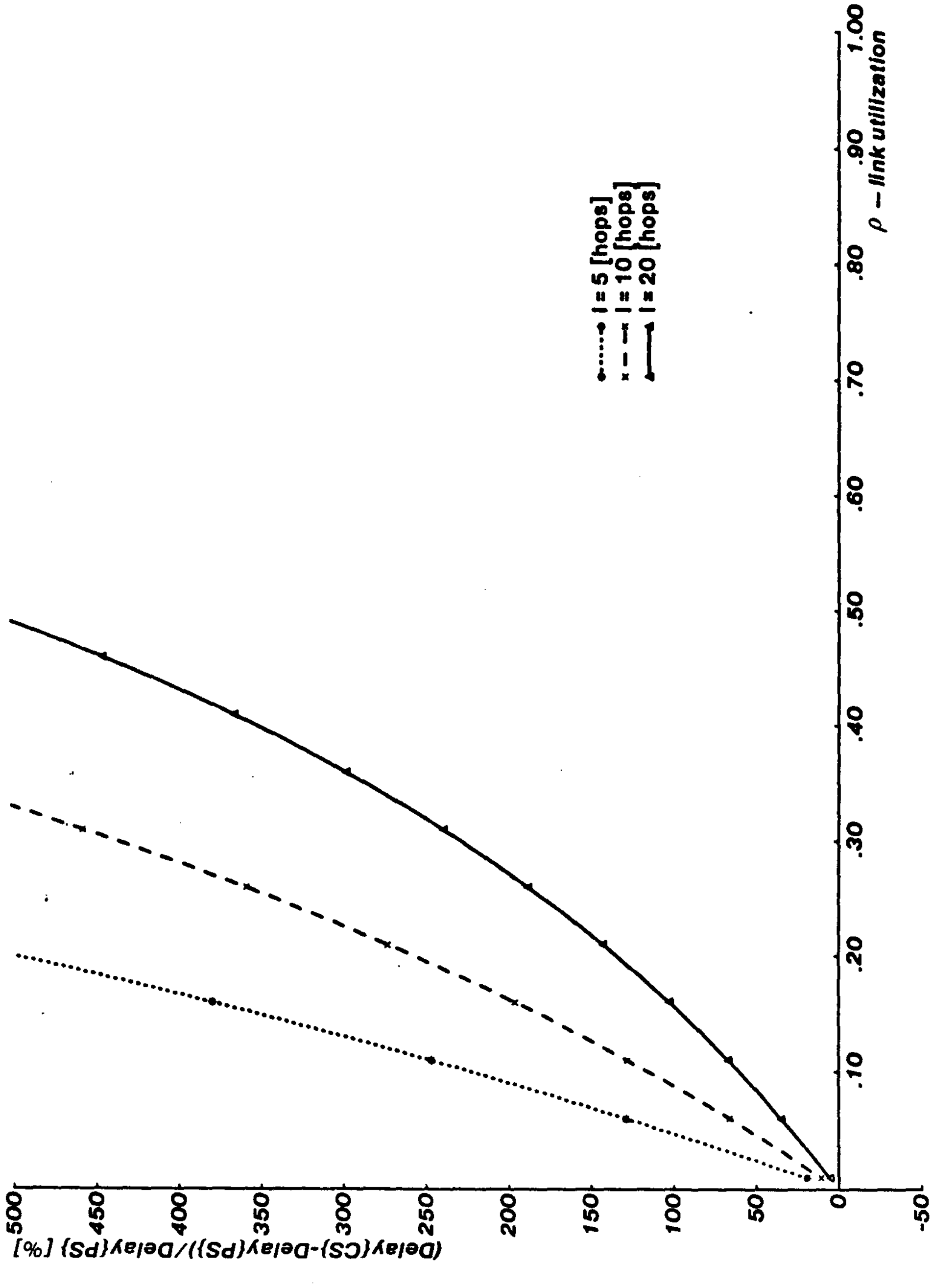


Figure 4.22: Effect of variations in l; Net communication delay; l=variable, α=0.100 [sec], d/C=1E-4 [sec]

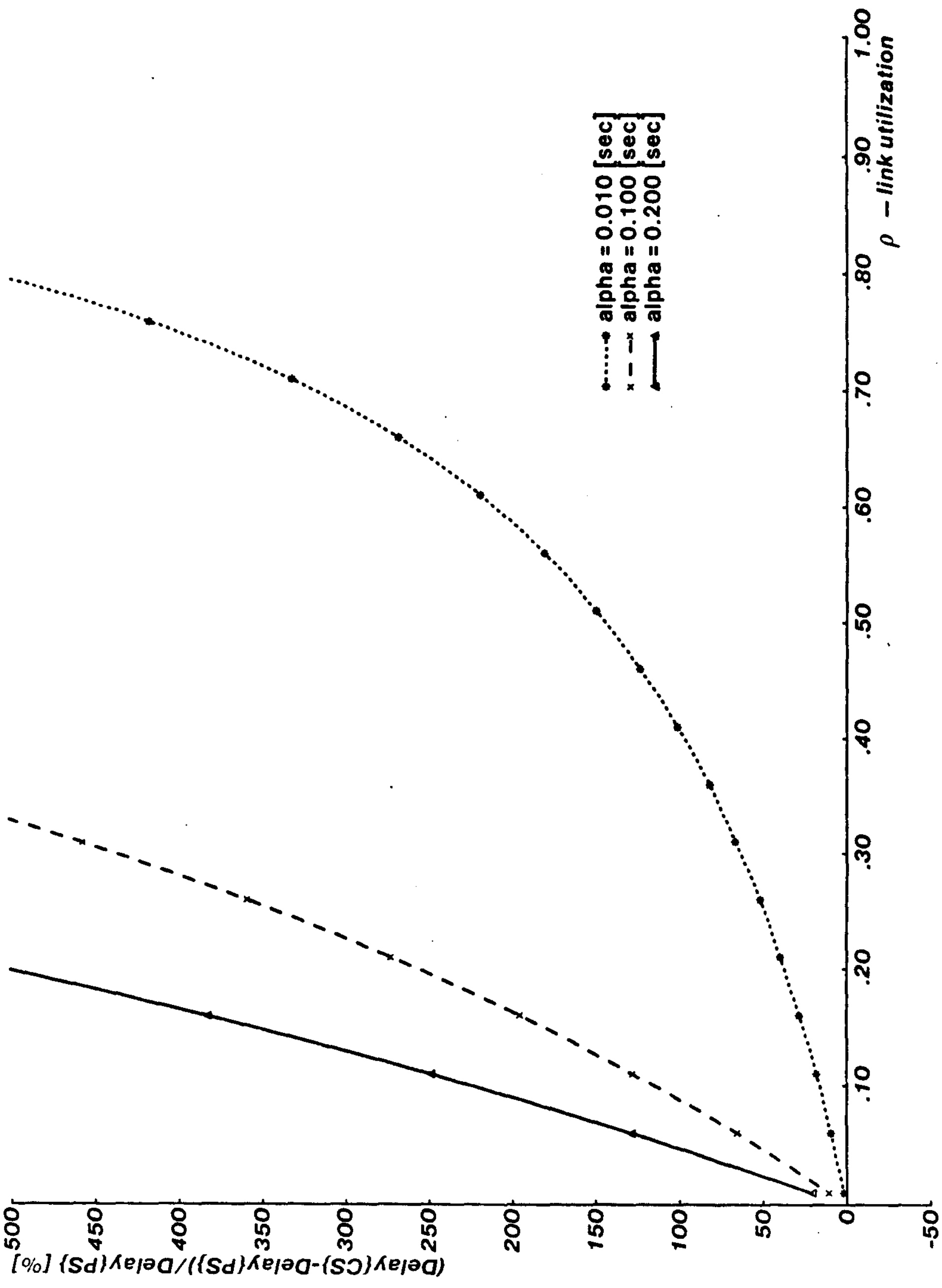


Figure 4.23: Effect of variations in α; Net communication delay; l=10 [hops], α=variable, d/C=1E-4 [sec]

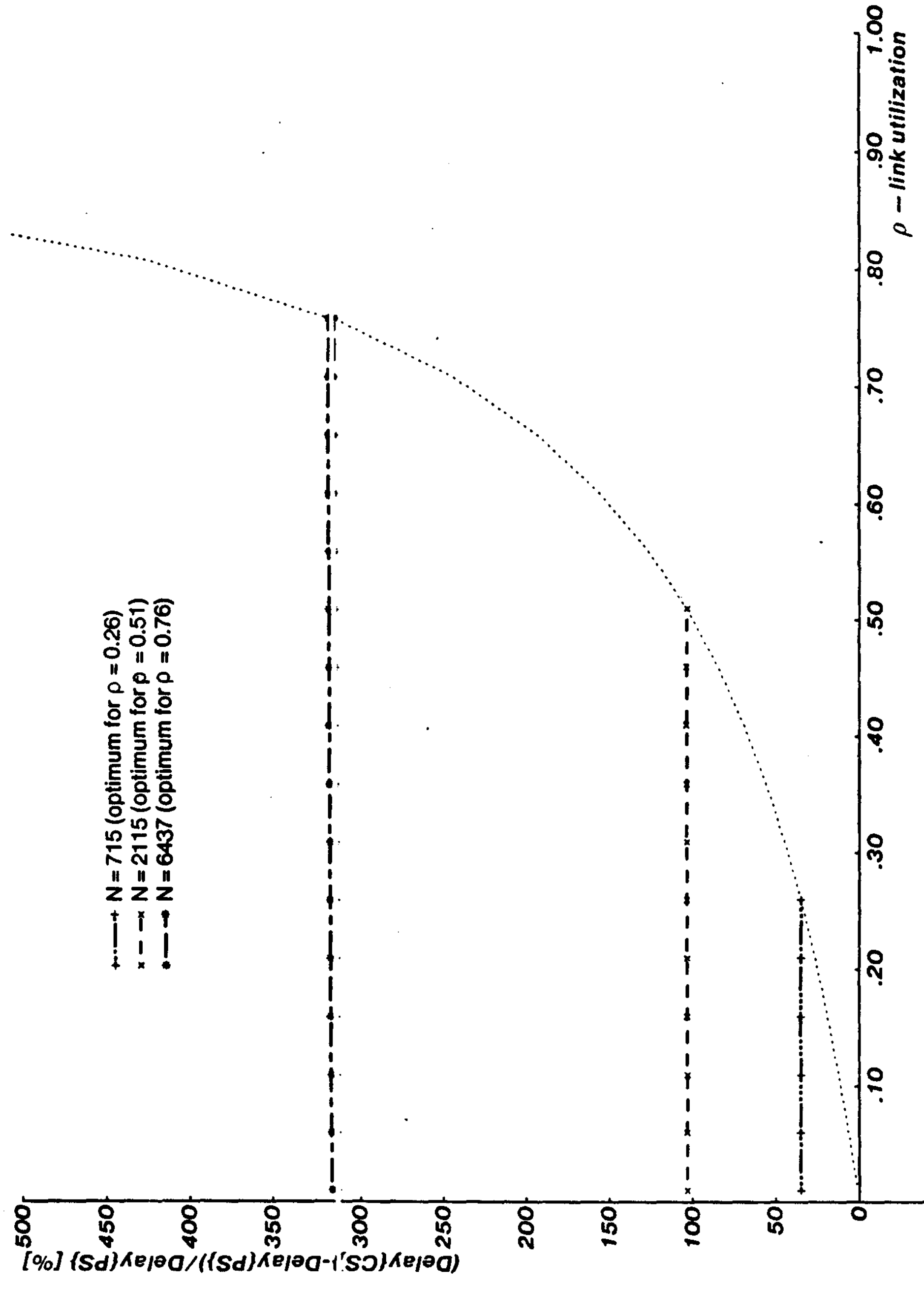


Figure 4.24: Effect of fixing N; Total delay: l=5 [hops], α=0.200 [sec], d/C=1E-4 [sec]

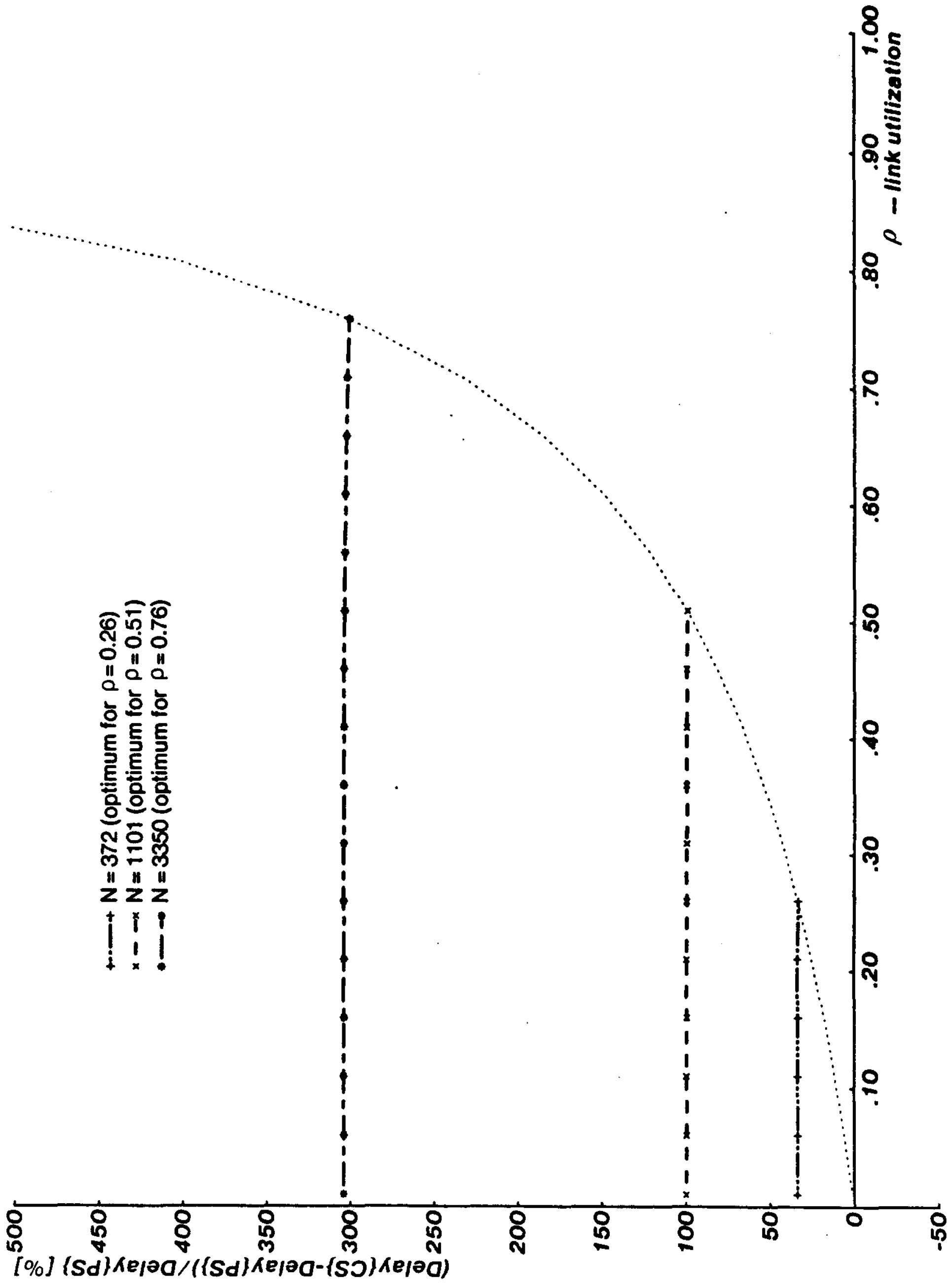


Figure 4.25: Effect of fixing N; Total delay: l=10 [hops], α=0.100 [sec], d/C=1E-4 [sec]

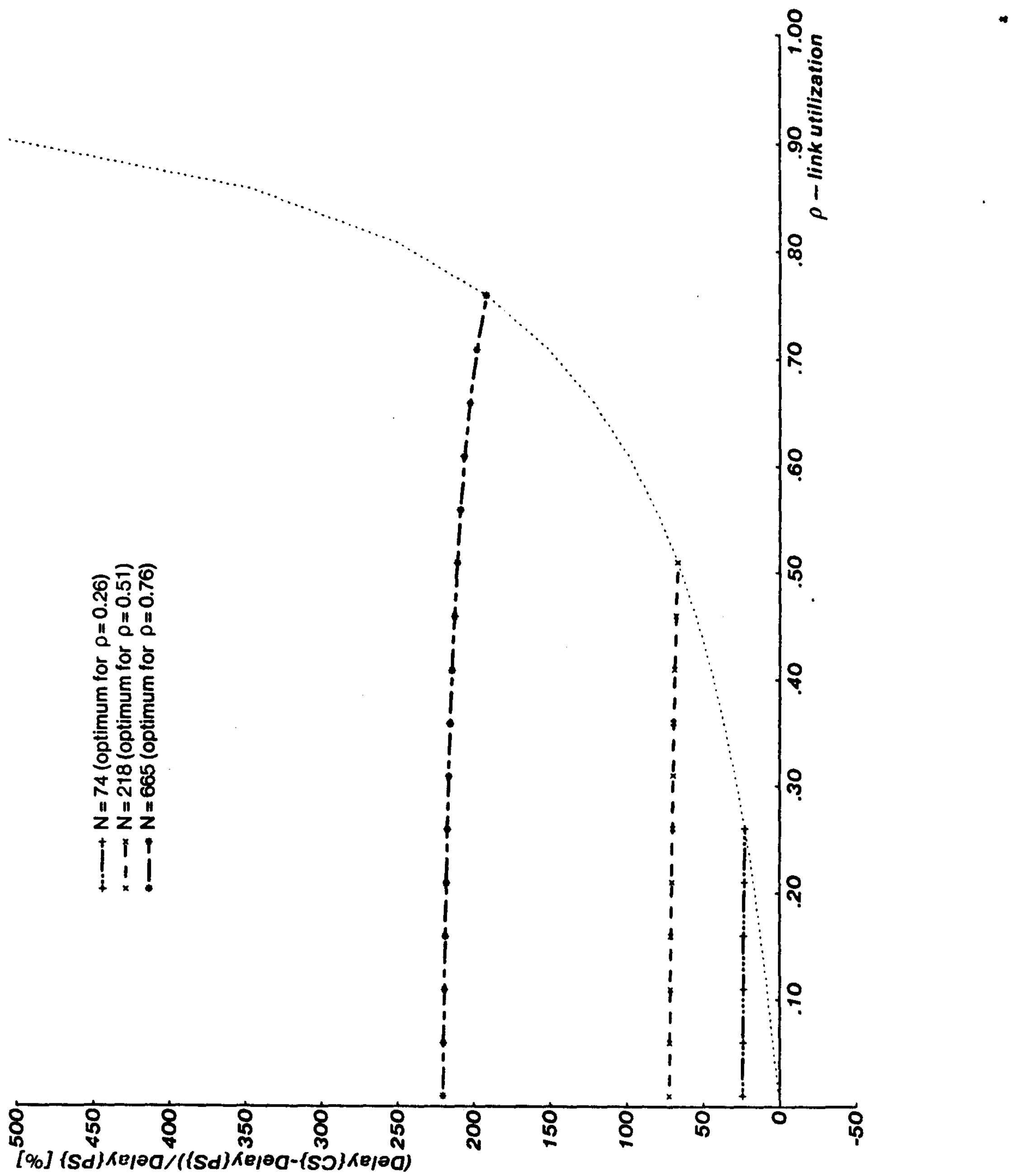


Figure 4.26: Effect of fixing N; Total delay: l=20 [hops], α=0.010 [sec], d/C=1E-4 [sec]

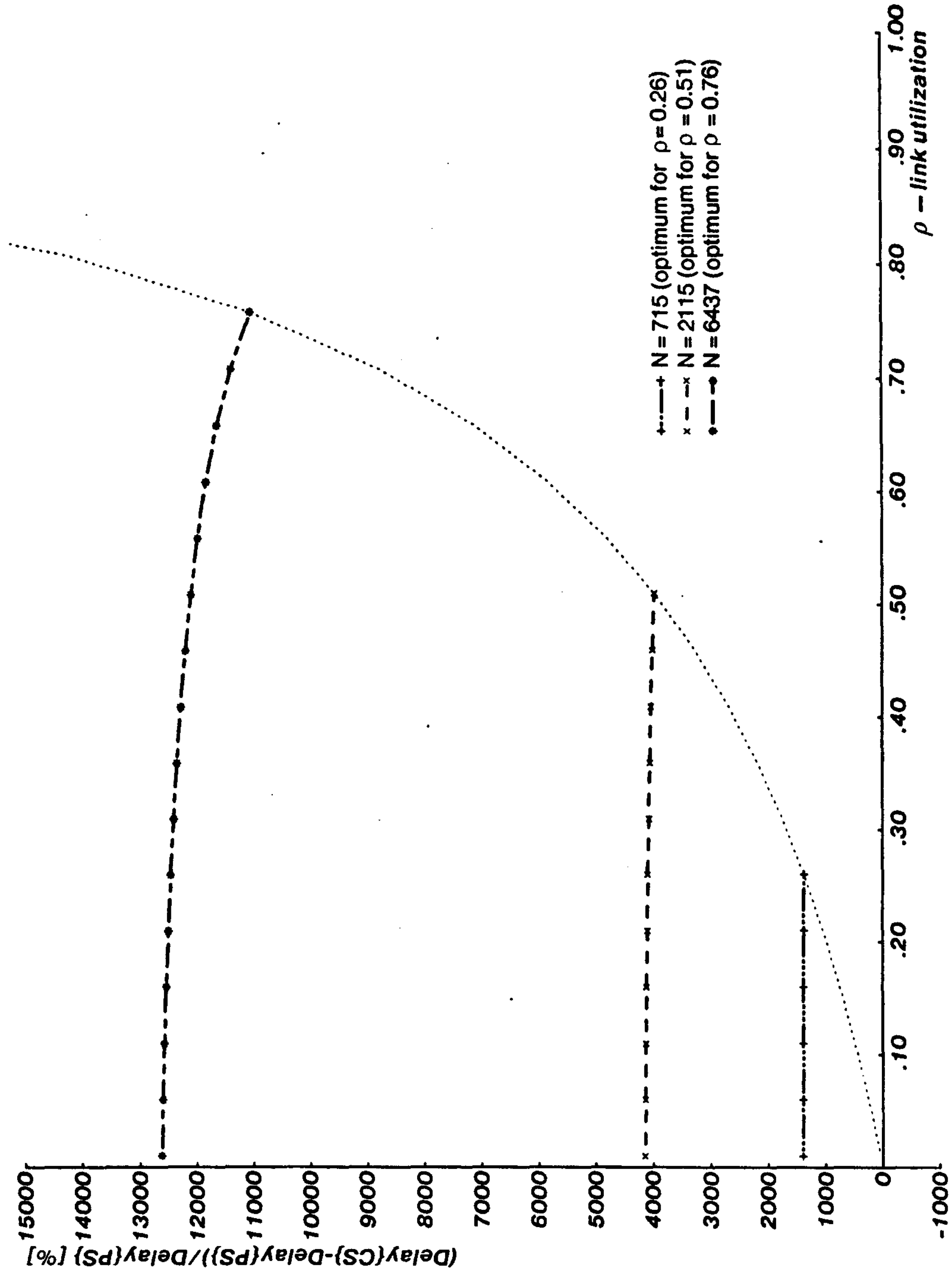


Figure 4.27: Effect of fixing N; Net communication delay: l=5 [hops], α=0.200 [sec], d/C=1E-4 [sec]

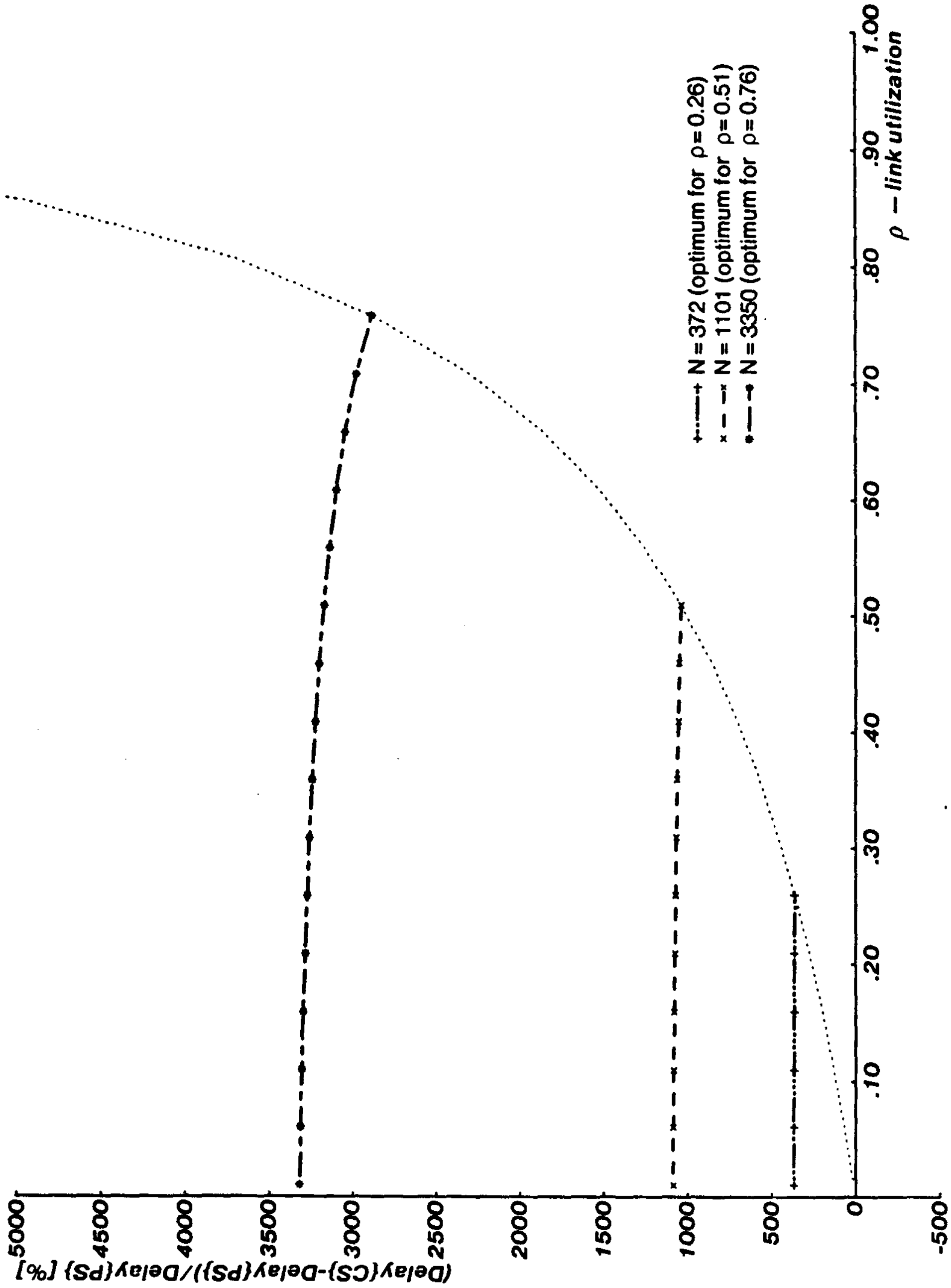


Figure 4.28: Effect of fixing N; Net communication delay: l=10 [hops], α=0.100 [sec], d/C=1E-4 [sec]

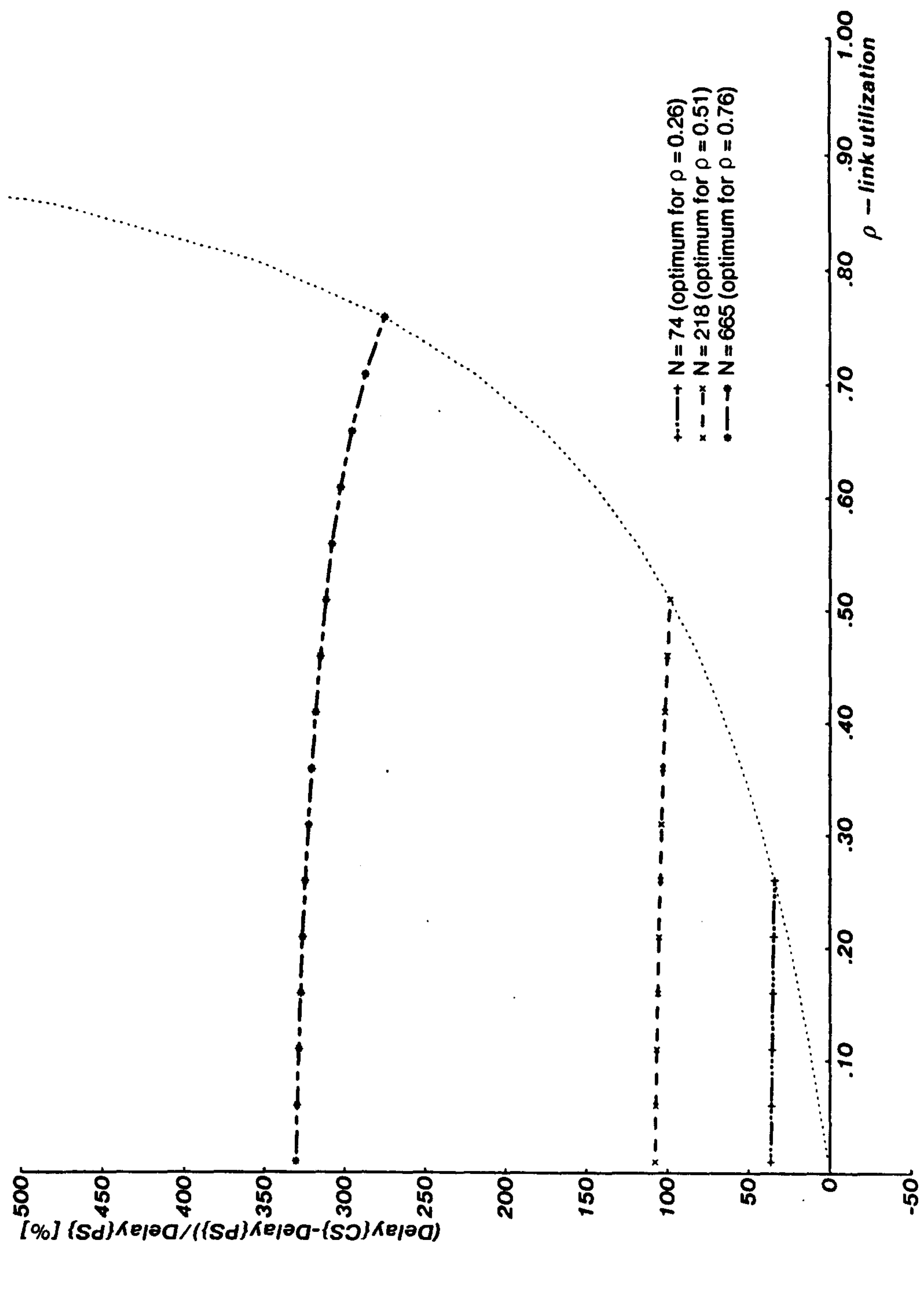


Figure 4.29: Effect of fixing N; Net communication delay: l=20 [hops], α=0.010 [sec], d/C=1E-4 [sec]

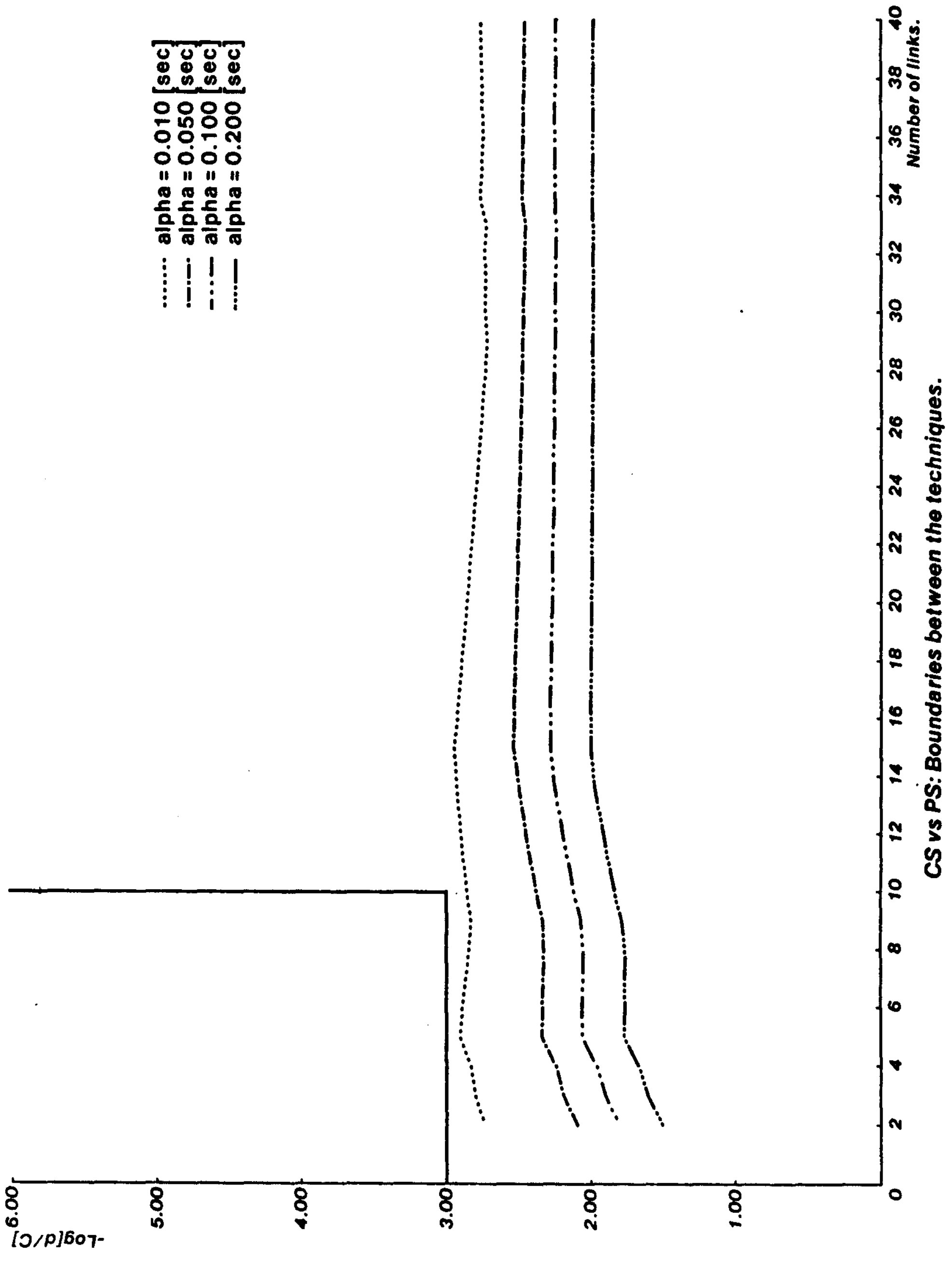


Figure 4.30: Boundaries between CS and PS

and high values of α. For example, in the case of l=5 [hops], α=0.200 [sec], and $d/C = 10^{-4}$ [sec] (Figure 4.15) the CS net communication delay is about four times larger than that of the PS scheme for ρ=0.1. Moreover, this set of graphs reveals that as the channel capacity increases the relative increase in the net communication delay of CS over PS also increases. Thus, as the bandwidth increases, PS becomes a more attractive switching method.

This sensitivity to the value of α is also demonstrated in Figure 4.22. (In Stanford V-system, the average values of α during periods of high activity were in the range of 50-70 msec, and the average value over all times was about 40 msec, as discussed in [40]. Of course, servers in wide-area networks serving larger user community may experience higher load, which results in longer processing delay.) The sensitivity of the comparison between the two switching techniques to the path length, l, is shown in Figure 4.23. The conclusion that I draw from these graphs is that the parameters l, α, and ρ strongly affect the relative performance of both schemes. The parameter d/C is less influential for and beyond the value of 10^{-4} [sec], especially for low ρ.

As pointed out in [15], long path and long messages (equivalent in the present work to large d/C parameter) favor CS in the sense of lower delay. However, as can be noticed from the graph in Figure 4.30, the range of the two parameters over which CS has lower delay than PS can hardly be met in high-performance systems. For example, for $d/C \leq 10^{-2}$ [sec] (which corresponds to a 10 Mbit message over a 1 Gbps channel), $\alpha \geq 0.100$ [sec], and any value l, PS always has lower delay than CS. As the channel capacity increases, the network operation point moves further from the boundary. This behavior again demonstrates that increasing the channel capacity favors PS.

The basic reason for this behavior is that a message that appears as stream traffic on a low-capacity link, occupies a high-capacity link for a short duration, thus appearing more bursty on high-speed media. Therefore, on a high-speed channel such a short burst strongly decreases the utilization of the link, especially in wide-area networking where the propagation delay (which does not scale down with increased capacity) forces a long channel reservation. (Also, the processing time may not scale down as much as the capacity does, and this may add an additional source of sub-channel holding time in the CS case.) Such long holding times imply long waiting times for a free sub-channel. To compensate for this behavior, the analysis dynamically increases the number of sub-channels by dividing the total capacity into smaller sub-channels. However, this increase in the number of sub-channels leads to increased transmission time and increased total delay.

The dynamical optimization of the number of sub-channels with the network load needs to be approached with caution. Such dynamic adjustment might be impractical or highly expensive in some situations. In those situations where the number of sub-channels is fixed, there exists a ρ_{max}, beyond which the CS scheme saturates. (The value of ρ_{max} is close to the value of ρ for which this particular fixed number of sub-channels optimizes the CS delay.) The CS delay remains relatively constant with ρ, up to ρ_{max}, where no further increase in input traffic can be achieved (as shown in Figures 4.24 to 4.29). However, this constant value of the CS delay is worse than the CS delay that results when the number of sub-channels is optimized. (For $\rho \ll \rho_{max}$ this constant value is much worse than the optimized value.) This behavior is shown in Figures 4.24 to 4.29 for different values of the parameters. In each case three values for the number of sub-channels have been used, the values that optimize the CS delay for three values of ρ: 0.26, 0.51, and 0.76. Thus, the tradeoff here is between the maximal available utilization, ρ_{max}, and the lowest (nearly constant) delay. Consequently, for the case of a fixed number of sub-channels in the CS scheme, the superiority of PS over CS becomes even more apparent.

As mentioned in Section 4.3, the model used for the comparison of the switching techniques assumes small requests and large responds. I will explain now why this assumption is a conservative one. Assume an environment in which requests are large and comparable in size to responds. I claim that in this case circuit-switching without the conventional set-up procedure cannot be used, because the switching nodes cannot guarantee immediate storage allocation for large requests. Since the delay of the set-up procedure equals at least twice the round-trip propagation time through a network and since the round-trip propagation time is significant in wide-area networks, the comparison of the packet-switching technique with the circuit-switching technique that includes the set-up procedure favors even more the packet-switching technique. Assume now that both requests and responds are small. The transmission of a respond from a server to a client occupies a reserved reverse sub-channel for a shorter time. However, because of the constant (and long) propagation delay, which is part of the sub-channel reservation time, the decrease in the reservation time is relatively less than the decrease in the respond's transmission time. Thus the channel utilization decreases even more, resulting in longer delays for the circuit-switching than for the packet-switching technique. Consequently, I conclude that the analysis presented here is the "worst-case" scenario for packet-switching.

In addition to a higher transactional delay, CS has other disadvantages over PS when used in high-performance networks. One such disadvantage is that some features are less flexible when integrated into a network that is implemented with the CS technique. For example, in the CS technique it is difficult to change the priority of a traffic stream or to change the multicast address list, in the middle of a session, since these operations require closing the established circuit. (Multicast is sending the same data to many destinations). Also, packet-switching offers a flexible way for bandwidth allocation ([9]). In other words, in circuit-switching there is need to allocate the required bandwidth prior to the actual information exchange. In packet-switching the bandwidth requirements can be dynamically changed and adapted to the changing requirements of an application (changing coding data rate in video compression).

Another disadvantage of the CS scheme occurs when multicasting is required. Suppose it is necessary to multicast to k destinations. If the link multiplexes N circuits in the circuit-switching method and $k > N$, some of the circuits will have to be closed and re-opened to new destinations, in order to perform the multicast operation. Thus, the delay significantly increases in such a case. In contrast, packet-switching allows a multicast packet to travel as a single packet up to the point where it has to be replicated. Thus, delay, as well as throughput, are improved in the packet-switched multicast as opposed to the circuit-switched one.

Yet another CS disadvantage comes from the fact that practical CS realizations require traffic synchronization. In PS, on the other hand, the traffic is switched asynchronously (see also Section 5.3).

The reasoning of some authors, favoring circuit-switching that supports traffic of long duration with high bit rate for the future high-speed networks, is that the switching speed will be limited by "... the time required to electronically calculate or look-up in a table the next path configuration to be realized by the switching array" [112]. However, traffic that looks like *stream* traffic may turns out to be more of the bursty type when used on a very high speed channel ([9]). (Stream traffic is traffic with average to peak data rate ratio close to unity. In bursty traffic the ratio is considerably smaller then 1. The ratio of a traffic depends on the channel characteristics and is measured on a channel that the traffic is transmitted on.) For example, traffic like packetized voice, which on a 64 kbps channel is apparently of the stream type, changes to bursty traffic on 100 Mbps link. Consequently, high-speed links in future networks will see traffic that is more of the bursty type. In Chapter 3, I presented the design of *Blazenet*, a packet-switching network that can perform packet switching on the fly, and is, therefore, not limited by the calculation or look-up procedure of a packet routing operation. *Blazenet* design shows that fast packet-switching is, in fact, possible.

As C increases the transmission time decreases and the propagation delay dominates in the value of the total delay of a packet-switched network. Thus, it is reasonable to ask what is the point beyond which any further increase in total channel capacity yields only a marginal improvement in the delay. The breaking point (defined as the value of C which results in transmission time to be equal to 10% of the propagation time) can be estimated by

$$C_{break} = \frac{10 \cdot d}{T_{prop}}. \tag{4.23}$$

Suppose 10 Mbit need to be communicated over a path of length 1000 km. Assuming 200 000 km/sec as a speed of light on optical fiber, results in $C_{break} = 20$ Gbps. Therefore, to reach the breaking point, even for such a long path, requires large capacity. However, if very large capacities will become available with the progress of the technology in the future, the argument presented in this section supports the use of a number of packet-switched networks operating concurrently, rather then a single circuit-switched network. Such a solution possesses the additional advantages of increased reliability and lower dependency on future growth of the network.

In Chapter 3, I described *Blazenet*, which is packet-switching network. *Blazenet* is not an exact implementation of the packet-switching technique as modeled in this chapter. In particular, *Blazenet* do not store the blocked packets in a queue and blocked packets are not immediately forwarded when the output link becomes free. Moreover, blocked packets are not guaranteed to be forwarded on the first-in-first-out basis. Consequently, the average packet delay of *Blazenet* is longer than the delay predicted for the packet-switching scheme by the analysis presented in this chapter. Therefore, it is interesting to know if *Blazenet* still outperforms the circuit-switching technique. The following shows that this is indeed the case.

The number of sub-loops that a single hop is composed of, has a crucial effect on the average packet delay of this hop. In particular, for some link utilization the delay decreases as the number of sub-loops increases till the number of sub-loops reaches some value, referred to here as the optimum value of sub-loops for this link utilization. Further increase in the number of sub-loops has only a minor effect on the average packet delay. Also, as a function of the link utilization, the larger the load, the larger the optimum value of sub-loops for this hop. Consequently, knowing the expected load of a link, *Blazenet* can be designed to have the average packet delay close to the delay of the packet-switching model presented in this section.

4.8 Summary

In summary, the following two factors contribute to the increased transaction delay in the CS technique:

- Dividing of the channel into smaller sub-channels, thus increasing the message transmission time. This is an important factor in high-performance networks because of the prediction that future high-performance networks will connect a large number of local-networks.

- Dedicating resources, thus lowering the channel utilization and increasing the delay. This is an important factor in high-speed networks because the transmission time shortens with increased link speed but the propagation delay remains constant.

Also, other considerations such as requirements for multicast, for priority delivery, and variable bandwidth allocation favor packet-switching.

From the comparison of the packet-switching technique with the circuit-switching technique I conclude that PS is the preferred switching technique to be used in future networks planned to provide high-performance communication.

Chapter 5

Photonic implementation of WAN

5.1 Introduction

An increased interest in photonic switching and photonic processing is apparent in the recent technical literature ([93, 94, 95, 96, 97, 98, 99]). In particular, photonic implementation of high-performance communication networks has been proposed ([100]). In such an implementation the information is entered into the network as light, is amplified and possibly regenerated, and is switched, without being converted to an electrical signals at any time. An extension of the idea of a photonic network is a totally photonic network implementation, in which the switching nodes' control consists too of optical devices (without any electrical components). Such an implementation has some salient advantages over the conventional electronic implementation. These are immunity to electro-magnetic interference, increased speed and bandwidth, higher security, lower design complexity, and increased design flexibility. However, a fully photonic implementation of a switching node is still not commercially feasible due to as-yet-unavailable optical amplifiers and optical logic devices. Moreover, optical RAMs are, and are expected to remain expensive, too expensive to be readily used as a large storage device ([43, 101, 103]). Consequently, the conventional electronic architecture of a switch needs to be replaced by an architecture suitable for photonic implementation. *Blazenet* is an appropriate solution for a photonically implementable switching node. Using the *Blazenet* design, the data path of the switching node can be fully photonically implemented with the today's state of the art.

In this chapter, I argue that future wide-area networks must be very high-speed, low delay, packet-switched, and that photonics is essential for the implementation of these wide-area networks.

This chapter is organized as follows: Section 5.2 argues for very high-performance of wide-area networks. Section 5.3 presents some of the specific problems of photonic circuit-switching, problems that do not exist in photonic packet-switching. Section 5.4 discusses the problems and difficulties associated with the electronic implementation of high-speed, packet-switching network. Finally, Section 5.5 summerizes the presented ideas.

5.2 Why high-performance networks?

The focus of this work is the design and implementation of wide-area networks to interconnect high-performance local networks and high-performance computer systems. Local networks in the 100 Mbps to 1 Gbps range and possibly higher will be available in the near future. I see local networks being driven into these performance ranges by several factors. First, realization of high-performance communication, introduced in Chapter 1.2, requires networks that can successfully cope with the special characteristics of high-performance communication. Of particular importance is the increase in the size of the data required to be conveyed over the local networks, as illustrated by the following examples: real-time color graphics simulations performed by supercomputers, documents sent with font definitions instead of being formated with the font definition at the destination, or chunks of frequently-used software that are moved between machines' main memory instead of being read from a local disc. Because of the data size and because of the requirement to keep reasonable response time, it is necessary to increase the speed of the media. Second, the increasing speed of the applications requires faster network performance in order to prevent the network from being the "bottleneck" of an operation. For example, fast file and database access on increasingly fast workstations requires high data rates to minimize the delay for accessing significant amounts of data. Third, use of clusters of machines on a local network for parallel computation and real-time control requires low-delay, high-performance communication. Finally, the fiber optic transmission and switching technology exists, and its rapid development is improving the economics of use.

If local networks are interconnected by a lower-speed wide-area network, internetwork traffic (typically of a bursty nature) must be buffered and queued in gateways to the wide-area network. Also, congestion control techniques are required in the gateways to avoid overloading the backbone network. The consequences are increased gateway costs, increased delays for the queuing and congestion control, and poorer return on investment for the local network resources.

In contrast, if the interconnecting network is of higher performance than the local networks (so that the average packet transmission time plus possible queuing delay is much smaller than the packet inter-arrival time, i.e., the network utilization is low), a packet is typically forwarded on the interconnecting network without being queued, buffered or delayed. This reduces the need for significant memory and processing power in the gateways and switching nodes. The transmission-induced delay is also reduced. Finally, assuming a common performance profile and load of the originating and receiving local networks, the rate at the receiving gateway is matched to the rate of the receiving local network. Thus the wide-area connection appears to be "nearly" transparent to communication between local networks.

In general, delays introduced by wide-area networks waste the resources of local networks and degrade the performance seen by end users. As fiber optics technology, photonic switching, and photonic processing make higher performance feasible, it can be expected that data rates on wide-area networks will be pushed as high as possible, and perhaps will be limited by the costs of host/gateway interfaces.

5.3 Photonic packet- vs. circuit-switching

In Chapter 4, I compared packet-switching with circuit-switching for high-performance communication. In this chapter, I concentrate on additional issues in this comparison, issues specific to photonic implementation.

5.3.1 Time synchronization in photonic wide-area networks

Traditionally, circuit-switching uses the Fixed-Time-Division-Multiplexing (FTDM) scheme to multiplex many traffic streams on a single channel. FTDM requires traffic synchronization in switching nodes at least at the level of slots. In this scheme the switching operation consists of the interchange of information between time slots of different traffic streams. In order for such an interchange to be possible, the time slots of all the traffic streams need to be synchronized. The synchronization can be achieved by introducing appropriate delay on the input lines that assures that all the frames arrive at the switching element at the same time. Note that there is no essential need to synchronize the traffic to the bit level, because the content (light) of a whole slot is interchanged without the necessity to know the slot's actual content.

Of course, in practical systems such fixed synchronization cannot stay permanently stable. Drift of clocks, changes in hardware operational parameters (due to aging, for example), hardware malfunction, or changes in optical length (as the result of temperature changes) drive the system out of synchronization in a short time. In today's electrical telephone system the synchronization is done on the bit level. The clock is extracted from the signal by Phase-Locked-Loop devices, which continuously correct the clock by filtering the incoming waveforms. Furthermore, the differences in clock frequencies of different streams are compensated by bit-stuffing techniques, in which a marked bit is added every so often to adjust the drift in clocks. These bit-stuffing techniques require *elastic memories*, whose size increases with the speed of the data stream.

For photonic switching systems the synchronization techniques used in electrical systems are currently impractical because of their complexity, especially at the very high-speed at which photonic systems are expected to operate ([104, 105]). (This assertion will need to be reevaluated with the progress of the photonic technology.) Also, the need for *elastic memory* makes this synchronization technique unattractive in photonic implementations. At present, the solution for FTDM synchronization in a photonically switched networks is to use a central clock and to include "guard bands" in the slots ([105]). A "guard band" is an additional, initially unused delay in the slot format (see Figure 5.1). As the slot is interchanged and travels through a

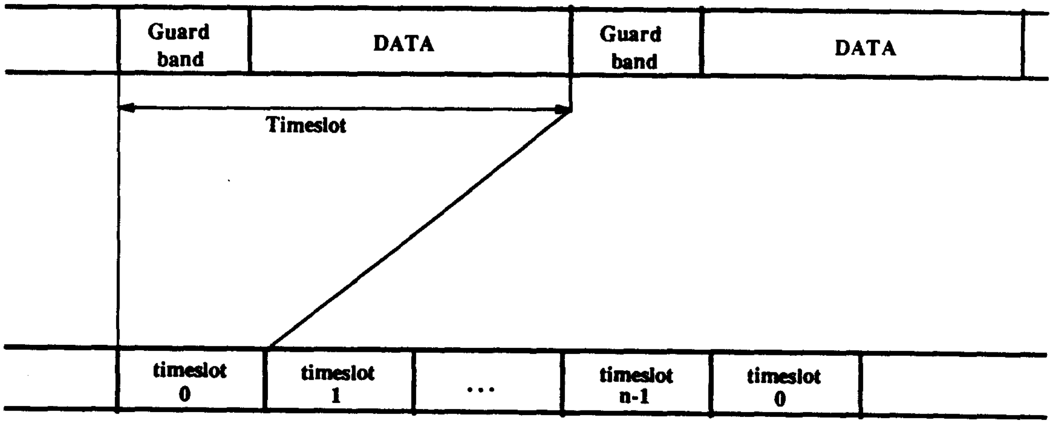


Figure 5.1: "Guard bands"

network, the data content of the slot floats within the space defined by two adjacent guard bands. In this way overlapping of any two adjacent slots is prevented. Thus, theoretically the slot synchronization can be maintained with a central (fixed) clock. The size of the guard bands is determined according to the operational parameters of the network. (As mentioned in [106], additional guard bands might also be needed to compensate for the reconfiguration time of the switching elements.) However, central clock distribution poses a serious difficulty in the network implementation. Moreover, because of the variations of the optical length with temperature, the guard-bands technique has an additional drawback when implemented in wide-area networks. This is illustrated by the following example.

Assume a network with a span of 1000 km and the coefficient of thermal linear expansion of optical fibers $1.5 \cdot 10^{-5}$ $/^{o}C$ ([107]). Assuming a maximum change of temperature of $10^{o}C$, the change in the light path amounts to as much as 150 m, which equals 750 nsec. If the traffic considered is voice and the maximum permissible delay in the packetizing is of the order of 20 msec, then the maximum packet size is 1280 bits. Assuming a link of 1 Gbps, the guard band constitutes 37% of the total slot size. With increase in the data rate the situation gets worse, since the change in the light path remains the same and the data field shrinks in time. The calculations in this example take into account only the effect of temperature changes. However, in practical systems other factors, such as material aging, might affect the synchronization even more. Packet-switching, on the other hand, does not require time synchronization between traffic streams, since the switching is done on the per-packet basis. Thus, the above problems do not exist in the packet-switching scheme.

5.3.2 Switching in a photonic wide-area network

Circuit-switching sets up a path between the source and the destination, a path that continues to exist as long as the session exists. Once a connection is set up, the switching of a slot from one link to another is accomplished by reading an appropriate entry from a look-up table. Thus the actual routing algorithm is performed only once, in the setup stage. In the packet-switching scheme the routing decision is done on a per-packet basis. Consequently, some researchers believe that for very high-speed communication the complexity of the circuit-switching routing is lower. Thus, the feasibility of the photonic implementation is increased. My belief is, however, that in gigabit networks (circuit- or packet-switched) it will be extremely difficult to perform routing decisions "on the fly." Hence, source routing is a good candidate to be used in photonic, high-speed, packet-switched networks. Comparison between the routing in source-routing, packet-switching networks and the routing in table-lookup, circuit-switching networks favors packet-switching, since, as demonstrated by *Blazenet*, the source-routed packet-switching can be implemented with no memory and with no memory-access bottleneck.

5.4 Photonic vs. electronic packet-switching

Photonic switching is important for very high-performance networks because of some major disadvantages and difficulties associated with the electronic design of a switch and with its interfacing to optical fibers. The steps that are involved in switching of a packet are:

- Packet synchronization.
- Reading of the header information.
- Switching decision.
- The actual switching operation.

In the totally photonic network these operations are performed in the following manner: Packet synchronization is done by an optical correlator ([33, 108]). The header information is read and the switching decision is made by optical logic ([109, 110, 99]). Since source routing is assumed, the switching decision is a simple operation. The switching operation is performed by optically driving a photonic switching element. Also, somewhat related to the above operations is the need to regenerate the signal. In photonics, signals are regenerated by some sort of a threshold amplifier ([106, 110]). (It is assumed that the cost of such a threshold amplifier is comparable to the cost of a laser modulator.)

5.4.1 Design complexity

The use of a serial-to-parallel shift register as the first stage of an electronic design can decrease the requirements of memory speed by factor equal to the length of the shift register. In other words, the claim is that if a memory of the required speed is not available, then by buffering the information into words and processing words rather then bits, the speed requirements can be matched. To examine this claim, I concentrate on a simple switching node design, a design that performs only the forwarding operation of packets that arrive on one of its 10 input links and that are destined to one of the 10 output links. Links are assumed to operate at 10 Gbps. Source routing is used as the addressing scheme. Thus the output link number is embedded in the packet header. Consider the design in Figure 5.2. Each incoming stream of bits is fed into an input shift register that is x-bits long. Then the x-bit words are transferred into an input FIFO, each transfer occurring within a single clock time duration. A packet is, therefore, stored as consecutive words in the FIFO. The switching element connects the input FIFOs to the output FIFOs, an operation

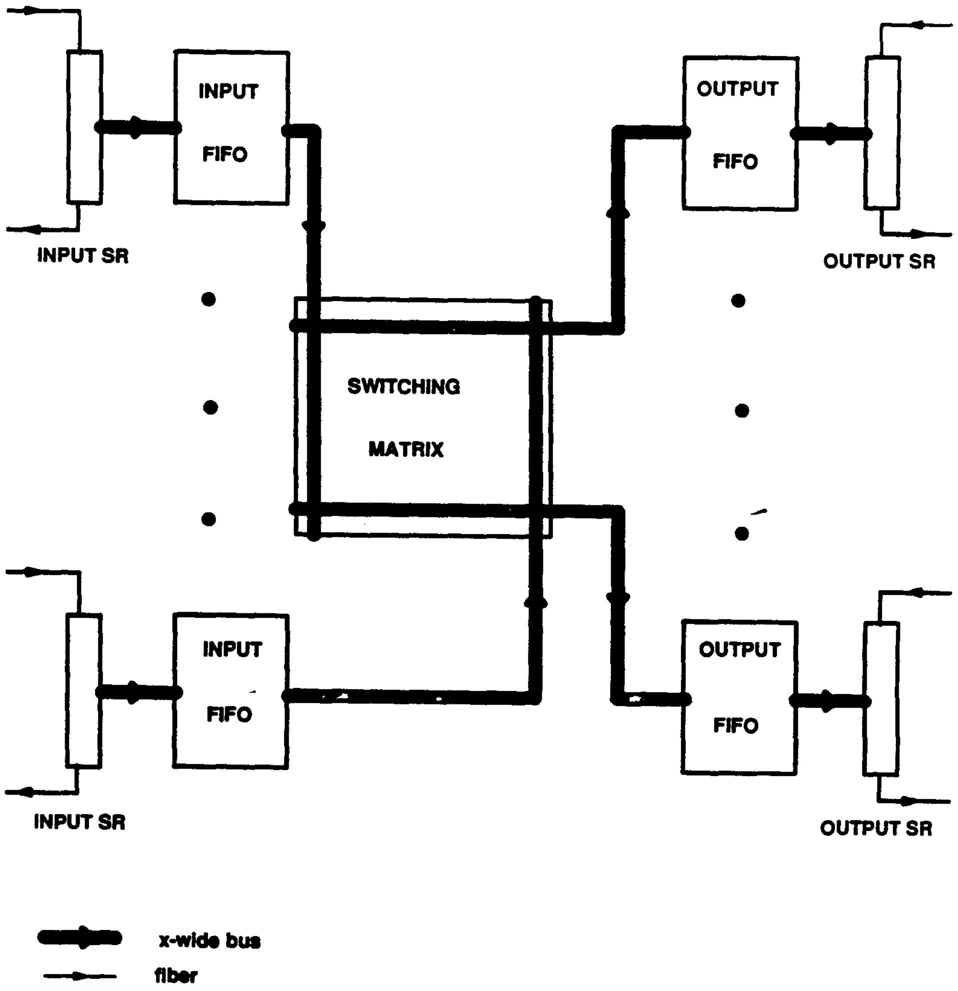


Figure 5.2: Simple switching node design.

that is controlled by the address field in the packet header. Words from an output FIFO are transferred to an output shift register connected to the FIFO, are parallel-to-serial converted, and are transmitted on an output link at the high speed. The input FIFOs allow for the time to interpret the packet header and for the time to perform the actual switching, and provide storage for packets in contention. The output FIFOs provide buffering for the output links queues.

This simple design enables us to evaluate the complexity involved in designing the operation of serial-to-parallel conversion, an operation required to increase the switch (memory) bandwidth. I shall show that x cannot take arbitrarily large values, and thus indefinitely increase the switch bandwidth. The switching element can be implemented as a simple switching matrix and will consist in this case of $n \cdot n \cdot x$ switches, where n is the number of inputs to the switching element (10 in the example). A smaller number of switches can be achieved by choosing some restrictive architecture for the switch ([65, 66]). I assume here that the switch size is $O(n \cdot \log n)$. Thus in this example the switching element consists of $33 \cdot x$ switches. These switches operate at the speed of $10/x$ Gbps. Since the shift registers operate at 10 Gbps, the connections between adjacent bits in a shift register cannot be longer than 1.5 mm, which equals approximately one tenth of the 10 GHz wavelength. Consequently, a shift register must be implemented on a single IC and, because of the maximum IC's pin count, $x \approx 100$ is a reasonable limit ([111]). Suppose x=100. Then as many as 3300 switches operating at a speed on the order of 100 Mbps are needed. These 3300 switches must be connected to the FIFOs with wires not longer than 15 cm, which equals one tenth of the 100 MHz wavelength. (It is assumed that level and not pulsed logic is used.) A signal of 100 MHz requires a transmission line, possibly a coaxial cable. A design of 3300 transmission lines connecting closely spaced 100-bit-wide FIFOs is of unrealistic complexity. More reasonable is the case $x \approx 10$. In this case the number of switches required is about 330. These need to operate at a speed on the order of 1 Gbps. Thus, a reduction in the operation speed on order of 10 seems to be a reasonable limit. Note that the solution of integrating the whole structure on a single wafer is not realistic due to the high power requirements of the whole configuration (see also Section 5.4.3).

5.4.2 Speed of operation

Optical processing (such as correlation) can be performed at a much higher rate than the corresponding electronic processing. Rates in excess of 100 GHz are possible ([33, 108, 32]). These higher rates are especially important, because some processing is needed on the data before the serial-to-parallel conversion is performed. For example, the recognition of the synchronization field, as an indication of the beginning of a packet, needs to be performed at the fiber transmission rate. Thus, by employing optical techniques increased signaling speed can be achieved.

Optical switching systems, when used in the continuous mode of operation, cannot provide switching speeds that are orders of magnitude faster than those of electronic systems ([112]). However, considerable increase in switching speed can be achieved for intermittent operation ([113]). The intermittent operation region is applicable for packet-switching networks, because the large packet size expected leads to long time between consecutive activation of the switching element.

5.4.3 Switch implementation issues

Several issues arise in the implementation of the switch.

Signal conduction, interference, and isolation

An electrical signal conducted over distances on the order of the wavelength of the signal needs a transmission line to be effectively propagated. A design using transmission lines as wires is expensive and occupies considerable space, especially when the signal is parallel processed. An optical signal, on the other hand, is conducted by a low-volume fiber. Several signals can be wavelength multiplexed on a single fiber, and thus space requirements are further reduced. Also, the loss of the optical signal in the propagation process in fiber is lower than the corresponding loss of the electrical signal propagated in a transmission line. Consequently, the signal can be propagated over longer distances.

Photons do not easily interact with photons. The electrical interference and signal isolation problems plaguing electrical systems are not present in optical systems. Thus signals are more protected in optical design, and the task of designing interconnections within a circuit is more predictable. Also, the greater immunity to EMP of a totally photonic design leads to a network with an increased level of survivability.

Data synchronization

The serial-to-parallel conversion, which is used in electronic design for rate reduction, requires close synchronization of the parallel-propagated data. In other words, the difference in propagation speed of the different bits in a word needs to be negligible compared to the bit length. At 1 Gbps the length of a bit is approximately 20 cm. Thus the maximal difference in electrical length between any two lines of the bus needs to be smaller than 2 cm. This issue, similar to the "clock skew" problem, poses a difficult problem in the design of connections within the switch. (Also, the bus may be composed of separate coaxial cables requiring coaxial connectors, a situation that further complicates the data synchronization problem.) Photonic designs usually do not employ parallel data propagation, thus the data synchronization problem does not exist.

Memory requirements

A design that uses high-speed electronic memories of the size required in a switching node in wide-area networks results in prohibitive power requirements. If useful flow control on the data link layer is required, then the storage capacity per input link needed in a switching node is at least of the size of the information that can be in transit on a link during a round trip delay. For example, a 10 Gbps transmission rate of a 100 km link results in 10 Mbit of memory per input link.

The Table 5.1 summarizes the power requirements of a 10 Mbit memory. The table was compiled by referring to the graph of access time vs. power dissipation for SRAMs (for various technologies) from [114].

Technology	*Chip size*	*Access time*	*Power/Bit*	*Power/10 Mbit*
HBT (0.2 μm)	1 Kbit	100 ps	1.5 mW	15,000 W
Si-Bipolar (0.2 μm)	1 Kbit	200 ps	1 mW	10,000 W
Si-Bipolar (0.2 μm)	4 Kbit	400 ps	0.5 mW	5,000 W
HFET (0.2 μm)	16 Kbit	200 ps	0.2 mW	2,000 W
GaAs (0.2 μm)	16 Kbit	300 ps	0.2 mW	2,000 W
GaAs (0.5 μm)	16 Kbit	1.5 ns	0.1 mW	1,000 W
GaAs (0.2 μm)	64 Kbit	600 ps	0.04 mW	400 W

Table 5.1: Access time vs. power dissipation for various SRAM technologies

Using the GaAs (0.2 μm) technology that yields a 1.67 Gbps clock, requires as much as 400 W or 200 A current at 2 V power supply ! Also the low-integration scale of these high-speed devices leads to about 156 ICs for the 10 Mbit memory in the

GaAs (0.2 μm)-64 Kbit-per-IC technology !

Large random-access, photonic memory is difficult to implement ([43, 115, 101]). However, some network architectures, such as *Blazenet*, avoid using large memory banks and are thus particularly attractive for photonic implementation.

5.4.4 Additional issues

Security

A photonically implemented switch is relatively immune to monitoring and jamming in comparison to an electronic switch. In addition, photonic components including fiber are more difficult to tamper with, without the tampering being detected. Thus a photonic implementation is also attractive in providing some security guarantees for the network.

Interfacing with electronics

Since the data rate in photonic network is very high-speed, appropriate interfacing methods with the slower electronic world need to be design. One such a possibility is the use of optical multiplexing/demultiplexing ([113]), allowing slow signals to be time-division multiplexed on high-speed optical media. On the receiving part, the high-speed optical signals are demultiplexed into slower data rate, a data rate suitable for optical detectors and electronic devices.

Computer boards and devices, and possibly even gates on an IC of the future, may well be connected by means of optical fibers ([96, 116]). With the growing popularity of optical connections, a photonic network will be able to interface directly with computer optical buses, eliminating the need for optics/electronics conversion. Devices (as Analog-to-Digital-Converter in [117]) that produce as their output high-rate optical data are another instance of possible direct interfacing to a photonic network. In addition, fully optical computers have been proposed ([95], for example). In these cases, direct connection of the photonic network to the optical source of data can be of great advantage in terms of increased bandwidth, design simplicity, and cost.

5.5 Concluding remarks

In this chapter, I argue that photonics is not just an attractive technology for achieving the performance levels offered by fiber optics, but is a necessary direction to pursue, given the limits of performance and the difficulties in the implementation of high-speed electronics. These limits and difficulties are avoided by using photonic switching and photonic processing.

I have focused here on three key issues:

- How important is high-performance in the range of multi-gigabits to future wide-area networks?
- What are the advantages of packet-switching over circuit-switching for photonic implementation?
- How important is a photonic implementation?

In particular, I have demonstrated some limits of the performance of high-speed electronics, and some difficulties associated with high-speed electronic implementation. These limits and difficulties can be overcome with the use of photonic switching and photonic processing, and with the introduction of new network architectures. One such architecture, *Blazenet*, overcomes the main limitation of a photonic design: lack of large optical memories.

In general, I see *Blazenet* as a possible candidate for the next generation of wide-area networks. Moreover, I argue that any competing network design must also provide very high-performance packet-switching that is amenable to fully photonic implementation. Hopefully, this focus on a smaller design space will bring about more concentrated research and development efforts, leading to a successful realization of this class of networks.

Chapter 6

Summary and conclusions

The purpose of this work is to investigate the design of high-performance wide-area networks. First, by introducing *Blazenet*, the feasibility of packet-switching in high-speed networks was demonstrated. In other words, the *Blazenet* design shows that it is not necessary to resort to circuit-switching to handle the data rates made possible by optical fiber. The three key ideas behind *Blazenet*'s design are: source routing, packet loopback on blockage, and photonic implementation made possible by the first two. Loops that constitute *Blazenet*'s links provide the temporary storage for blocked packets in transit; thus the storage inherently present in the optical fiber links is used. In general, the *Blazenet* switching node is a simple, universal, high-performance component suitable for photonic implementation. Specifically, *Blazenet* provides high-speed switching of multi-gigabit per second data rates, low delay, flow control by the back-pressure mechanism, and good behavior under load, with support for multicast, for priority traffic, and for real-time traffic—features that appear essential for the next generation of communication applications.

Second, I showed that under a set of reasonable assumptions and based on the transactional model of communication, the delay of a transaction in the packet-switching scheme has lower delay than the corresponding delay in the circuit-switching case. In particular, as the bandwidth increases and the path length decreases, the delay of a transaction in circuit-switching model increases more than the corresponding delay in the packet-switching model. The results show that even for large response size on the order of 1 Mbit sent over 1 Gbps channel, small request processing time on the order of 10 msec, large path of 20 hops, and the conservative assumptions favoring circuit-switching, the packet-switching technique has lower delay. Moreover, for reasonable parameter range the delay overhead of circuit-switching is hundreds of percents greater than of packet-switching. For example, for 1 Mbit response sent over 1 Gbps channel, request processing time of 100 msec, path of 10 hops, channel utilization of 25%, and the conservative assumptions favoring circuit-switching the delay of a transaction in circuit-switching technique is nearly three times longer than the corresponding delay of the packet-switching technique. Also, in packet-switching in combination with source routing, the routing-per-packet overhead is comparable with the routing-per-session overhead of the circuit-switching scheme. Moreover, other considerations such as requirements for multicast, for priority delivery, and variable bandwidth allocation favor packet-switching.

Third, packet-switching is more suitable for photonic implementation in wide-area networking than circuit-switching, because the latter presents some major obstacles, such as need for traffic synchronization. The all-photonic design of a communication network is an attractive alternative to the conventional electronic design, because of the higher processing speed, increased immunity to electro-magnetic interference, higher security, and possibly lower cost of the photonic design. Some of the limitations associated with high-speed electronic switching are: design complexity, data synchronization, and memory power requirements. However, the present lack of large photonic memory banks may impose difficulty in the implementation of conventional packet-switching architecture. Consequently, the lack of conventional memory requirement in the *Blazenet* switching node design increases the feasibility of the network's photonic implementation.

Blazenet was presented as a photonically implementable backbone wide-area network. However, with some constraints on the packet size, the *Blazenet* concept can be extended to metropolitan or even local-area networks. *Blazenet* can be seen also as a large distributed switch, whose inputs are the network's entrances. Adopting this view, the collection of local-area networks connected by *Blazenet* resembles in performance characteristics the local environment in which users are connected by a local-area network. These characteristics are: data rates of hundreds of Mbps, end-to-end delay of tens of msec, and error rates on the order of 10^{-9}. And since *Blazenet* is designed to provide high-performance communication, the motivation for this work: to provide LAN-grade of performance in WAN, can be realized.

Considerations for the future

The time is ripe for the development of a new generation of high-speed wide-area networks for computer communication. The characteristics of this new generation of networks will strongly determine the economics and functionality of wide-area distributed systems of the future. The cost of these networks suggests that design choices made now will have implications for years, if not decades, to come. It is not an overstatement to say that these networks will be a key component of a nation's economic infrastructure and a strong determinant in future international competitiveness, both commercially and economically.

I argue that a variety of factors call for data rates well beyond that justified by basic capacity utilization arguments. In particular, very high data rates increase the network capacity, reduce the delay, reduce the gateway connection cost, and reduce the response times for local network and computing resources. *Blazenet* is an example of a network design that can provide these performances. Network such as *Blazenet*, can revolutionize the opportunities for distributed command and control, information and resources sharing, real-time conferencing, and wide-area parallel computation, to mention but a few applications. I see *Blazenet* as a representative of a future class of networks that behave as passive "light pipes" for data, offering high throughput, low delay, and high reliability.

However, further research and development are required to make this perception of the future a reality. First, with the introduction of this class of wide-area networks, I expect that the computer interfaces, rather than the networks, will be the performance and functionality bottlenecks of the communication process. Consequently, work has to be done in developing high-performance host interfaces and gateways.

Second, work is needed in order to estimate the traffic mixture that future WANs will carry. The actual traffic mixture has a crucial impact on the design parameters and the performance of any proposed design and traffic integration is an important issue that has to be successfully supported by the high-performance communication network.

Third, once the predicted traffic mixture is known, further comparison between the various designs of high-performance networks (that can be done by means of a simulation) are needed to evaluate their merits with respect to delay, capacity, cost, design complexity, reliability, and miscellaneous features as multicast, priority traffic, and real-time traffic.

Finally, actual implementation and experimentation with networks like *Blazenet* in a real environment is required to gain deeper understanding of the issues involved in high-performance wide-area networking.

Although there is considerable research to be done, as described here, I regard this work as setting a new directions for WAN design and a first step to revolutionizing computer communication exploiting the enormous potential offered by optical fibers.

Chapter 7

Appendices

7.1 Appendix A: Derivation of formula for reassembly delay

Our model is presented in Figure 7.1. We start from the formula for reassembly time cited in [36] and also derived in [42]:

$$n_m = n_{m-1} + (n_{m-1} + 1) \cdot \lambda_p^* - m_{k-1} \cdot q_m \cdot P_m \qquad k \geq 0$$

$$n_0 = 0\,, \tag{7.1}$$

where n_m is the average length of the inter-packets gap at the m^{th} node on the message path, λ_p^* is the average number of packets that arrive on other input links to the output link that the message is forwarded during the transmission of a packet of the message, q_m is the fraction of total input traffic to the node that leaves on other output links, and P_m is the probability that the output link of the m^{th} node on the message path is busy.

I assume a totally symmetric network with equal utilization for all links, ρ. Consequently,

$$\lambda_p^* = \frac{n-1}{n} \cdot \rho; \qquad q_m = \frac{n-1}{n}; \qquad P_m = \rho\ . \tag{7.2}$$

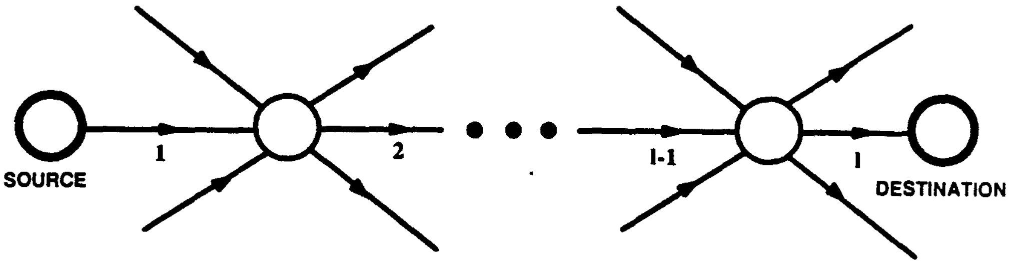


Figure 7.1: Message path model for reassembly time derivation

In order to abolish the dependence on the number of input links, n, it is assumed that $n \rightarrow \infty$. This assumption increases the inter-packets gap estimation. Therefore,

$$n_m = n_{m-1} + \rho \ . \tag{7.3}$$

The message is, thus, delayed by a $k-1$ inter-packets gap, where k is the number of packets the message is divided into. Consequently, the total reassembly delay of a message, $D_{reassembly}$, is

$$D_{reassembly} = (k-1) \cdot n_{l-1} \cdot T_p + k \cdot T_p = k \cdot (n_{l-1}+1) \cdot T_p - n_{l-1} \cdot T_p \text{ [sec]} , \tag{7.4}$$

where T_p [sec] is the transmission time of a single packet and l is the number of links on the message path. In order to abolish the dependence of the reassembly time on the packet size, I assume that $k \gg 1$. Consequently,

$$D_{reassembly} = l \cdot \rho \cdot T_m \text{ [sec]} , \tag{7.5}$$

where T_m [sec] is the total message transmission time. Thus the inter-packets gaps increase the effective transmission time of a message by a factor of $l \cdot \rho$.

7.2 Appendix B: Message- and packet-switching with fixed message size

In Appendix 7.1 the formula for the inter-packets gap was developed. The derivation assumed that there is no correlation between the packets of different messages. This

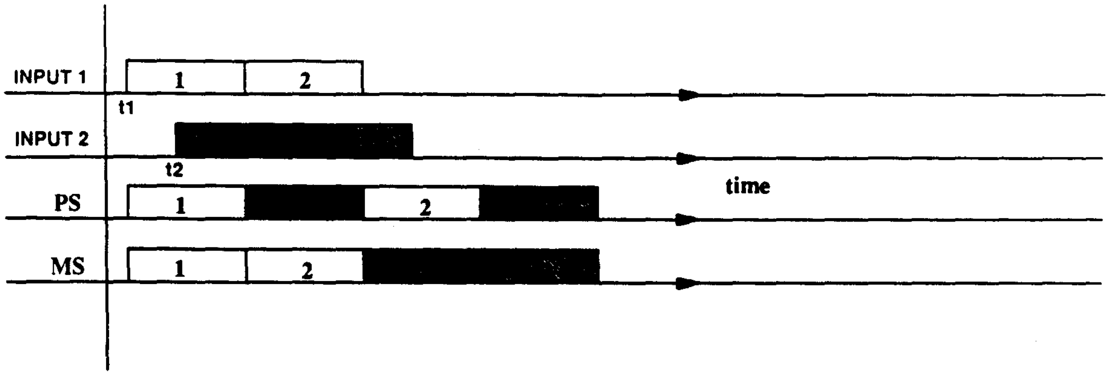


Figure 7.2: PS vs. MS: 2 messages contending on an output link

assumption is a reasonable one for messages that have traveled over a large number of hops within the network. In such cases the inter-packets gaps become large enough to make the consecutive packets of any two overlapping messages on a link only weakly correlated. However, near the network entries the assumption of a weak correlation between two overlapping messages is incorrect. In this Appendix, I show that at the network entrances, because of this correlation, message-switching has always shorter delay than packet-switching (or an equal delay, in the case of a single input line), when the messages are of equal length and when *general-cut-through* switching mode is used.

To prove the above claim, I concentrate on a single switching node with m messages contending for an output link and calculate the delay associated with the switching process, averaged over all the messages involved. It is assumed that the packet header is a negligible fraction of the total packet size.

Obviously, the claim is true for a single message (m=1), since in this case no contention occurs and the delay of the message under both schemes is equal.

Now let us examine the case of two messages (m=2) divided into two packets each and contending on the same output link. Assume the messages are of equal length with transmission time equal to one unit. Suppose the first packet of message 1 arrives at time t_1 and the first packet of message 2 at t_2, as shown in Figure 7.2. The arrival of the second packet of each message immediately follows the first one. The first packet of message 1 is immediately forwarded on the output link. It is followed

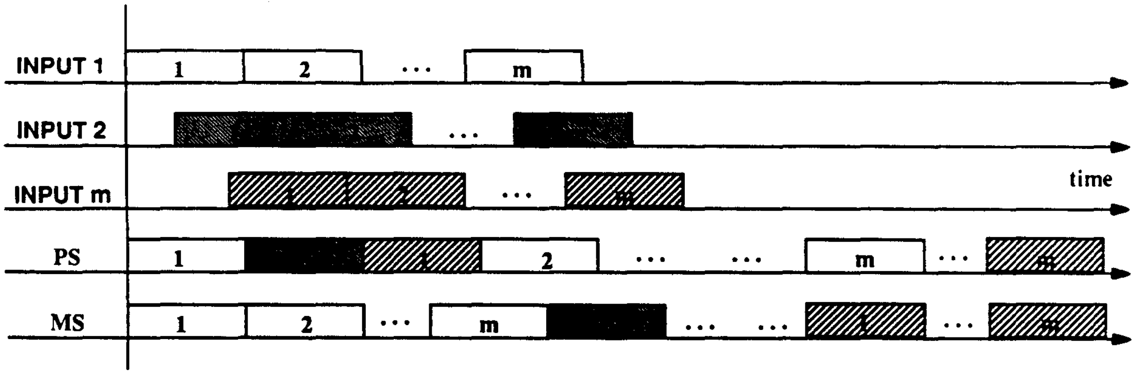


Figure 7.3: PS vs. MS: m messages contending on an output link

by the first packet of message 2, which is followed by the second packet of message 1, which finally, is followed by the second packet of message 2. Message 1 is delayed by $\frac{1}{2}$ unit and message 2 is delayed by $\frac{1}{2} + t_2 - t_1$ units. Thus the average delay is $\frac{1}{2} + \frac{1}{2} \cdot (t_2 - t_1)$ units. Note that the delay is measured according to how much the last bit of a message is delayed.

Consider now the message-switching case. Message 1 is immediately forwarded upon its arrival, thus encountering no delay. Message 2 is delayed by $\frac{1}{2} + t_2 - t_1$ units. Thus the average delay is equal to $\frac{1}{2} \cdot (t_2 - t_1)$ and therefore, the packet-switching delay is longer than the message-switching delay, in this case.

Consider now the general case of m messages. The messages are of equal length, composed of k packets each, and arriving at the same output link, as described in Figure 7.3. The arrival times of the messages are assigned values t_1 to t_m. Without loss of generality, assume that $t_1 = 0$. Then the delays encountered by the various messages in the packet-switching scheme are:

Message	Delay
1	$\frac{m(k-1)}{k} - 1 + \frac{1}{k}$
2	$\frac{m(k-1)}{k} - 1 + \frac{2}{k} - t_2$
3	$\frac{m(k-1)}{k} - 1 + \frac{3}{k} - t_3$
.	
.	
.	
m	$\frac{m(k-1)}{k} - 1 + \frac{m}{k} - t_m$.

The average delay is, therefore:

$$Delay\{PS\} = \frac{m \cdot (k-1)}{k} - 1 + \frac{(1+m)}{2k} - \frac{1}{m} \cdot \sum_{i=1}^{m} t_i \text{ [units]} . \tag{7.6}$$

For the message-switching arrangement the following are the delays of the various messages:

Message	Delay [sec]
1	0
2	$1 - t_2$
3	$2 - t_3$
.	
.	
.	
m	$(m-1) - t_m$,

which averages to

$$Delay\{MS\} = \frac{(m-1)}{2} - \frac{1}{m} \cdot \sum_{i=1}^{m} t_i \text{ [units]} . \tag{7.7}$$

It is easily verified that $Delay\{PS\} \geq Delay\{MS\}$, with equality only for $m = 1$ (and $k = \infty$). Thus, I conclude that message-switching has lower delay than packet-switching when *general-cut-through* switching is used, provided all the messages (and all the packets) are of equal size.

The above proof assumes that the first packet of the next arriving message arrives at the switching node before the transmission of the first packet of the previous message is completed (that is, all the first packets are transmitted first, then the second packets, etc). If this is not the case, the message-switching delay remains the same and the packet-switching delay is decreased. (The limiting case is when the first packet arrives during the transmission of the last packet of the previous message; in which case the messages are transmitted exactly as in the message-switching case.) However, if the above assumption is not true, it is easy to show that the packet-switching delay is still longer than the message-switching delay (with equality in the limiting case described above).

The above claim is, obviously, not valid when messages are of different sizes. As an example, consider a very long message arriving before a single-packet-size message. Obviously, letting the short message interleave the transmission of the long message results in considerable decrease in the short message's delay, and has only a minor effect on the long message's delay. However, the above claim is valid in a *transactional* environment with *responses* of equal size (like a data base of images, for example) and where a separate channel is provided for the *requests*, which are assumed to be short. Consequently, message-switching used in *general-cut-through* switching is advantageous in such a situation.

The conclusion from the analysis presented in this Appendix is that the reassembly delay time is, in general, longer than the reassembly delay developed in Appendix 7.1. Consequently, the packet-switching delay, as opposed to the message-switching delay, is larger than can be predicted by the analysis in Appendix 7.1.

7.3 Appendix C: Derivation of the composite holding time

The formula for average holding time of a link at a switching node is calculated here by use of the residual path length.

The holding time at a particular switching node is composed of:

- Twice the propagation delay from the node to the destination.
- Contention time (for a free sub-channel) for each of the remaining hops on the packet path, $W_{M/G/N}$ [sec].
- Request processing time, α [sec].
- Response transmission time, d/C [sec].

The composite holding time, $\tilde{h}$ [sec], is, therefore, given by the equation

$$\tilde{h} = 2\tilde{l}_r \cdot t_{prop} + (\tilde{l}_r - 1) \cdot W_{M/G/N} + \alpha + d/C \text{ [sec]}, \tag{7.8}$$

where t_{prop} [sec] is the propagation delay of a single hop (assuming all hops are equal) and $\tilde{l}_r$ [hops] is the remaining number of hops on a packet's path.

In order to find the average composite holding time, $\overline{h}$, the $\overline{l_r}$ need to be calculated. This is done by use of the residual life argument of [88], first by finding the probability function of l_r, which is given by

$$Pr[l_r = n] = \frac{1}{\bar{l}} \cdot \sum_{i \geq n} Pr[l = k], \tag{7.9}$$

where $\bar{l}$ is the average path length within the network.

Then, the mean of $\overline{l_r}$ is readily found by averaging over all possible values

$$\overline{l_r} = \sum_{n \geq 1} \{n \cdot Pr[l_r = n]\} = \frac{1}{\bar{l}} \sum_{n \geq 1} \{n \cdot \sum_{k \geq n} Pr[l = k]\} \text{ [hops]} . \tag{7.10}$$

And, after changing the order of summation the following result is easily obtained:

$$\overline{l_r} = \frac{1}{2\bar{l}}(\overline{l^2} + \bar{l}) \text{ [hops]}, \tag{7.11}$$

which, when substituted into the averaged form of 7.8, results in

$$\overline{h} = \frac{\overline{l^2} + \bar{l}}{\bar{l}} \cdot t_{prop} + \frac{\overline{l^2} - \bar{l}}{2\bar{l}} \cdot W_{M/G/N} + \alpha + d/C \text{ [sec]} . \tag{7.12}$$

In the simple case of a constant path length, $\bar{l} = l$, and 7.3 simplifies to

$$\overline{h} = (l + 1) \cdot t_{prop} + \frac{l - 1}{2} \cdot W_{M/G/N} + \alpha \text{ [sec]} . \tag{7.13}$$

7.4 Appendix D: Distribution of interdeparture times from an M/M/1 queue, operating in the *general-cut-through* mode

A somewhat surprising result is the density function of interdeparture times of the output of a queue working in *cut-through* mode. Burke's well known theorem ([118]) states that the steady state output of a stable M/M/m queue is a Poisson process with parameter equal to the parameter of the arrival Poisson process. Of course, the theorem considers departures as the end of service instances. In analysis of the *general-cut-through* mode of operation, however, one needs to consider departure as the instances of the beginning of a service.

One would think that such a "shift in time" is immaterial as far as the distribution of interdeparture times is considered. The fact is that the interdeparture times are not exponentially distributed random variables (even though being close to exponential distribution). An exact analysis follows.

Theorem: *The probability density function of interdeparture times, as seen by the first bit, from an M/M/1 queue is given by*

$$f_T(t) \equiv Pr(T = t) = \mu \cdot e^{-\mu t} + (1 - \rho^2) \cdot (\lambda - (\lambda + \mu) \cdot e^{-\mu t}) \cdot e^{-\lambda t} . \qquad (7.14)$$

Proof:

From the theorem of total probability it follows that

$Pr(T = t) =$

$Pr(T = t \mid$ queue is free at last departure$)\cdot$

$Pr($queue free at last departure$)+$

$Pr(T = t \mid$ queue is busy at last departure$)\cdot$

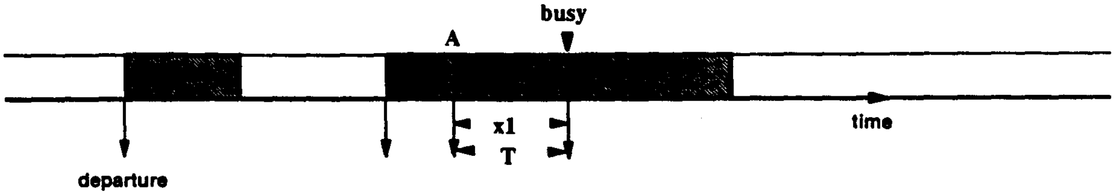


Figure 7.4: Calculation of $Pr(T = t \ \mid$ queue is busy at last departure)

$$Pr(\text{queue busy at last departure})\,, \tag{7.15}$$

$$Pr(\text{queue free at last departure}) = 1 - \rho\,, \tag{7.16}$$

and

$$Pr(\text{queue busy at last departure}) = \rho\;. \tag{7.17}$$

Now concentrate on finding the $Pr(T = t \mid$ queue is busy at last departure) and $Pr(T = t \mid$ queue is free at last departure).

Calculation of $Pr(T = t \ \mid$ queue is busy at last departure)

Figure 7.4 refers to the following derivation:

Given that the queue is busy at last departure, it is obvious that $T = x_1$ and therefore

$$Pr(T = t \mid \text{queue is busy at last departure}) =$$

$$Pr(x_1 = t \mid \text{queue is busy at last departure}).$$

Using Bayes Theorem results in

$$Pr(x_1 = t \mid \text{queue is busy at last departure}) =$$

$$\frac{Pr(\text{busy} \mid \ x_1 = t) \cdot Pr(x_1 = t)}{Pr(\text{busy})}\,, \tag{7.18}$$

where "busy" means "queue is busy at last departure."

Since packet length is exponentially distributed,

$$Pr(x_1 = t) = \mu \cdot e^{-\mu t} \, . \tag{7.19}$$

To find $Pr(\text{busy} \mid x_1 = t)$ proceed as follows:

$$Pr(\text{busy} \mid x_1 = t) = Pr(\text{more then one in queue at A}) +$$

$$Pr(\text{0 or 1 in queue at A and at least one arrival during } t) \, . \tag{7.20}$$

Since the queue is M/M/1,

$$Pr(\text{k in queue}) = (1 - \rho) \cdot \rho^k \, . \tag{7.21}$$

And so,

$$Pr(\text{0 in queue}) = (1 - \rho) \, , \tag{7.22}$$

$$Pr(\text{1 in queue}) = (1 - \rho) \cdot \rho \, , \tag{7.23}$$

and

$$Pr(\text{more than 1 in queue}) =$$

$$1 - Pr(\text{0 in queue}) - Pr(\text{1 in queue}) = 1 - (1 - \rho) - \rho \cdot (1 - \rho) = \rho^2 \, . \tag{7.24}$$

Also

$$Pr(\text{at least one arrival during } t) = 1 - Pr(\text{no arrivals during } t)$$

$$= 1 - e^{-\lambda t} \, . \tag{7.25}$$

Therefore, substituting equations (7.22) – (7.25) into equation (7.20) and using the independence between the arrival process and the state of the queue, I obtain the following equation:

$$Pr(\text{busy} \mid x_1 = t) = \rho^2 + (1 - \rho^2) \cdot (1 - e^{-\lambda t}) = 1 - (1 - \rho^2) \cdot e^{-\lambda t} \, . \tag{7.26}$$

Substituting equations (7.17), (7.19), and (7.26) into equation 7.18 results in

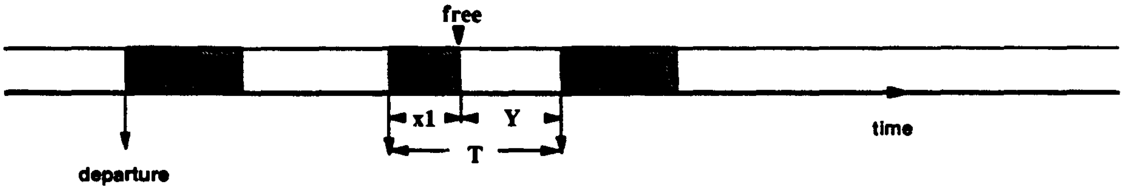


Figure 7.5: Calculation of $Pr(T = t \mid \text{queue is free at last departure})$

$$Pr(x_1 = t \mid \text{busy}) = \frac{[1-(1-\rho^2)\cdot e^{-\lambda t}]\cdot \mu \cdot e^{-\mu t}}{\rho} . \tag{7.27}$$

I concentrate now on finding $Pr(T = t \mid \text{queue is free at last departure})$.

Calculation of $Pr(T = t \mid$ queue is free at last departure)

The following derivation refers to Figure 7.5.

Removing the conditioning on length of the just finished packet $(x = \tau)$ by using the independence between the arrival process and state of the queue, results in:

$$Pr(T = t \mid \text{free}) = \int_0^t Pr(T = t \mid x = \tau, \text{ free}) \cdot Pr(x = \tau \mid \text{free}) \cdot d\tau =$$

$$\int_0^t Pr(Y = t - \tau \mid \text{free}) \cdot Pr(x = \tau \mid \text{free}) \cdot d\tau =$$

$$\int_0^t Pr(Y = t - \tau) \cdot Pr(x = \tau \mid \text{free}) \cdot d\tau . \tag{7.28}$$

Since the unconditioned arrival process is Poisson then

$$Pr(Y = t - \tau) = \lambda e^{-\lambda(t-\tau)} \tag{7.29}$$

and

$$Pr(x = \tau \mid \text{free}) = \frac{Pr(\text{free} \mid x = \tau)\cdot Pr(x = \tau)}{Pr(\text{free})} =$$

$$\frac{[1 - Pr(\text{busy} \mid x = \tau)]\cdot Pr(x = \tau)}{Pr(\text{free})} =$$

$$\frac{(1-\rho^2)\cdot e^{-\lambda\tau}\cdot \mu \cdot e^{-\mu\tau}}{(1-\rho)} = (1+\rho)\cdot \mu \cdot e^{-(\mu+\lambda)\tau} . \tag{7.30}$$

Substituting (7.29) and (7.30) into (7.28) and performing the integration results in

$$Pr(T = t \mid \text{free}) = \int_0^t \mu \cdot (1+\rho) \cdot e^{-(\mu+\lambda)\tau} \cdot \lambda \cdot e^{-\lambda(t-\tau)} \cdot d\tau =$$

$$\lambda \cdot (1+\rho) \cdot \mu \cdot \int_0^t e^{-\lambda t} \cdot e^{-\mu\tau} \cdot d\tau = \lambda \cdot (1+\rho) \cdot e^{-\lambda t} \cdot (1 - e^{-\mu t}) \,. \qquad (7.31)$$

In summary, we obtained that

$$Pr(x_1 = t \mid \text{busy}) = \frac{[1 - (1-\rho^2) \cdot e^{-\lambda t}] \cdot \mu \cdot e^{-\mu t}}{\rho} \qquad (7.32)$$

and

$$Pr(T = t \mid \text{free}) = \lambda \cdot (1+\rho) \cdot e^{-\lambda t} \cdot (1 - e^{-\mu t}) \,. \qquad (7.33)$$

Inserting equations (7.31) and (7.32), as well as equations (7.16) and (7.17), into equation (7.15) results in

$$Pr(T = t) = \mu \cdot e^{-\mu t} + (1-\rho^2) \cdot (\lambda - (\lambda+\mu) \cdot e^{-\mu t}) \cdot e^{-\lambda t} \,. \qquad (7.34)$$

Q.E.D

Another way to prove the theorem is to make use of the theorem of total transform. The reasoning follows. Concentrate on departure points. Assign the variable q to the number of packets in a queue just before the departure instances (see Figure 7.6). Then $q = 0$ means the queue was empty before the current departure and $q = 1$ means there was one packet left when the current departure took place.

If $q \geq 2$ then the following interdeparture interval is of exponential distribution with parameter μ.

If $q = 0$ or $q = 1$ then the actual distribution of the following interdeparture interval depends on the number of arrivals during the current service period. If there were no arrivals during the service time period then the service time is exponentially distributed with parameter $(\lambda + \mu)$, i.e.,

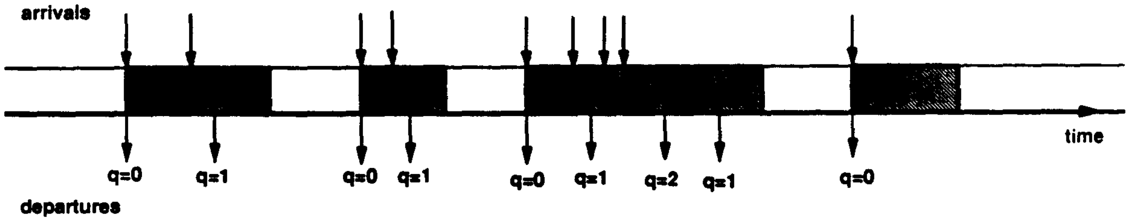


Figure 7.6: Definition of the variable q

$$Pr(x = t \mid \text{no arrivals during service period}) =$$

$$\frac{Pr(\text{no arrivals during a service period} \mid x = t) \cdot Pr(x = t)}{Pr(\text{no arrivals during a service period})} =$$

$$\frac{e^{-\lambda t} \cdot \mu \cdot e^{-\mu t}}{\frac{\mu}{\lambda+\mu}} = (\lambda + \mu) \cdot e^{-(\lambda+\mu)t} . \tag{7.35}$$

If there was at least one arrival during the service time period then the service time has the following distribution:

$$Pr(x = t \mid \text{at least one arrival during a service period}) =$$

$$\frac{Pr(\text{at least one arrival during a service period} \mid x = t) \cdot Pr(x = t)}{Pr(\text{at least one arrival during a service period})} =$$

$$\frac{(1 - e^{-\lambda t}) \cdot \mu \cdot e^{-\mu t}}{\frac{\lambda}{\lambda+\mu}} = (\frac{\lambda + \mu}{\lambda}) \cdot (\mu \cdot e^{-\mu t} - \mu \cdot e^{-(\lambda+\mu)t}) . \tag{7.36}$$

Now using the theorem of total transform, i.e.,

$$D(s) = D(s \mid q \geq 2) \cdot Pr(q \geq 2) + D(s \mid q = 0 \text{ or } 1) \cdot Pr(q = 0 \text{ or } 1) =$$

$$D(s \mid q \geq 2) \cdot Pr(q \geq 2)+$$

$$D(s \mid q = 0 \text{ or } 1, \text{ and no arrivals}) \cdot Pr(q = 0 \text{ or } 1, \text{ and no arrivals})+$$

$$D(s \mid q = 0 \text{ or } 1, \text{ and at least one arrival})\cdot$$

$$Pr(q = 0 \text{ or } 1, \text{ and at least one arrival}) =$$

$$D(s \mid q \geq 2) \cdot Pr(q \geq 2) + \tag{7.37}$$

$$D(s \mid q = 0 \text{ or } 1, \text{ and no arrivals}) \cdot Pr(q = 0 \text{ or } 1) \cdot Pr(\text{no arrivals}) +$$

$$D(s \mid q = 0 \text{ or } 1, \text{ and at least one arrival}) \cdot Pr(q = 0 \text{ or } 1) \cdot$$

$$Pr(\text{at least one arrival}) ,$$

where "no arrivals" and "at least one arrival" mean no arrivals during the exponentially distributed service time, and at least one arrival during the exponentially distributed service time, respectively.

Substituting the corresponding transforms of (7.35) and (7.36) into (7.37), and using the facts that

$$Pr(\text{no arrivals}) = \frac{\mu}{\mu + \lambda} , \tag{7.38}$$

$$Pr(\text{at least one arrival}) = \frac{\lambda}{\mu + \lambda} , \tag{7.39}$$

$$Pr(q \geq 2) = \rho^2 , \tag{7.40}$$

and

$$Pr(q = 0, \textit{ or } 1) = 1 - \rho^2 \tag{7.41}$$

results in

$$\begin{aligned}
D(s) &= \rho^2 \cdot (\frac{\mu}{\mu + s}) + (1 - \rho^2) \cdot \frac{\mu}{\mu + \lambda} \cdot \left(\frac{\mu + \lambda}{\mu + \lambda + s} \cdot \frac{\lambda}{\lambda + s} \right) + \\
&(1 - \rho^2) \cdot \frac{\lambda}{\mu + \lambda} \cdot \frac{\lambda + \mu}{\lambda} \cdot \left(\frac{1}{\mu + s} - \frac{\mu}{\mu + \lambda + s} \right) = \\
&\frac{\mu}{\mu + s} + (1 - \rho^2) \cdot \left[\frac{\mu \cdot \lambda}{(\mu + \lambda + s) \cdot (\lambda + s)} - \frac{\mu}{\mu + \lambda + s} \right] = \\
&\frac{\mu}{\mu + s} + (1 - \rho^2) \cdot \left[\frac{\lambda}{(\lambda + s)} - \frac{(\mu + \lambda)}{\mu + \lambda + s} \right] .
\end{aligned} \tag{7.42}$$

Taking now an inverse transform of equation (7.42) results in

$$D(s) \longleftrightarrow \mu \cdot e^{-\mu t} + (1-\rho^2) \cdot (\lambda - (\lambda+\mu) \cdot e^{-\mu t}) \cdot e^{-\lambda t} \; . \tag{7.43}$$

Q.E.D.

Integration of equation (7.14) gives the probability distribution function

$$F_T(t) \equiv Pr(T \leq t) =$$

$$\int_0^t f_T(\tau) \cdot d\tau = 1 - e^{-\mu t} - (1-\rho^2) \cdot e^{-\lambda t} \cdot (1 - e^{-\mu t}) \, , \tag{7.44}$$

for $t \geq 0$, and $F_T(t) = 0$ for $t < 0$.

The average interdeparture time is easily found by using equation (7.14) and is, of course,

$$\overline{T} = E(T) = \frac{1}{\lambda} , \tag{7.45}$$

as required.

The distribution of interdeparture times for extreme ρ's resembles the exponential distribution, i.e.,

$$\lim_{\rho \to 0} f_T(t) = \lambda \cdot e^{-\lambda t} \tag{7.46}$$

and

$$\lim_{\rho \to 1} f_T(t) = \lim_{\rho \to 1} \mu \cdot e^{-\mu t} = \lambda \cdot e^{-\lambda t} \; . \tag{7.47}$$

7.5 Appendix E: Probability of a packet blockage in a switch

It is shown in this Appendix that the probability of a packet blockage at a single attempt to cross a switch is

$$Pr(\text{packet blockage in a switch}) =$$

$$1 - \frac{(1-\gamma)}{(M-1)} \cdot [1 + (M-2) \cdot (1 - \frac{\delta}{M-1})^{(M-2)}] \, , \tag{7.48}$$

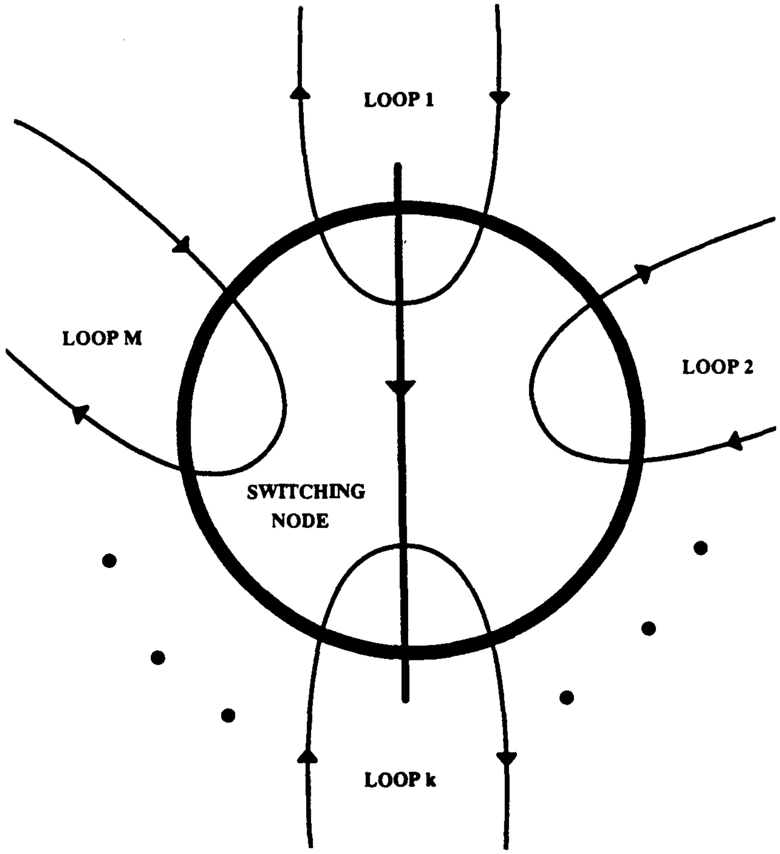


Figure 7.7: Model of a switching node

where δ is the traffic arriving to the switch from every loop and competing in the switch and γ is the returned traffic on each loop. In the double-loop configuration, γ consists of the traffic blocked at the other end of the loop. In the single-loop configuration, γ includes the traffic blocked at the other end of the loop as well as the traffic blocked at the current switching node.

Figure 7.7 will assist in explaining the analysis. The number of input/output loops connected to the switch is labeled M. It is assumed that the traffic matrix is totally symmetric (i.e., traffic on every loop is equally destinated to all the other loops). However, incestuous traffic is forbidden.

Without loss of generality let us concentrate on a packet arriving to the switch on loop number 1 and assume that the packet needs to be forwarded on loop number k. Furthermore, let us assume that there is no returned traffic on loop k, that is, loop k

serves as a perfect sink. (The last assumption will be relaxed later.) Thus the packet arriving on loop 1 can be blocked only if upon its arrival some other traffic is being forwarded to loop k. The probability that at any particular moment there is traffic from loop i to loop k is given by $\delta_i/(M-1)$. Thus the probability that at the moment of the packet's arrival there is no other traffic destinated to loop k from any of the other $(M-2)$ loops (loop k cannot direct traffic to itself) is given by

$$\prod_{i=2;\ i\neq k}^{M} (1 - \frac{\delta_i}{M-1}) \ . \tag{7.49}$$

If all the loops carry the same traffic, then $\delta = \delta_i$, and the above probability simplifies to

$$(1 - \frac{\delta}{M-1})^{(M-2)} \ . \tag{7.50}$$

The probability that the packet of loop 1 makes it through the switch (under these conditions) is given by

$$\begin{aligned}
&Pr(\text{packet makes it through the switch}) = \\
&Pr(\text{no traffic to loop } k \ \mid \text{last packet on loop } k \text{ came from loop } 1)\cdot \\
&\quad Pr(\text{last packet on loop } k \text{ came from loop } 1)+ \\
&Pr(\text{no traffic to loop } k \ \mid \text{last packet on loop } k \text{ came from other than loop } 1)\cdot \\
&\quad Pr(\text{last packet on loop } k \text{ came from loop other than loop } 1) \ .
\end{aligned} \tag{7.51}$$

If the last packet came from the same loop from which the current arriving packet comes, then

$$\begin{aligned}
&Pr(\text{no traffic to loop } k \ \mid \text{last packet on loop } k \text{ came from loop } 1) \\
&= 1 \ .
\end{aligned} \tag{7.52}$$

Also, the traffic arriving at any loop comes equally from all the other loops, just as the traffic on each loop is equally destinated to all the other loops. Substituting the appropriate terms in equation 7.51 results in

Pr(packet makes it through the switch | destination loop is a sink) =

$$\frac{1}{(M-1)} \cdot [1 + (M-2) \cdot (1 - \frac{\delta}{M-1})^{(M-2)}] \,. \tag{7.53}$$

Relax now the condition that the destination loop has no returned traffic. Assign γ to the returned traffic on loop k that leaves the switching node. Then, the probability that a packet makes it through the switch has to be multiplied by the probability that the output loop is free, i.e., $(1-\gamma)$. Thus

Pr(packet makes it through the switch) =

$$\frac{(1-\gamma)}{(M-1)} \cdot [1 + (M-2) \cdot (1 - \frac{\delta}{M-1})^{(M-2)}] \,. \tag{7.54}$$

Or

Pr(packet blockage in a switch) =

$$1 - \frac{(1-\gamma)}{(M-1)} \cdot [1 + (M-2) \cdot (1 - \frac{\delta}{M-1})^{(M-2)}] \,. \tag{7.55}$$

Q.E.D

7.6 Appendix F: Analytical solution for double-loop *Blazenet*

7.6.1 General remarks

The following analytical treatment evaluates the capacity and the average packet delay of a totally symmetric and balanced double-loop *Star Blazenet* configuration.

A totally symmetric network is defined as network which possesses the following property: each node in the network sees exactly the same constellation of other network nodes. This means that if the nodes' ids are erased then one could not distinguish between the network nodes.

A balanced network is defined as a network having exactly the same traffic pattern between each ordered pair of nodes. This means that in the traffic matrix, Λ, all $\lambda_{i,j}$ are equal. Therefore, define $\lambda \overset{\text{def}}{=} \lambda_{i,j}$ for $i \neq j$.)

General structure of a double-loop *Star Blazenet* is shown in figure 7.8. Some properties of this configuration are: The central node is assumed to perform the switching function only, and does not generate traffic. Number of bi-directional links (loops) in the network is equal to the number of peripheral nodes, N. The central node is labeled with number 0. No node is allowed to transmit to itself. Each packet has a path of exactly two hops. No blockage can occur on the second hop, provided that the destination is always ready to accept traffic. (This assumption on destination availability (the "perfect sink" assumption) enables to easily calculate the network capacity and the average packet delay.)

I consider here two separate cases: the slotted and the non-slotted version of *Blazenet Star*. For the slotted case I assume that the slot size is equal to the duration of transmission of a single constant length packet. For the non-slotted case I consider two cases of packet size: constant and exponentially distributed.

Packets arrive randomly to the network and their arrival rates are according to the traffic matrix. The arrivals are treated independently for each pair of nodes in the network.

Parameters:

N – number of nodes in the network.

L – number of unidirectional links (loops) in the network.

w_i [km] – length of link nr. i. Equal to half the total length of loop i.

C [bits/sec] – single fiber link transmission capacity.

C_{link} [bits/sec] – the actual capacity of a single link.

C_{total} [bits/sec] – total network capacity.

λ [packet/sec] – average packet arrival rate between any source-destination pair.

$\bar{l}$ – number of hops in an average path in the network.

v [km/sec] – speed of light in the fiber link.

$\bar{p}$ [bits] – average packet length. For constant packet length: $\bar{p} = p$.

s [sec] – slot time, $= \bar{p}/C$. For slotted version only.

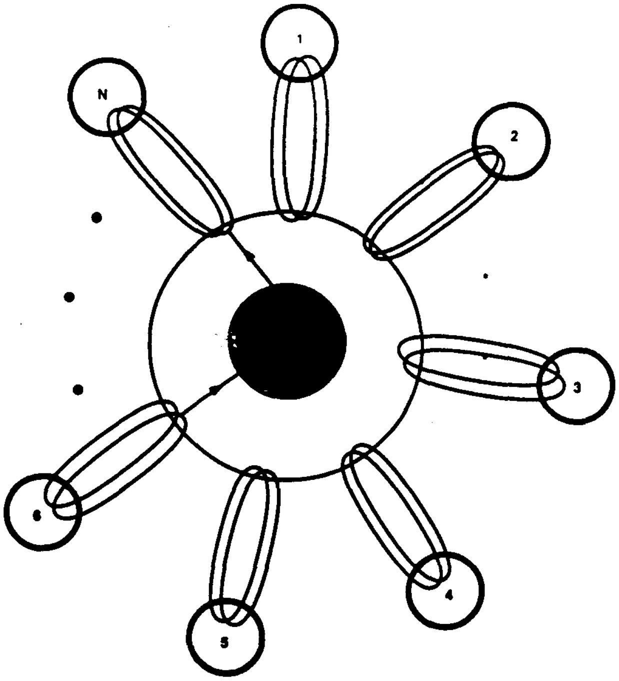


Figure 7.8: General *Star* configuration

$\bar{r}$ – the average number of attempts to cross a node for a single packet, before forwarded to the next loop. Equals average number of blockages a packet will experience at the node.

7.6.2 Slotted double-loop *Blazenet Star* analysis

In the slotted *Blazenet* version the slot arrivals to the central node are synchronized. I assume that the probability of more than one packet arrival during a single slot time is negligible. Therefore, I assume Bernoulli distribution of packet arrivals with parameter equal to $arrival\ rate * slot\ time$. Packets are assumed to be of fixed total length, p.

Capacity analysis

The analysis depends on the type of line hunting scheme that is used in the switching node. The two possibilities are: sequential or random line hunting. In sequential line hunting the Control in the switching node always tries first to route the packets of the link nr.1. After then, the Control serves the link nr.2, then nr.3, etc. In random hunting scheme the Control has no preference of any link, and the probability of successfully forwarding any packet from any link is equal for all links. Random hunting is a scheme giving equal priorities to all lines, but sequential hunting is probably a more practical scheme for realization.

The first condition bounds the total traffic in the network to total available fiber capacity of the network links:

$$\frac{C}{\bar{p}} \geq \frac{\lambda \cdot N \cdot (N-1) \cdot \bar{l} \cdot \bar{r}}{L}. \tag{7.56}$$

The condition is general and permits the calculation of the maximum possible λ, given that $\bar{r}$ is known.

I evaluate now $\bar{r}$ for *Star Blazenet* . In particular, I assume that $N \gg 1$.

First let assume sequential hunting. The probability of a blockage of a packet from link i trying to get to link j depends on the value of i. For link nr.1 blockage cannot occur and therefore $P_{block}(1) = 0$. For link nr.2. blockage occurs only if packet from link nr.2 is destined for the same link as packet from link nr.1. Therefore, $P_{block}(2) = 1 - \frac{N-2}{N-1}$. Continuing the argument in the same way one obtains that

i	$P_{block}(i)$
1	0
2	$1 - \frac{N-2}{N-1}$.
3	$1 - (\frac{N-2}{N-1})^2$.
$\vdots$	
N	$1 - (\frac{N-2}{N-1})^{(N-1)}$.

A blocked packet returns after one loop trip time to be considered again for forwarding. The same P_{block} governs his behavior this time. Therefore, a blocked packet persistently tries to be forwarded until success. Probability of a success on each trial is $1 - P_{block}$. Thus, the average number of trials for packet from link i , $\bar{r}(i)$, is $1/(1 - P_{block})$. Therefore,

i	$\bar{r}(i)$
1	1
2	$\frac{N-1}{N-2}$.
3	$(\frac{N-1}{N-2})^2$.
$\vdots$	
N	$(\frac{N-1}{N-2})^{(N-1)}$.

I refer to the the average number of transmissions on the forward portion of a loop required to forward a single packet (also labeled $\bar{r}$) as the average number of retransmissions. The average number of retransmissions can be calculated as follows:

$$\bar{r} = \frac{1}{N} \cdot \sum_{k=0}^{N-1} (\frac{N-1}{N-2})^k = \frac{1}{N} \cdot \frac{1-(\frac{N-1}{N-2})^N}{1-\frac{N-1}{N-2}} = \frac{N-2}{N} \cdot [(\frac{N-1}{N-2})^N - 1]. \qquad (7.57)$$

As N increases, $\bar{r}$ approaches a limit, that is

$$\lim_{N \to \infty} \bar{r} = e - 1 \cong 1.718. \qquad (7.58)$$

The minimum value of the average number of retransmissions is, therefore, achieved for large N, and is approximately 1.718.

In some networks which are not totally symmetric the important parameter is the maximum number of required retransmissions on the forward loop to get a packet through. This parameter, $\bar{r}_{max}$, is simply the $\bar{r}(N)$, since the N^{th} input loop consists now a bottleneck. Therefore,

$$\bar{r}_{max} = (\frac{N-1}{N-2})^{(N-1)}. \qquad (7.59)$$

An interesting property of $\bar{r}$ is that it is quite insensitive to increase in N. The following calculated values of $\bar{r}$ as a function of N ($N \geq 3$) reveal this phenomenon.

N	$\bar{r}$
3	2.333
5	1.928
10	1.798
15	1.767
20	1.754
25	1.746
30	1.741
35	1.738
40	1.735
45	1.733
50	1.732

As can be observed the value of $\bar{r}$ for wide range of values of N remains between 1.7 and 1.8.

Now let us assume random hunting. Here the probability of a packet getting through the node to the next loop is independent of the input loops ordering.

The probability of a single packet successfully being forwarded by the router in the node is given by

$$Pr_{succ} = \sum_{i=0}^{N-2} \binom{N-2}{i} \cdot \left(\frac{1}{N-1}\right)^{i} \cdot \left(\frac{N-2}{N-1}\right)^{(N-2-i)} \cdot \frac{1}{i+1}. \tag{7.60}$$

As can be easily verified, Pr_{succ} is an increasing function of N, and has a limit as N reaches infinity, that is

$$\lim_{N\to\infty} Pr_{succ} = 1 - \frac{1}{e} \cong 0.6321. \tag{7.61}$$

And since

$$\bar{r} = \frac{1}{Pr_{succ}}. \tag{7.62}$$

Therefore,

$$\lim_{N\to\infty} \bar{r} \cong 1.582. \tag{7.63}$$

Thus, the average number of retransmissions is an increasing function of N, and its maximum is achieved as N tends to infinity.

The number of required retransmissions till a packet is forwarded by the Control to the next loop is given by $1/Pr_{succ}$. Since the value of Pr_{succ} is equal for every input loop, this is also the average number of retransmissions.

Also in the case of random hunting, as in the case of sequential hunting, the average number of retransmission of a packet till it is forwarded to the next loop, $\bar{r}$, is relatively insensitive to the number of input loops, N.

The following table shows $\bar{r}$ for some values of N ($N \geq 2$):

N	$\bar{r}$
2	1.000
3	1.333
5	1.463
10	1.530
15	1.549
20	1.558
25	1.563
30	1.566
35	1.568
40	1.570
45	1.571
50	1.573

For sequential hunting the $\bar{r}$ is a decreasing function of N with its minimum value of 1.718. In the case of random hunting scheme, $\bar{r}$ is an increasing function of N with its maximum value of 1.582. Consequently, I conclude that the average number of retransmissions, $\bar{r}$, is always greater for sequential hunting scheme than for the random scheme.

The capacity of the double-loop *Star Blazenet* can be calculated using equation 7.56 with equality, where the number of links, L, is equal to the number of "bottleneck" links, i.e., the links connecting the sources with the central nodes. Thus $L = N$ and the average path length, $\bar{l}$, is equal to 1, since we consider now only the sub-graph, which consists of the "bottleneck" links only. As a good approximation for the average number of retransmissions, I assume $\bar{r} = 1.7$ for sequential hunting and $\bar{r} = 1.5$ for random hunting.

$$\lambda = \frac{C}{\bar{p}\cdot(n-1)\cdot\bar{r}} \text{ [packets/sec]} . \tag{7.64}$$

For sequential hunting one gets the following approximation:

$$\lambda \cong \frac{C}{1.7\cdot\bar{p}\cdot(N-1)} \text{ [packets/sec]} . \tag{7.65}$$

And for random hunting

$$\lambda \cong \frac{C}{1.5\cdot\bar{p}\cdot(N-1)} \text{ [packets/sec]} . \tag{7.66}$$

The total network capacity, C_{total}, is in general case:

$$C_{total} = \frac{C\cdot N}{\bar{r}} \text{ [bits/sec]} . \tag{7.67}$$

And since $\bar{r}$ is quite constant with N, the total network capacity increases linearly with N.

Delay analysis of the network

The delay in *Blazenet* is composed of two factors. The delay encountered while waiting to be admitted to the network, i.e. the queuing delay d_{queue}, and the delay encountered by the packet when traveling through the network $d_{network}$. As far as the user is concerned, the interesting quantity is the $d_{queue} + d_{network}$. From the point of investigating the performance of *Blazenet* , the interesting parameter is the $d_{network}$.

The value of $d_{network}$ depends on the traffic load in the network. For underloaded networks very few packets are blocked. Consequently, only few packets are returned and $d_{network}$ is simply the propagation delay of a packet through the network in addition to the transmission time. The propagation delay is the total length of the forward portion of the loops on the packet's path, divided by the speed of light on the optical links.

$$d_{network|light\ load} \cong \sum_{k=1}^{\bar{l}} \frac{w_k}{v} + (\bar{l}-1)\cdot\frac{\bar{p}}{C} \text{ [sec]} . \tag{7.68}$$

In particular for symmetric networks when all links are of equal length, $w_i = w$,

$$d_{network|light\ load} \cong \bar{l} \cdot \frac{w}{v} + (\bar{l}-1) \cdot \frac{\bar{p}}{C}\ [\text{sec}]\ . \tag{7.69}$$

When the network traffic load increases, some of the packets are blocked and returned on the return portion of loops. Consequently, the average delay increases and can be calculated according to the following formula:

$$d_{network|load} = \sum_{k=1}^{\bar{l}} \frac{(2\bar{r}_k - 1) \cdot w_k}{v} + (\bar{l}-1) \cdot \frac{\bar{p}}{C}\ [\text{sec}]\ . \tag{7.70}$$

It should be pointed out that the term $(\bar{l}-1) \cdot \frac{\bar{p}}{C}$ appears when the implementation of a node includes delay line to store the forwarded packet. This way of implementation is required for realization of some of the extended features of *Blazenet.* In implementations with no such a delay line, so that a packet can be immediately forwarded upon arrival (given the output loop is free, of course), this term does not exist.

Considering again the particular case of the *Star* topology, the network delay can be found by using equation 7.70, i.e.,

$$d_{Star} = \frac{2\bar{r} \cdot w}{v} + \frac{\bar{p}}{C}\ [\text{sec}]\ . \tag{7.71}$$

Since $\bar{r}$ is quite insensitive to changes in N, the average packet delay is quite constant with N.

To summerize, the slotted *Star Blazenet* have the property of total capacity linearly increasing with N, and the average delay essentially constant with N.

7.6.3 Non-Slotted double-loop *Blazenet Star* analysis

I assume here that packets arrive at each input loop according to the Poisson distribution, that the arrivals are unsynchronized, among themselves, and that the traffic matrix is totally symmetric. The switch will achieve its capacity when the packets on every loop are back-to-back, that is $\rho = 1.0$ for every loop.

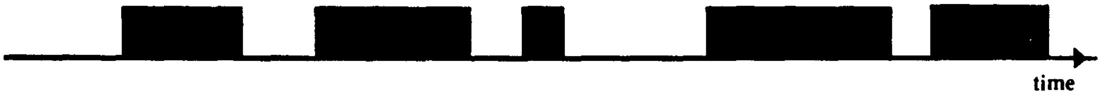


Figure 7.9: Typical busy-idle pattern of an output line

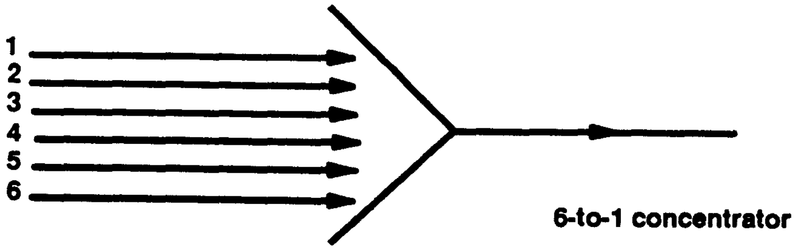


Figure 7.10: Model of an output line as a concentrator

Exponential packet length

In order to evaluate the switch capacity, concentrate on one output line j. A typical busy-idle pattern of the output line is shown in figure 7.9. I will show that the average idle period equals the average busy period and thus the capacity equals 0.5.

Line j gets its traffic from each one of the $N-1$ loops. Each loop i contributes $\rho_{i,j} = \frac{1}{N-1}$ of its total input traffic $\rho_i = 1.0$. Thus, each output loop can be modeled as a contention concentrator of $N-1$ input lines, as shown in Figure 7.10. Each such input line contributes traffic of intensity $\rho_k = \frac{1}{N-1}$ to the concentrator.

The distribution of the idle period can be calculated by noticing that the output line j has an idle period of at least x seconds iff there is no new arrival on any of the inputs to the concentrator for at least x seconds. Let $F_I(x)$ be the distribution of the idle period length. Then,

$$F_I(x) = Pr(I \leq x)\,, \tag{7.72}$$

$$1 - F_I(x) = Pr(I \geq x) =$$

$$\prod_{k=1}^{N-1} [Pr(\text{no new arrival on line } k \text{ for at least time } x)] =$$

$$= \prod_{k=1}^{N-1} [Pr(k \text{ idle for time} \geq x \mid k \text{ idle at the beginning of idle period})$$

$$\cdot Pr(k \text{ idle when idle period starts})$$

$$+Pr(\text{time till new arrival on } k \geq x \mid k \text{ busy at the beginning of idle period})$$

$$\cdot Pr(\text{input } k \text{ busy at the beginning of the idle period})]. \tag{7.73}$$

Since $\rho_k = \frac{1}{N-1}$, then

$$Pr(\text{input } k \text{ idle at the beginning of the idle period}) = 1 - \frac{1}{N-1} \tag{7.74}$$

and

$$Pr(\text{input } k \text{ busy at the beginning of the idle period}) = \frac{1}{N-1}\,. \tag{7.75}$$

When an input is idle, then the time till the next arrival is exponentially distributed with parameter $\lambda_k = (\mu C)/(N-1)$. So that

$$Pr(\text{input } k \text{ idle for time} \geq x \mid \text{input } k \text{ idle when the idle period starts})$$

$$= e^{-\lambda_k x}. \tag{7.76}$$

When an input is busy then the time till the next arrival is composed of a sum of two exponentially distributed times with parameters: $(\mu C)/(N-1)$ and μC. Consequently, (see Appendix G for proof):

$$Pr(\text{time till new arrival on } k \geq x \mid \text{input } k \text{ busy when idle period starts}) =$$

$$= \frac{\lambda_2 e^{-\lambda_1 t} - \lambda_1 e^{-\lambda_2 t}}{\lambda_1 - \lambda_2}. \tag{7.77}$$

Substituting all the factors into the equation 7.73 results in

$$1 - F_I(x) =$$

$$= \{e^{-\lambda_k x}.[\frac{(N-2)\cdot(\mu C - \lambda_k) - \mu C}{(N-1)\cdot(\mu C - \lambda_k)}]+e^{-\mu C x}.[\frac{\lambda_k}{(N-1)\cdot(\mu C - \lambda_k)}]\}^{N-1}. \tag{7.78}$$

If $N \gg 1$ and since $\lambda_k = \frac{\mu C}{N-1}$ the above formula reduces to a simple exponential distribution

$$1 - F_I(x) = e^{-\mu C x}. \tag{7.79}$$

Therefore,

$$F_I(x) = 1 - e^{-\mu C x} \tag{7.80}$$

and

$$f_I(x) = \mu C e^{-\mu C x}. \tag{7.81}$$

Thus,

$$\bar{I} = \frac{1}{\mu C}. \tag{7.82}$$

And since $\bar{B} = \frac{1}{\mu C}$, I conclude that

$$\frac{\bar{B}}{\bar{B}+\bar{I}} = 0.5\,, \tag{7.83}$$

i.e, the capacity of a loop is $0.5 \cdot C$.

Consequently, the total capacity of a non-slotted double-loop *Star* with N input and N output loops and with exponentially distributed packet length is

$$C_{total} = \frac{N \cdot C}{2}\ [\text{bits/sec}]\,. \tag{7.84}$$

Thus the link capacity is limited to half its transmission rate and the total capacity of a non-slotted *Star* increases linearly with the number of the peripheral nodes.

I will now evaluate the packet delay, encountered when a packet attempts to cross the switch, when the switch operates at its capacity. More precisely I will evaluate the number of retransmissions a single packet will undergo on the average, until it succeeds in crossing the switch.

Probability of a single packet from an input loop to enter a destination output loop is equal to the ratio of the idle period to the total time, as seen by packets on the input loop. The idle periods on a destination line, as seen by packets on any input loop, are longer than the total idle periods on the destination line, since the total idle periods include the traffic from this specific input line. The difference is equal to $\frac{1}{N-1}$ of the total idle periods. Thus, the probability of a packet to make it through the switch on a single attempt is given by

$$Pr_{success} = (1 + \frac{1}{N-1}) \cdot \frac{1}{2} = \frac{N}{2(N-1)}. \tag{7.85}$$

A blocked packet will continuously try to cross the switch, until it is successful. Thus, the average number of retransmissions (NOR) until a packet makes it through the switch is equal to $1/Pr_{success}$, or

$$NOR = \frac{2(N-1)}{N}. \tag{7.86}$$

Thus for example, NOR equals 1.6 for $N = 5$ and 1.8 for $N = 10$. As N increases the average number of retransmissions increases, being bounded by the value of 2 when N approaches infinity, i.e:

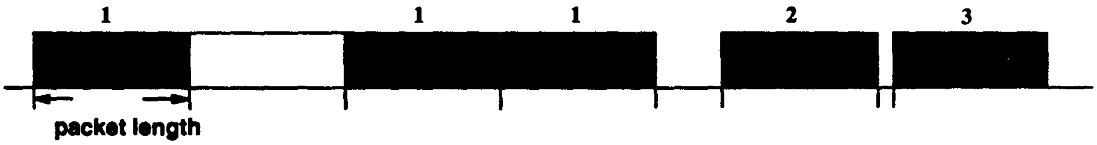


Figure 7.11: Typical busy-idle pattern for constant length packets

$$\lim_{N\to\infty} NOR = 2. \tag{7.87}$$

NOR varies from the value of 1.5 to 2.0 as N varies from 3 to infinity. NOR is, therefore, a slowly increasing function of N. Thus the average delay of the switch is not a strong function of the number of input and output loops.

Constant packet length

Solution for the case of constant packet length is slightly more difficult, since the probability of an arrival of a new packet during an idle period depends on the input loop. That is, if, after a packet is done, a new packet arrives from the same input line as the just transmitted one, the idle period will be in multiples of the packet length. If, however, the new packet comes from a different input loop, and since arrival on different input lines are unsynchronized, the idle period can be of any length. This idea is presented in figure 7.11, where the busy-idle pattern for an output line is shown. The numbers on the packets represent the number of an input loop that the packet comes from. Note that on an input line packets are arranged back-to-back, since $\rho = 1$ on these lines.

The time axis is divided into packet length intervals. A new packet can arrive from the same source as the just transmitted one. In this case the idle period is of length 0. The probability of such an event is given by

$$Pr(\text{next arrival from the same source}) = \frac{1}{N-1} \overset{\text{def}}{=} p_1. \tag{7.88}$$

The same source can have a packet to other destination with probability

$$Pr(\text{next arrival not from the same source}) = \frac{N-2}{N-1} \overset{\text{def}}{=} q_1. \qquad (7.89)$$

In this case the next arrival to the output loop can come during the following interval from any of the other $N-2$ input lines. This can occur with probability

$$Pr(\text{arrival from different source during an interval}) =$$

$$= 1 - \prod_{k=1}^{N-2} Pr(\text{no arrival from input } k \text{ to the output line}) =$$

$$1 - [\frac{N-2}{N-1}]^{N-2} \overset{\text{def}}{=} p_2. \qquad (7.90)$$

Since all the input loops are unsynchronized, this arrival can take place equally at any instant during the interval. Thus, in this case, the mean idle period is half of the packet length.

No arrival from any of the other $N-2$ input lines to the output line during an interval can occur with a probability

$$Pr(\text{no arrival from different source during an interval}) =$$

$$= \prod_{k=1}^{N-2} Pr(\text{no arrival from input } k \text{ to the output line}) =$$

$$[\frac{N-2}{N-1}]^{N-2} \overset{\text{def}}{=} q_2. \qquad (7.91)$$

Therefore, the idle period can take on values $0, 1, 2, 3, ...$ in the case the next arrival comes from the same source as the previous packet, or takes on the average values $0.5, 1.5, 2.5, ...$ in the case the next arrival comes from a different source. The average idle period is found by calculating the expected value of $\bar{I}$.

$$\bar{I} = \sum_{\text{all values I}=j} l \cdot Pr(\text{I}=j) =$$

$$0 \cdot p_1 + \frac{1}{2} \cdot q_1 p_2 + 1 \cdot q_1 q_2 p_1 + 1\frac{1}{2} \cdot q_1 q_2 q_1 p_2 + 2 \cdot q_1 q_2 q_1 q_2 p_1 + ... =$$

$$= \sum_{k=0}^{\infty} k \cdot p_1 q_1^k q_2^k + p_2 \cdot \sum_{k=0}^{\infty} (\frac{1}{2} + k) \cdot q_1^{k+1} q_2^k = \dots =$$

$$= \frac{p_2 q_1 + p_2 q_1^2 q_2 + 2 p_1 q_1 q_2}{2(1 - q_1 q_2)^2}. \tag{7.92}$$

Substituting for p_1, q_1, p_2, q_2 the following formula can be obtained:

$$\bar{I} = \frac{(\frac{N-2}{N-1}) \cdot [1 - (\frac{N-2}{N-1})^{N-2}] \cdot [1 + (\frac{N-2}{N-1})^{N-1}] + 2(\frac{N-2}{N-1})^{N-1} \cdot \frac{1}{N-1}}{2[1 - (\frac{N-2}{N-1})^{N-1}]^2} . \tag{7.93}$$

As N approaches infinity $\bar{I}$ converges to a limit

$$\lim_{N \to \infty} \bar{I} = \frac{e+1}{2(e-1)} \simeq 1.082. \tag{7.94}$$

The capacity of a single link is calculated by

$$C_{link} = \frac{1}{1 + \bar{I}} \cdot C \text{ [bits/sec]} . \tag{7.95}$$

As N increases C_{link} converges to a limit

$$\lim_{N \to \infty} C_{link} = 0.4803... \cdot C \text{ [bits/sec]} . \tag{7.96}$$

It can be easily verified that link capacity remains practically stable as a function of N. Consequently, the total capacity of the configuration increases linearly with N and is equal, for large values of N, to

$$C_{total} \simeq 0.48 \cdot C \cdot N . \tag{7.97}$$

An interesting point is that the link efficiency is an increasing function of N in the constant packet length case, whereas it is a decreasing function in the exponential packet length case. I also note that the link efficiency is slightly higher for small values of N for the constant packet length case than for the exponential case. As N increases, the reverse becomes true.

The delay through the switch for the constant packet length is calculated in the similar way as for the exponential case. For large values of N the NOR reaches a limit, i.e.,

$$\lim_{N \to \infty} NOR = 2.0820... \tag{7.98}$$

Once again, the NOR is a slowly increasing function of N.

7.7 Appendix G: Sum of two exponential random variables

Assume x_1 and x_2 are two *independent, exponentially-distributed* random variables with parameters λ_1 and λ_2, respectively. I will find the distribution of $y = x_1 + x_2$.

$$f_1(x_1) = \lambda_1 \cdot e^{-\lambda_1 x_1}. \tag{7.99}$$

$$f_1(x_2) = \lambda_2 \cdot e^{-\lambda_2 x_2}. \tag{7.100}$$

$$\begin{aligned}
F_{x_1+x_2}(t) &= Pr(x_1 + x_2 \leq t) = \\
&= \int_0^t Pr(x_1 \leq t - s \mid x_2 = s) \cdot \lambda_2 \cdot e^{-\lambda_2 s} \cdot ds = \\
&= \int_0^t Pr(x_1 \leq t - s) \cdot \lambda_2 \cdot e^{-\lambda_2 s} \cdot ds = \\
&= \int_0^t [1 - e^{-\lambda_1 (t-s)}] \cdot \lambda_2 \cdot e^{-\lambda_2 s} \cdot ds = \\
&= 1 - e^{-\lambda_2 t} + \frac{\lambda_2}{\lambda_1 - \lambda_2} \cdot (e^{-\lambda_1 t} - e^{-\lambda_2 t}) = \\
&= 1 + \frac{\lambda_2 e^{-\lambda_1 t} - \lambda_1 e^{-\lambda_2 t}}{\lambda_1 - \lambda_2}.
\end{aligned} \tag{7.101}$$

$$f_{x_1+x_2}(t) = \frac{d}{dt} F_{x_1+x_2} = \frac{\lambda_1 \cdot \lambda_2}{\lambda_1 - \lambda_2}(e^{-\lambda_2 t} - e^{-\lambda_1 t}). \tag{7.102}$$

Q.E.D

Chapter 8

Bibliography

[1] Z. Haas and D. R. Cheriton, "*Blazenet*: A High-Performance Wide-Area Packet-Switched Network Using Optical Fibers," in Proceedings of the *IEEE Pacific RIM Conference on Communication, Computers and Signal Processing,* Victoria, B.C., June 4-5, 1987.

[2] Z. Haas and D. R. Cheriton, "*Blazenet*: A Photonic Implementable Wide-Area Network," *Department of Computer Science, Stanford University,* technical report no. STAN-CS-87-1185, October 1987.

[3] Z. Haas and D. R. Cheriton, "A Case for Packet-Switching in High-Performance Wide-Area Networks," in Proceedings of *SIGCOMM '87 Workshop,* Stowe VT, Aug 11-13, 1987.

[4] C. Gordon Bell, "Gordon Bell calls for a U.S. research network," *IEEE SPECTRUM,* vol.25, no.2, February 88.

[5] J. S. Turner, "New Directions in Communications," in Proceedings of the *International Seminar on Digital Communications, New Directions in Switching and Networks,* Zürich, Switzerland, March 11-13, 1986.

[6] Stephen B. Weinstein, "Telecommunications in the coming decades," *IEEE SPECTRUM,* vol.24, no.11, November 87.

[7] Abraham Peled, "The Next Computer Revolution," *Scientific American,* vol.257, no.4, October 87.

[8] Glenn D. Rennels and Edward H. Shortliffe, "Advanced Computing for Medicine," *Scientific American,* vol.257, no.4, October 87.

[9] Lloyd R. Linnell, "A Wide-Band Local Access System Using Emerging-Technology Components," *IEEE Journal on Selected Areas in Communications,* vol.SAC-4, no.4, July 86.

[10] Robert P. Freese, "Optical disks become erasable," *IEEE SPECTRUM,* vol.25, no.2, February 88.

[11] Heinrich Armbrüster and Gerhard Arndt, "Broadband Communication and Its Realization with Broadband ISDN," *IEEE Communications Magazine,* vol.25, no.11, November 87.

[12] N. S. Jayant and Peter Noll, "Digital Coding of Waveforms: Principles and Applications to Speech and Video," *Prentice-Hall, Inc.,* 1984.

[13] Tekla S. Perry, "Hypermedia: finally here," *IEEE SPECTRUM,* volume 24, number 11, November 87.

[14] P. Kermani and L. Kleinrock, "Virtual Cut-Through: A New Computer Communication Switching Technique," *Computer Networks 3,* 267-286, 1979.

[15] P. Kermani, "Switching and Flow Control Techniques in Computer Communication Networks," Ph.D. thesis, *Computer Science Dept., School of Engineering and Applied Science, University of California,* Los Angeles, February 1978.

[16] M. Ilyas and H. T. Mouftah, "Quasi cut-through: New hybrid switching technique for computer communication networks," *IEE PROCEEDINGS,* vol.131, no.1, Jan 1984.

[17] Ahmed Abo-Taleb and Hussein T. Mouftah, "Delay Analysis Under a General Cut-Through Switching Technique in Computer Networks," *IEEE Transactions on Communications,* vol.COM-35, no.3, March 1987.

[18] P. E. Jackson and C. D. Stubbs, "A Study of Multi-access Computer Communications," in Proceedings of *AFIPS Conference*, Spring Joint Computer Conference, **34,** 1969.

[19] E. Fuchs and P. E. Jackson, "Estimates of Distributions of Random Variables for Certain Computer Communications Traffic Models," *Communications of the ACM,* **13**, 1970.

[20] Leonard Kleinrock, "Queueing Systems, Volume 2: Computer Applications," *Wiley-Interscience,* New-York, 1976.

[21] David R. Cheriton, "VMTP: a transport protocol for the next generation of communication systems," in Proceedings of *ACM SIGCOMM'86,* Aug 5-7, 1986.

[22] Donald B. Keck, "Fundamentals of Optical Waveguide Fibers," *IEEE Communications Magazine,* vol.23, no.5, May 1985.

[23] L. C. Blank *et al.,* "120-Gbit · km lightwave system experiments using 1.478-μm and 1.52-μm distributed feedback lasers," in Proceedings of the *Conference on Optical Fiber Communication,* pp. 86-87, 1985.

[24] S. R. Nagel, "Optical Fiber–the Expanding Medium," *IEEE Communications Magazine,* vol.25, no.4, April 87.

[25] Kiyoshi Nakagawa, Norihisa Ohta, and Tetsuya Kanada, "Overview of Very High Capacity Optical Transmission Systems," *IEEE Global Telecommunications Conference,* San Diego, California, Nov.28-Dec.1, 1983.

[26] Paul S. Henry, "Introduction to Lightwave Transmission,“ *IEEE Communications Magazine,* vol.23, no.5, May 1985.

[27] F. Guterl and G. Zorpette, "Fiber optics: poised to displace satellites," *IEEE SPECTRUM,* August 1985.

[28] J. Salz, "Modulation and detection for Coherent Lightwave Communications," *IEEE Communications Magazine,* vol.24, no.6, May 1986.

[29] E. E. Basch and T. G. Brown, "Introduction to Coherent Optical Fiber Transmission," *IEEE Communications Magazine,* vol.23, no.5, May 1985.

[30] I. W. Stanley, G. R. Hill, and D. W. Smith, "The Application of Coherent Optical Techniques to Wide-Band Networks," *IEEE Journal of Lightwave Technology,* vol.LT-5, no.4, April 1987.

[31] A. Bellman, "Switching Architectures Towards the Nineties," in Proceedings of the *International Seminar on Digital Communications, New Directions in Switching and Networks,* Zürich, Switzerland, March 11-13, 1986.

[32] Paul R. Prucnal, Mario A. Santoro, and Sanjay K. Sehgal, "Ultrafast All-Optical Synchronous Multiple Access Fiber Networks," *IEEE Journal on Selected Areas in Communications,* vol.SAC-4, no.9, December 86.

[33] Kenneth P. Jackson, Steven A. Newton, Behzad Moslehi, Moshe Tur, C. Chapin Cutler, Joseph W. Goodman, and H. J. Shaw, "Optical Fiber Delay-Line Signal Processing," *IEEE Transactions on Microwave Theory and Techniques,* vol.MTT-33, no.3, March 1985.

[34] R. D. Rosner and B. Springer, "Circuit and packet switching," *Computer Networks 1,* 7-26, 1976.

[35] K. Kümmerle and H. Rudin, "Packet and Circuit Switching: Cost/Performance Boundaries," *Computer Networks 2,* 3-17, 1978.

[36] H. Miyahara, Y. Teshigawara, and T. Hasegawa, "Delay and Throughput Evaluation of Switching Methods in Computer Communication Networks," *IEEE Transactions on Communications,* vol.COM-26, no.3, March 1978.

[37] P. Kermani and L. Kleinrock, "A Tradeoff Study of Switching Systems in Computer Communication Networks," *IEEE Transactions on Computers,* vol.C-29, no.12, December 1980.

[38] M. Ilyas and H. T. Mouftah, "Delay analysis of a modified cut-through switching for multipacket messages," *IEE PROCEEDINGS,* vol.132, no.1, Jan 1985.

[39] Byung Cheol Shin and Chong Kwan Un, "Performance Analysis of a Quasi-M/D/1 Cut-Through Switching Network with Noisy Channels,"

IEEE Transactions on Communications, vol.COM-34, no.9, September 1986.

[40] David R. Cheriton and Carey L. Williamson, "Network Measurement of the VMTP Request-Response Protocol in the V Distributed System," *Department of Computer Science, Stanford University,* technical report no. STAN-CS-87-1145, February 1987.

[41] David R. Cheriton, "VMTP: Versatile Message Transport Protocol," Protocol Specification, *Computer Science Department, Stanford University,* January 10, 1988.

[42] G. D. Cole, "Computer Measurements; Technical and Experiments," Ph.D. Dissertation, *Computer Science Department, UCLA,* June 1971.

[43] Alan Huang, "The Relationship Between STARLITE, a Wideband Digital Switch, and Optics," in Proceedings of *International Conference on Communication,* Toronto, June 22, 1986.

[44] Alan Huang and Scott Knauer, "STARLITE: A Wideband Digital Switch," in Proceedings of *IEEE Global Telecommunications Conference,* Atlanta, vol.1, November 1984.

[45] Chet Day, Jim Giacopelli, and Jason Hickey, "Applications of Self-Routing Switches to LATA Fiber Optic Networks," in Proceedings of *ISS'87,* Phoenix, Arizona, March 15-20, 1987.

[46] T. T. Lee, R. Boorstyn, and E. Arthurs, "The Architecture of a Multicast Broadband Packet Switch," in Proceedings of *INFOCOM'88,* New Orleans, LA, March 27-April 1, 1988.

[47] T. T. Lee, "Non-blocking Copy Networks for Multicast Packet Switching," in Proceedings of *International Zürich Seminar on Digital Communications,* Zürich, Switzerland, March 8-11, 1988.

[48] P. Newman, "A broad-band packet switch for multi-service communications," *The University of Cambridge Computer Laboratory,* July 1987.

[49] Yu-Shuan Yeh, Michael G. Hluchyj, and Anthony S. Acampora, "The Knockout Switch: A Simple, Modular Architecture for High-Performance Packet Switching," *IEEE Journal on Selected Areas in Communications,* vol.SAC-5,no.8, October 1987.

[50] E. Stewart Lee and Peter I. P. Boulton, "The Principles and Performance of Hubnet: A 50 Mbit/s Glass Fiber Local Area Network," *IEEE Journal on Selected Areas in Communications,* vol.SAC-1,no.5, November 1983.

[51] Harvey Ikeman, E. Stewart Lee, and Peter I.P. Boulton, "High-Speed Network Uses Fiber Optics," *Electronics Week,* October 22, 1984.

[52] N. F. Maxemchuk, "The Manhattan street network," in Proceedings of *GLOBECOM'85,* New Orleans, LA, December 1986.

[53] Nicholas F. Maxemchuk, "Routing in the Manhattan Street Network," *IEEE Transactions on Communications,* vol.COM-35, no.5, May 1987.

[54] Yousuke Yamamoto, Hiroshi Miyanaga, Yoshiji Kobayashi, Yasukazu Terada, and Naoaki Yamanaka, "A Novel Concept for High-Speed Time Switch Approaching Memory Read Cycle Limit," *IEEE Transactions on Communications,* vol.COM-34, no.9, September 1986.

[55] R. A. Thompson, "An Experimental Photonic Time-Slot Interchanger using Optical Fibers as Re-Entrant Delay-Line Memories," in Proceedings of the *Optical Fiber Communication Conference, Technical Digest,* 1986.

[56] P. R. Prucnal, D. J. Blumenthal, and P. A. Perrier, "Self-Routing Photonic Switching Demonstration with Optical Control," *The Center for Telecommunications Research, Columbia University,* technical report no. CU-CTR-TR-33, 1987.

[57] P. R. Prucnal, D. J. Blumenthal, and P. A. Perrier, "Photonic Switch with Optically Self-Routed Bit Switching," *IEEE Communication Magazine,* vol.25, no.5, May 1987.

[58] Paul R. Prucnal and Mario A. Santoro, "Spread Spectrum Fiber Optic Local Area Network Using Optical Processing," *The Center for*

Telecommunications Research, Columbia University, technical report no. CU/CTR/TR-2, 1987.

[59] Paul R. Prucnal, "VLSI Fiber Optic Local Area Network," *The Center for Telecommunications Research, Columbia University,* technical report no. CU-CTR-TR-27, 1987.

[60] P. R. Prucnal, "All-optical ultra-fast networks," *SPIE Fiber Telecommunications and Computer Networks,* vol.715, 1986.

[61] J. S. Turner, "Design of a Broadcast Packet Switching Network," *Washington University, Computer Science Department,* technical report no. WUCS-85-4, March 1985.

[62] Jonathan S. Turner and Leonard F. Wyatt, "A Packet Network Architecture for Integrated Services," in the Proceedings of *GLOBECOM '83,* San Diego, Nov.28-Dec.1, 1983.

[63] A. Tanenbaum, "Computer Networks," *Englewood Cliffs: Prentice Hall,* 1981.

[64] Vineet Singh, "The Design of a Routing Service for Campus-Wide Internet Transport," M.Sc. thesis, *Laboratory for Computer Science, MIT*, MIT/LCS/TR-270, August 1981.

[65] Krishnan Padmanabhan and Arun N. Netravali, "Dilated Networks for Photonic Switching," *IEEE Transactions on Communications,* vol.COM-35, no.12, December 1987.

[66] Ron A. Spanke, "Architectures for Guided-Wave Optical Space Switching Systems," *IEEE Communication Magazine,* vol.25, no.5, May 1987.

[67] The MATHLAB Group, "MACSYMA Reference Manual," *Laboratory for Computer Science, MIT,* Version Ten, January 1983.

[68] David R. Cheriton and Steve E. Deering, "Host Groups: A Multicast Extension for Datagram Internetworks," in Proceedings of the *9th Data Communication Symposium,* IEEE Computer Society and ACM SIGCOMM, September 1985.

[69] Yogen K. Dalal and Robert M. Metcalfe, “Reverse Path Forwarding of Broadcast Packets,” *Communication of the ACM,* vol.21, no.12, December 1978.

[70] Stephen E. Deering, “Multicast Routing in Internetworks and Extended LANs,” submitted to *ACM SIGCOMM ‘88.*

[71] M. I. Belovolov, E. M. Dianov, V. I. Karpov, A. P Kryukov, A. A. Kuznetsov, and A. M. Prokhorov, “Fiber-Optic Dynamic Memory. First Demonstration,” in Proceedings of the *Tenth European Conference on Optical Communication,* Stuttgart, FRG, September 3-6, 1984.

[72] Lixia Zhang, “Designing a New Architecture for Packet Switching Communication Networks,” *IEEE Communications Magazine,* vol.25, no.9, September 1987.

[73] P. Green, ed., “Computer Network Architecture and Protocols,” *New York: Plenum Press,* 1982.

[74] Takakiyo Nakagami and Teruo Sakurai, “Optical and Optoelectronic Devices for Optical Fiber Transmission Systems,” *IEEE Communications Magazine,* vol. 26, no.1, January 88.

[75] W. J. Dally and C. L. Seitz, “The Torus Routing Chip,” *Journal of Distributed Computing*, vol. 1, no. 3, 1986.

[76] Paul Bratley, Bennett L. Fox, and Linus E. Schrage, “A Guide to Simulation,” *Springer-Verlag,* Second Edition, 1987.

[77] Charles H. Sauer and K. Mani Chandy, “Computer Systems Performance Modeling,“ *Prentice-Hall, Inc.,* 1981.

[78] M. Aoki, T. Uchiyama, S. Tonami, A. Hayakawa, and H. Ichikawa, “Protocol Processing for High-Speed Packet Switching Systems,” in Proceedings of the *International Seminar on Digital Communications, New Directions in Switching and Networks,* Zürich, Switzerland, March 11-13, 1986.

[79] Ira Jacobs, “Design Considerations for Long–Haul Lightwave Systems,” *IEEE Journal on Selected Areas in Communications,* volume SAC-4, number 9,

December 1986.

[80] Stewart E. Miller and Alan G. Chynoweth, ed., "Optical Fiber Telecommunications," *Academic Press, Inc.,* 1979.

[81] Helmut F. Wolf, ed., "Handbook of Fiber Optics: Theory and Applications," *Garland Publishing, Inc.,* 1979.

[82] Edward A. Lacy, "Fiber Optics," *Prentice-Hall, Inc.,* 1982.

[83] H. S. Hinton, "Photonic Switching Using Directional Couplers," *IEEE Communication Magazine,* vol.25, no.5, May 1987.

[84] Tadahiko Yasui and Hirokazu Goto, "Overview of Optical Switching Technologies in Japan," *IEEE Communications Magazine,* vol.25, no.5, May 1987.

[85] S. F. Su, L. Jou, and J. Lenart, "A Review on Classification of Optical Switching Systems," *IEEE Communications Magazine,* vol.24, no.5, May 1986.

[86] James E. Rogalski, "Evolution of Gigabit Lightwave Transmission Systems," *AT&T Technical Journal,* vol.66, issue 3, May/June 1987.

[87] S. Y. Suh, S. W. Granlund, and S. S. Hegde, "Fiber-Optic Local Area Network Topology," *IEEE Communications Magazine,* vol.24, no.8, August 1986.

[88] Leonard Kleinrock, "Queueing Systems, Volume 1: Theory," *Wiley-Interscience,* New-York, 1975.

[89] Leonard Kleinrock, "Communication Nets: Stochastic Message Flow and Delay," *Dover Publications, Inc.,* New-York, 1972.

[90] S. A. Nozaki and S. M. Ross, "Approximations in Finite-Capacity Multi-Server Queues with Poisson Arrivals," *Journal of Applied Probability* **15**, pp.826-834, 1978.

[91] Sheldon M. Ross, "Introduction to Probability Models," *Academic Press, Inc.,* Third Edition, 1985.

[92] Ryszard Syski, "Introduction to Congestion Theory in Telephone Systems," *Elsevier Science Publishers B. V., Amsterdam,* second edition, 1986.

[93] *IEEE Communications Magazine*: Photonic Switching, vol.25, no.5, May 1987.

[94] *IEEE Journal on Selected Areas in Communications*: Photonic Switching, August 1988, to be published.

[95] Trudy E. Bell, "Optical Computing: A Field in Flux," *IEEE SPECTRUM,* volume 23, number 8, August 1986.

[96] Lynn D. Hutcheson, Paul Haugen, and Anis Husain "Optical interconnections replace hardwire," *IEEE SPECTRUM,* March 1987.

[97] T. G. Giallorenzi, J. A. Bucaro, A. Dandridge, and J. H. Cole, "Optical-fiber sensors challenge the competition," *IEEE SPECTRUM,* vol.23, no.9, September 1986.

[98] *Proceedings of the IEEE*: Special Issue on optical computing, vol.72, no.7, July 1984.

[99] Francis T. S. Yu, "Hybrid optical computing," *IEEE POTENTIALS,* December 1987.

[100] Stewart D. Personick, "Photonic Switching: Technology and Applications," *IEEE Communications Magazine,* vol.25, no.5, May 1987.

[101] M. Sakaguchi and K. Kaede, "Optical Switching Devices Technologies," *IEEE Communication Magazine,* vol.25, no.5, May 1987.

[102] M. Sakaguchi and K. Kaede, "Optical Switching Devices Technologies," in Proceedings of the *International Seminar on Digital Communications, New Directions in Switching and Networks,* Zürich, Switzerland, March 11-13, 1986.

[103] L. Thylén, "High Speed, Wide Bandwidth Digital Switching and Communications Utilizing Guided Wave Optics," in Proceedings of the *International Seminar on Digital Communications, New Directions in Switching and Networks,* Zürich, Switzerland, March 11-13, 1986.

[104] H. Goto, *et al.*, "An experiment on optical time-division digital switching using bistable laser diodes and optical switches," in Proceedings of the *IEEE GLOBECOM,* vol.2, November 1984.

[105] J. R. Erickson, R. A. Nordin, W. A. Payne, and M. T. Ratajack, "A 1.7 Gigabit-per-Second, Time-Multiplexed Photonic Switching Experiment," *IEEE Communications Magazine,* vol.25, no.5, May 1987.

[106] W. A. Payne and H. S. Hinton, "Design of Lithium Niobate Based Photonic Switching Systems," *IEEE Communication Magazine,* vol.25, no.5, May 1987.

[107] John E. Midwinter, "Optical Fibers for Transmission," *John Wiley & Sons, Inc.,* 1979.

[108] Behzad Moslehi, Joseph W. Goodman, Moshe Tur, and Herbert J. Shaw, "Fiber-Optic Lattice Signal Processing," *Proceedings of the IEEE,* vol.72, no.7, July 1984.

[109] P. Bruce Berra and Nikos B. Troullinos, "Optical Techniques and Data/Knowledge Base Machines," *IEEE COMPUTER,* October 1987.

[110] S. D. Smith, "An introduction to optically bistable devices and photonic logic," *Phil. Trans. R. Soc. Lond., Great Britain,* A-000-000, 1984.

[111] James D. Meindl, "Chips for Advanced Computing," *Scientific American,* vol.257, no.4, October 87.

[112] H. Scott Hinton, "Photonic Switching Technology Applications," *AT&T Technical Journal,* vol.66, issue 3, May/June 1987.

[113] P. W. Smith, "On the Physical Limits of Digital Optical Switching and Logic Elements," *The Bell System Technical Journal,* vol.61,no.8, October 1982.

[114] T. Sugeta, T. Mizutani, M. Ino, and S. Horiguchi, "High Speed Technology Comparison -GaAs vs Si-," *IEEE GaAs IC Symposium,* Grenelefe, Fl, October 28-30, 1986.

[115] Alan Huang, "Architectural Considerations Involved in the Design of an Optical Digital Computer," *Proceedings of the IEEE,* vol.72, no.7, July 1984.

[116] J. Goodman *et al.,* "Optical Interconnections for VLSI Systems," *Proceedings of IEEE,* July 1984.

[117] Richard A. Becker, Charles E. Woodward, Frederick J. Leonberger, and Richard C. Williamson, "Wide-Band Electrooptic Guided-Wave Analog-to-Digital Converters," *Proceedings of the IEEE,* vol.72, no.7, July 1984.

[118] P. J. Burke, "The Output of a Queueing System," *Operations Research,* **4**, pp.699-704, 1966.