Recent developments in optimization of the computer communication networks.

Galina Peyros
RECENT DEVELOPMENTS IN OPTIMIZATION
OF THE COMPUTER COMMUNICATION NETWORKS

by

Galina Peyros

A Thesis
Presented to the Graduate Committee
of Lehigh University
in Candidacy for the Degree of
Master of Science
in
Information Science

Lehigh University
1983
This thesis is accepted and approved in partial fulfillment of the requirements for the degree of Master of Science.

September 8, 1983
Date

Professor in Charge

Chairman of Department
ACKNOWLEDGMENTS

I am very grateful to Dr. Andrew J. Kasarda for his help and advice in writing this thesis.

I thank Western Electric for its financial support.
<table>
<thead>
<tr>
<th>TABLE OF CONTENTS</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table of Figures</td>
<td>vii</td>
</tr>
<tr>
<td>Abstract</td>
<td>1</td>
</tr>
<tr>
<td>1. Introduction</td>
<td>2</td>
</tr>
<tr>
<td>2. Survey of the Latest Research Areas</td>
<td>8</td>
</tr>
<tr>
<td>in the Computer Communication Networks</td>
<td></td>
</tr>
<tr>
<td>2.1 Congestion Control Issues in the</td>
<td>9</td>
</tr>
<tr>
<td>Computer Communication Networks</td>
<td></td>
</tr>
<tr>
<td>2.2 Routing Considerations in Computer</td>
<td>14</td>
</tr>
<tr>
<td>Communication Networks</td>
<td></td>
</tr>
<tr>
<td>2.3 Local Area Networks</td>
<td>16</td>
</tr>
<tr>
<td>2.4 Voice/Data Integration - Future of</td>
<td>17</td>
</tr>
<tr>
<td>the Computer Communications</td>
<td></td>
</tr>
<tr>
<td>2.5 Network Computers</td>
<td>19</td>
</tr>
<tr>
<td>3. Congestion Control Mechanisms for Computer</td>
<td>22</td>
</tr>
<tr>
<td>Networks</td>
<td></td>
</tr>
<tr>
<td>3.1 Composite Congestion Control Schemas</td>
<td>22</td>
</tr>
<tr>
<td>3.2 Window Flow Control</td>
<td>27</td>
</tr>
<tr>
<td>3.3 Analysis of a Feedback Scheme for</td>
<td>33</td>
</tr>
<tr>
<td>Congestion Control</td>
<td></td>
</tr>
<tr>
<td>4. Routing in Computer Communication</td>
<td>38</td>
</tr>
<tr>
<td>Networks</td>
<td></td>
</tr>
<tr>
<td>4.1 Dynamic Behavior of Shortest Path</td>
<td>38</td>
</tr>
<tr>
<td>Routing Algorithms for Communication Networks</td>
<td></td>
</tr>
</tbody>
</table>
4.2 Guided - Adaptive Routing
Techniques for Packet-Switching Networks

4.2.1 The Periodic Queue Exchange Algorithm

4.2.2 The Asynchronous Traffic - Splitting Algorithm

4.2.3 Performance Study

5. Local Area Networks - Performance Comparison

5.1 Performance Issues of Ring Techniques for the Local Area Networks

5.1.1 Token Ring

5.1.2 Slotted Ring

5.1.3 CSMA Collision - Detection Bus

5.1.4 MLMA Ordered - Access Bus

5.1.5 Performance Measurements Results

5.2 The Conflict-Free Protocols for Local Area Networks

5.2.1 Review of BRAM and MSAP

5.2.2 GBRAM

5.2.3 GBRAM Performance

6. Integrated Voice/Data Computer Communication Networks

6.1 Packet-Switching in Computer Communication Networks
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.2</td>
<td>Properties of VCP and Datagram Protocol in Both Data and Voice</td>
<td>79</td>
</tr>
<tr>
<td></td>
<td>Packet-Switching Node</td>
<td></td>
</tr>
<tr>
<td>6.3</td>
<td>Channel Capacity Utilization in Integrated Voice/Data</td>
<td>85</td>
</tr>
<tr>
<td></td>
<td>Communications Networks</td>
<td></td>
</tr>
<tr>
<td>7.</td>
<td>Allocation of Task Forces in the Computer Communication Networks</td>
<td>91</td>
</tr>
<tr>
<td>7.1</td>
<td>Wave Scheduling Technique for a Distributed Task Force Scheduling</td>
<td>91</td>
</tr>
<tr>
<td>7.1.1</td>
<td>Description of Wave Scheduling</td>
<td>93</td>
</tr>
<tr>
<td>7.1.2</td>
<td>Efficiency of Wave Scheduling</td>
<td>95</td>
</tr>
<tr>
<td>7.2</td>
<td>Task Allocation Scheme for Time-Critical Applications</td>
<td>96</td>
</tr>
<tr>
<td>7.2.1</td>
<td>Problem Description</td>
<td>98</td>
</tr>
<tr>
<td>7.2.2</td>
<td>Summary of the Theoretical Branch and Bound (BB) Method</td>
<td>100</td>
</tr>
<tr>
<td>7.2.3</td>
<td>Application</td>
<td>102</td>
</tr>
<tr>
<td>8.</td>
<td>Conclusions and Areas for Future Research</td>
<td>107</td>
</tr>
<tr>
<td></td>
<td>Bibliography</td>
<td>120</td>
</tr>
<tr>
<td></td>
<td>Vita</td>
<td>130</td>
</tr>
</tbody>
</table>
# TABLE OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>The Basic Structure of a Computer Communication Network</td>
<td>8</td>
</tr>
<tr>
<td>3.1</td>
<td>Configuration of Stage K</td>
<td>23</td>
</tr>
<tr>
<td>3.2</td>
<td>Queueing Model of Virtual Route</td>
<td>29</td>
</tr>
<tr>
<td>3.3</td>
<td>Power Curves for Finite Sources</td>
<td>31</td>
</tr>
<tr>
<td>3.4</td>
<td>Model for &quot;Half-Duplex Flip-Flop&quot; Terminals</td>
<td>32</td>
</tr>
<tr>
<td>3.5</td>
<td>Two Consecutive Nodes on a Communication Path</td>
<td>34</td>
</tr>
<tr>
<td>3.6</td>
<td>Decision Curves Vs. Buffer Occupancy</td>
<td>36</td>
</tr>
<tr>
<td>4.1</td>
<td>Design Issues for Adaptive Routing Algorithm</td>
<td>43</td>
</tr>
<tr>
<td>4.2</td>
<td>Main Table for ATS Using Two Alternative Routes</td>
<td>52</td>
</tr>
<tr>
<td>4.3</td>
<td>Average Delays for Different Routing Algorithms</td>
<td>54</td>
</tr>
<tr>
<td>4.4</td>
<td>Values of Average Delays Relative to Optimum Delays</td>
<td>57</td>
</tr>
<tr>
<td>5.1</td>
<td>Token-ring Model</td>
<td>61</td>
</tr>
<tr>
<td>5.2</td>
<td>Slotted-ring Model</td>
<td>63</td>
</tr>
<tr>
<td>5.3</td>
<td>Transfer Delay-Throughput Characteristics of the Four Subnetworks at 1Mbit/sec</td>
<td>66</td>
</tr>
<tr>
<td>5.4</td>
<td>Transfer Delay-Throughput Characteristics of the Four Systems at 10 Mbit/sec</td>
<td>68</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Figure 5.5</td>
<td>The Throughput-Delay Performance of GBRAM</td>
<td>75</td>
</tr>
<tr>
<td>Figure 5.6</td>
<td>Comparison of Offered Load Versus Throughput Between GBRAM and Other Protocols</td>
<td>76</td>
</tr>
<tr>
<td>Figure 6.1</td>
<td>Network of Cascaded Nodes</td>
<td>80</td>
</tr>
<tr>
<td>Figure 6.2</td>
<td>Data Packet Delay</td>
<td>82</td>
</tr>
<tr>
<td>Figure 6.3</td>
<td>Data Packet Delay</td>
<td>82</td>
</tr>
<tr>
<td>Figure 6.4</td>
<td>Voice Packet Delay</td>
<td>84</td>
</tr>
<tr>
<td>Figure 6.5</td>
<td>Voice Packet Delay</td>
<td>84</td>
</tr>
<tr>
<td>Figure 6.6</td>
<td>Integrated Digital Voice/Data System</td>
<td>87</td>
</tr>
<tr>
<td>Figure 7.1</td>
<td>Allocation to Achieve Lower IPC and Waiting Time</td>
<td>103</td>
</tr>
<tr>
<td>Figure 7.2</td>
<td>Allocation to Achieve Lower IPC, but Higher Waiting Time</td>
<td>103</td>
</tr>
<tr>
<td>Figure 7.3</td>
<td>Average Waiting Time for P1</td>
<td>105</td>
</tr>
<tr>
<td>Figure 7.4</td>
<td>PTP Time for Thread Containing Tasks</td>
<td>105</td>
</tr>
</tbody>
</table>

- viii -
ABSTRACT

Recent accomplishments and developments in the area of optimization of computer communication networks are described, analyzed and criticized. Experimental and simulation results are presented. A large number of articles is surveyed, numerous examples and case studies are considered.

Research focuses of five topics in the Computer Communication area: Congestion Control in Computer Communication Networks; Optimization of Routing Algorithms; features and performance issues of the Local Area Networks; integration of voice and data over common Communication Network; resource allocation and task scheduling in the Network Computers.

The performance of the considered algorithms is analyzed in terms of delay-throughput characteristics, total system throughput, average network delay, power and blocking probabilities. For routing algorithms such characteristics as stability, traffic types, convergence and speed of convergence are of interest.
CHAPTER 1

INTRODUCTION

Communication systems starting from telegraph and telephone have entered the era of Computer Communication Systems. The growing economic, social and political requirements coupled with the technological advances are leading to the development of large information processing time shared systems and data bases. The invention of sophisticated powerful switching computers is the spirit behind the integration of communication and computer technologies into a common technology called computer communication. As a result, cost effective computer communication networks are proliferating to provide accession of data bases at various information centers to each host computer in the network.

In this thesis, recent accomplishments and developments in the area of optimization of computer communication networks are described, analyzed and criticized. Wherever available, the experimental or simulation results are presented.
Research concentrates on five major aspects of the computer networks, that received a great deal of attention recently and represent a tremendous potential for the future research.

A large number of articles are surveyed, numerous examples and case studies are considered.

The five major topics that are explored in this thesis are: the congestion control for computer communication networks; the optimization of routing algorithms in the computer networks; the features and performance issues of the local area networks; the integration of voice and data in the same network; the resource allocation and task scheduling in the computer communication networks.

The remainder of this thesis is organized as follows:

Chapter 2 surveys the recent developments and raises issues in each of the five areas mentioned above. It also gives an overview of network types that will be considered.

Chapter 3 focuses on specifics of the congestion control problems and recent optimization methods.
A composite congestion control scheme adopting isarithmic control and input buffer limiting control is analyzed and testing results are presented.

Until recently, virtual routes with window control have been modeled as loss system. This has led to a good representation of transit network performance measures (e.g., total throughout), but has not yielded results on admission delays. In Chapter 3, a queueing analysis of these admission delays, based on decomposition is given. Simple models yielding to an effective computational evaluation are surveyed and evaluated. Also, the performance of a feedback congestion control scheme is evaluated, and implications of using this mechanism in practical applications are discussed.

Chapter 4 surveys and discusses several proposed routing algorithms for the computer communication networks. One of them is the shortest path algorithm, currently in operation in the ARPANET, routes messages along shortest path computed by using some set of link length. When these lengths depend on current traffic conditions as they must in an adaptive algorithm, dynamic behavior questions such as stability, convergence, and speed of convergence are of interest.
Also issues involved in designing adaptive routing techniques are discussed. Two new guided-adaptive routing techniques are investigated: Periodic Queue Exchange and Asynchronous Traffic Splitting. Both techniques use more than one route between each source/destination pair selected carefully to avoid routing conflicts and to allow proper distribution of the traffic on the available links, especially the critical links of the network. The new algorithms have excellent performance compared to the ARPANET routing algorithm and the optimal bifurcated routing algorithm.

Chapter 5 focuses on performance and optimization issues of the local area networks. This chapter provides a comparative evaluation of the performance of ring and bus systems constituting subnetworks of local-area network. Performance is described in terms of the delay-throughput characteristics. System investigated include token-controlled and slotted rings as well as random-access busses (CSMA with collision detection) and ordered access busses (MLMA).

Also an analysis of a local network media access protocol called GBRAM is given. This protocol is an extension of BRAM and MSAP, which are also discussed.
The GBRAM is decentralized and conflict-free, and is suitable for a common channel local network with a large number of users. Several performance issues are discussed in this chapter: the throughput-delay relationship, the channel load versus throughput, the selection of optimal grouping, and the performance of an "asynchronous" GBRAM.

Chapter 6 investigates problems related to sharing the resources in an integrated voice/data packet-switching network. The four major alternatives for integrating voice and data are described and analyzed: circuit-switching, packet-switching, hybrid-switching, and message-switching. Two control schemes - Virtual Circuit Protocol (VCP) and Datagram Protocol - are analyzed for performance.

The Store-and-forward buffer design problem in an integrated digital voice-data system is described. Finally, the buffer behavior in terms of overflow probability and average queuing delay is studied for different values of traffic intensity and buffer length.
Chapter 7 deals with the resource allocation and task scheduling in the computer networks. To be general purpose problem solvers in a variety of environments, network computers need distributed procedures for scheduling competing task forces. In Chapter 7, the new Wave Scheduling technique is described and analyzed. The new technique distributes task force scheduling by recursively subdividing and issuing wavefront-like requests to worker nodes capable of executing user tasks. It uses a hierarchical high-level operating system control structure to partition competing task forces among nodes in any network structure. Also a task allocation model designed for distributed computing system in time-critical applications is discussed.

Chapter 8 presents the conclusions and identifies future research areas in each of the discussed fields.
SURVEY OF THE LATEST RESEARCH AREAS IN
THE COMPUTER COMMUNICATION NETWORKS

In the last decade, we have seen the development of a number of computer communication networks. Typical examples are the ARPANET (Advanced Research Projects Agency), network in the USA [1], and the DATAPAC network in Canada [2]. Such developments have made it possible for a user to access resources such as hardware units, software packages and data files in a remote computer system.

The basic structure of a computer communication network can be shown in the following picture:

![Diagram of a computer communication network]

Figure 2.1 The Basic Structure of a Computer Communication Network
It can be portioned into a communication network and a user-resource network. The communication network consists of the switching computers (or nodes) and the communication channels. Its function is to deliver messages from one node to another. The collection of terminals and computing resources forms the user-resource network. These resources are connected to the switching nodes and communicate with each other via the communication network.

2.1 Congestion Control Issues in the Computer Communication Networks

Any system sharing resources has a finite processing capacity, so if it is highly utilized, congestion must be inevitable. But congestion causes excessive delay, degradation of throughput and possibly, what is worse, system deadlock. System deadlock is a situation where each component of system is blocked and gives up operating. Flow control is a system of algorithm which prevents congestion, and is classified into two categories, that is, routing control and congestion control. The former is a scheme determining customer’s traversable route dynamically or statically and aims at attaining maximum throughput or
minimum delay. However, this scheme is powerful only at light traffic load, and its usefulness degrades as traffic load level becomes higher. Because of their high sensitivity to traffic intensity along every communication link in the network adaptive, dynamic or semidynamic routing techniques are very difficult to implement [3]. They require a great deal of processing, and many control messages will be circulating in the network before a stable state could be reached. This stable, or steady state, must be reached quickly enough for the network to perform satisfactory [4]; this is sometimes extremely difficult to accomplish. Also, recent results in the literature [5] have shown that dynamic routing improves the delay performance of the network by about only 10 percent, and should dynamic routing be used at all, it must be implemented with great care. Recent developments in routing techniques are discussed in Chapter 4.

Congestion control scheme is a scheme which is devised to keep a system operating efficiently in case some portion of offered load must be lost. It has merits, that is, prevention from an increase of
delay and a decrease of throughput, reducing the possibility of deadlock and so on. Congestion control schemes being used in practical situations are as follows [6]:

(1) Local Control: This scheme is used on the basis of informations concerning local traffic data and immediate neighbor's states. For example, if a stage keeps being busy without stopping for prescribed interval, an input is rejected. From the microscopic standpoint, it may prevent local congestion, but it is useless for global congestion. What is worse, if local congestion is caused by users remote from the congested point, it is still less effective.

(2) Central Control: A whole network system is overviewed by a central controller. However this scheme introduces additional traffic in gathering information from local users and announcing the decisions from the center. So possibly, it may not operate the system efficiently.

(3) Congestion Control Table Scheme [7]: Congestion control table containing the data about the whole network circulates in the network. Each user selects the information necessary for its own decision and revised the table. This scheme also increases loads.
(4) Hop Priority Technique: Each job is assigned priority according to the number of nodes which it has passed. Therefore the longer job proceeds, the higher priority it has.

(5) End-to-End Control (Fixed Window): The maximum number of acceptable jobs is predetermined for each origin-destination pair group. When jobs in the network are saturated to a limit with respect to some pair, incoming jobs in the same group are rejected to enter. The stochastic behaviors of networks with this scheme are well analyzed through a closed queueing network theory. Such control scheme was used in TYMNET and is well analyzed in [8] and [5].

(6) Input Buffer Limiting Control [9] [10]: External jobs and transit jobs are separated at each stage. The former first enters input buffers where maximum possible queue length is limited to a finite level and may be rejected because of lack of holding space. If this scheme is used without any other scheme, this is not effective.

(7) Link Buffer Limiting Control: At each stage, the maximum possible queue length of transit jobs plus accepted input jobs are restricted. But with this
scheme blocking may occur [11]. Blocking is a phenomenon that after the completion of a service a transit job cannot enter the next immediate neighboring stage because there is no buffer available for it. Blocking decreases service efficiency and causes system deadlock if it spreads over the network. Therefore this scheme must be incorporated with other schemes.

(8) Isaritmic Control: Fixed number of permits circulate according to the rule in a network. External jobs first join an input buffer queue and reach available permits (accompanying no job) passing through source stage from the head of queue. After receiving a permit a job is able to proceed through the network to a destination stage.

Chapter 3 focuses on detailed description and analysis of different congestion control schemes, that received a great deal of attention recently. Such schemes are analyzed as the composite congestion control schemes, window flow control scheme and a feedback scheme.
2.2 Routing Considerations in Computer Communication Networks

A central operational problem of a communication network involves the choice of routes used by messages to travel from origin to destination. It is possible of course, to choose a fixed route for each origin-destination pair, but this precludes the possibility of adjusting routes to alleviate congestion due to variation in average traffic conditions.

A routing procedure in general is a decision rule to determine the next node that a message will visit in its way through the network from source to destination. A good routing procedure is essential for the successful operation of a computer network. A poor routing procedure could cause inefficient utilization of network resources and excessive delay for messages.

Routing techniques as it was mentioned before are divided into two main categories: deterministic and adaptive. Adaptive routing is suitable for unknown and unpredictable traffic environments [12] and is characterized by the existence of an updating process which enables the network to adapt both to the rapid
changes in traffic load and to the rather slow changes in topology. Deterministic routing procedures route packets in fixed predetermined routes. These procedures do not adapt to changing traffic pattern.

For this reason attention has focused on adaptive routing strategies whereby congestion in the network is continuously monitored and routes between origin-destination pairs are modified in real time so as to keep average delay per message at a reasonable level. A routing scheme of this type was implemented in the ARPANET in 1969 and attracted considerable attention, [13] [14].

The routing algorithms used in a number of operating networks (ARPANET, TYMNET, TRANSPAC) as well as in commercial network architectures (IBM SNA, DEC DNA) all turn out to be variants, in one form or another, of shortest path algorithms that route packets from source to destination over a path of least cost [15]. Interestingly, shortest path single routes produced by these routing procedures turn out not to be optimum if the main objective is to minimize the average end-to-end message delay [15].
Chapter 4 presents and analyzes different techniques that are used to find the shortest and the most efficient path in the network.

2.3 Local Area Networks

In the past few years, the need has increased for computer systems with high performance and reliability within a local environment. Such areas of application include office and laboratory use as well as university and industrial use. The existing computer capacity should thus be made available to a larger number of users. It is also become desirable to exchange with other users data and programs at a high speed. These requirements are fulfilled by local area networks, in which not only hardware is physically distributed, but also processing and control are completely decentralized. Local computer networks vary mainly with respect to their architecture and to the method of transmission. The most common network topologies are star-, ring-, and linear bussystems. Controversial arguments are used in the discussion about the qualities of the different systems. As pointed out in [16], in particular the performance issues of ring techniques for the local area do not seem to be well understood, neither is their relation to the
performance of local-area bus systems. As it is pointed out in [17], in the local area, subnetworks like rings and busses which operate in a broadcast fashion represent attractive alternatives to store-and-forward and circuit-switching networks. Since for the time required to transmit a packet the whole transmission medium (or at least a significant portion of it) is allocated to one static, a suitable method to resolve access conflicts has to be used. There are two basic categories of access schemas [18]: ordered access and random access. In case of ordered access, a control scheme is employed which resolves access conflicts in such a way that no packet collisions occur on the transmission medium, whereas collisions are possible in systems using random access.

Performance issues related specifically to local area networks are discussed in Chapter 5.

2.4 Voice/Data Integration - Future of the Computer Communications

The integrated voice and data traffic transmission with the evolution of digital communication networks has received a great deal of attention recently [19, 20]. As voice digitization technology and the packet-switching techniques improve,
the inherent cost savings of integrating voice and data become feasible and more attention is focused on the problem by the agencies operating communications networks which carry the two traffic types. As it is identified in [21], many studies so far have put all the effort on the four major alternatives for integrating voice and data:

(1) Circuit-Switching: The traditional switching scheme for telephony which is also relatively useful for bulk traffic. Due to long circuit set up time, this method is not efficient for interactive data transmission.

(2) Packet-Switching: An efficient switching scheme for interactive as well as bulk type traffic. With additional efforts for constructing appropriate protocol structures, this switching technique happens to be very attractive for digital voice transmission in the form of packets as well [22, 23]. Packet-switching technology [24, 25] achieves the effective sharing of expensive communication resources for the performance and cost optimization. It is expected that this will lead to the integration of data and speech messages over a common computer communication network.
Hybrid-Switching: Operating cost of a communication network is naturally a dominant factor in pursuing different design technologies for the particular application. It has become evident that [19], circuit switching is most cost effective when used for traffic requiring high bandwidth, but not so efficient for traffic characterized in bursts. In this latter case, packet-switching techniques then provide a desirable alternative for integrating data and voice.

Message-Switching: Although this approach may only represent an efficient technique in specific applications, recent studies [26, 27] have shown its usefulness when implemented through intelligent control and decision making processes at communication nodes. Some studies suggest its applicability in integrated data and voice environments.

Issues related to data/voice switching in computer communication networks are discussed in Chapter 6.

2.5 Network Computers

The explosive rate of progress of microprocessor technology has made the distributed computing networks economically attractive for many
computer applications. However, use of distributed computing networks requires effective distribution of application tasks in a network topology. The effective task distribution would maximize system throughput by minimizing the time and cost of computer communication.

A network computer is a general purpose computer build from a collection of independent, asynchronously executing, and loosely-coupled microcomputers. Each processing element, node, consists of at least one CPU attached to a local RAM which contains instructions and data. The memory of one node is not directly accessible by any other node. Moderately wide-band connections, currently on the order of 1 Mbyte/sec, link each node to a limited number of neighboring nodes, often forming the basis for a packet-switching communications subnetwork. Communication between nodes is performed by message passing. The objective of such a machine is to provide a facility for taking advantage of parallel algorithms in numerical analysis, artificial intelligence, simulation, and many other application.

A general purpose network computer with hundreds of thousands of nodes will be a very complex computing ensemble, regardless of the simplicity of
processor interconnection in local regions. This complexity is most noticeable with respect to the scheduling of cooperating tasks for execution by the computer. Collections of cooperating tasks are known as task forces [28]. To be useful as general purpose problem solvers in a variety of environments, network computers need procedures for selecting competing task forces and assigning them to available processors.

However, to achieve the goals of fault-tolerance and extensibility, the procedures must be distributed in such a way that local scheduling decisions are the norm, with centralized control held to a minimum, or eliminated completely. Such requirements are distinctly different from those which hold in conventional uniprocessors and call for radically different scheduling techniques. Those techniques, in particular, wave scheduling, and techniques applicable in the time-critical environment, are discussed in details in Chapter 7.
Flow and congestion control problems have been identified as one of the important issues in the design of computer communication networks. Networks with no control imposed on the use of their resources have been shown to exhibit a throughput-delay relation similar to contention system. That is, the throughput increases with the applied load up to some maximum (optimum) value, beyond which, due to unpredictable behavior by users and additional user-user and user-server interaction and overhead, more load causes reduction in throughput. The physical reason behind this phenomenon is that when a network is overloaded, its resources are either wasted and not utilized for useful purposes or processors may be blocked. In any case, work is not conserved and throughput degradation occurs.

3.1 Composite Congestion Control Schemas

Multiple level flow controls have been studied recently. Two level flow control scheme is proposed [29], where a limit is set on the total number of jobs in the network and separate limits are set on the number of jobs belonging to each group of source-
destination pairs. Moreover three level flow control scheme is given [30], where in addition to the limits of two level flow control scheme each stage blocks all the jobs from neighboring stages if a threshold is exceeded. In [6] a composite congestion control scheme is proposed an analyzed. In this scheme isaritmic control incorporated with input buffer limiting control. This scheme can be illustrated as follows. Each stage consists of two substages, that is one is the link substage and the other is the input substage as can be shown in following figure:

![Configuration of stage k](image)

*Figure 3.1 Configuration of stage k*
External job can join the input buffer queue at their stage of origin when the queue length on their arrival is less than the capacity of the buffer. When an overflow occur the overflown jobs will be rejected and lost. From the head of the input queues, each accepted job catches a permit if one is available at the permit queue and then goes into the link buffer queue to wait to be served by a link server. If there is no available permit at the permit queue, the accepted jobs in the input buffer must wait for permits coming to this stage. After completion of service, the job leaves the system and moves to an immediate neighboring stage according to predefined probabilities. On leaving a job at its destination stage, the permit accompanying the job goes to the permit queue at a particular stage. Permit returning rules can return permit to the original stage of each permit (fixed permits), to a stage at random (free permits), to the most congested stage and so on.

As for network performance, the following characteristics were estimated:
(1) Average network delay: Delay experienced by a job consists of admission delay and link delay. The former is the waiting time at the input buffers and the latter is the trading time among the link. Admission delay of a stage can be calculated easily by adding of stage probabilities of input substages. Average link delay was calculated in the model as: [number of permits (equal to active jobs) in the link] / (accepted traffic rate), and average delay = (average admission delay) + (average link delay). And average network delay is a weighted average of each class job's averaged by throughput.

(2) Throughput: It was defined that throughput is the accepted input at each stage. And the total network throughput is the sum of throughput at each stage. From the simulation model was found that total throughput increases according to arrival rate and number of permits. In fact, the more permits there are, the more easily jobs are accepted and subsequently, greater throughput is achieved. However, it is unavoidable that more throughput causes more network delay. Throughput and delay are contrary to each other.
(3) Power: An interesting performance ratio was introduced in [29], which is called power. It compromises two extreme performance measures, that is, a throughput and delay into the following simple measure,

\[
\text{power} = \frac{\text{total throughput}}{\text{average network delay}}
\]

It was shown that when input buffer site is 0 and number of permits is 1, it seems to be better setting than the other as a whole. However this selection causes unacceptable blocking when the arrival rate increases.

(4) Blocking Probability: Network blocking probability \( P \) is obtained from following relation:

\[
\sum_{i=1}^{M} \lambda^{(i)} (1 - P_0) = \Gamma, \quad \text{where}
\]

\( M \) - total number of stages in the network

\( \Gamma \) - total throughput

Changing the approach it can be found such \( N \) which makes power high or attained an aimed throughput keeping blocking probability low. If the number of permits is fixed, delay gets longer with the size of input buffers because of an increase in admission delay. On the other hand, throughput hardly increases, and power is dominated mainly by delay. Moreover, it
was shown, that under the same parameters, free permit routing scheme where permits are sent back to the most congested stage dynamically, does not improve system parameters remarkably.

3.2 Window Flow Control

One flow control protocol which enjoys great popularity due to its simple implementation is the window protocol. Many existing networks and architectures use this protocol. (e.g. ARPANET [1], [4], X.25 [30], SNA [31], etc.)

The basis for window control is the virtual connection, i.e., an end-to-end protocol which establishes a protected point-to-point connection. An example of an architecture which embodies flow control through windows and virtual routes is given by SNA [31, 32].

M. Reiser modeled a network with various virtual routes employing window flow control by means of queuing network model with multiple closed chains [11, 33]. However, the closed-chain approach required the assumption of a less system, i.e., messages which arrive when the virtual route blocks admission to the
transit network are lost. Contrarily, real data networks are all wait systems, i.e., messages not admitted are queued at the first network node. Such kind of networks is analyzed in [11]. First the transit networks are solved in [11], assuming saturated load for all possible window assignments (this can be done with the method described in [33]). Then each virtual route is modeled individually, replacing network with a "Northon equivalent server" [34]. This method was first used by Schwartz [35] who established its validity through extensive simulation. The model considered in [11] is the following: Traffic originates from a set of terminals, say, which are controlled by a remote control unit (called TIP in ARPANET and Cluster Controller in SNA). This control unit enables the terminals through poll commands, and schedules traffic through the terminal controller with a host through the transit network. Terminals are assigned to a virtual route which connects the terminal controller with a host through the transit network. The virtual route dispatches traffic onto an explicit route assumed to be fixed. The virtual route employs a sliding window which can be described as follows: A
pool of K permits is kept by the sending end of the virtual route. As a message enters, it obtains a permit and is dispatched. The permit is removed from the pool. If no permit is available, the message waits (the virtual route is said to be blocked). As messages arrive to the destination node, acknowledgements are returned to the sender. It is assumed that each message is individually acknowledged and that acknowledgements are transmitted with higher priority than messages. As acknowledgements arrive at the source node, permits are returned to the send pool. Evidently, the total number of messages and permits is at all times fixed to K. Thus, the virtual route can be modeled as closed queueing network. Figure 3.2 shows the basic queueing model.

Figure 3.2 Queueing Model of Virtual Route
The decision box lets a message proceed $j > 0$, i.e. 
$i \to i-1$, $j \to j-1$, $k \to k + 1$,

Where $K$: window size  
$N$: number of sources (terminals)  
i: number of messages waiting for permits  
$K$: number of messages in transit  
j = K-k: number of permits  
$\phi$: delay of acknowledgements over return chain

From the analyzing the data interesting conclusions were made. First it was observed that the graph $P_0$ (optimum power) vs. $K$ is monotonically increasing. Thus, the virtual route has not distinct window maximizes the virtual route power, as was the case with the loss model. From the virtual-route point of view, the larger the window the better. This is not surprising. However, the network need is a window as small as possible in order to minimize congestion. A compromise can be defined at the "knee" of the curve $P_0$ vs. $K$. The data of Figure 3.3, argumented by many more cases which were solved, provides the following interesting results:

- The "knee" of the curve $P_0$ vs. $K$ is located at the window size $K_0 = 2L$. Thus, a window yielding near-
maximum VR power is given by $K_0 = 2L$.

- The optimum power $P_0$ is achieved at the load

$$\lambda = 0.5 \text{ (units such that } \pi, l = 1, 2, \ldots, L)$$

Figure 3.3  Power curves for Finite Sources (N=10).

$P_0$ is defined as $\max_\lambda \{P(\lambda)\}$

The result contrasts with earlier results [35] which give the window minimizing VR power for the loss model as $K_0 = L-1$. It is very difficult to minimize VP power, but it is possible to arrive at a good compromise between the network and the virtual-route point of view at $K_0 = 2L$ which compares to $K_0 = L-1$ for the loss model. The value $K_0 = 2L$ was suggested by Deaton [36] on other heuristic grounds.
It is not difficult to consider the effect of end-to-end delays on the virtual route performance. In this case a terminal gets polled, but that once a message is sent the terminal has to wait for a reply from the host (even though it may get polled in the interim). This mode of conversation is called "half-duplex flip-flop" [37]. Its queueing model is quite simple and can be easily solved with standard methods [38]. The graphical representation of this model is given in figure 3.4.

![Figure 3.4 Model for "Half-Duplex Flip-Flop" terminals](image-url)
3.3 Analysis of a Feedback Scheme for Congestion Control

There are basically two classes of congestion controls: global vs. local. Global congestion control schemas are the ones which stop input to the network as a means to prevent the network from reaching a congested stage. All considered before congestion control schemas are examples of global control schemas.

Local congestion controls, are the ones which are carried out by individual nodes of a network based on their own traffic data and resource utilization. Basically, when a node is short of resources, it stops accepting any more traffic from its neighbors as well as from external sources. In [39] a local control scheme, referred as feedback congestion control (FDBKCC), is analyzed and its implications are studied. This control mechanism is suitable for virtual circuit networks with fixed routing.

The basic idea behind the feedback congestion control is to enable each node of a network to control the rate of traffic which it receives from its neighbor
nodes. In VC networks this control is imposed on virtual circuits which pass through a node to limit throughput of individual VC's out of the node.

There is a control table at each node which has $I + 1$ entries, which are referred by $T(i), 0 \leq i \leq I$. The entries are such that $T(1) < T(2) < \ldots < T(I)$. Associated with a virtual route, $j$, which passes through the node there is a pointer $p_j$, $0 \leq p_j \leq I$. The entry $T(p_j)$ associated with VC$j$ determines the time gap between successive messages of VC$j$ which are sent out to the next node on VC$j$'s path. Schematically this mechanism could be shown in figure 3.5.

![Figure 3.5 Two consecutive nodes on a communication path.](image)

Figure 3.5 Two consecutive nodes on a communication path.
Occasionally, each node sends S control messages to its neighbor nodes. The message contains a field for each VC which leaves the neighbor node and passes through this node. Based on the resource availability at a node, the control field contains -1, 0, or 1. When a VC receives a -1 it reduces its pointer to the control table by 1, hence reducing its inter-packet gaps and increasing its output rate. A 0 result in no change and a 1 results in increasing the pointer by one and reducing the output rate. It is necessary to point out that control messages can be piggy backed to outgoing messages to reduce the overhead. The above mechanism is a powerful tool to divert traffic from a congested node and, more generally, from a congested region of a network. If the congestion is temporary, this control is sufficient to keep the load in nodes within a desired limit.

Aside from local congestion control, this mechanism can be used in networks which provide some sort of guaranteed services in terms of network delay. By choosing proper values for buffer site at each queue one can restrict the model delays to certain limits. This in turn establishes an upper bound on the network delay.
In [39] it was revealed two basic drawbacks of FDBKCC, namely:

(1) The dependency of the optimal control signals in a node on the control table pointer at its neighbor nodes. This is a result of system state specification, \((n, i)\), where \(n\) - number of messages at a given node and \(i\) — pointer into control table.

(2) The dependency of optimal control signals on the parameter \(\lambda\), where \(\lambda\) is a control factor and is defined as: \(T(i) = \lambda^{-i}\) \(0 \leq i \leq I\), and \(\lambda > 1\) (Figure 3.6).

![Figure 3.6 Decision curves vs. buffer occupancy for varying \(\lambda\): \(\lambda = 8, \gamma = 4, j = 40\) (msg/sec); \(B=I=10\).

curve A: \(\lambda = 1.25\), B: \(\lambda = 1.5\), C: \(\lambda = 2\), D: \(\lambda = 3\), E: \(\lambda = 4\)
With respect to 1 one can provide a field in the message headers to pass the necessary information to the adjacent nodes; an approach introduces some overhead. Another alternative is a heuristic solution which does not require the value of the control table pointer to determine the optimal control signals. Research in this direction now in progress.

The second drawback seems to be more serious.
CHAPTER 4

ROUTING IN COMPUTER COMMUNICATION NETWORKS

The design of an optimal routing scheme to implement and maintain the minimum message delay behavior is difficult, with the network suffering from topological changes and the traffic exhibiting statistical non-stationarity due to user trends, e.g., peak and off-peak loads. The optimal routing scheme must therefore be capable of adapting to any conditions experienced.

Adaptive routing schemas provide a mechanism which can carry out routing decisions on the basis of feedback from the network, reacting to congestion and facility failures with intelligently modified routing actions.

4.1 Dynamic Behavior of Shortest Path Routing Algorithms for Communication Networks

The implementation of the minimum delay path idea in the original ARPANET algorithm had a number of flaws allowing for example, the formation of loops. For this reason, alternative schemes based on the same idea were studied, and a new algorithm called SPF has been developed and implemented [40, 14, 41].
In [42] author focuses not on the ARPANET and the SPF algorithm in particular, but rather is geared towards understanding the effect of feedback and the nature of the dynamic behavior of shortest path algorithms where link lengths depends on current traffic conditions. It is necessary to note that algorithms proposed in [42] are far from optimal since they are single path algorithms in the sense that at any given time there is only one path per origin-destination pair along which messages can travel. Better performance can be achieved by allowing multiple path as for example in the optimization algorithm of Gallager [43] or its second derivative versions [44, 45]. Very interesting algorithm of optimum routing based on shortest path generation have been given recently in [46].

On the other hand, the hardware limitations of the most presently existing networks preclude the use of such more sophisticated algorithms.

In [42] author focuses on analysis of dynamic behavior of routing algorithm that has been successfully implemented in a network as interesting and influential as the ARPANET. This is reinforced by the fact that the behavior exhibited by the algorithm
is quite interesting and can pose nontrivial design problems.

In [42] author provides a complex mathematical and logical study of a deterministic finite-state Markov chain framework just for a simple version of the algorithm. He shows that for a ring networks the algorithm may tend to oscillate between poor routing path and become itself a major contributor to congestion. He also demonstrates how the use of a bias factor can provide a mechanism for damping oscillations as confirmed by experience with the original ARPANET algorithm.

The finite-state model does not lend itself to analysis of more sophisticated routing schemes and more general network topologies. Therefore later author introduces a model of a ring network with a continuum of nodes and a single destination. This allows to employ techniques of stability analysis of discrete-time systems with continuous state space, and enables to further quantify the relationship between choice of link lengths and algorithmic behavior.

Also author shows that oscillations can also be damped effectively by making the link lengths dependent on several preceding routing paths via some averaging
mechanism such as an exponential fading memory scheme or asynchronous link length updating. It is now believed that the significant degree of averaging inherently present in the SPF algorithm is in large measure responsible for the stable dynamic behavior observed in experiments conducted thus far [41].

The ring topology is central for the extension of all results to more complex network topologies. This extension is discussed in [42] under the assumption that an equilibrium routing exists. However, by contrast with ring networks, an equilibrium routing need not always exist for more complex topologies. It was demonstrated via example the mechanism by which such a phenomenon can occur.

The results and analysis of the named paper can be generalized to the case where there are more than one destinations. This analysis is straightforward, but considerably more complex technically and may be found in [47]. The validity of found in [14] results have been proven valid for finite-mode networks by extensive computational experimentation, the results of which are given in [47] and [14]. In particular, the validity of qualitative results regarding the role of a
bias factor and averaging as damping mechanisms have been demonstrated.

4.2 Guided-Adaptive Routing Techniques for Packet-Switching Networks

Routing schemes that adapt to change in traffic and topology by switching from one route to another are called guided-adaptive. Guided-adaptive technique have the advantage of using simple mechanisms which exchange a small amount of information for adaptation, thus keeping the overhead of route adapting reasonable. Due to the constrained adaptation of guided-adaptive techniques, the distribution of traffic flow over different links can be estimated. This offers the advantage of a relatively easier and more tradable analytical approach to the evaluation of performance, task that proved to be formidable for other adaptive techniques.

The adaptive routing techniques, routing of packets is adjusted to traffic fluctuations and to node and channel failures. A process called "updating" makes this adjustment possible. The detailed design issues for adaptation and updating according to [48] are illustrated in fig. 4.1.
Fig. 4.1 Design Issues for Adaptive Routing Algorithm
A. **Objective of Updating**

One of the main objectives of updating is determining network connectivity. In addition, the routing technique can be required to support interactive traffic, high-throughput traffic or real-time traffic. In case of interactive traffic the main objective is to minimize delay experienced by packets in their way from source to destination. Reaching this objective may require a scheme that gives priority to packets which have stayed more than a specified time limit in route [49].

In case of high-throughput traffic the main objective is to maximize the throughput.

In case of real-time traffic the objective is to compromise between the previous two objectives.

Other possible objectives can be the minimization of cost or the maximization of reliability.

The objective of updating determines, to a large extent, the kind of information to be collected for adaptation.
B. How Updating is Performed

According to [48], there are three main strategies for performing the updating process, local, distributed and centralized. Local techniques operate independently within each node, using only locally available dates for routing packets. Each node has access only to the information from the packets flowing through it. No explicit information about traffic is exchanged between nodes.

In distributed techniques, nodes cooperate in the route selection by exchanging special packets called "routing update packets" containing information about the network status (traffic load and topology changes). All the nodes are involved in the decisions that concern routing after they process the available information.

In centralized techniques, a center (or centers) receives routing information from other nodes in special packets for updating. The control center(s) processes the available information and dictates the routing decision to other nodes.
C. Type of Adaptation

As it is mentioned in [48], there are two types of adaptation: guided and free. In guided-adaptive techniques, multiple routes are specified at each node. The available information for adaptation is used in switching from one route to another. In free-adaptive techniques, available updating information is processed and is used in choosing the routes to every destination.

D. When Updating is Performed

The frequency of performing the updating process has to be adjusted to detect important changes in the network. Updating can be worked either periodically or asynchronously.

A natural way to detect network changes is to invoke the updating process periodically. The period may be fixed or adjustable. One reason to use an adjustable period [50] is to speed the propagation of packets when the nodes and channels can afford additional overhead in the case of light loads. The frequency of updating may vary from one link to another.
The updating process can be invoked by some event (asynchronously) such as a line failure or a traffic estimate that exceeds a threshold level.

E. Techniques for Reducing Routing Overhead

The storage requirements of routing tables and updating these tables represents an overhead that increases with the number of nodes in the network. Thus overhead reduction techniques are important for large networks. In general, there are two policies for reducing routing overhead for such large networks.

The first policy is based on clustering. In this policy, the size of the routing tables is reduced by keeping in every node complete information about nodes which are close to it in terms of a specified nearness measure such as the hop distance, and lesser information about nodes located farther away. This can be realized by providing one entry per destination for the closer nodes and one entry per set of destinations for the remote nodes [51].

The second policy is based on favoritism. Real measurements on operational network syndicate [52] that nodes prefer to send their traffic to a limited number of nodes for a relatively long period of time.
This is the motivation, a technique based on this phenomenon [48]. Users in computer centers prefer other centers as their favorite destinations since these centers are usually specialized in hardware or software applications suitable for a limited number of subscribers.

Favoritism in large network can be used to an advantage for overhead reduction. Complete routing tables need not be stored in the node’s primary storage all the time. Only a small set of favorite destinations need to be present in the primary memory. A detailed solution for the reduction of updating overhead is presented in [48], and as it is possible to see, the route updating packets in this scheme contain only one fit for every destination.

4.2.1 The Periodic Queue Exchange Algorithm

In this technique the primary and the alternative routes are stored in each node and are selected according to certain network rules. An emergency route is also chosen to be used in the case of failures or severe congestion.

Every node periodically sends the "best local link" table to its adjacent neighbors. The table contains one link for every destination representing the link of the primary or alternative routes with the minimum service queue length. Every node also sends the queue length on its links with the best local link table. The node might multiply the queue lengths by fixed weights to give the illusion of longer queues so as to solve some problems like those related to channels having different capacities.

Each node utilizes the information available locally and the information offered from its neighbors to decide what route to use for every destination. Method can be extended to more than one alternative route straight-forward.
4.2.2 The Asynchronous Traffic-Splitting Algorithm

A routing technique is required to adapt rapidly to changes without the large overhead usually accompanying periodic updating. This overhead can be decreased by increasing the updating period, but this slows down the adaptation. Rapid adaptation in network areas where the traffic load is high, requires short updating periods which in turn increase the cost of updating and affect the packet delay and throughput and the utilization of the network resources. However, the network areas where the traffic load is low, the overhead may be kept at a considerably reduced level while still allowing for a suitable performance to be achieved. Overhead reduction and rapid adaptation suggest binding the updating process with the detection of specified conditions about the traffic or topology. This is the main idea of asynchronous updating techniques.

As it is pointed by authors in [48], the design of an asynchronous technique should include specifying schemes for three main functions. First, a scheme for detection of a specified threshold level to
invoke the updating operation. The threshold level can be a queue length, a link utilization, an estimate of a traffic parameter. It can also be the difference between the current value of a specific parameter and its value at the last updating.

Second, a scheme is required for limiting the number of updates once they are invoked. An upper bound is imposed on the number of updating operations so that the updating overhead does not affect the performance of the network adversely, otherwise, the updating process would degrade rather than improve the network utilization.

Third, a scheme is required for preparing information that will be exchanged and for recognizing the nodes to which the updating packets are to be sent.

In [48] queue lengths are selected to be threshold parameter because the dominant delay component under load is the queueing delay. The proposed algorithm utilized two threshold levels, TH1, and TH2, where TH1 is a short queue length and TH2 is a long queue length. When a queue exceeds TH1, congestion is expected to build up, whereas a queue exceeding TH2 signifies that severe congestion already exists.
The ATS algorithm utilizes primary and alternative routes, where primary route is accompanied by a 2-bit indicator and every alternative route is accompanied by a 1-bit indicator (see fig. 4.2). A 2-bit counter may accompany every route for traffic-splitting.

<table>
<thead>
<tr>
<th>P</th>
<th>A1</th>
<th>A2</th>
<th>Ip</th>
<th>Il</th>
<th>I2</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1</td>
<td>3</td>
<td>00</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>4</td>
<td>01</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>10</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Bits controlled by external information

Bits controlled by local information

Ip - 2 bit primary indicator

Il - 1 bit alternative route i indicator

P - primary route

Ai - alternative route i

Fig. 4.2 Main Table for ATS using Two Alternative Routes
There are three updating mechanisms in each node. These are internal updating, external updating, and congestion routine.

The internal updating mechanism is invoked periodically, the external updating mechanism is invoked within a node when a congestion is expected to build up (when at least one of the node's queues exceeds TH1), and the congestion routine is triggered within that node when a severe congestion already exists (when at least one of the queues in the node exceeds TH2 or at least one of the queues in the node remains greater than TH1 for a long specified period).

For example, when the internal updating algorithm is invoked, it sets the right bit of every 2-bit primary indicator corresponding to a link having a queue that exceeds TH1 and resets the other right bits. The indicator bits of the alternative routes are set locally in the same way. The indicator bits will help in route selection for packets and in traffic splitting for high loads.
4.2.3 **Performance Study**

In [48] performance of proposed algorithms was quantitated in terms of delay, the throughput, the cost, and the reliability.

Fig. 4.3 shows the average packet delay of the two proposed algorithms compared to the old ARPANET routing algorithm and to the optimum bifurcated routing algorithm.

![Fig. 4.3 Average Delays for Different Routing Algorithms](image)

**Fig. 4.3 Average Delays for Different Routing Algorithms**
The details of ARPANET old routing algorithm can be found at [13].

The new ARPANET routing algorithm seems to show some advantages over the old in terms of speed of response to changing topology, stability, and suppression of looping. These advantages are apparently attained without undue overhead [15]. Interestingly, the old ARPANET routing algorithm is shown in [12] to have better average delay since the new algorithm uses only one route to every destination for a relatively long time (10s). The simulation of the new ARPANET routing algorithm is currently under way.

The optimum bifurcated routing algorithm is a theoretical algorithm whose average packet performance represents the lower bound of the average packet delay achieved by any practical adaptive or deterministic algorithm [53].

As can be seen from Fig. 4.3, both of the proposed in [48] algorithms have excellent performance with respect to delay.

At light leads, the delay curves of the ARPANET (ARPA), the Periodic Queue-Exchange (PQE), and the Asynchronous Traffic Splitting (ATS) algorithms are
very close to each other and within 3% of that of the optimum bifurcated algorithm (OB). At this loads adaptation to traffic loads has no significant gain.

A moderate loads the ARPA performance curves starts deviating significantly from the two other curves which are still very close to each other and to the OB algorithm.

At high loads, the ARPA algorithm continues deviating more sharply, approaching a vertical asymptote. The ATS algorithm deviates from PQE algorithm in this load range.

At larger loads, the ATS algorithm performance curve increases rapidly and cannot support a load. This considerably high load needs careful splitting of traffic among the numbers of the critical links. The queueing time on these links is the dominant factor affecting the average delay value of packets. Any improper distribution of loads may cause severe accumulation of packets on one or more of these critical links. Thus more information for adaptation is necessary for supporting higher loads that utilized by ATS. The PQE algorithm uses more accurate information for adaptation. This is why it can support a higher load.
Figure 4.4 shows the average packet delay of various algorithms relative to that achieved by the OB.

<table>
<thead>
<tr>
<th>Load</th>
<th>0.3</th>
<th>0.45</th>
<th>0.6</th>
<th>0.7</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARPA</td>
<td>1.03</td>
<td>1.27</td>
<td>1.94</td>
<td>3.05</td>
</tr>
<tr>
<td>PQE</td>
<td>1.03</td>
<td>1.13</td>
<td>1.16</td>
<td>1.31</td>
</tr>
<tr>
<td>ATS</td>
<td>1.03</td>
<td>1.15</td>
<td>1.16</td>
<td>1.60</td>
</tr>
</tbody>
</table>

Figure 4.4 Values of Average Delays Relative to Optimum Delays

The main reason for the average delay improvement of the new algorithms are the distribution of traffic load over available routes and the elimination of loops, but not the low overhead of these algorithms. This result is reached after purposely increasing the overhead to a level equal to that of the old ARPANET routing algorithm and observing that the performance did not significantly changed.

Although low delay is the main objective for most of the networks, two proposed algorithms achieve a goal throughput level that has been studied under different loads. The algorithms score quite well with respect to
reliability too. This is achieved because of the rapid reporting of failures to all nodes and the relatively short queues that the algorithms maintain at all nodes. It is necessary again to emphasize the importance of using guided-adaptive techniques for efficient routing, which allow to utilize multiple routes and can adapt to changes in traffic by switching from one route to another.
5.1 Performance Issues of Ring Techniques for Local Area Networks

As it was pointed out in [16], the performance issues of ring techniques for local area do not seem to be well understood. The author of [17] tries to shed more light onto this subject by considering the performances of four important candidates for local-area subnetworks: token ring, slotted ring (empty-slot technique), random-access bus (CSMA with collision detection) and ordered-access bus (MLMA reservation scheme).

The investigation of all types in [17] is based on analytic models which describe the various topologies and access mechanisms to a sufficient level of detail.

The performance measure of prime interest is the delay-throughput characteristic of the systems. Delay is measured as the mean transfer time of the packets which is defined as the time interval from the generation of a packet at the source station until its reception at the destination. This means that the
transfer time includes the queueing and access delay at the sender, the transmission time of the packet, and the propagation delay.

5.1.1 Token Ring

In a token ring [16,54], access to the transmission channel is controlled by passing a permission token around the ring. When the system is initialized, a designated station generates a free token which is passed around the ring until a station ready to transmit changes it to busy and puts its packet onto the ring. The sending station is responsible for removing its own packet from the ring. At the end of this transmission, it passes the access permission to the next station by generating a new free token. Basically, it is a single-server queueing model with as many queues as stations attached to the ring. The queues are serviced in a cyclical manner symbolized by the rotating switch which stands for the free token.

With respect to the order of service several policies can be distinguished, e.g., that a queue is serviced until it is empty ("exhaustive service") or that only a limited number of packets is serviced per access possibility ("non-exhaustive service").
Although, in principle, performance differences exist between these policies [55], it can be generally observed that for the parameters valid in local networks, these differences are very small if the traffic is equally spread over all stations. The token ring model is graphically represented in Figure 5.1.

![Token-ring Model](image)

Figure 5.1 Token-ring Model

(S = 4 stations; $\lambda_i$: packet arrival rates; $C$: round-trip delay; $T_p$: packet service time)

5.1.2 Slotted Ring

In a slotted ring [16, 18, 56], a constant number of fixed-length slots circulates continuously around the ring. A full/empty indicator within the slot header is used to signal the state of a slot. Any station
wanting to transmit occupies the first empty slot by setting the full/empty indicator to "full", and places its data in the slot when the sender receives back the occupied slot, it changes the full/empty indicator to "free". This prevents hogging of the ring and guarantees fair sharing of the bandwidth among all stations. Rings used in the local area are usually relatively short which means that a packet or message has to be transmitted within several slots.

In a simple case, where the ring carries one slot only, the system can be described by means of the queueing model with S+1 queues, where S is the number of stations attached to the ring. Packets waiting in one of the queues are served in sequence for a short time quantum at which corresponds to the slot length. For analytic convenience, it is good to adopt the point of view which was introduced by Kleinrock in the context of time-sharing systems, that the service quantum at shrinks to zero. Since in the local environment, packets are usually at least ten times longer on the average than the slots, this condition is normally fulfilled. A graphical representation of the model is given in Figure 5.2.
5.1.3 CSMA Collision—Detection Bus

Amount the numerous random-access schemes, for example, Carrier-Sense Multiple-Access (CSMA) combined with collision detection appears to be a very attractive solution for use on a bus system [17]. Under a CSMA protocol, every station wanting to transmit a packet must listen to the bus in order to find out whether any transmission is in progress. In this case, it defers its transmission until the end of the current transmission. In spite of carrier sensing, packet collisions cannot be completely avoided because of the non-zero propagation delay of the bus. The occurrence
of a collision can be detected by the sending station by comparing the transmitted with received data. Upon detection of a collision, transmission is aborted and the station reschedules its packet by determining a random retransmission interval. To avoid accumulation of retransmissions, in other words, to achieve stability, the retransmission interval is adaptively adjusted to the actual traffic load. Ethernet [57], for example, uses the so-called Binary Exponential Back-off algorithm which, in principle, means that the average of the retransmission interval is doubled every time the transmission attempt of a certain packet ends in collision. Of course, other control policies are also possible [58,59].

A performance analysis of the system described, given in [17] appears to be in excellent accordance with simulation results given by Lam [58]. A further profound investigation of a similar system with a slightly different CSMA protocol was performed by Tobagi and Hunt [59].

5.1.4 MLMA Ordered - Access Bus

As an example of a bus system employing ordered access, the reservation-type scheme MLMA (Multi-Level
Multiple-Access) proposed by Rothauser and Wild, can be considered [60]. In its simplest version, this access mechanism works as follows. Information transmission occurs in variable-length frames, the structure of which consists of a controller, request slot and an arbitrary number of packets. A controller provides start flags at appropriate time intervals which signal the beginning of a frame. A frame is divided into two parts: a request slot and an arbitrary number of packets. In the version of MLMA considered here, every station attached to the bus owns one bit within the request slot. By setting its private bit, a station indicates that it wants to transmit a packet within this frame. At the end of the request cycle, all stations know which of the stations will make use of this frame. The transmission sequence is given by a priority assignment known to all stations. Stations indicate the end of their transmission by a unique marker so that any active station can simply determine when its time to transmit has come.

For a scheme similar in concept, mark [61] derived approximate results for the queueing delay assuming constant packet length and an upper boundary of the number of packets a station is allowed to
transmit in a single frame. In [62], exact results for the mean transfer time of every station (depending on its priority) are derived under the assumption of generally distributed packet lengths if no boundary on the number of packets per frame and station is imposed.

5.1.5 Performance Measurements Results

Figure 5.3 compares the delay-throughput characteristic of the four local subnetworks described above for the following parameters: transmission rate of 1 Mbit/sec, a cable length of 2 km, which is considered as a worst case in the local area 50 systems attached to either system; exponentially distributed packet lengths with mean 1000 bits, and a header length of 24 bits. The uniform header-length value of 24 bits is optimistic for the bus systems because they usually require a longer preamble of the packets to achieve bit synchronization.

![Graph comparing delay-throughput characteristics of slotted ring, MLA ordered access bus, token ring, and CSMA collision detection bus. The x-axis represents throughput rate/transmission rate, and the y-axis represents mean transfer time/mean packet transmission time.](image-url)
Fig. 5.3 Transfer delay-throughput characteristics of the four subnetworks at 1Mbit/sec

The slotted ring (1 slot) exhibits considerably worse performance than the other systems because of the following two effects [17]:

(1) The buffering capability of a ring in the local area is relatively small. This leads to an unfavorable ratio of the header length to the data field per slot.

(2) The time needed for passing around access permission - in the form of an empty slot - is significant in slotted rings. Consider, for instance, the situation where only one station is transmitting. In order to guarantee equal sharing of the bandwidth, the access policy requires that a station, after having used a slot, passes it on with the busy/free indicator set to "free". Thus, only half of the bandwidth is used in this case.

In Figure 5.4, only one parameter has been changed as compared to the previous example, namely, the transmission rate which is now 10 Mbit/sec instead of 1 Mbit/sec.
Fig. 5.4 Transfer delay - throughput characteristics of the four systems at 10 Mbit/sec.

This one change, changes the situation considerably. The most remarkable difference is observed for the bus with CSMA and collision detection. Whereas, for loads less than 0.2, it shows the best performance, the transfer delay increases rather rapidly for loads greater than 0.4 and has a vertical asymptote at about 0.6. The obvious reason for this behavior is that for higher throughput values the frequency of transmission attempts during the vulnerable period of the propagation delay is such that a significant portion of transmission attempt end in collision. Thus, the transfer delay increases due to the need for retransmissions.
Investigations of other policies indicate that
the maximum-achievable throughput can be slightly
increased by using a non-persistent strategy [59]
instead of the persistent one assumed in examples
above. Nevertheless, the conclusion is, that even a
sophisticated random-access scheme like CSMA with
collision detection leads to inefficient operation if
the propagation delay to packet-transmission time is
too high.

In contrast to the random-access bus, the token
ring performs almost ideally also for higher speeds.
Its slightly higher delay as compared to CSMA bus for
low throughput is due to the access delay of the ring
until a free token arrives.

The transfer time of the MLMA ordered - access
bus is slightly longer than that of the token ring over
the whole range of throughput values. This is mainly
due to the bus scheduling overhead.

5.2 The Conflict-Free Protocols for Local Area
Networks

It has been widely recognized that for a
broadcast multi-access network with low to medium
traffic loads, random access schemas such as CSMA [17,
63], and CSMA with collision detection [17,59]
protocols provide a simple and efficient means of
communications between a group of users nodes attached to the network. However, data collision can occur if two or more nodes attempt to transmit at nearly the same time. The colliding packets must then be retransmitted after some random delays in order to avoid further collisions. The performance of these protocols is, therefore, directly related to how well they avoid message collisions and how well they handle retransmissions. The problem of data collision worsens drastically as the traffic load goes higher.

The unstable and nondeterministic nature of the transmission delay introduced by data collision in random access techniques has prompted the study of many collision-free protocols (see [64] for numerous examples), among them the broadcast recognized access method (BRAM) [65], the mini-slotted alternating priorities (MSAP) [66], and more recently, the group BRAM (GBRAM) [67, 68] protocols.

BRAM and MSAP employ a virtual-token passing scheme, in which each node gains network control (the virtual token) at a unique time slot determined by a decentralized scheduling function, hence avoiding data collisions completely.
Both BRAM and MSAP work well with a small number of users, particularly under heavier loads. The delay performance of these protocols degrades, however, as the number of users nodes increases. The parametric BRAM [65] remedies this by partitioning the nodes into groups, letting the nodes in a group share a single time slot, with the cost of possible intragroup collisions.

GBRAM employs different grouping criterion than is used in the parametric BRAM; one that is able to avoid collisions altogether [68]. The analysis in [68] indicates that GBRAM has a significant performance improvement over BRAM, and yet is simple to implement.

5.2.1 Review of BRAM and MSAP

There are two variants to the BRAM [65], namely, fair BRAM and prioritized BRAM (which is essentially the MSAP). The former limits each mode to transmitting at most one packet at a time; the latter allows each node to transmit all the packets in its buffer before relinquishing channel control. The channel state under BRAM can be viewed as consisting of a sequence of cycles composed of idle, scheduling and transmission periods. Analysis of BRAM shows that for
small values of the \((K \times a)\) product \((K - \) total number of nodes on the network, \(a - \) period of time), BRAM provides fair allocation of the channel and good throughput - delay and throughput - traffic load performance [65]. However, as the \((K \times a)\) product increases, the length of the scheduling period increases as well resulting in degradation of performance.

One solution to the above problem is a variation of BRAM called parametric BRAM [68, 65], which partitions the \(K\) nodes into \(M\) groups and lets the nodes in the same group contend for the same time slot. Although theoretically parametric BRAM does indeed reduce the scheduling overheads, collision detection and resolution mechanisms must be added to BRAM, and the optimal grouping is difficult to achieve in practice.

The complications of the parametric BRAM lead one to wonder if there is an easier way to partition the nodes into groups and yet maintain conflict-free transmissions. A straightforward answer [68] is that such groupings may have already existed in many local networks in the form of node physical locations. For example, there could be a global data bus that runs through several buildings, or from one
floor to another. The terminal-nodes or device-nodes in a single room may be viewed as a "natural group" of nodes on the bus. Typically, the intragroup transmission delay is a small fraction of the total bus delay. This is especially true if the data bus spans several buildings. It would seem possible, then, to further subdivide the group slot into mini-slots of the size of an intragroup delay so that each mode in the group can be allocated a unique mini-slot for transmission attempts. In [68] it is shown how this can be implemented by giving the notion of the group-GBRAM (GBRAM) and its variants.

5.2.2 GBRAM

In GBRAM there are basically K nodes connected to a common communication medium. These nodes are partitioned into M groups such that each group occupies only a fraction of the total bus length. Each node can sense the bus status (i.e., carrier-sensing) in a negligible time. The channel is noiseless and the acknowledge processing time in negligible. These assumptions are made in [68], where author gives the performance analysis of GBRAM.

GBRAM [68] can be envisioned as having a "virtual token" circulation from node to node, such that if all nodes in one group have been visited
by the token, the token is passed from that group to the next group in sequence. The channel periods of GBRAM are similar to those of BRAM, except that the scheduling period is slightly more complicated.

It is also possible to extend both fair and prioritized GBRAM to extreme cases. Extreme-fair GBRAM allows only one node in each group to transmit per group slot. To ensure fairness among the nodes in the same group, the intra-group priority can be constantly shifted so that a transmitting node will have the lowest priority in the next group slot.

Extreme-prioritized GBRAM allows the nodes in a group to be visited more than once by the token; and the token will be passed to the next group only if all nodes in the current group are idle.

5.2.3 GBRAM Performance

The throughput-delay performance of GBRAM is compared with BRAM and CSMA/CD delay performance in Figure 5.5.

It should be noted here that the CSMA/CD discussed in this paper is taken from the v-persistent back off scheme analyzed by Tobagi and Hunt [59], and is slightly different from the Ethernet protocol, which
uses its own binary exponential back-off scheme [69].

Figure 5.5 The Throughput-Delay Performance of GBRAM

It is clear that GBRAM is far more superior than the BRAM overall, and is very close to the CSMA/CD with optimal v-value in light traffic. In heavy loads, it exceeds its collision-prone counterpart.

The GBRAM throughput, versus offered channel traffic is given in Figure 5.6.
Throughput

Throughput

Figure 5.6 Comparison of Offered Load Versus Throughput between GBRAM and Other Protocols

GBRAM, obviously, exhibits stability over other protocols under heavy loads, while under low traffic it gives performance close to both BRAM and CSMA/CD protocols.
INTEGRATED VOICE/DATA COMPUTER COMMUNICATION NETWORKS

Packet switching technology achieves the effective sharing of expensive communication resources for the performance and cost optimization. This technique for the transmission of data is now being introduced in the field of speech communication.

6.1 Packet-Switching in Computer Communication Networks

Packet-transmission networks have been developed over the past ten years in an attempt to increase the channel utilization [70,71]. The basic idea is to allocate some or all of the system capacity (along some path between subscriber) to one customer at a time, but only for a very short period of time. Customers are required to divide their messages into small units (packets) to be transmitted one at a time. Each packet is accompanied by the identity of its intended recipient. In packet-switching networks, each packet is passed from one packet switch to another until it arrives at one connected to that recipient, whereupon it is delivered. Packets arriving at a switch may need to be held temporarily until the transmission line that
they need is free. The resulting queues require that packets be stored in the switches and it is not unusual that all packet buffers are occupied in a given switch. Thus both the switch capacity (processing and storage) and transmission capacity between switches is statistically multiplexed by subscribers. The designers of such packet networks are faced with the problem of choosing line capacities and topologies that will result in relatively high utilization without excessive congestion.

Packet-switching provides two basic protocol alternatives in communication networks [21].

(1) Virtual Circuit Protocol (VCP): Although shared by both data and voice packets, VCP uses the following concepts: A source which is the originator of data or voice packets requests a connection through an appropriate control packet. The control packet establishes the connection by setting up a proper pointer to routing tables while identifying the connection requested through a connection identification. When the source destination connection is approved, the originator is notified and transmission starts. Its basic characteristic is that the destination receives packets in the same sequence
transmitted by the source. This makes the VCP an interesting procedure for packetized voice applications due to critical timing and sequence of packets in recovering the voice information. Basic differences of this procedure with that of circuit switching are discussed in [19]. The virtual circuit is disconnected by issuing a connection termination command through either source or destination. From a link failure point of view, this scheme has some drawbacks [72].

(2) Datagram Protocol: Each packet is delivered from source to destination independent of all others. Packets consult the routing tables at each node and proceed until reaching the destination node. Although in communication environments with link failure this procedure proves more efficient than VCP, it has problems of high packet overhead. This is because each packet should find its way through the network independent of all others (no connection identification is required).

6.2 Properties of VCP and Datagram Protocol in Both Data and Voice Packet-Switching Node

In the experimental work performed on the ARPANET, the problem associated with packetized voice transmission in store-and-forward tandem link was in
the area of end-to-end delay [73]. In [21], in order to examine the probability of sharing the communication links between voice and data packets, the integrated network is proposed, where each node is capable of storing packets and forwarding them to the next node. The architecture of this network is presented in Fig 6.1

![Network of Cascaded Nodes](image)

In [21] authors examine several techniques for controlling the flows of voice and data packets in order to minimize the voice delay.

A priority queuing model is analyzed with high priority given to voice packets, since they are more sensitive to a variable delay.

The system considered in [21] represents an infinite queue which receives a mixture of packetized voice and data each following a Poisson process with fixed traffic rates $\lambda_4$ and $\lambda_2$ respectively. It is
assumed that the two arrival processes, although identically distributed, represent two independent arrival. The service policy is as follows:

A switch is set for transmission of voice for only \( P \) fraction of the time and for data during the remaining time which accounts for \( 1-P \) fraction of the time. The static model utilizes full channel capacity \( C \) for transmission of voice and data packets and it preserves the priority scheme. Only during \( P \) fraction of the time whereas, during \( 1-P \) fraction of the time voice packets may arrive and experience delay. The effect of the delay is embedded in the complicated set of equations, and it is controlled by \( P \). When the system is looked at in steady state, on the average at least \( P \) fraction of the time switch is dedicated to voice transmission and \( 1-P \) to data.

In the dynamic model, voice packets are given higher priority over those of data. Whenever a voice packet arrives at the system if a data packet is already in process, it is finished being processed and then switched to voice packets (non-preemptive). The switch is turned to those of data packets if and only if there are no voice packets available in the system.
The analysis of static model that represents the normalized delay per voice and packet separately versus static switching probability \((P)\) shows that for all values of \((\lambda)\) as the portion of time dedicated to voice increases voice packet delay decreases, while that of data is concave. \((\lambda)\) is defined a ratio \((\frac{\mu_1}{\mu_2})\) of average data packet length to average voice packet length (both independently exponentially distributed). Also from an analysis comes out an optimal static control probability \((P^*)\) which minimizes data delay while maintaining an acceptable voice delay.

Fig. 6.2 and 6.3 show the results of normalized data packet delays versus \((\lambda)\) for different traffic intensities.

--- dynamic control; --- static control

Fig. 6.2 Data Packet delay vs. \((\lambda)\)

--- dynamic control; --- static control

Fig. 6.3 Data Packet delay vs. \((\lambda)\)
In these figures $\rho'_1 = \rho_1 = \frac{\lambda_1}{\mu_1 C}$
$\rho'_2 = (1-p) \rho_2 = \frac{\lambda_2}{\mu_2 C}$, traffic intensity for voice and data packets with full channel capacity consequently.

Fig. 6.2 and 6.3 show that as ratio of data packet length to that of voice increases, service time of each voice increases, service time of each voice packet compared to those of data will be more evident. In static control the average data delay will decrease for large voice packet sizes since voice has higher priority, this will increase for average data delay. These figures also show the results of data delay in the dynamic model which generally represent the same characteristics. Data delay under dynamic control is always better than that of static control. It should be noticed that in all these results, the switching time between data and voice packets is not included.

Fig. 6.4 and 6.5 show average voice delay versus $(\lambda)$ for different traffic intensities. As it can be seen from these figures, average voice delay decreases and is always lower for higher probability of switching. The dynamic model holds the switch at data queue until the data packet already in the transmitter is served and then switch to voice.
Due to longer data packets compared to those of voice, voice delay in dynamic control increases. A very interesting property shown by these results is that the delay trends always increase at $\lambda < 1$ for data packets, and for $\lambda > 1$ data delay is almost unchanged. For voice packets, the reverse of the above is observed for $\lambda < 1$. Dynamic control characteristics also represent similar ($\lambda$) trends. It is immediately noticed that the static control scheme presents a more attractive model than the dynamic since almost always


As a practical contribution, a hardware implementation of the multiplexer is not given anywhere in the literature and requires further study.

6.3 Channel Capacity Utilization in Integrated Voice/Data Communications Networks

It has been observed that the utilization of a telephone channel is around 40% only [74], because of the occurrence of silence periods in speech. To cater to the rapidly increasing demand for communication facility asynchronous interpolation of non-real time data in speech gaps is suggested in [34] the transmission of data in speech gaps through proposed asynchronous data interpolation (ADI) technique, improves the channel utilization to almost 100% at higher loads.

All it needs is the incorporation of a store-and-forward buffer for the storage of data messages and a switching mechanism at the nodes of the communication network. The store-and-forward buffer handles statistical peaks in random messages. Whereas, the switching mechanism detects the occurrence of gaps to the waiting data messages. Since the data from the buffer is transmitted at random points of time for
random duration, it is said that the data is asynchronously interpolated in speech gaps. The random availability of the finite capacity communication channel together with the random nature of the input data traffic causes queue formation in the buffer. Consequently, the data message is delayed. The buffer design and its performance evaluation in such store-and-forward system for data communication where the channel capacity is being shared, is one of the most important aspects in computer communication network design. Queueing models [75, 76, 77] are developed as a tool for the analysis of such a buffer at a node. The integrated speech data system with a synchronous time division multiplexing (STDM) for a large number of speech sources and Erlang arrival process data messages is modeled in [74] as a discrete time multiserver queueing system with Erlang input traffic and server interruption described by a binomial process. Graphical representation of this model given in Fig. 6.6
In the earlier studies of the buffer performance with Erlang arrival process, the buffer state equations were written in terms of stages rather than characters [78]. The solution of state equations in terms of stages is not possible in a closed form even for a single server case [78]. In [74], the buffer state equations are written directly in terms of characters by converting state arrival process into character arrival process. An interactive algorithm is developed for the solution of these equations.

In [74], the system considered for the multiserver queueing model, which is the store-and-forward buffer of an integrated digital voice-data system shown in Fig. 6.6.
The transmission channel is shared by N unbuffered synchronously time division multiplexed speech sources and non-real time data stored in the buffer. The digital voice has no buffering facilities and has the priority for the transmission over the data to prevent the degradation of speech quality. The two registers of site N each record the status of the fixed assigned time slots of each user in a frame and monitor this information to the CPU. The CPU thus knowing the number of idle slots in a frame operates two registers and the buffer at the clocking epoch. As a result, characters equal to or less than the number of idle slots subject to its availability are taken out from the buffer by the server and the CPU synchronous interpolates them in the idle slots. The clocking interval equals the frame duration and is taken as one unit of service-time.

The scheme suggested in Fig. 6.6 is feasible in view of the advances in digital switching and signaling. The integration of voice and data over a common channel is discrete possibility in the near future. Tsuda et. al. [79] have discussed the development of a packetized voice/data terminal to be used in digital network.
To illustrate the application of the model, the twenty four voice sources were synchronously multiplexed in [74] (STDM) and the data was interpolated in the speech gaps. The state of the fixed assigned time slots in a frame depends upon the speech source pattern. In [80], the author has given the average duration of the talkspurt and pause as 1.366 and 1.803 seconds respectively. The steady state probability of the slot being busy, $P_{54}$ can be obtained from the same reference [80] as $P_{54} = 0.5688$.

According to analysis in [74], the overflow probability, $P_0$ decreases with buffer length, $L$ for a given traffic intensity, $\rho$ and Erlang parameter, $r$ and it also decreases with $r$ for a given $\rho$ and $L$.

It also was established that the average queueing delay increases with the traffic intensity for a specified overflow probability $P_0$ of and Erlang parameter, $r$. The effect of increasing Erlang parameter $r$ is to reduce the overflow probability and thus the buffer length requirement is reduced. However, there is not much appreciable change in overflow probability for Erlang parameter beyond $r=3$. 

- 89 -
Decrease of average queueing delay with increase of $r$ is almost nil below the traffic intensity $\rho = .5$. It is also observed that by computing few values of overflow probabilities and average queueing delays using exact solution for specified values of traffic intensity and buffer length and comparing those with approximate solution shows that the difference is very small. The difference between the delays is also negligible.

It is also observed that in both the cases the computational time required for the convergence is more and more as $\rho$ increases and it is much more higher in the case of exact method than the other. This leads to the conclusion that approximate solution is not only superior on the basis of computing time requirement but also more reliable as it slightly overestimates the buffer length.
In this chapter a new distributed task force scheduling technique for networks of homogenous computers is discussed. The technique is called Wave Scheduling.

7.1 Wave Scheduled Technique for a Distributed Task Force Scheduling

In [81] before the Wave Scheduling Technique is discussed the following assumption is made: that the task processors of the network are homogenous and, aside from peripheral connections and advantages of physical location, all processors are equally capable of executing each task. While this homogeneity assumption is restrictive, it is not debilitating: the network computers currently under construction such as MICRONET at SUNY/Buffalo [82], X-Tree at UC-Berkeley [83], and MuNet at MIT [84], all very nearly satisfy the constraint.

The restraint put in [81] can be explained as an elimination of special cases. One consequence will be that, in a network with heterogeneous nodes, task forces scheduled by Wave Scheduling may not execute
quite as efficiently as they might if a more detailed matching of tasks to processors were performed.

Another assumption is made that the network computer supports a hierarchical high-level operating system control structure such as described in several recent papers [85, 86, 87].

The Wave Scheduling uses the hierarchical control structure. The nodes at level 0 (workers) are available for user tasks. Those at higher levels (managers) are responsible for maintaining the integrity of the communications subnetwork and for performing resource allocation in a local region. The hierarchical control scheme does not imply that the physical connections in the network form a hierarchy.

As it is pointed out in [81], the basic motivations for hierarchical control are its resistance to isolated failures and its ability to make decisions by passing summarized information upward. By establishing many managers at many levels the danger of total collapse is greatly reduced. When a manager is disabled for whatever reason, another manager can allocate its functions to other nodes. Such a hierarchy is a good compromise between centralized
control and completely distributed control because it provides for many control sites with differing levels of responsibility.

The Wave Scheduling procedure depends on the tree-like structure of the control hierarchy to partition task forces by size. Scheduling is done completely independent of the network physical connection topology. It is possible to establish control hierarchies automatically in arbitrary networks [85]. Thus, Wave Scheduling is also applicable to arbitrary interconnection topologies.

7.1.1 Description of Wave Scheduling

Suppose that a task force which needs \(S\) nodes to execute, enters a queue of ready task forces at any arbitrary network node. Task forces may appear at any level of the hierarchy, at managers as well as worker nodes. All managers in the hierarchy will try to schedule task forces which are no larger than some dynamically changing fraction of the number of worker nodes in the subtree of which the manager is the rest.

If a task force enters the network at a level which cannot schedule a task force of size \(S\), the task force descriptor is passed up the tree until a suitably
enabled manager is reached. A manager at the appropriate level becomes the Task Force Master (TFM) for the task force. Task forces too large to be handled even by the entire network are rejected when they reach the top level.

Each TFM keeps track of the number of non-busy workers believed to be in that TFM's subtree. TFM's are also responsible for reserving enough nodes for the task forces which they control. The TFM for a task force which needs $S$ nodes wages the activity of the network in its subtree and computes a value $R > S$, which is the number of workers that it will try to reserve.

To prevent deadlock and cope with hardware failures, each level of the hierarchy observes timeout rules. For example, a submanager reports the number of worker nodes reserved for its manager after a fixed time cut interval, regardless of whether all its subrequests have been answered.

If the number of workers which are actually reserved for a requesting TFM is less than 5, the scheduling is considered a failure, and the unscheduled task force return to the queue for execution by the local node.
7.1.2 Efficiency of Wave Scheduling

It is shown in [81] that at reasonable levels of network utilization the efficiency of Wave Scheduling is comparable to that of centralized scheduling. For example, in a network of 1000 worker nodes in which the worker service rate is 1 task/sec, the task force arrival rate is 500 tasks/sec, and the cost of reserving a worker node before successful scheduling is .1 sec, the relative efficiency of Wave Scheduling is found to be about 82%.

In [88], the efficiency of Wave Scheduling is compared to a central scheduler using a Markov queueing model. The measure of efficiency which is used in that analysis is the total system time which a task force can expect to spend in the network. This measure includes both execution time and queue delay. The primary result of that analysis is that the relative efficiency of Wave Scheduling with respect to central scheduling, $E_{rel}$ is given by:

$$E_{rel} = \frac{\frac{m \times |Nw|}{1 + \frac{m \times w}{m}} - \beta}{\frac{m \times |Nw|}{m} - \beta}$$

where
\( m \) is the service rate of a single worker node,
\( b \) is the arrival rate of task forces,
\( |N_W| \) is the number of worker nodes in the network
and \( \bar{W} \) is the average cost (in wasted time) of reserving
a worker node before a task force is successfully
scheduled.

7.2 Task Allocation Scheme for Time-Critical Application

In the time-critical environment, the
development of software allocation technique is
subjected to more constraints [89]. The tasks in the
time-critical application are divided into several
major processing threads. Because of the real time-
critical nature these threads may be executed
concurrently to meet the port-to-port (PTP) processing
time. PTP time confines the total execution time of
the specified tasks in a thread within a certain
prescribed period. Therefore, an application of the
tasks to the processing elements in the time-critical
application must first satisfy these PTP time
requirements to qualify as a valid application toward
development of a task allocation model for the time-
critical distributed system, the authors in [90]: (1) identify the major contributing factors to the failure of the PTP time requirement; (2) derive an algorithm to incorporate these contributors and minimize the PTP time; (3) test the results derived by the algorithm on a simulator; (4) suggest additional research to improve both the algorithm and overall design methodology.

However, two major considerations which have significant impact on the PTP time requirement are not addressed in [90]: (1) change of implementation or operational environment as defined by a given scenario; (2) topology of the computer network.

[90] addresses only the development of the task allocation model based on a given scenario and network topology. However, with the help of the simulator, important feedback information can be provided on the effects of a given scenario and/or network topology. Additionally, if the worst case scenario is assumed, then the allocation should easily satisfy the less stringent scenario requirement.
7.2.1 Problem Description

For the time-critical applications, all tasks are arranged into several major processing threads. These threads must meet PTP time requirement. The execution time of a thread, as it defined in [90], consists of the following factors:

(1) Execution time of tasks on the processor. This is determined by the task size and the processor MIPS rate. The task size is in machine language instructions. The execution time of task \( i \) is determined by \( E_i = \frac{\text{size of task } i}{\text{MIPS rate of processor}} \).

(2) Network operating system overhead (NO). In addition to conventional operating system functions, distributed operating systems must perform concurrency control, integrity checking, recovery checkpoint update, among others, requiring additional overhead.

(3) Interprocessor Communication (IPC) cost. It occurs when tasks resident in different processors must communicate with each other. The IPC cost of tasks that interact with each other should be high when they reside in separate processors.
Waiting time (WT). It is the amount of time a task waits in the processor's enablement queue. The waiting time is strongly influenced by the number and size of tasks, the number of enablement, and processor loads. In particular, it is increased if large tasks are assigned to the same processor.

The sum of these factors determines the PTP time of a thread, ET:

$$ET = E_i + NO + IPC + WT.$$ 

For a given network topology and scenario, the network operating system overhead (NO) and the number of enablements are relatively constant. Therefore, in order to reduce the task waiting time, large tasks should be allocated to different processors. To reduce $E_i$, large tasks should be allocated to higher MIPS rate processor. To reduce IPC cost, tasks with high IPC cost should be placed in the same processor.

In [90], the allocation model for time-critical application is described. All the necessary information related to task and network is provided as follows:
The relevant information on tasks includes coupling factors among tasks, task size, and the number of enablements of each task. The coupling factor between tasks is a measure of the number of data units transferred from one task to the other \([84, 91]\). The unit of data is dependent on the application task; for example it may be work or byte.

The relevant information on network is the interprocessor distance and processor constraint. The interprocessor distance is conceptually the physical distance between two processors. It may represent any type of costs, however, that can be measured in time or in dollars \([92]\). Every computer has its hardware constraints which are called processor attributes. For example, the attribute bound may be the processor in MIPS, storage space, etc.

7.2.2 Summary of the Theoretical Branch and Bound (BB) Method

To employ the BB technique \([90]\) the allocation problem is formulated in terms of a depth-first search tree with nodes and branches. Consider a problem of allocating \(m\) tasks among \(n\) processors. Starting with task 1, each task is allocated to one of the \(n\) processors, subject to the constraints imposed on
the relations on tasks and processors. The allocation decision represents a branching at the node corresponding to the given task. For each node there are \( n \) branches. Some of which might be non-feasible choices. And \( m \) tasks correspond to \( m \) levels of branching in the search tree. A feasible sequence of successive branches is called a path. A path of length \( m \) (from the root node to the last node) corresponds to a complete allocation; otherwise it is a partial allocation. The cost of a path is determined by the branches and computed by the cost equation. The partial cost of node \( k \) is the cumulative cost of all branches in the path leading from node 1 to node \( k \). The cost associated with a complete allocation is called a complete cost.

The allocation algorithm [89] has two phases: initial allocation and backtracking. The initial allocation, starting with the first node (task) is obtained by selecting the branch (allocation) for each node one by one. When the last node is reached, a complete cost solution is determined. Then a systematic backtracking, starting with the last node, is activated to evaluate other paths of the search tree by investigating every branch for each node.
The BB Method defined by Kohler and Steiglitz [93] is applied. Its application for generating a search tree is concisely explain in [94].

7.2.3 Application

The Ballistic Missile Defense (BMD) case study [89] involves the design and allocation of a set of real time software tasks into a network of processors. Twenty-three tasks are identified (only 20 are used). These tasks are divided into seven major processing threads. Each thread involves a number of tasks. Because of the real time critical nature of the BMD application, each thread must meet the port-to-port processing times. Therefore, any allocation of the tasks to the processor must first satisfy these port-to-port time requirements to be qualified as a valid allocation. For example, one thread contains tasks 1, 8, 9, 13 and 22, and the PTP time limit of this thread is 40 milliseconds (ms). This means the total execution time for these 5 tasks must be within the 40ms time limit.

In order to justify the allocation result, a simulator [95] was designed to implement this BMD application data processing system which includes
twenty tasks and three processors. The three processors were simulated as homogeneous computers, fully connected with 1.5 MIPS rate each. Task information was provided by the simulator during a 900 ms simulation run [89]. This information includes the size of each task, the number of enablements of each task within the simulator run, and the coupling factor among the tasks. Group 1 is the result derived from the allocation algorithm to achieve lower IPC cost and lower waiting time. This assignment is shown in Fig. 7.1.

The assignment for Group 2 is to achieve lower IPC cost, but higher waiting time. This assignment is shown in Fig. 7.2.

<table>
<thead>
<tr>
<th>Tasks</th>
<th>P1</th>
<th>P2</th>
<th>P3</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>6</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>8</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>10</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>16</td>
<td>22</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>18</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>21</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Pi - processor i

<table>
<thead>
<tr>
<th>Tasks</th>
<th>P1</th>
<th>P2</th>
<th>P3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>6</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>16</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>18</td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>19</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>20,21</td>
<td>17</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Three system parameters are used in [89] to evaluate these two groups of results. These parameters are interprocessor communication (IPC) waiting times and port-to-port times. All three parameters are sampled and recorded by the simulator. The IPC is measured by K words/ms, where K is the average number of words transmitted among the three processors within one millisecond. Waiting times are sampled ten times every 100 ms in each processor’s enabling queue. PTP time is the total execution time for all tasks of that threads. Since the tasks in one thread may allocate onto different processors, the thread PTP time includes processing time, wait time and interprocessor communication time.

Two allocation assignments, A1 and A2 were tested on the simulator. The IPC for A1 and A2 is 114 and 119 words/ms, respectively.

Since processor P1 was the most heavily loaded, the wait time data are shown in Figure 7.3 for both assignments A1 and A2. The data were sampled 10 times every 100 ms, the waiting time is an average of the 10
samples during the 100ms interval.

<table>
<thead>
<tr>
<th>time interval of simulation</th>
<th>0 to 100</th>
<th>100 to 200</th>
<th>200 to 300</th>
<th>300 to 400</th>
<th>400 to 500</th>
<th>500 to 600</th>
<th>600 to 700</th>
<th>700 to 800</th>
<th>800 to 900</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average max. waiting time in Pi in ms</td>
<td>A1</td>
<td>.34</td>
<td>.48</td>
<td>.85</td>
<td>2.3</td>
<td>1.64</td>
<td>3.05</td>
<td>.97</td>
<td>1.26</td>
</tr>
<tr>
<td></td>
<td>A2</td>
<td>.2</td>
<td>.27</td>
<td>.44</td>
<td>2.1</td>
<td>3.8</td>
<td>5.72</td>
<td>10.6</td>
<td>20.2</td>
</tr>
</tbody>
</table>

Fig. 7.3 Average Waiting Time for Pi

In Figure 7.4, the PTP times for an important thread involving tasks 22, 1, 8, 9, and 13 are shown. The PTP limit is 40ms and the time indicated is the time when such measurements were made by the simulator and is approximately the mid point of the thread, since each thread was repeatedly executed many times during the simulation.

<table>
<thead>
<tr>
<th>approximate time of simulation</th>
<th>250 350 450 550 650 750 850 900</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>300 400 500 600 700 800 900</td>
</tr>
<tr>
<td>Total PTP time for the given thread in ms</td>
<td>A1</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>A2</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- 105 -
Fig. 7.4 PTP Time for Thread Containing Tasks (22,1,8,9,13).

Time is necessarily an important parameter because the scenario as well as the processing load vary greatly within the simulated time interval. From data given in Fig. 7.3 and Fig. 7.4, a definite trend toward increasing PTP time with increasing waiting time can be seen.

The IPC's for A1 and A2 are almost the same. However, the waiting time for A2 is increasing significantly after 400ms time interval. Therefore, the waiting time of tasks in each processor is a major contributor to the PTP time.

The results of an allocation model described in [90], show that the allocation model developed there, uses the constraint efficiently to generate allocations that reduce the waiting time and pass the PTP time requirement.
CHAPTER 8

CONCLUSIONS AND AREAS FOR FUTURE RESEARCH

In this thesis, in an area of congestion control, composite congestion control scheme is represented and its performance is described. The scheme is mainly composed of isarithmic control method and input buffer limiting. From numerical results and simulation, it can be seen that composite congestion control scheme is effective. When some requirements are made, for example, attaining predetermined throughput, preventing delay from exceeding maximum acceptable level, suppressing blocking probability and so on, the composite control scheme and approximation method may be one of a useful tool from which nearly optimal control strategy can be derived.

It was pointed out earlier and needs to be stressed again that neither local nor global congestion control by itself is sufficient as an effective control tool. The basic idea when a congestion starts to develop in a node, an attempt is initiated to simultaneously throttle the sources of traffic and to slow down the traffic from the neighbor nodes of the troubled node.
Feedback congestion control mechanism can be used for that. If congestion prolongs and the local control measure does not turn out to be sufficient, the global control tool becomes effective. On the other hand, if the congestion is temporary, the local congestion measure is sufficient to eliminate it and the global congestion control never comes into picture.

Also, the importance of efficient routing should be underlined again. The transmission of digital information has shown tremendous growth over the last decade and will undoubtedly continue to expand in the future. Primarily, this can be attributed to the growing awareness and rapidly developing use of computer networks, which serve to provide distributed computed facilities to geographically dispersed users. Also relevant is the growth of digital electronic switching techniques and through these the implementation of more complex network protocols and routing schemas.

Especially promising seem to be two new guided-adaptive techniques analyzed - PQE and ATS. Both of those algorithms utilize multiple routes and can adapt to changes in traffic by switching from one route to another. The routes are selected such that
they pass through the smallest number of intermediate nodes and avoid the creation of routing conflicts. The new guided-adaptive algorithms have excellent performance compared to the ARPANET-like algorithm and the optimum bifurcated algorithm. The average packet delay corresponding to the PQE (Periodic Queue Exchange) algorithm is approximately half that of the ARPANET-like algorithm under high loads and is within 15% to 30% of the optimal theoretical lower bound (depending on the traffic load). The throughput of the PQE is also better by about 10% of the ARPANET-like algorithm.

The average delay corresponding to ATS (Asynchronous Traffic Splitting) algorithm is also superior to the ARPANET-like algorithm, especially at medium and high loads, and has a very low overhead at the same time. The ATS algorithm can be used in large networks where a low overhead is essential for efficient operation.

The reporting of topology changes asynchronously and the separation of the topology updating procedure from the traffic updating procedure
are important features of the new algorithms. They are necessary features for medium and large networks. They increase the network reliability and performance, and result in significantly reducing the possibility of severe congestion.

Local networks are intended to provide improved communication capabilities among terminals and computers within a limited geographical area like an establishment or a single building.

The performance measure of prime interest is the delay-throughput characteristic of the systems. The prime objective of the Chapter 5 investigation of the local area networks, was to provide a comparative performance evaluation of five important candidates for local area networks: token ring, slotted ring, CSMA collision-detection bus, MLMA ordered-access bus, and group-BRAM. The major conclusions from comparison of the transfer delay-throughput characteristics for the five major systems are:

The token ring performs almost ideally over the whole range of parameters which is of intent in the local area. A necessary condition to achieve this behavior is that the delay caused within the stations be kept low.
The slotted ring shows comparatively high transfer-delay values for two reasons: (1) Due to the short slots of local-area rings, the overhead for addressing and control is exceedingly high; (2) The time needed to pass empty slots around the ring to guarantee fair bandwidth sharing is significant. Positive features of the slotted ring from a performance point of view are: (a) The expected transfer time of a packet is proportional to its length. (b) The overall mean transfer time is independent of the packet-length distribution.

The bus with CSMA and collision detection behaves ideally as long as the ratio of propagation delay to mean packet transmission time is sufficiently low. If for reasonable traffic loads this ratio exceeds 5% (as a rule of thumb), the increasing collision frequency causes significant performance degradation.

As a rule, the MLMA ordered-access bus shows slightly higher transfer delay than the token ring. This difference - which in most cases is insignificantly small - is caused by the overhead required for scheduling of the packets.
GBRAM is basically a two-level BRAM; a high level for group scheduling and a low level for node scheduling. It is decentralized and conflict-free. Analysis shows that GBRAM has a significant performance improvement over BRAM and MSAP, particularly in situations where the number of nodes on the network is large.

Several switching techniques for integration of voice and data packets have been investigated in the literature. Among those, reference [96] considers the performance of circuit-switching for voice and packet-switching for data traffic, whereas reference [97] considers integration of packetized voice and data over packet-switched networks. Based on suggested in [21] control schemes the sensitivity of the network performance, in particular packetized voice and data packet delays, to the switching parameters along with different packet lengths were investigated in [21]. Results of this study depends on the ratio of data and voice packet sizes.
Numerical values from analytical equations along with simulation experiments, described in Chapter 6, resulted in a collection of statistical measurements which provides an intuitive approach to the design of such networks. The presented results also provide a quantitative basis for more efficiently using the resources while maintaining an acceptable voice speech.

Other research related subjects are needed to include preemptive schemas and their effects on different switching models. It may be needed to establish special routing and protocol schemas in the event of voice and data traffic recovery due to partial transmission. It is shown [98] that, for two different arrival rates with no priority imposed on all class of traffic, there exists an optimum switching probability (local control) that minimizes the average system waiting time.

Network computer is an integrated part in the computer networks. To be general purpose problem solvers in a variety of environments, network computers need distributed procedures for scheduling competing task forces.
Several task force scheduling techniques have recently been proposed. In contract bid scheduling [99], the resource required for a requested task are broadcast until an idle node with enough resources responds. Task forces are scheduled on task at a time.

In diffusion scheduling [100], requests to load tasks are passed to neighboring nodes until an idle node is found. Neighbors may be chosen randomly or by following a fixed cycle [101] through the nodes. Requests tend to migrate from areas of high activity to areas of low activity. In fixed assignment scheduling [82,102], a programmer must explicitly specify the node that will execute each task.

Wave Scheduling, discussed in Chapter 7, differs from other proposed task force scheduling techniques in several ways. First, since the assignment is dynamic and not fixed once and for all, the reliability of task force scheduling is improved over that for fixed assignment schemas.

Secondly, Wave Scheduling includes a mechanism for avoiding static deadlock. In diffusion scheduling, on the other hand, static deadlock can occur because competing task forces are not made aware that each
holds resources which the others need. Fixed assignment scheduling can also lead to deadlock if different programmers try to use the same network nodes simultaneously. Contract bid scheduling also does not include any provisions for detecting nor avoiding deadlocks.

Finally, Wave Scheduling is an extensive scheduling technique. Increasing the site of the host network has not effect on the basic scheduling procedure. In fixed and cyclic diffusion scheduling, on the other hand, changes in network topology are catastrophic because they can make it impossible to schedule some task forces even when plenty of nodes are actually available. Also, as the number of network nodes (and consequently the number of network users, increases, fixed assignment schemas become unwieldy and inefficient).

Wave Scheduling is currently being implemented in the MICROS Operating System of the MICRONET network computer.

The features of task allocation in a time-critical environment are also discussed. An allocation model to allocate the software tasks into processors in
distributed computing system for time-critical applications is discussed in Chapter 7. The allocation model employs the branch and bound technique and allows the use of many constraints to reduce the IPC and the task waiting time.

The allocation result was tested on the MBD case study simulator. The simulator result shows that the task waiting time is a major contributor to PTP time. The allocation model can reduce the waiting time by the use of task exclusive matrix, task size and the number of enablements of each task. By using these constraints on the model, a better balanced allocation assignment A-1 was obtained, which was shown to have a much improved PTP time execution requirement for an important thread in the case study.

Many advances in modeling and measurements of different types of computer networks have been made since this concept emerged in the late sixties. We described in this thesis many of the recent techniques and illustrated their use by calling on specific examples. It is clear, however, that we are far from having answered all needs. In the following, we
briefly discuss some open areas, just to name a few, that emerged from studying a subject and where more work is in order.

In addition to the many design problems (topological, capacity assignment, routing, etc.), network behavior is believed to be greatly affected by the flow and congestion control algorithms in use; the modeling and analysis of these techniques are still in their infancy. Perhaps the most elaborate work so far achieved consists of the analysis of a dynamic decision process relating to the number of messages outstanding in the network to the destination buffer occupancy in an environment where the changes in the messages do not affect the network response time. Unfortunately, such limitations in the model render it only a first approximation. In summation, there is a definite shortage of analytic work in this area. The measurement of end-to-end protocol has been also extremely limited due to the difficulty in interpreting the results which depend not only on the characteristics of the communication subnet.
Although in this thesis we have mostly focused on packet-switched networks suitable mainly for computer-to-computer communications, we observe today an important trend towards the design of integrated packet-circuit switching communications to satisfy a broader class of users with a large variety of traffic characteristics (interactive data, long files or facsimile, real time data such as digitized voice, etc). The design an analysis of such systems are only at their start.

Finally, we get to the problem of network interconnection. It is all too evident that the behavior of communication network varies with its type (land based, ground radio, satellite), as well as, with the specific implementation and control techniques used. The interconnection of networks exhibits the need for a simple and accurate characterization and classification of these networks and for the development of analytic tools which help predict the performance of various interconnection topologies. Moreover, measurement facilities which allow for coordination, control and collection of simultaneous measurements in several interconnected networks in view of future internetworking experiments will be of utmost
importance. These experiments include among others the
evaluation of interwork protocols and the end-to-end
user performance in a multinetwork environment.

Thus, we conclude here that in this
exciting area of measurements and optimization of data
communication networks, we are still faced with many
problems of the most challenging kind.


20. T. G. Gruber, Delay Related Issues in Integrated Voice and Data Transmission, Sixth Data
Communications Symposium, November, 1979.


29. L. Kleinrock, On Flow Control in Computer Networks, ICC '78 (1978) 27.2.1-27.2.5.


- 122 -


53. G. L. Fultz, Adaptive Routing Techniques for


75. M. B. Kerke and C. L. Saxena, Finite Buffer Behavior


85. A. M. VanTilborg and L. D. Wittie, High-Level Operating System Formation in Network Computers,


Galina Peyros was born to Esther and Alexander Furman on December 21, 1958, in Kaunas, Lithuania. In 1979 she received an equivalent of B. A. in Management Information System Engineering from Riga Polytechnical Institute, Riga, Latvia. In 1979 she emigrated to United States of America, and in fall 1980 she entered Lehigh University. Since 1982, she has been working at Western Electric in Reading, Pennsylvania. Her work there involves development of computerized Production Control Systems.