

NRL Report 7846

Improvements in Routing for Packet-Switched Networks

CALDWELL MCCOY, JR.

*Advanced Projects Group
Acoustics Division*

February 18, 1975



NAVAL RESEARCH LABORATORY
Washington, D.C.

Approved for public release; distribution unlimited.

This report is a facsimile of a dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the School of Engineering and Applied Science, George Washington University, Washington, D.C., 1975. The work for this dissertation was directed by Raymond L. Pikholtz, Professor of Engineering and Applied Science.

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM															
1. REPORT NUMBER NRL Report 7846	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER															
4. TITLE (and Subtitle) IMPROVEMENTS IN ROUTING FOR PACKET-SWITCHED NETWORKS		5. TYPE OF REPORT & PERIOD COVERED Interim report on a continuing NRL problem.															
		6. PERFORMING ORG. REPORT NUMBER															
7. AUTHOR(s) Caldwell McCoy, Jr.		8. CONTRACT OR GRANT NUMBER(s)															
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Research Laboratory Washington, D.C. 20375		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS NRL Problem S01-61 ARPA Order 2275															
11. CONTROLLING OFFICE NAME AND ADDRESS Department of Defense Defense Advance Research Projects Agency Arlington, Va. 22209		12. REPORT DATE February 18, 1975															
		13. NUMBER OF PAGES 121															
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified															
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE															
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.																	
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)																	
18. SUPPLEMENTARY NOTES																	
19. KEY WORDS (Continue on reverse side if necessary and identify by block number)																	
<table style="width: 100%; border: none;"> <tr> <td style="width: 33%;">Adaptive routing</td> <td style="width: 33%;">Queueing</td> <td style="width: 33%;">Throughput factor</td> </tr> <tr> <td>ARPA computer network</td> <td>Pocket switching</td> <td>Deterministic routing</td> </tr> <tr> <td>Computer network performance</td> <td>Priority assignments</td> <td>Stochastic routing</td> </tr> <tr> <td>Computer networks</td> <td>Message throughput</td> <td>Routing algorithms</td> </tr> <tr> <td>Communication networks</td> <td>Average message delay</td> <td style="text-align: right;">(Continued)</td> </tr> </table>			Adaptive routing	Queueing	Throughput factor	ARPA computer network	Pocket switching	Deterministic routing	Computer network performance	Priority assignments	Stochastic routing	Computer networks	Message throughput	Routing algorithms	Communication networks	Average message delay	(Continued)
Adaptive routing	Queueing	Throughput factor															
ARPA computer network	Pocket switching	Deterministic routing															
Computer network performance	Priority assignments	Stochastic routing															
Computer networks	Message throughput	Routing algorithms															
Communication networks	Average message delay	(Continued)															
20. ABSTRACT (Continue on reverse side if necessary and identify by block number)																	
<p>Adaptive routing algorithms in store-and-forward communication networks are demonstrated to decrease average message delay, to increase message throughput, and to decrease the number of undelivered messages. Comparisons of existing adaptive routing techniques are presented to provide the basis for extension of the theory in two areas: priority assignment for messages reaching an assigned aging threshold, and prevention of messages looping within the network.</p> <p>Optimization of a defined measurement parameter, throughput factor, $\phi(t,k)$ is performed in terms of the priority threshold setting k. A closed form solution for $\phi(t,k)$ is obtained on</p> <p style="text-align: right;">(Continued)</p>																	

19. Key Words (Continued)

Store-and-forward networks
Computer simulation of network routing strategies

20. Abstract (Continued)

a 3-node network assuming infinite buffers and fixed routing. Included are simulations on damaged and undamaged systems for specific inter-connected networks with finite buffers and adaptive routing. Comparisons are made for the predicted $\phi(t,k)$ versus the simulated results on an undamaged network.

TABLE OF CONTENTS

	Page
ACKNOWLEDGMENTS	ii
TABLE OF CONTENTS	iii
LIST OF TABLES	vii
LIST OF FIGURES	ix
ABSTRACT	x
I. INTRODUCTION AND OUTLINE	1
1.1 Historical Background	1
1.2 Problem Considered	4
1.3 Outline of the Dissertation	5
II. ROUTING STRATEGIES	6
2.1 Routing Classification	6
2.2 Deterministic Routing	8
2.2.1 Flooding	8
2.2.2 Selective Flooding	8
2.2.3 Fixed Routing	8
2.2.4 Split Traffing	8
2.2.5 Ideal Observer	8
2.2.6 Ideal Routing Table and Shortest Path	8
2.2.7 Dynamic Programming	9
2.2.8 Butrimenko's Method	9
2.3 Stochastic Routing	9
2.3.1 Random Routing	10
2.3.2 Isolated	10
2.3.3 Distributed	10
2.3.4 Baran's "Hot Potato"	10
2.3.5 Backwards Learning	10
2.3.6 Negative Reinforcement	11
2.3.7 Bi-adaptive	11
2.3.8 Superposition	11
2.4 Flow Control Routing	11
2.4.1 Isarithmic Network	11
2.4.2 Buffer Storage Allocation Schemes	11
2.4.3 Special Route Assignment Algorithms	11
2.5 Adaptive Routing	12
2.6 Routing Scenarios Compared in this Research	12
2.6.1 ARPA Routing Strategy	13
2.6.2 Shortest Queue+Bias+Periodic Updating (SQ+B+PUD)	14
2.6.3 Pre-assigned Links (PAL)	15
2.6.4 Backwards Learning (BL)	16

TABLE OF CONTENTS (Continued)

	Page
2.6.5 Negative Reinforcement (NR).....	17
2.6.6 Superposition (SP).....	17
2.6.7 Bi-adaptive (BA).....	17
2.6.8 No Updating (NU).....	18
2.6.9 Dynamic Programming (DY).....	18
2.6.10 Last M Nodes Visited (LMNV).....	18
2.6.11 Best Three Links (BTL).....	18
2.6.12 Priority Assignment Based on Threshold (km).....	18
2.6.13 MOD ₁	19
2.6.14 MOD ₁ +50%.....	19
III. PRIORITY ASSIGNMENT STRATEGIES.....	20
3.1 Priority Queues.....	20
3.2 Waiting Time Distribution in a Two Class Priority System.....	21
3.3 Priority Assignments.....	23
3.4 Minimizing the Average Message Delay.....	24
3.5 Throughput Factor $\phi(t,k)$	26
3.5.1 Numerical Solution for (k_{op}).....	33
IV. SIMULATION ON AN 8-NODE HIGHLY CONNECTED NETWORK.....	39
4.1 Computation of Message Volume (MV).....	39
4.2 Objectives of the Simulation.....	40
4.3 No Link or Node Failures.....	41
4.4 Simulation With 50% Link Destruction.....	46
4.5 Routing Tables.....	49
4.6 Summary of 8-Node Simulation.....	51
V. CDC-3800 SIMULATION ON A 19-NODE ARPA NETWORK.....	55
5.1 19-Node ARPA Simulation.....	55
5.2 ARPA Network With 3 Links Disabled.....	60
5.3 ARPA Network With One Node Disabled.....	63
5.4 Summary of 19-Node ARPA Network.....	66
VI. THEORETICAL MODEL.....	68
6.1 Average Message Delay.....	68
6.1.1 Average Message Delay With No Priorities in the System.....	68
6.1.2 Average Delay for a Two Class Priority System.....	69
6.2 Effect of a High Utilization Factor ρ	72
6.3 Effect of Priority Assignment.....	72
6.3.1 Balking and Reneging.....	74
6.3.2 Throughput Factor $\phi(t,k)$	75
6.4 Comparison of Predicted to Simulated Results.....	75
VII. SUMMARY AND CONCLUSIONS.....	78
7.1 Summary.....	78
7.2 Conclusions.....	78
7.3 Future Work.....	79

TABLE OF CONTENTS (Continued)

	Page	
APPENDIX		
A	QUEUEING THEORY AND FORMULATIONS	80
A.1	Queueing Theory	80
A.2	Imbedded Markov Chain	82
A.3	Pollaczek-Khintchine Formula	82
A.4	Waiting Time Distribution	83
B	ADAPTIVE ROUTING PROGRAM FOR COMPUTER SIMULATION . .	90
B.1	Flow Diagram of Adaptive Routine	90
B.2	Program ARPSIM	93
B.3	Explanation of Variables	102
B.4.1	Simulation Runs for the DDP-24 (No Destruction of Network)	105
B.4.2	Simulation Runs for the DDP-24 (With 50% Destruction on Links, No Node Damage)	106
C	SIMULATION ON THE CDC-3800	107
C.1	Simulation Runs for the CDC-3800 (No Destruction of Network)	107

LIST OF TABLES

Table	Page
3.1 Fraction of Undelivered Messages $v(t,k)$	35
3.2 Fraction of Undelivered Messages $v(t,k)$	36
3.3 Fraction of Undelivered Messages $v(t,k)$	37
4.1 Program Algorithms for DDP-24 Simulation.....	41
4.2 Average Time for Messages in an 8-node Highly Connected Network.....	43
4.3 Message Throughput in an 8-node Highly Connected Network.....	44
4.4 Average Time for Messages on an 8-node Network With 50% Link Destruction.....	47
4.5 Message Throughput for an 8-node Network With 50% Link Destruction.....	48
4.6 Routing Tables for DY Routine With 50% Disabled Links and $[1/5 MV]=5$	52
4.7 Routing Tables for BL Routine With 50% Disabled Links and $[1/5 MV]=5$	53
5.1 Program Algorithms for CDC-3800 Simulation.....	56
5.2 Average Time for Messages With a Uniform Input Distribution	57
5.3 Message Throughput for Messages With a Uniform Input Distribution.....	57
5.4 Number of Messages Finding the Queue Full	58
5.5 Average Time for Messages With a Poisson Input Distribution.....	59
5.6 Average Time for Messages With a Poisson Input Distribution.....	59
5.7 Message Throughput for Messages With a Poisson Input Distribution	60
5.8 Message Throughput for Messages With a Poisson Input Distribution	60

LIST OF TABLES (Continued)

Table	Page
5.9 Average Time for Messages on the 19 node ARPA Network With Three Links Disabled	61
5.10 Message Throughput for the 19 node ARPA Network With Three Links Disabled	62
5.11 Message Throughput for the 19 node ARPA Network With Three Links Disabled	62
5.12 Average Time for Messages on a 19 node ARPA Network With One Node Disabled	65
5.13 Average Time for Messages on a 19 node ARPA Network With One Node Disabled	65
5.14 Message Throughput for a 19 node ARPA Network With One Node Disabled	66
5.15 Message Throughput for a 19 node ARPA Network With One Node Disabled	65
6.1 Computed vs. Simulated Throughput Factor $\phi(t,k)$	76

LIST OF FIGURES

Figure	Page
2.1 5 Node Network	14
3.1 Three Node Tandem Net.....	26
3.2 Flow Pattern in a 3 Node Net with a Single Destination Node	28
3.3 Fraction of Undelivered Messages vs. Priority Setting	34
3.4 Fraction of Undelivered Messages vs. Priority Setting	36
3.5 Fraction of Undelivered Messages vs. Priority Setting	38
4.1 Highly Connected 8 Node Network	39
4.2 Average Message Delay for 8 Node Net	42
4.3 8 Node Network With 50% Disabled Links	45
4.4 Average Message Delay for Disabled 8 Node Net	46
5.1 ARPA Network 19 Nodes	55
5.2 Average Message Delay for 19 Node Net With Uniform Input Distribution.....	56
5.3 Average Message Delay for 19 Node Net With Poisson Input Distribution.....	58
5.4 ARPA Network With Three Links Disabled	61
5.5 Average Message Delay on 19 Node Net With Three Links Disabled.....	62
5.6 ARPA Network With One Node Disabled.....	64
5.7 Average Message Delay on 19 Node Net With One Node Disabled	64
5.8 Average Message Delay on 19 Node Net With One Node Disabled	64
6.1 Computed vs. Simulated Throughput Factor $\phi(t,k)$	77

ACKNOWLEDGMENTS

I am deeply indebted to Prof. Raymond Pickholtz for suggesting this topic and for his infinite patience in guiding my research efforts. To Mr. William Finney of the Naval Research Laboratory I am sincerely grateful for his support and recommending me for the Edison Memorial Graduate Training Program that made this course of study possible. My warmest thanks to my associate Dr. David A. Swick who afforded me the much needed sympathy and empathy throughout this program.

To my wife Doris, and my children, Michele and Wanda, I offer my most heartfelt thanks and sincerest apology for their having to suffer through such a long period of "defacto" neglect over these many years.

Thanks are also due to Mr. Wendell Anderson for his helpful discussion regarding the numerical aspects of this work and Mrs. Louise Maiden for her untiring efforts in typing the numerous drafts of this study.

I would like to thank the NRL Graphic Arts Branch, Mr. W. H. Ramey and Mrs. D. Wilbanks along with their staff for their professional expertise on the completion of this dissertation. This research was supported by NRL Problem 81S01-61.

I. INTRODUCTION AND OUTLINE

1.1 HISTORICAL BACKGROUND

The seventies has ushered in the era of computer communication networks that are essentially networks of computers interconnected for the purpose of resource sharing. The concept of resource sharing, wherein users can access the vast software and computer hardware on a global basis is a fitting extension of the time-sharing systems of the sixties. The unrelenting pursuit of more powerful, faster and versatile computer capability has resulted in the union of time-shared systems sharing each other's resources by interconnecting them over a network to effect a computer communication network.

Time-shared systems are not in the process of being phased out, but rather are being expanded depending on users need. Computer systems for airline reservations, hotel reservations, military fire control, credit checking, on-line banking and savings, etc., are in effect groups of remote terminals feeding a central processor, where each CPU is dedicated to a particular problem. The quest for bigger, faster, and more reliable computers is asymptotically bounded by the speed of an electron thru a conductor; hence, CPU access times or cycle times in nano-seconds is the upper limit and physical size of the computer becomes somewhat moot.

Space technology paved the way for miniaturization and high reliability by virtue of micro logic. The direct consequence of micro logic in the computer field was the mini-computer; this increased considerably the volume of computers along with their associated users and subsequently increased the demand for software. The demand for more sophisticated software, taxed the capability of the mini-computer; hence another boundary was being approached, that being a given computer complex could only be efficient for certain classes of problems, whether the complex was made up of mini or maxi computers or a combination of both. The problem of effecting a computer facility with maximum throughput, minimum cost, high reliability, remote accessing, computational versatility, computational efficiency, etc., was rapidly approaching an impasse. Dr. L. G. Roberts [RO67] was initiating the concept that existing computer systems could be better utilized when interconnected in a network so that the resources (hardware, software, and data) could be shared.

The construction of a computer network enabled each computer system oriented to some specialized effort to be interconnected on a network, but each system maintains its autonomy. The result of this interconnecting of computer systems saves duplication of effort and hopefully minimizes cost to the user, while maximizing his computational efficiency.

The computer communication network while heralding in the era of resource sharing brings with it many new problems associated with computer networks. Computer network problems reduce essentially to the problem of moving messages from source S to destination D in a minimum time frame (error free) and is very similar to a transportation or traveling salesman problem.

The computer network should look transparent to the individual computer system and provide the necessary message handling capability required to transmit and insure rapid delivery of error free messages from source to destination. Since computer networks are not fully connected for economic consideration, there exists a requirement for storing and forwarding of messages between relay nodes within the network and as a consequence a routing strategy for rapid delivery of messages that can adapt to network traffic loads or blockage must be established. Thus, the problem, simply stated but not simply solved, is the storing and forwarding of messages (packets) along some given network configuration.

A historical review of the evolution of computer networks is contained in the work by Fultz [FU72] and NBS bibliography [NBS73]. It will suffice for the present to use the definition by Cole [CO71] for a "computer network": "A computer network is a set of interconnected processors which can be utilized jointly in a productive manner, but which normally are controlled by separate operating systems, and can perform in an autonomous manner."

The analytic treatment of stochastic flow of message traffic in connected networks of communication centers was studied by Kleinrock [KL64]. Kleinrock considered networks that were channel-capacity limited, and his measure of performance was taken to be the average delay encountered by a message in passing through the net. He addressed questions pertaining to the assignment of channel capacities, effect of priority discipline, choice of routing procedure, and design of topological structure. Although Kleinrock's initial efforts preceded present day computer networks, his concepts are the basis for most analytic studies in this area. The system to which he directed his techniques were associated with the automatic telegraph switching system of Western Union for Air Force military traffic [VER58].

Kleinrock used the discipline of the early investigators of queueing processes, Johannsen,¹ Erlang [BRO48] and numerous other contributors to the early development of queueing theory (Molina [MO22, 27] and O'Dell [O'D20,27]) and incorporated these disciplines along with modern day researchers (see Kleinrock [KL64] for further listing) to study communication nets. His efforts are notable in establishing some of the mathematical rationale used in evaluating communication networks.

The work of the Rand Corporation in a series of memoranda titled "On Distributed Communications to a Low-Data-Rate Military Command and Control System" was primarily the work of Baran [BA64A-64H]. Other contributors in the Rand study were Boehm [BO64] and Smith [SM64] and the initiation of store-and-forward message switched digital networks. This series of memoranda brought out Baran's definition of a "packet" along with his "hot potato" routing algorithm.

The Rand series was primarily interested in the survival of communications after an attack on the system. The series concerned itself with stochastic adaptive routing techniques that used information from messages passing through the network to adjust routing tables; deterministic techniques, which use dynamic

1. Reference to Johannsen's work can be found in [Brockmeyer 48].

programming or graph-theoretic algorithms to recalculate changes in the tables from observed changes in the network.

The groundwork set down in the Rand series led to the development of computer communication networks, the most ambitious effort undertaken to date being the Advanced Research Projects Agency (ARPA) Computer Network, or more simply the ARPA NET. The evolution of this network is rather extensive and a very good bibliography on the ARPA NET is given by Wolf [WO73], who also gives an excellent overview of the ARPA network.

The prime mover behind the creation, growth, and development of the ARPA computer system is credited to Roberts [RO67]. He indicated that existing computer systems could be more effectively utilized by interconnecting them in a network so that their resources could be shared. The idea of resource sharing was not entirely a new concept, but instead a natural outgrowth of earlier interconnected systems, such as the Semi-Automatic Ground Environment (SAGE) air defense system of the fifties [Everett 57].

There were numerous computer-connected systems that existed prior to ARPA and are historically reviewed by Fultz [FU72]. One of the early systems was the: American Airlines SABRE Reservation System [PL61], that led to the present systems used by other airlines, hotels, etc. The need for improved military data communications led to the Automatic Digital Network (AUTODIN) Communications System in 1963 [Millar 68]. This system depended on both line and message switching facilities with added constraints of survivability and vulnerability.

In addition to computer systems listed by Fultz, there is a tutorial write-up of four currently operating computer-communication networks [Schwartz 72]. They include the TYMNET network, the G.E. Information Services Network, the NASDAQ over-the-counter stock-quotation system, and the Computer Sciences Infonet. These systems all contain some mixture of computers and networks and possess to various degrees the need for store-and-forward capability. This store-and-forward technique of sending and retrieving data as opposed to dedicated connections while handling data is the uniqueness of computer-communications. It should be noted that in the TYMNET network, they prefer to describe the function of their network as virtual line switching as opposed to message switching.

Computer networks, while solving possible cost effectiveness of computer systems by bringing massive hardware and software resources to users with moderate investments, also brought massive problems of network structure, handling and transmission of message routing procedures, and CPU protocols. This then is the challenge of the seventies, to extend to more users the vast computer resources on a global basis, with minimum cost, high reliability, remote accessing, maximum throughput, all with a minimum of effort and time for the user.

1.2 PROBLEM CONSIDERED

The intent of this research effort is to establish a tractable adaptive routing procedure applicable to store-and-forward computer-communication networks. The routing assignment will be on a node-to-node basis and will essentially be an extension of Baran's [BOEHM and BARAN 64] "hot potato" routing technique with priority assignments. The concepts of adaptive routing are not new, but are lacking in true adaptability due to network topology changes (new links or blocked links), or the "deadly embrace" of "looping" as well as "ping-ponging".* It is unlikely that there can be an all encompassing optimal routing technique for all network designs and all system constraints. The definition of optimal is open to debate, since it could be conditioned on cost, time-delay, guaranteed delivery, minimum path, maximum throughput, etc.

The definition of optimality that will be adopted here will be in the context given under dynamic programming principles as applied to optimum paths and states:

The optimal path from X to A has the property that, for any node Y along the path, the remainder of the path is the optimal path from Y to A . If we define $N(X)$ as the set of neighbors of X connected to X by links, we have

$$D(X) = \text{Min}_{Y \in N(X)} [d(X, Y) + D(Y)]. \quad (1.1)$$

The procedure will be to simulate some routing strategy for a given network topology or general class of networks and measure the effective adaptability of the routing, when it is compared against various performance measures. Performance will be measured on the basis of average message delay, throughput, and percent of undelivered messages. When dealing with an operational network, a measurement of average message delay includes such items as: queuing delays, line and node blockages, node processing delays, and message retransmission due to channel errors.

The measure of performance will generally be obtained from computer simulation, since mathematical models do not lend themselves to analysis of network performance. This absence of a mathematical model to effectively evaluate a network in closed form, means one has to represent the system parametrically, simulate the system, analyze the results, and ascertain what changes should be made.

The major emphasis in this research will be to demonstrate how some relatively simple add-ons to already existing adaptive routing techniques can decrease the average message delay and correspondingly increase message throughput in the network. A further objective of this study will be to assess the effects of priority assignment to messages that have reached some specified aging threshold while in the network and note the effects of such priority assignment on network performance. The technique will involve working on a node-to-node

* "Looping" is when a message repeatedly travels the same set of nodes. "Ping-ponging" is where a message goes back and forth between the same two nodes.

basis within some specific network topology and optimize the routing while attempting to minimize the hardware and software requirements at the nodes.

1.3 OUTLINE OF THE DISSERTATION

Chapter II is an overview of routing strategies and routing classifications. Chapter III treats priority assignment strategies and optimization of a defined measurement parameter, throughput factor, $\phi(t,k)$ and how it relates to the priority threshold setting k . A closed form solution for $\phi(t,k)$ is obtained on a 3-node network assuming infinite buffers and fixed routing. The problem of analytically modeling a store-and-forward system with adaptive routing is also addressed in Chapter III.

Chapter IV and V are simulation studies on an 8-node and 19-node network respectively. The simulations are a means of verifying the performance of some selected algorithms with the modifications suggested in this study. The adaptability of the routing algorithms were tested on undamaged and damaged networks of the class mentioned above. Chapter VI is a comparison of the numerical throughput factor to the simulated one for the 19-node ARPA network.

This work is summarized in Chapter VII along with final conclusions and recommendations for future work.

II. ROUTING STRATEGIES

An excellent overview of the basic routing problem, routing requirements, and routing classification is given by Fultz [FU72] and a brief summary of his efforts are worth reiterating here.

The basics of the deterministic routing problem have been formulated mathematically and some general techniques exist for solutions to some specific classes of problems which have been cited by Ford and Fulkerson [FO62]. Their solutions are associated with maximum flow problems, the Hitchcock transportation, project planning, and the shortest route problem.

The standard switched circuit communication network has a routing problem associated with finding free communication links that maintain a fixed path for the duration of the message flow. On the other hand, in a message switching computer communication network, the messages must be routed under a different strategy. Message routing involves queueing delays due to the store-and-forward aspects of routing strategies. This class of nets utilizes message delay estimates as a means of measuring network performance; hence, routing strategies are of prime concern.

General requirements for message routing can be listed in three categories. One, the routing technique should adapt to changes in network topology resulting from node or line failures. Second, routing strategy should adapt to varying source-destination traffic loads to insure minimal message delays. Third, contingency plans should exist for dropping of undeliverable messages; i.e., messages should be dropped after some pre-set time lapse.

Store-and-forward routing is in the same context as packet-switched networks. In either vernacular, the basic unit of information exchange is the "packet" and varies from 1008 to 1024 bits depending on whether the header is included. The ARPA system accepts messages up to 8095 bits in length and divides these into 1008 bit packets.

2.1 ROUTING CLASSIFICATION

Before analyzing some of the popular routing algorithms, particularly those pertaining to adaptive techniques, it would be instructive to list the routing classification.

It was stated previously that there were three major categories for classifying routing techniques and they were given by Fultz, Boehm and Mobley [BO66], along with a quick summary. Fultz's list augmented with the RAND series are as follows:

- A. Deterministic Routing
 - 1. Flooding
 - a. All links
 - b. Selective links

2. Fixed routing
 - a. Table look-up
 3. Split traffic
 - a. Optimum bifurcated
 - b. Suboptimum bifurcated
 4. Ideal observer (scheduling problem)
 - a. Present
 - b. Future
 5. Ideal routing table and shortest path
 6. Dynamic programming (solution of the shortest path)
 7. Butrimenko's method (of updating shortest paths)
- B. Stochastic Routing
1. Random routing
 2. Isolated
 - a. Local delay estimates
 - b. Shortest queue + bias
 3. Distributed
 - a. Periodic updating (nearest neighbor)
 - b. Asynchronous updating (Percolation)
 4. Baran's "hot potato"
 5. Backwards learning
 6. Negative reinforcement
 7. Bi-adaptive
 8. Superposition
- C. Flow Control Routing
1. Isarithmic network
 2. Buffer storage allocation schemes
 3. Special route assignment algorithms

Any attempt to go into any significant detail of all the possible routing schemes in the various categories would be too much of an undertaking and would detract from the issue at hand and can be found in the cited references. The listing is to help keep techniques in their proper perspective and a given routing scheme will be elaborated on when deemed appropriate for a given discussion.

A brief resume of some of the routing techniques will be given next; details can be found in Fultz [FU72], Boehm and Mobley [BO66, 69], or Fultz and Kleinrock [FU71].

2.2 DETERMINISTIC ROUTING

Deterministic routing is where the routes are selected from the network structure, based upon the minimum time delay to reach a destination under ideal, loop-free conditions.

2.2.1 Flooding

Each node receiving or originating a message transmits it over every outgoing link, after checking to see that it has not previously transmitted this same message, or that it is not the destination of the message. Inefficiency is the price paid for this technique.

2.2.2 Selective Flooding

Each station will transmit only over links heading in the general direction of the destination. Storage of tables at each node for the best outgoing link is the price paid by this method. In addition, if nodes and links are disabled, some sources and destinations would be connected only by round-about paths excluded from consideration by the selective flooding technique.

2.2.3 Fixed Routing

Fixed routing specifies a unique path over a selected set of links $\pi(N_i, \dots, N_j)$ for a given source node N_i and destination node N_j . This is accomplished by a table look-up from a routing table contained at each node. Fixed routing requires completely reliable nodes and links.

2.2.4 Split Traffic

This technique has been reported by Fultz [FU72], and as opposed to fixed routing, allows traffic to flow over more than one set of paths between a given source-destination node pair. This splitting of traffic is called traffic bifurcation. The suboptimal bifurcation is used since it's more simple in structure and requires less computation time than the optimal.

2.2.5 Ideal Observer

This technique is essentially a scheduling problem. Each new packet entering the system has its minimum route computed for the given source-destination pair. This newly computed route is based upon the complete *present* information about the packets already in the network and their known routes. If the ideal observer has information about *future* events, this is also utilized in computing the route. This technique is more theoretical than practical, but serves as a basis for comparing other routing techniques.

2.2.6 Ideal Routing Table and Shortest Path

Let X , Y , and A denote arbitrary nodes in a network, and let $d(X, Y)$ denote transit time along the link.

If node A is the destination, the ideal routing table entry gives the shortest distance from node X to node A starting along link XY , and may be represented as a function $R(X,Y)$ of X and Y alone. If one defines $D(Y)$ as the distance along the shortest path from Y to A , it follows that

$$R(X,Y) = d(X,Y) + D(Y), \text{ for node } A. \quad (2.1)$$

Hence, if $D(Y)$ is determined for all nodes Y , then one can compute the table entries $R(X,Y)$ of the ideal routing table from equation (2.1). The report by Boehm and Mobley [BO66] elaborates on many of these routing procedures.

2.2.7 Dynamic Programming

The dynamic programming principle of optimality states:

The optimal path from X to A has the property that, for any node Y along the path, the remainder of the path is the optimal path from Y to A . If we define $N(X)$ as the set of neighbors of X connected to X by links, then

$$D(X) = \text{Min}_{Y \in N(X)} [d(X,Y) + D(Y)] \quad (2.2)$$

Dynamic programming requires storing routing tables at each node as well as a map of the network. Any changes in the net are sensed by neighboring nodes that can communicate these changes to all other nodes. Dynamic programming is treated in Bellman and Dreyfus [BEL62].

2.2.8 Butrimenko's Method

Butrimenko's [BU64] method of updating the shortest path uses perturbation techniques that adjust to simultaneous, or near simultaneous, failures or additions of nodes and links.

The Butrimenko method checks for possible link or node failures in the network by verifying at each node N_i along a message's route whether or not it is still the minimum route to take.

2.3 STOCHASTIC ROUTING

Stochastic routing operates on a probabilistic set of decision rules. Routes are based on the actual state of the network or estimates of the present state. Stochastic routing as well as deterministic routing were initially suggested by Boehm [BO66].

2.3.1 Random Routing

Random routing uses a decision rule for next node to visit based on some probability distribution over the set of neighbor nodes. The set of nodes can be all or some selective set that are in the general direction of the message's destination.

Prosser [PR62] and Kleinrock [KL64] investigated some of these random routing techniques. They concluded that although such routing may be relatively unaffected by small changes in the net structure, they were highly inefficient in terms of message delay.

2.3.2 Isolated

The source of information available for developing the delay table estimates determines whether the routing technique belongs to the isolated or distributed class. If the only information available to a node comes via packets flowing through it, then the procedure is designated as isolated, or a local delay estimate.

Shortest queue + bias is a technique where a packet's route is selected by placing it in the shortest output channel queue. This technique is used in the ARPA system [Heart 70] and is extensively dealt with in Fultz's [FU72] work. This procedure will be further developed in section (2.6.2).

2.3.3 Distributed

Operates in the same basic way as the isolated technique. In this routing technique, neighboring nodes exchange status information as regards to delay and routing information. This information exchange is accomplished by special information packets; hence, routing data percolates throughout the network.

2.3.4 Baran's "hot potato"

In a manner analogous to selective flooding there is a technique whereby the table at each node has not only the best outgoing link for each destination, but also the second best through n th best, enumerating all n outgoing links. Thus an incoming message can be immediately routed out on the best available link at any given instant. This selection of outgoing links is called "hot potato" routing, and was extensively investigated by Paul Baran in the RAND Memoranda series [BA64A-64I].

2.3.5 Backwards Learning

In the report by Boehm and Mobley [BO66], the value of the hand-over number (HN) contained in a message received along the incoming link (LI) estimated the shortest time currently necessary to go from the source S to the present node X . The backwards learning technique (Used by Baran [BA64] in the "On Distributed Communications" series) utilizes the fact that HN also estimates the shortest time $T(S,LI)$ currently necessary to go from X to S , using LI as the outgoing link (LO). The value $T(S,LI)_{OLD}$ of the routing table at X is then updated by a simple averaging process.

2.3.6 Negative Reinforcement

Negative reinforcement is a class of techniques that penalizes various routing table entries by various amounts whenever the delay coming into a node is equal to the delay going out from said node via the same link. It adds some constant delay value to its old routing table values whenever this equality occurs. The intent of this operation is to reduce the “ping-pong” effect.

2.3.7 Bi-adaptive

Bi-adaptive is still a further synthesis of backwards learning and negative reinforcement, where table entries can only be increased by negative reinforcement and decreased by backward learning.

2.3.8 Superposition

Superposition is the linear combination of backwards learning and negative reinforcement.

2.4 FLOW CONTROL ROUTING

Flow control routing involves techniques to coordinate the network message routing algorithm and the activities of the software at source and destination nodes regarding sequencing of messages, network connectivity, buffer storage allocation, etc. Flow control routing was studied and classified by Fultz [FU72].

2.4.1 Isarithmic Network

Davies [DA71] proposed the isarithmic network as a solution to the network congestion problem. He observed that the level of traffic within the network could be expressed by the number of packets in transit. His solution was to prevent congestion by placing a limit on the total number of packets in the network. Thus, he defined an “isarithmic” network as one where the total number of packets is held constant. The idea was basically to have a fixed number of packets in the system either empty or carrying data. When data is removed from a packet an empty one is available to acquire new data.

2.4.2 Buffer Storage Allocation Schemes

Buffer storage allocation is the technique of coordinating network routing with that of congestion control algorithms to eliminate bottlenecks at the buffer queues. Effective buffer storage schemes enable high throughput levels for the net as well as acceptable average message delays. Buffer techniques are discussed by Zeigler [ZE71].

2.4.3 Special Route Assignment Algorithms

Route assignment algorithms are the attempts to avoid establishing instantaneous alternate routes in response to rapid changes in traffic flow within the network. They control the rate at which traffic is allowed

to increase on paths in a net and the interval before additional alternate routes are established. This subject has been investigated by Fultz [FU72], Kahn and Crowther [KA71] and Frank, et al [FR72].

2.5 ADAPTIVE ROUTING

The present day concept of adaptive routing can probably be attributed to the RAND memorandum by Boehm and Mobley [BO67]. Their objective was to reroute messages in a military communication system where the lines of communication had been damaged or destroyed. They were investigating an effective adaptive routing technique that would adjust the routing tables of the message-switching control system to a changing network topology. Present day computer communication networks need to employ the same adaptive routing strategies, not for safeguarding against attack, but for keeping data from getting blocked at overloaded nodes.

Summarizing the previous sections we can say that adaptive routing techniques fall into two basic classes: 1) Stochastic techniques that *estimate* the time required to reach a destination from observations of messages passing through nodes, and 2) deterministic techniques that *compute* the time required to reach a destination under ideal conditions based on network structure. Fultz lists a third technique, called flow control routing that incorporates buffer storage allocation schemes and special route assignment algorithms.

Routing requirements for packet switched networks can be listed briefly as follows:

1. Message routing should insure rapid and error-free delivery of messages.
2. The routing strategy should adapt to changes in the network topology resulting from node and communication link failures.
3. The routing technique should adapt to varying source-destination traffic loads.
4. Packets should be routed around nodes that are congested or temporarily blocked due to full storage.
5. Packets should be routed to their destinations via the fewest intermediate nodes.
6. A technique for detecting "loops and ping-ponging" should be incorporated in the routing strategy.
7. The routing technique should be as simple as possible to minimize hardware and software requirements at each node.

2.6 ROUTING SCENARIOS COMPARED IN THIS RESEARCH

The following algorithms are briefly outlined along with their associated acronyms. The outlines are intentionally brief, since there are ample references cited in this and previous sections.

Routing tables are initially calculated by computing the minimum path matrix on an empty network. Floyd's algorithm [FLO62] is generally used to solve the shortest paths for setting up the node delay tables. [Frank 71] has developed a computationally more efficient routing matrix that the author claims would yield

a near optimal routing strategy.* These initializing algorithms would be essentially optimum if the network either did not become too congested and there were no changes in net topology due to link or node failures.

In all cases of routing the following specifications should serve as a guide.

1. Route selection should be independently performed at each node, based on nearest neighbor information and traffic pattern in the immediate area.
2. Individual routing should be globally sensible.
3. Updating should be conditioned on queue size at the next best nodes; i.e., update if traffic (messages) start to overflow at a given queue.
4. Paths with fewest nodes to destination should be considered first.

2.6.1 ARPA Routing Strategy

The ARPA network utilizes Inter Message Processors (IMP's) that are autonomous to the system. Each IMP maintains its own routing table that it updates every half second using nearest neighbor estimates of delay to each destination. ARPA routing is adaptive, although it is not optimal. There is also an instability in the ARPA system caused by such things as node *A* routes messages to node *C* via node *B* and *B* believes the best path to *C* is via node *A*. This instability in routing is generally called ping-pong effect and in this example would be ping-ponging between nodes *A* and *B*.

The routing algorithm directs each packet to its destination along a path for which the total estimated transit time is minimum [Heart 70]. This path is *not* determined in advance; instead, each IMP individually decides which of its output lines should transmit a packet addressed to another destination. The selection is made by a relatively fast and simple lookup procedure, where, for each possible destination, an entry in the table designates the appropriate next leg. These entries reflect line or IMP trouble, traffic congestion, and current subnet connectivity. This routing table is updated every half second as follows:

Each IMP estimates the delay it expects a packet to encounter in reaching every possible destination over all of its output lines. The minimum delay estimate is selected for each destination and periodically (about twice a second) passes these estimates to its immediate neighbors. Each IMP then constructs its own routing table by combining its neighbors' estimates with its own estimates of delay to that neighbor. The estimated delay to each neighbor is based on both queue lengths and recent performance of the connecting communication network. For each destination, the table is then made to specify that selected output line for which the sum of the estimated delay to the neighbor plus the neighbor's smallest delay, is a minimum for a given destination.

The routing table is consistently and dynamically updated to adjust for changing conditions in the network. This essentially makes the system adaptive to fluctuations of line traffic, IMPs, and congestion. It is

* Optimal routing as used in this study implies routing that minimizes the average message delay.

not necessary for an IMP to know the topology of the network. Indeed, an IMP need not know the identity of its immediate neighbors. This enables the leased circuits to be reconfigured to a new topology without requiring any changes to the IMP's.

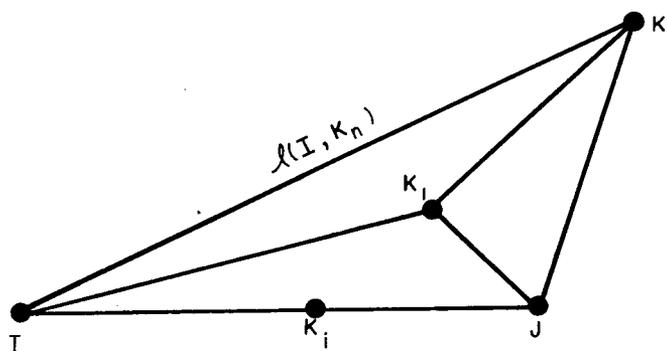
In the ARPA network the basic unit of information passed between any pair of nodes is called a "packet" with maximum size of approximately 1000 bits. Messages that originate at a HOST computer (any number of large computers serving the ARPA net are called HOST) have a maximum length of approximately 8000 bits. The IMP breaks these large message packets into 1000-bit segments or packets. These packets are then handled by the network as independent entities until they reach their destination node. Packets of a multi-packet message have headers associated with them for error checking, how many packets in a message and their ordering, and what their destination node is. When packets reach their destination node, they are reassembled before transferring to the destination HOST. Single packet messages are given higher priority than multipacket messages so that the network can support interactive users.

2.6.2 Shortest Queue + Bias + Periodic Updating ($SQ + B + PUD$)

The ARPA system uses an algorithm for routing called shortest queue + bias + periodic updating ($SQ + B + PUD$). This scheme uses minimum delay vectors to propagate from node to node any changes in the system. The routing strategy uses a combination of the present table entries and queue lengths at the node to choose the "best" outgoing link, and its scenario is as follows:

For each point in time, calculate minimum time delay (vector) to go from say node I to node J ; i.e., $\text{Min}(I, J)$ (Fig. 2.1).

Node I will ask all of its neighboring nodes K_i what is their min. delay vector; i.e., (K_i, J) .



All K_i 's are connected directly to I , but not necessarily to J .

Fig. 2.1 – 5 Node network

Take minimum of all neighboring nodes K_i and this establishes the following:

$$\text{Min}(I, J) = \text{Min} \{ (K_i, J) + l(I, K_i) + Q(I, K_i) \} \quad , \quad (2.3)$$

where

$(K_i, J) \equiv$ path length from node K_i to node J ,

$\ell(I, K_i) \equiv$ link time from node I to node K_i ,

$Q(I, K_i) \equiv$ queue length at node I on output link to node K_i ,

and

$K_i \equiv$ a neighboring node of I .

New routing table now uses the new delay values

$$\Delta t_i = \{ (K_i, J) + \ell(I, K_i) \} \quad (2.4)$$

Hence, the new routing table becomes:

$$\begin{array}{c|ccc} I & K_1 & K_2 & \dots K_n \\ \hline J & \Delta t_1 & \Delta t_2 & \dots \Delta t_n \end{array} \quad (2.5)$$

The next node K_ρ is chosen such that

$$Q(I, K_\rho) + \Delta t_\rho = \underset{i}{\text{Min}} \{ Q(I, K_i) + \Delta t_i \}, \quad (2.6)$$

Hence

$$\text{Min}(I, J) = SQ + B + PUD. \quad (2.7)$$

2.6.3 Preassigned Links (PAL)

Preassigned links (PAL) is a technique where outgoing links for incoming messages (either external or internal) are assigned as soon as messages get on the queue; i.e., buffer storage. Outgoing links are chosen from the routing tables; hence, for no updating, link assignment is fixed routing. When updating is employed, link assignments change according to what constitutes the new minimal path.

It would take considerably more programming to check messages in queue and to recalculate optimum path, than it takes to check available outgoing links against those messages that have that link assignment.

2.6.4 Backwards Learning (BL)

In the report by Boehm and Mobley [BO66], the value of the hand-over number (HN) contained in a message received along the incoming link (LI) estimated the shortest time currently necessary to go from the source S to the present node X . The *backwards learning* technique (used by Baran in the "On Distributed Communications" series) utilizes the fact that HN also estimates the shortest time $T(S, LI)$ currently necessary to go from X to S , using LI as the outgoing link (LO). The value $T(S, LI)_{OLD}$ of the routing table at X is then updated by a simple averaging process:

$$T(S, LI)_{NEW} = T(S, LI)_{OLD} + [K(HN-T_0)] \cdot [HN-T_0], \quad (2.8)$$

where

$$K(HN-T_0) = \begin{cases} k_1, & \text{if } HN-T_0 < 0 \Rightarrow \text{learning constant} \\ k_2, & \text{if } HN-T_0 \geq 0 \Rightarrow \text{forgetting constant,} \end{cases}$$

and

$$0 \leq k_1, k_2 \leq 1.$$

The new estimate becomes a certain fraction $K(0 \leq K \leq 1)$ of the value between the old and the current HN . This procedure includes a "learning constant" k_1 , used to decrease $T(S, LI)_0$, and a "forgetting constant" k_2 , used to increase it. If more sophisticated statistical estimations are employed, these would require extra information on each table entry, thus increasing the computer cycle requirement at each node and defeating the simplicity strived for in this study.

Backwards learning (BL) is then defined as:

$$T_N = T_0 + K[HN-T_0], \quad (2.9)$$

where

$$K = \frac{1}{2} .$$

The major disadvantages of the backwards learning technique are that it utilizes no direct information on the progress of messages through the system, and the indirect information supplied by the above-mentioned method is often inadequate or misleading. BL updates toward S and not toward D , and in cases where there is node failure either due to hardware or software failure, BL will not adjust to these new constraints and looping will occur or "ping-ponging" between nodes until the message has reached its maximum time to be in the system and is subsequently discarded. "Ping-ponging" occurs whenever $LI = LO$.

NOTE: HN is the actual time of message transit from point S to X and T_0 is value in table at X_1 to go from X to S .

2.6.5 Negative Reinforcement (NR)

In the following topology, if in going from node S to node D via J , and message is then sent back from J to I (Fig. 2-2), the new table value at J becomes:

$$T_N(J, I, D) = T_0(J, I, D) + 2 \lambda(i, j) \quad (2.10)$$

and at node I

$$T_N(I, J, D) = T_0(I, J, D) + 2 \lambda(i, j). \quad (2.11)$$

Thus NR penalizes various routing table elements whenever $LI = LO$.

The disadvantage of negative reinforcement is that it has no way of adjusting to improvements, such as replacement of a damaged link or the creation of a new link. Routing table entries could have little or no relation to the actual transmission times between nodes.

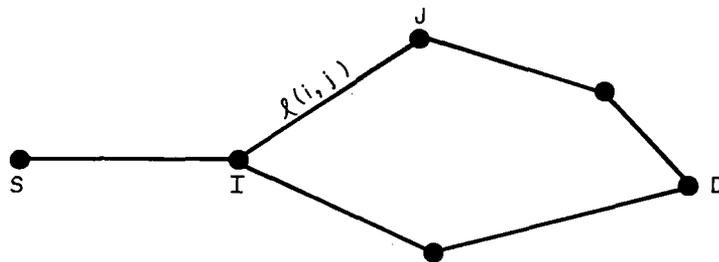


Fig. 2.2 - 6 Node Network

2.6.6 Superposition (SP)

Superposition \Rightarrow Backward Learning and Negative Reinforcement. This would be equation (2.9) used in combination with equations (2.10) and (2.11), where the latter two equations are used to offset “ping-ponging.”

2.6.7 Bi Adaptive (BA)

Bi Adaptive \Rightarrow Backward Learning and Negative Reinforcement, such that in equation (2.8) $K_1 = 1/2, 1/3, 1/4$, etc. (some assigned value) and $K_2 = 0$.

2.6.8 No Updating (NU)

On a lightly loaded net the ideal routing tables would be essentially optimum routing. The no updating routine (NU) was used to evaluate the performance of other routing algorithms.

2.6.9 Dynamic Programming (DY)

This programming technique is analogous to the “ideal” observer and assumes that its knowledge of what links and/or nodes are inoperable does not change. With this quasi all encompassing overview of the network, this type of programming should yield the optimum path from any source S to any destination D via any intermediate nodes N_i with a minimum time delay.

2.6.10 Last M Nodes Visited (LMNV)

LMNV is a technique in the program routing strategy to prevent the message from returning to the last node it has visited. This scheme can be extended to include the last M nodes visited ($M = 1, 2, \dots, K$) and will be designated as (LMNV). Preventing messages from taking routes previously visited, is intended to reduce message transit time by cutting out the “looping” or “ping-pong” effects.

2.6.11 Best Three Links (BTL)

There is a tag associated with each message such that when it is prevented from taking the last node visited, it will be given an option to take one of three best links from a given node N_i towards its destination node D . The use of last M nodes visited (LMNV) and best three links (BTL) are generally used together in the programming. If the message becomes trapped in the network where it can't proceed without backtracking over previous routes, it may be “killed” or discarded from the network and tallied as an undelivered message. The point at which it is discarded from the net is determined by a clocking

mechanism* that keeps a tally of how long a given message has been in the system and when this tally reaches some prescribed MDT_{MAX} and the message has not been delivered, then it is discarded from the system.

2.6.12 Priority Assignment Based on Threshold (km)

An aging parameter has been incorporated in the program whereby after a message has reached some percentage of MDT_{MAX} it is given a priority while in the queue so that it can be dispatched through the network on a priority basis. Once a message has been upgraded to a priority level it continues to have this priority at all succeeding nodes. The priority assignment is non-preemptive, in that a message being serviced at a given node will continue to be serviced until completion, regardless of its priority status. Note also that messages are not generated with a priority, they are only upgraded to a priority after having spent some time $X\%$ of MDT_{MAX} in the system ($D_T \geq km$).

The idea behind the upgrading or priority assignment is to enhance the delivery of messages already in the system. Messages already in the system represent some investment as regards to time, servicing, and use of system capacity. Once this investment has been made it would be desirable to expedite the handling of such messages when they become stalled in the network, due to looping, congestion at various nodes, etc. The level of setting the priority threshold km will be accomplished by simulation and ascertaining where message throughput peaks when network loading ρ increases, i.e., what level of thresholding is necessary to peak the throughput and minimize the time delay.

2.6.13 MOD_1

The routines associated with last M nodes visited (LNMV) and best three links (BTL) will be designated as modification sub one (MOD_1).

2.6.14 $MOD_1 + 50\%$

When the aging parameter that sets up priorities is included with any programming routine, it will be added on as some percentage; thus $MOD_1 + 50\%$ indicated that an aging parameter of $k = 50\%$ was used.

It should be noted here that the techniques of preassigned links (PAL), MOD_1 , and $MOD_1 + 50\%$ are strategies being offered in this study to improve existing routing algorithms. Further, these relatively simple add-ons are essentially asynchronous and serve as congestion control and as devices for "unblocking" heavily loaded networks. The effectiveness of the above mentioned schemes, as regards to minimizing the average message delay while increasing total message throughput, will be demonstrated in the simulation runs of Chapters IV and V.

* Clocking is not synchronous, but is a numerical incrementation given to each packet in the system during every operational cycle of the simulation routine.

III. PRIORITY ASSIGNMENT STRATEGIES

The analysis of queueing theory is essentially derived from probability theory and as such, queueing theory concerns itself with congestion due to random flows. The parameters of concern in a queueing system differ according to how the queues in a system are structured and the mode of operation of such systems. The general lack of knowledge of the system specifications renders extensive analysis to be almost intractable except for a few idealized systems. The limitations on the analysis is directly due to limitations of the mathematics itself. The system can't be completely specified due to no systematic analytic modeling and the analytical modeling has to be too restrictive if it is to be solved. The consequence of this dilemma results with the mathematical modeling of physically unrealizable systems.

This chapter attempts to bring into focus some of the standard analysis applicable to a two-class priority network. The analysis addresses the procedures for establishing the priority threshold and its subsequent effect on the network as regard to message delay and message throughput.

A review of queueing theory along with some of the standard queueing formulations are given in Appendix A.

3.1 PRIORITY QUEUES

The subject of priorities needs to be further investigated due to the special effects of assigning such priorities and the unusual impact such assignments have on the analysis problem. Priority queueing assumes that the class of messages now arriving at a given service facility, need special handling either within the queue or with their service requirements. The priority classification can be established with some parameter $p(1, 2, \dots, k)$, where 1 denotes the class with highest priority and k the lowest. The discipline by which a server selects the next message and serves it is termed the "priority discipline." The decision rules whereby any priority scheme must act on are generally covered by the following rules:

- A. Whether the server continues or discontinues the servicing of a message already being serviced.
- B. What message should be next selected once the server is free to take one.

The service selection may be made exogenously, i.e., dependent only upon knowledge of priority class to which a message belongs. Alternatively, the service selection may be endogenously, i.e., the decision could be based on the existing state of the system, such as type of message last serviced or the waiting time of the messages. Endogenous priority disciplines are the least known disciplines and are seldom treated in the literature; it is this class that we attempt to analyze later in this Chapter.

With exogenous priority disciplines the decision rule for servicing the next message depends only upon its priority class. A message of the i^{th} class if present is always taken for service before a unit of the j^{th} class ($i < j$). In the event a message of the j^{th} class is in service and a message of i^{th} class ($i < j$) arrives, further alternatives may be invoked.

- A. Preemptive: The service of the j^{th} class message is immediately interrupted and the service is given the i^{th} class message.
- B. Non-preemptive: The service of the j^{th} class message is continued to completion.

In this study, there will be no preemption; hence, no further discussions upon the manner by which the j^{th} class message is serviced on its re-entry, need be further examined. Service discipline for any class will be on a first-come-first-served (FCFS) basis and is the same as first-in-first-out (FIFO), where both systems are servicing the oldest message (in the queue) first. Jiunn Hsu [HS71] gives an overview of some of the standard service disciplines. James Martin [MA72] gives a very comprehensive treatment of queueing effects related to the design and analysis of data transmission systems.

There will be only two types of systems addressed in this report. The Poisson input, exponential service, and single-server ($M/M/1$) or the Poisson input, general service, and single-server ($M/G/1$). The notation of M is for the memoryless Markovian process and this is indicative to Poisson or exponential distributions.

3.2 WAITING TIME DISTRIBUTION IN A TWO CLASS PRIORITY SYSTEM

In Appendix A the waiting time distribution for the steady state single class queue was given by Eq. (A.16). The problem here is to obtain the waiting-time distribution in a two class priority system.

We should like to avoid attempting to solve what could be a very difficult queueing problem. The intent here is to use some of the already existing theory to get an approximation of what to expect when the simulation is performed. Miller [MI59] made some very significant contributions in his study of "Priority Queues." He solved the waiting-time distribution for a two class priority in a head-of-the-line priority queue. His analysis dealt with the $M/M/1$ and $M/G/1$ single server system.

Kesten and Runnenburg [KES57] have derived a generalized characterization for the Laplace transform of the steady state waiting-time distribution of a K class priority system. Their efforts have been duly referenced by most writers investigating priority queues. Although Kesten and Runnenburg developed what appears to be a very informative piece of theoretical analysis, they apparently made an error when taking the inverse for the two priority class case.

In 1965 a group of researchers at Stanford Research Institute [NEE65] noted, and corrected the error on the inversion formula of Kesten and Runnenburg. The waiting-time distribution for the priority case, as given by Kesten and Runnenburg becomes ($t \geq 0$):

$$H_1(t) = 1 - \lambda\mu \exp \left\{ - \frac{(1 - \lambda_1\mu)t}{\mu} \right\}, \rho < 1 \quad (3.1)$$

where the service distributions for both priority classes are equal and given by

$$F_1(x) = F_2(x) = 1 - \exp(-x/\mu) . \quad (3.2)$$

The corrected waiting-time distribution for the non-priority class for the same service distribution is given by [NEE65],

$$H_2(>t) = \frac{\rho^2 - \rho_1}{\rho - \rho_1} e^{-\alpha t} + \frac{1 - \rho}{2\pi} \int_{\alpha_1}^{\alpha_2} f(r) dr, \text{ for } \rho^2 \geq \rho_1 \quad (3.3a)$$

$$= \frac{1 - \rho}{2\pi} \int_{\alpha_1}^{\alpha_2} f(r) dr, \text{ for } \rho^2 < \rho_1 , \quad (3.3b)$$

where

$$\begin{aligned} \rho &= \lambda/\mu , \\ \rho_1 &= \lambda_1/\mu_1 , \\ \alpha &= [(\rho - \rho_1)(1 - \rho)]/\rho\mu , \\ \alpha_1 &= (1 - \sqrt{\rho_1})^2\mu , \\ \alpha_2 &= (1 + \sqrt{\rho_1})^2\mu , \\ f(r) &= e^{-rt} \sqrt{(\alpha_2 - r)(r - \alpha_1)}/r(r - \alpha), \end{aligned}$$

and

$$H_1(t) = H_2(t), \text{ for the single-class case.}$$

Since both distributions are for exponential head-of-the-line priority queuing system, the analytic model will be for a $M/M/1$ system. The inversion formula for the $M/G/1$ system is not available in the literature and its solution would be too esoteric for this discussion and would not be any more informative than the $M/M/1$ analysis. R. H. Davis [DA65] has developed an explicit formula for the waiting-time distribution of any number of parallel, negative-exponential servers subject to Poisson demands from any number of priority classes.

3.3 PRIORITY ASSIGNMENTS

The priority assignment will be based on how long a message has been in the system, such that when $Pr(\text{wait in } Q \geq T)$ the message is given a priority. There will be only two classes of messages; those of priority ($p = 1$) and those without ($p = 2$). The probability that a message reaches some age T and is given a priority will be established from the "waiting time" distribution.*

The priority process to be considered in this study will be of the $M/G/1$ type, where the service distribution is a constant. The constant service distribution is a result of all messages being of equal length and having identical service requirements. The constant server requires analysis by the Imbedded Markov chain. Also, the analysis for a priority process has some additional difficulties, since the state description has to incorporate information about each priority class, and thus becomes multidimensional.

The waiting-time distributions for a $M/M/1$ system will be used to get an analytic approximation to the $M/G/1$ simulation.

The selection of priority assignments in this investigation is based upon the waiting-time distribution as opposed to service-time distribution. Numerous authors have addressed the problem of priority selection based on minimum service time (e.g., [Phipps 56], [Jackson 60 and 62], and [Morse 58]). Cole [CO71] based his priority selection upon message length, yielding the same results as a selection based on service time.

Conway, et al [CON67] asserts that the optimal priority assignment for different service time requirements would be:

$$\frac{E(P_1)}{\mu_1} \leq \frac{E(P_2)}{\mu_2} \leq \dots \leq \frac{E(P_n)}{\mu_n}, \quad (3.4)$$

where

P_k = random variable for service-time of a k^{th} class message,
 u_k = some linear cost or other weighting factor associated
 with the waiting-time for messages of k^{th} class.

Cox and Smith [Cox61] assert there is some distinction to be made regarding a message's queueing-time and its waiting-time. The former being the time from his entering the system to the instant at which his service begins, and the latter is the time from entering the system to the instant at which his service is completed. Waiting-time equals queueing time plus service time. In many systems queueing-time and service-time are statistically independent, allowing one to obtain properties of waiting-time from those of queueing-time, and vice-versa.

* [Cole 71] investigated the case where priority assignment was based on message length distribution.

Message's queueing-time or waiting-time are of direct interest when there is an economic loss (message expires) if it is kept queueing. The effect of too long a delay either queueing or waiting results in giving a priority assignment to a message once it has been in the system more than km time units. The effect of this priority assignment will hopefully expedite delivery of messages that have already spent ample time in the system.

The introduction of a priority system of any form has no effect on the proportion of time for which the server is busy. This is solely determined by the total service-time of the messages arriving over a long period.

The priority assignment will be designated as $p(k)$ and defined as

$$p(k) = Pr(D_T \geq k MDT_{MAX}), 0 \leq k \leq 1, \quad (3.5)$$

where

$$\begin{aligned} D_T &= \text{total average message delay,} \\ k &= \text{priority threshold setting.} \end{aligned}$$

3.4 MINIMIZING THE AVERAGE MESSAGE DELAY

The average message delay, as has been suggested earlier, is one of the essential measurements deemed of vital importance in a store-and-forward communication system. The work of Cobham [COB54] formulates the equation for finding the expected waiting time for messages of each priority level of a single-channel (unsaturated) waiting line system. His general equation as would apply to an exogenous priority discipline has the form

$$W_p = \frac{1}{2} \lambda \int_0^{\infty} t^2 dB(t) / (1 - \sigma_{p-1})(1 - \sigma_p), \quad (3.6)$$

where

$$\begin{aligned} \rho_i &= \lambda_i / \mu_i \text{ and } \lambda = \sum_1^N \lambda_i, \\ \sigma_p &= \sum_1^k \rho_i < 1, \sigma_0 \equiv 0, \\ B(t) &= (1/\lambda) \sum_1^N \lambda_i B_i(t). \end{aligned}$$

$B_i(t)$ is the cumulative service time distribution function for the i^{th} priority. Thus for a two class priority system ($p = 1, 2$), the average message delay for the priority ($p = 1$) message becomes

$$W_1 = W_0 / (1 - \rho_1), \quad (3.7)$$

where

$$W_0 = \frac{1}{2} \lambda \int_0^{\infty} t^2 dB(t) \equiv \text{average time to complete existing service.}$$

For the non-priority ($p = 2$) messages, the delay is

$$W_2 = W_0 / (1 - \rho_1)(1 - \rho_1 - \rho_2). \quad (3.8)$$

When priority assignments are established within the system, then the priority message rate becomes:

$$\lambda_1 = p(k) \lambda, \quad (3.9)$$

where $p(k)$ is given by Eq. (3.5), and the non-priority message rate is

$$\lambda_2 = \lambda (1 - p(k)). \quad (3.10)$$

The unconditional delay at an arbitrary node becomes:

$$\begin{aligned} W &= p(k) W_1 + (1 - p(k)) W_2 \\ &= \frac{W_0}{1 - \rho} \left[\frac{1 - p(k) \rho}{1 - \rho_1} \right], \rho < 1 \end{aligned} \quad (3.11)$$

where

$$1 - \rho = 1 - \rho_1 - \rho_2.$$

In this investigation, all messages were assumed to be independent and are of equal length, hence have equal servicing. When $\mu_1 = \mu_2$, Eq. (3.11) is reduced to

$$W = W(\text{FCFS}). \quad (3.12)$$

It appears at first glance that Eq. (3.11) cannot be minimized by some optimal selection of k in establishing the $p(k)$ term. Subsequent developments will bear out that the average message delay can be reduced over the entire network, but cannot be directly assessed from the single node formulation given by (3.11).

The average message delay over the network is a function of the routing procedures, network utilization factor ρ , and network topology. The allowance of an adaptive routing procedure vs fixed routing, makes the indirect effects of priority assignment mathematically intractable; however, there is a technique to analytically assess the effect of priority assignment designated as the throughput factor.

3.5 THROUGHPUT FACTOR $\phi(t, k)$

The throughput factor $\phi(t, k)$ will be defined as the ratio of the delivered messages to the total number of messages handled by the network. Messages finding the queue full (balking) will not be included. It is desired to obtain the optimum value of k that maximizes $\phi(t, k)$ for all $\rho < 1$.

A closed form expression containing the parameters of interest can be obtained from a simplified network. Fig. (3.1) represents the 3-node tandem network under consideration. Non-priority messages with lengths drawn from an exponential distribution with mean $\mu = 1$, will arrive at nodes M and S at rates λ_1' and λ_2' respectively. It is assumed that the inter-arrival times of messages at either node are independent.

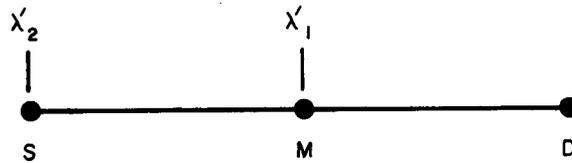


Fig. 3.1 – Three Node Tandem Net

We will assume that $1/2$ of λ_2' messages have M as a destination, and the remaining half have D as a destination. We desire to see what messages coming into M have D as a destination.

Neglecting the actual servicing, the probability that when a message leaves node S it is a priority message is given by

$$p(k) = P_r(k MDT_{MAX} \leq t \leq MDT_{MAX}), 0 \leq k \leq 1 \quad (3.13)$$

and using the distribution given by Eq. (3.1), we get

$$p(k) = \lambda_2' \exp\{-(1-\lambda_2') km\} - \lambda_2' \exp\{-(1-\lambda_2') m\}. \quad (3.14)$$

We define another variable called the "geriatrics" factor $g(m)$, as the probability of a message with source S and destination D , expires before it reaches M .

Then

$$g(m) = \Pr(t > MDT_{\text{MAX}}), \quad (3.15)$$

and since all messages into node S have no priority, we can use $H_1(t)$ as the distribution for a single class system, i.e., let $\rho_2 = 0$. Hence Eq. (3.15) becomes:

$$g(m) = 1 - H_1(m) = \lambda_2' e^{-(1-\lambda_2')m}. \quad (3.16)$$

Eq. (3.16) shows that $g(m)$ is a function of the maximum delay time m and message rate λ_2' at node S . Further $g(m)$ is not a function of the threshold parameter k , but effects the initial rate of undelivered messages as a function of m and λ_2' . We note that

$$p(k) + g(m) \leq 1, \quad (3.17)$$

this completes the analysis at node S .

At the second node M , the following variables are defined:

$$\begin{aligned} \lambda_1' &\equiv \text{arrival rate of new external messages at node } M \\ p(k) \lambda_2' / 2 &\equiv \text{arrival rate of internal priority messages at node } M \\ (1 - p(k) - g(m)) \lambda_2' / 2 &\equiv \text{arrival rate of non-priority messages at node } M. \end{aligned} \quad (3.18)$$

The non-priority message rate is designated by λ_2 and given by

$$\lambda_2 = (1 - p(k) - g(m)) \lambda_2' / 2 + \lambda_1' \quad (3.19)$$

and the priority message rate is designated by λ_1 and expressed as

$$\lambda_1 = p(k) \lambda_2' / 2. \quad (3.20)$$

The total arrival rate λ of messages into M becomes

$$\begin{aligned}\lambda &= (1 - p(k) - g(m))\lambda_2'/2 + \lambda_1' + p(k)\lambda_2'/2 \\ &= \lambda_2'/2 - g(m)\lambda_2'/2 + \lambda_1'.\end{aligned}\quad (3.21)$$

We further note that when the selection of m and the corresponding value of λ_2' is such that

$$g(m)\lambda_2'/2 \approx 0,$$

then

$$\lambda \approx \lambda_2'/2 + \lambda_1'.$$

The network flow pattern is as shown in Fig. (3.2). To calculate the probability that a message entering at node M is undelivered at node D , three cases must be considered.

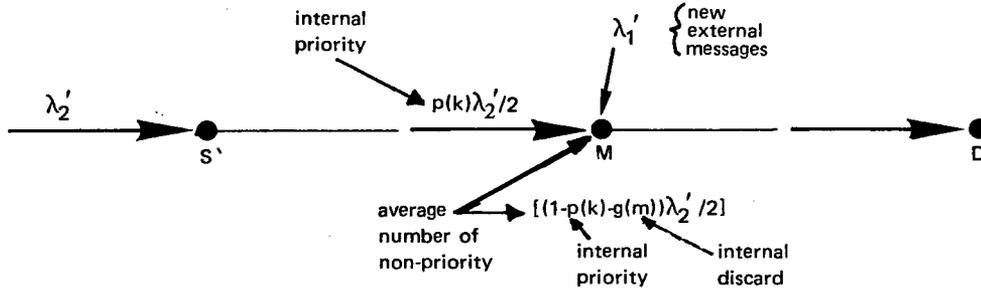


Fig. 3.2 – Flow Pattern in a 3 Node Net

Case 1

Messages that arrived from an external source or from node S that had *no wait* at node S .

We define the following random variables that are assumed to be independent

$$\begin{aligned}T_1 &= \text{time spent at node } S \\ T_2 &= \text{time spent at node } M \\ T &= T_1 + T_2 = \text{total time spent in the system.}\end{aligned}\quad (3.23)$$

From probability theory [PARZEN 62] we can express the following conditional probability of a random event A , given a random variable X by the Stieltjes integral as

$$P(A) = \int_{-\infty}^{\infty} P_r(A | X = x) dF_x(x), \quad (3.24)$$

then

$$P(A) = Pr(T > m) = \int_{-\infty}^{\infty} p_1(t) dF_{T_1}(t),$$

where

(3.25)

$$p_1(t) = Pr(T > m | T_1 = 0),$$

and

$$F_{T_1}(t) = \begin{cases} 0, & t < 0 \\ 1, & t \geq 0 \end{cases}, \text{ for any real } t.$$

Thus, the probability that a message expires before reaching D given that it had no wait at node S becomes:

$$\begin{aligned} P(A) &= Pr(T_2 > m) \\ &= 1 - H_2(m) \end{aligned} \quad (3.26)$$

since

$$T = T_2 \text{ for } T_1 = 0,$$

$H_2(t)$ is the waiting-time distribution of the non-priority messages in an exponential FCFS priority queueing system and is given by Eqs. (3.3a) and (3.3b). Hence Eq. (3.26) becomes:

$$P(A) = \begin{cases} \frac{\rho^2 - \rho^1}{\rho - \rho_1} e^{-\alpha m} + I, & \text{for } \rho^2 \geq \rho_1 \\ I, & \text{for } \rho^2 < \rho_1 \end{cases} \quad (3.27)$$

where

$$I = \frac{1-\rho}{2\pi} \int_{\alpha_1}^{\alpha_2} e^{-rt} \sqrt{(\alpha_2-r)(r-\alpha_1)} / r(r-\alpha) dr, \quad (3.28)$$

$$\alpha = (\rho - \rho_1)(1 - \rho) / \rho,$$

$$\alpha_1 = (1 - \sqrt{\rho_1})^2,$$

$$\alpha_2 = (1 + \sqrt{\rho_1})^2.$$

Case 2

Priority messages coming into node M .

Using the conditional probability given in Eq. (3.24) we define $P(B)$ as

$$P(B) = \Pr(T > m) = \int_{-\infty}^{\infty} p_2(t) dF_{T_2}(t) \quad (3.29)$$

where

$$\begin{aligned} p_2(t) &= \Pr(T > m \mid T_1 = t) \\ &= \Pr(T_2 > m - T_1 \mid T_1 = t) \\ &= \Pr(T_2 > m - t \mid T_1 = t), \end{aligned} \quad (3.30)$$

and due to the independence of T_1 and T_2 , Eq. (3.30) becomes:

$$P_2(t) = \lambda_1 e^{-(1-\lambda_1)(m-t)}. \quad (3.31)$$

The random variable associated with the distribution function $F_{T_2}(t)$ in Eq. (3.29) is that of a priority message and exists only in the interval from km to m . The probability density function (pdf) of a priority message is given by

$$f_{T_2}(t) = \begin{cases} 0, & t \geq m \\ (1 - \lambda_2') e^{-(1 - \lambda_2') t} \\ \hline [e^{-(1 - \lambda_2') km} - e^{-(1 - \lambda_2') m}] \\ 0, & t < km. \end{cases}, \text{ otherwise} \quad (3.32)$$

Hence,

$$\begin{aligned}
 P(B) &= \int_{km}^{\infty} \frac{m \lambda_1 (1 - \lambda_2') e^{-(1 - \lambda_1)(m-t)} e^{-(1 - \lambda_2')t}}{[e^{-(1 - \lambda_2')km} - e^{-(1 - \lambda_2')m}]} dt \\
 &= \frac{\lambda_1 (1 - \lambda_2') e^{-(1 - \lambda_1)m} [e^{-(\lambda_1 - \lambda_2')km} - e^{-(\lambda_1 - \lambda_2')m}]}{(\lambda_1 - \lambda_2') [e^{-(1 - \lambda_2')km} - e^{-(1 - \lambda_2')m}]}, \quad 0 \leq k < 1
 \end{aligned} \tag{3.33}$$

We note that Eq. (3.33) holds for all values of $0 \leq k < 1$ and $P(B) = 0$ for $k = 1$, as can be noted by the limits on the integral.

Case 3

A non-priority message that had to wait at node S.

We define $P(C)$ as:

$$P(C) = \Pr(T > m) = \int_{-\infty}^{\infty} p_3(t) dF_{T_3}(t), \tag{3.34}$$

where

$$\begin{aligned}
 p_3(t) &= \Pr(T > m \mid T_1 = t), \\
 &= \Pr(T_2 > m - t),
 \end{aligned} \tag{3.35}$$

and

$$f_{T_3} = \begin{cases} 0, & t > km \\ (1 - \lambda_2') e^{-(1 - \lambda_2')t} / [\lambda_2 (1 - e^{-(1 - \lambda_2')km})], & \text{otherwise} \\ 0, & t \leq 0. \end{cases}$$

The random variable associated with the *pdf* given in Eq. (3.34) exists only in the interval specified by Eq. (3.35).

Now

$$p_3(t) = \Pr(T_2 > m - t),$$

thus

$$p_3(t) = \begin{cases} \frac{\rho^2 - \rho_1}{\rho - \rho_1} e^{-\alpha(m-t)} + I, & \text{for } \rho^2 \geq \rho_1 \\ I, & \text{for } \rho^2 < \rho_1, \end{cases} \tag{3.36}$$

where I is defined by Eq. (3.28).

The solution for $P(C)$ as given in (A.27) and (A.28) allows us to write it in the following form:

$$P(C) = \begin{cases} I_1 + I_2, & \text{for } \rho^2 \geq \rho_1 \\ I_2, & \text{for } \rho^2 < \rho_1, \end{cases} \quad 0 < k \leq 1 \quad (3.37)$$

where

$$I_1 = \frac{(1 - \lambda_2')(\rho^2 - \rho_1) e^{-\alpha m}}{(\rho - \rho_1)(1 - \lambda_2' - \alpha)(1 - e^{-(1 - \lambda_2' - \alpha)km})} [1 - e^{-(1 - \lambda_2' - \alpha)km}], \quad (3.38)$$

and

$$I_2 = \frac{(1 - \lambda_2')(1 - \rho)}{2\pi(1 - \rho - (1 - \lambda_2')km)} \int_{\alpha_1}^{\alpha_2} \frac{e^{-mr} \sqrt{(\alpha_2 - r)(r - \alpha_1)}}{(1 - \lambda_2' - r)[r(r - \alpha)]} [1 - e^{-(1 - \lambda_2' - r)km}] dr.$$

The throughput factor $\phi(t, k)$ is a normalized function and related to the percent of undelivered messages $\nu(t, k)$ by the following:

$$\phi(t, k) = 1 - \nu(t, k). \quad (3.39)$$

$\nu(t, k)$ is the percentage messages handled by the system, but not delivered. From Fig. (3.2) we define the probability $r(t, k)$ as the probability that a message with source \underline{S} and destination D is undelivered. Let $S(t, k)$ define the probability that a message with source \underline{M} and destination D is undelivered.

Further, let $u(k) = (\lambda_2' - g(m) - \rho(k))$. By Eqs. (3.13), (3.16), and from the definition of λ_2' , it follows that $u(k)$ is the probability that a non-priority message has experienced some delay. Then $r(t, k)$ can be expressed by the following mutually exclusive independent events:

$$r(t, k) = g(m) + (1 - \lambda_2')P(A) + p(k)P(B) + u(k)P(C), \quad 0 \leq k \leq 1, \quad (3.40)$$

where

$$(1 - \lambda_2') = (1 - \rho) = \text{the probability that a message entering the system at node } S \text{ finds the queue empty at } S.$$

$$S(t, k) = P(A), \quad 0 \leq k \leq 1 \quad (3.41)$$

The percent of all undelivered messages $\nu(t, k)$ with destination D is

$$\nu(t, k) = [(\lambda_2'/2)r(t, k) + \lambda_1'S(t, k)]/(\lambda_2'/2 + \lambda_1'), \quad (3.42)$$

hence, the throughput factor $\phi(t, k)$ becomes

$$\phi(t, k) = 1 - [(\lambda_2'/2)r(t, k) + \lambda_1'S(t, k)]/(\lambda_2'/2 + \lambda_1'). \quad (3.43)$$

It should be noted that Eqs. (3.40) and (3.41) are continuous functions of k . However, it was shown for Case 2 and Case 3 that $P(B) = 0$ for $k = 1$ and $P(C) = 0$ for $k = 0$.

We would like to find the optimal value of the threshold parameter setting k that maximizes $\phi(t, k)$ for all $\rho < 1$. The optimization will be investigated in the following section.

3.5.1 Numerical Solution for (k_{op})

It is apparent that even if $\nu(t, k)$ is a convex function of k , it defines k as an implicit function of t , and k cannot be readily obtained (if at all) as an explicit function of t . Thus it becomes necessary to employ a numerical solution to find k_{op} .

Eq. (3.42) was solved using Simpson's rule [Krylon 62] to evaluate the integrals given by (3.27), (3.33), and (3.37). Computer runs were made on a fixed value of λ_2' and λ_1' and solving (3.42) for values of $k = 0.1, 0.2, \dots, 0.9$, and some appropriate value of $m = (MDT_{MAX})$. The selection of m was chosen to make $g(m)$ reasonably small, i.e., $g(m) < 5.0$ percent. All runs were made using 128 points in Simpson's rule and a value of $m = 20$.

Fig. (3.3) shows the fraction of undelivered messages $\nu(t, k)$ versus priority setting k . The utilization factor ρ was calculated from

$$\rho = \lambda_2'/2 - g(m)\lambda_2'/2 + \lambda_1', \quad (3.44)$$

where

$$g(m) = \lambda_2' e^{-(1 - \lambda_2')m}.$$

When m is fixed, $g(m)$ is directly effected by the selection of λ_2' , hence the curves given in Fig. (3.3) depicts the influence of $g(m)$ on the selection of k_{op} that minimizes $\nu(t, k)$. The range on ρ was from .37 to .49, but the range on $g(m)$ was from 35 percent down to .02 percent. The results indicate that when the value of the "geriatrics" factor $g(m)$ is very high, there is no well defined k value to minimize $\nu(t, k)$, as can be seen by the flatness of the bottom curve.

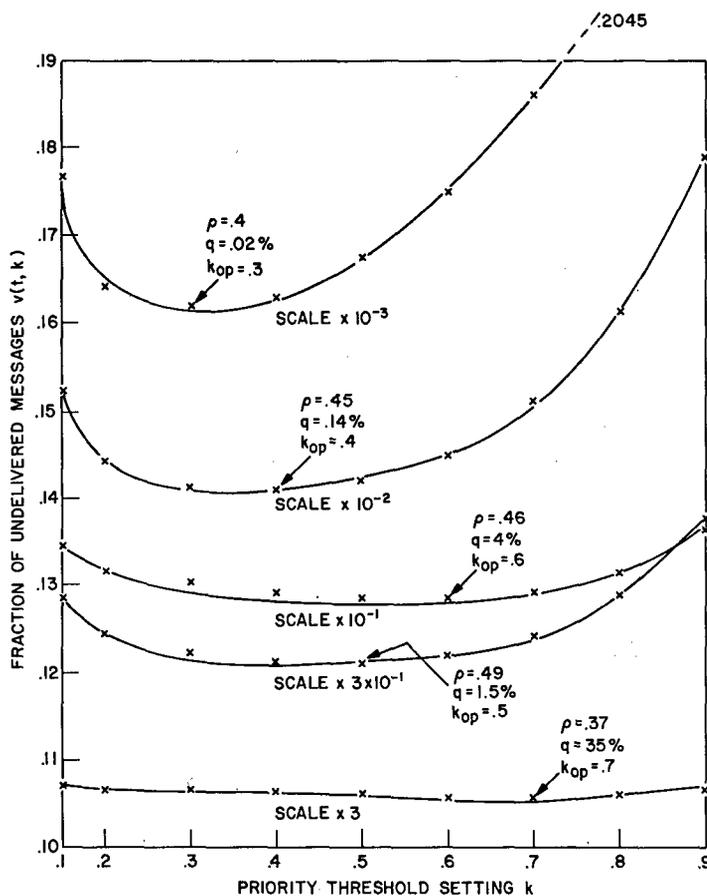


Fig. 3.3 – Fraction of Undelivered Messages Vs. Priority Setting

Table (3.1) is a tabulation of the curves in Fig. (3.3) and gives the associated values of λ_1' and λ_2' selected for each run. It should be noted how $g(m)$ varies with the selection of λ_2' and the subsequent deepening of the catenary in the curves as $g(m)$ decreases. Thus, when $g(m)$ is very small the selection of k_{op} is well defined and represents very fine tuning on the network parameters. The reason for this fine sensitivity is due to the very small value of $v(t, k) < .021$ percent, hence the selection of k_{op} represents what appears to be a vernier setting on $v(t, k)$. The average value of k_{op} in this set of runs was 0.5. The curves have been scaled against the ordinate values as indicated. Curves are assumed numbered from bottom to top as 1, 2, . . . , n.

TABLE 3.1
FRACTION OF UNDELIVERED MESSAGES $\nu(t, k)$

k	Curve #1 $\lambda_1' = .05$ $\lambda_2' = .95$	Curve #2 $\lambda_1' = .1$ $\lambda_2' = .8$	Curve #3 $\lambda_1' = .05$ $\lambda_2' = .85$	Curve #4 $\lambda_1' = .1$ $\lambda_2' = .7$	Curve #5 $\lambda_1' = .1$ $\lambda_2' = .6$
0.1	.3216	.1248E-01	.4037E-01	.1527E-02	.1768E-03
0.2	.3205	.1244E-01	.3957E-01	.1444E-02	.1643E-03
0.3	.3196	.1221E-01	.3907E-01	.1415E-02	<u>.1619E-03</u>
0.4	.3189	.1211E-01	.3875E-01	<u>.1410E-02</u>	.1633E-03
0.5	.3183	<u>.1210E-01</u>	.3859E-01	.1424E-02	.1676E-03
0.6	.3178	.1219E-01	<u>.3858E-01</u>	.1456E-02	.1751E-03
0.7	<u>.3177</u>	.1243E-01	.3880E-01	.1515E-02	.1868E-03
0.8	.3179	.1289E-01	.3944E-01	.1617E-02	.2045E-03
0.9	.3195	.1380E-01	.4097E-01	.1789E-02	.2310E-03

Fig. (3.4) are the curves associated with Table (3.2). This selection of runs were made over a wider range of network intensity ρ , but for the same $m = 20$ as before. The curves display the same trend as the previous results, indicating a single low point over the range of k . Flat curves are still indicative of no real sensitivity to the selection of k_{op} and deep catenaries indicate precise location of k_{op} for minimum $\nu(t, k)$. The average value of k_{op} for this run was 0.375.

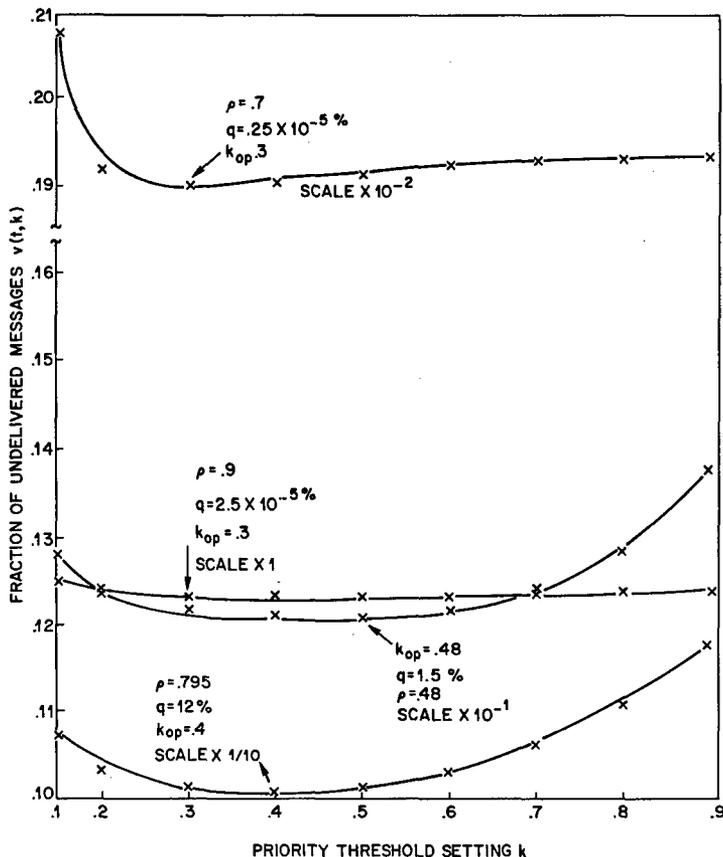


Fig. 3.4 – Fraction of Undelivered Messages Vs. Priority Setting

TABLE 3.2
FRACTION OF UNDELIVERED MESSAGES $v(t, k)$

k	Curve #1 $\lambda_1' = .4$ $\lambda_2' = .9$	Curve #2 $\lambda_1' = .1$ $\lambda_2' = .8$	Curve #3 $\lambda_1' = .7$ $\lambda_2' = .4$	Curve #4 $\lambda_1' = .5$ $\lambda_2' = .4$
0.1	.09692	.01284	.1253	.2076E-02
0.2	.09332	.01244	.1241	.1921E-02
0.3	.09134	.01221	<u>.1238</u>	<u>.1901E-02</u>
0.4	<u>.09071</u>	.01211	.1239	.1908E-02
0.5	.09129	<u>.01210</u>	.1239	.1917E-02
0.6	.09302	.01220	.1239	.1924E-02
0.7	.09599	.01243	.1240	.1928E-02
0.8	.1003	.01289	.1241	.1931E-02
0.9	.1063	.01380	.1241	.1932E-02

Table (3.3) are the values for Fig. (3.5) and indicate that family of curves are generated for the same value of ρ , and in this case for $\rho = 0.6$. It can be observed from Eq. (3.44) that there are numerous selections of λ_1' and λ_2' possible for a given ρ , the only restriction being that

$$\lambda_1' + \lambda_2'/2 \leq 1. \quad (3.45)$$

This selection of λ_1' and λ_2' keeps $\rho < 1$ on each link of the 3-node net given in Figs. (3.1) and (3.2). Since $g(m) > 0$ in all cases, this allows $\lambda_1 + \lambda_2'/2 = 1$, without violating the requirement of keeping $\rho < 1$. The average value of k_{op} for this set was 0.4.

TABLE 3.3
FRACTION OF UNDELIVERED MESSAGES $\nu(t, k)$

k	Curve #1	Curve #2	Curve #3
	$\lambda_1' = .5$ $\lambda_2' = .2$	$\lambda_1' = .15$ $\lambda_2' = .9$	$\lambda_1' = .2$ $\lambda_2' = .8$
0.1	.2123E-03	.9688E-01	.1183E-01
0.2	.2073E-03	.9504E-01	.1111E-01
0.3	<u>.2072E-03</u>	.9418E-01	.1081E-01
0.4	.2075E-03	.9364E-01	<u>.1074E-01</u>
0.5	.2077E-03	<u>.9338E-01</u>	.1084E-01
0.6	.2079E-03	.9345E-01	.1110E-01
0.7	.2079E-03	.9401E-01	.1157E-01
0.8	.2079E-03	.9544E-01	.1234E-01
0.9	.2079E-03	.9859E-01	.1354E-01

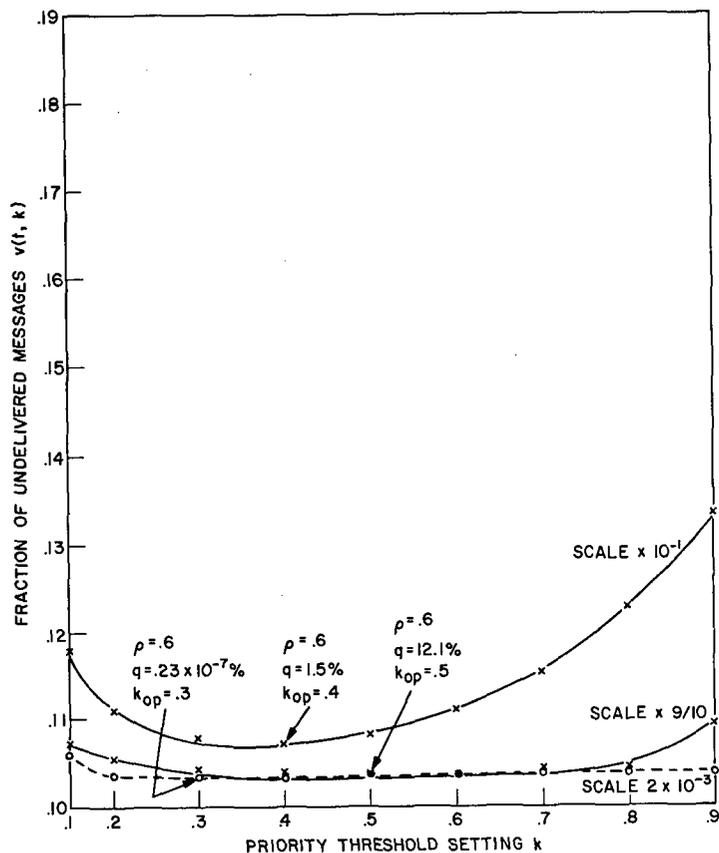


Fig. 3.5 – Fraction of Undelivered Messages Vs. Priority Setting

Recalling that $P(A)$ defined the probability of undelivered messages at D that arrived from an external source or from node S had no wait at S (Case 1). $P(B)$ defined the probability of undelivered messages that were in a priority class coming into node M (Case 2). $P(C)$ was for non-priority messages that had to wait at node S . In all cases tested, $P(A)$ and $P(B)$ decreased with increasing k (not tabulated), while $P(C)$ increased with k . From the definitions given it is clear that $P(A) < P(C)$ and $P(B) < P(C)$ for all $0 \leq k \leq 1$.

IV. SIMULATION ON AN 8-NODE HIGHLY CONNECTED NETWORK

The first series of simulation runs are for the 8-node network of Fig. (4.1). Appendix B contains the flow diagrams, Fortran listing, explanation of variables, and listing of the simulation runs used on the selected network.

RAND program originally had no queueing or preassigned links (PAL) and generated messages such that the network had constant loading. However, messages also had a dependency on whether a given node was busy, hence message volume (MV) would *not* always generate the same total number of messages.

A buffer was included in the RAND program to facilitate a common basis for comparing the various routing schemes. RAND had to use "Hot-Potato" since there were no buffers (queues). Buffers could hold up to 30 messages at each node.

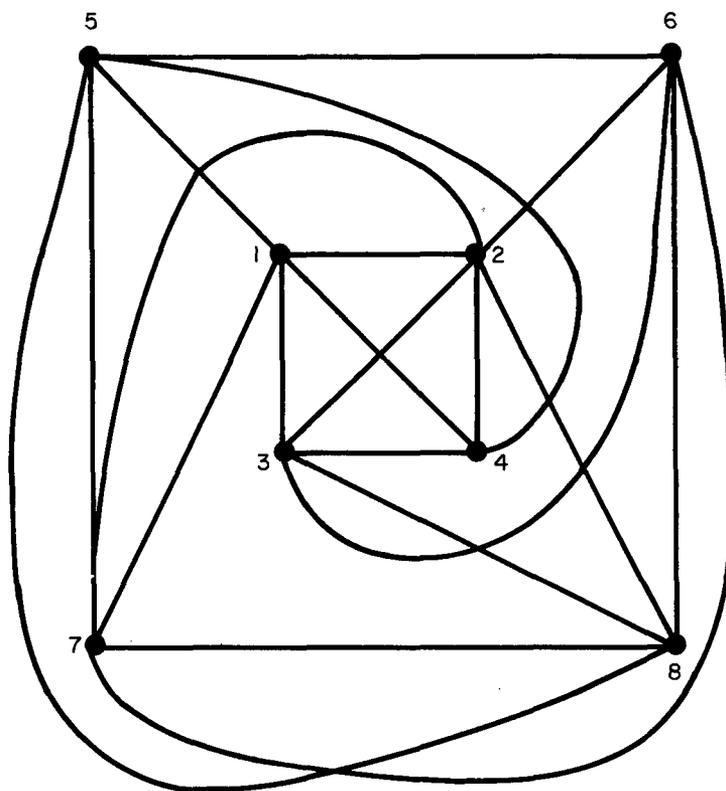


Fig. 4.1 – Highly Connected 8-Node Network

4.1 COMPUTATION OF MESSAGE VOLUME (MV)

Message volume (MV) is expressed as a percentage of links carrying messages per unit time. The number of messages generated in each time interval is equal to $[MV(\text{No. of links in network})/100]$. Count the number of links from a given net topology and double this number to account for duplexing.

Thus if number of links in topology = 12, then 24 becomes the total number of links in that net.

If $MV = 10$, then $\left\lceil \frac{24(10)}{100} \right\rceil = \lceil 2.4 \rceil = 2$ messages per unit time generated. If running time of simulation has $T = 500$ time units, then total number of messages generated during that run is 1,000. For any given MV and T , the total number of messages generated in the system will always be the same. However, the total number of messages *handled* by the system can be less than or equal to total number of messages generated.

For the 8-node net, there are 20 links and total number of duplexed links = 40. For $MV = 100$, then $\left\lceil \frac{40(100)}{100} \right\rceil = 40$ messages per unit time. If simulation time = 500 time units, there would be $40(500) = 20,000$ messages generated.

Messages are generated randomly with a random source S and random destination D such that $D \neq S$. Messages are generated on each computer cycle and the message volume is directly proportional to the value of MV selected when initiating the run.

Messages are allowed some maximum circulation time MDT_{MAX} and if they are not delivered before that time is reached, they are dropped from the system and tallied as undelivered messages. Dropping messages from the system after some maximum cycling time would hopefully relieve network congestion and allow the remaining messages to get delivered. The longer messages remain in the system without being delivered, the greater the average message delay becomes.

If the queue at S is full when a new message is generated, then a busy is detected and the message is dropped, and counter NZ incremented. It should be noted that messages finding the queue full are different from messages undelivered. When a significant number of messages find the queue full (say 10% of total messages generated), then this becomes an indication of the network approaching saturation.

The total number of messages generated for any given network loading was the same for all routing algorithms on any specified net configuration. Thus one can add the number of messages delivered, as tabulated in the message throughput Table (4.3), to the number undelivered (value given in parenthesis) and obtain the total messages handled by the system. The difference in the totals is attributed to the number of messages finding the queue full (tabulated in Chap. V) and the messages in transit. Hence, for light net loading, messages delivered plus undelivered will be about the same for all program routines.¹

The program has an initialization (warm up) of $T = MDT_{MAX}$ and is also the maximum time a message can be in the network. Thus for a message time $> MDT_{MAX} = 50$, those messages are considered undeliverable.

4.2 OBJECTIVES OF THE SIMULATION

The objectives of this simulation study was to find an adaptive routing algorithm that could maximize message throughput while minimizing average message delay. Since the routing would be stochastic in nature and essentially utilize information about the state of the network via node delay tables, a systematic analytical method of solution was not possible. Ideally we would want a routing algorithm that could adapt to network topology changes, either due to congestion at some given nodes, or adjust to physical changes in node or link structure.

1. Also [average time] x [messages delivered] + $[MDT_{MAX}]$ x [number of undelivered messages] \approx a constant for a given value of MV .

The stochastic techniques rely on information exchanges between nodes, and this additional traffic reduces the network capacity to send data. Since each node must make inquiries of its neighboring nodes to calculate new delay table updates, this requires some type of processor to be available at each node. Another objective of this study was to minimize hardware and software requirements and still maintain some high degree of operating efficiency.

The simulation affords a method to test the effectiveness of various routing schemes and possibly how to optimize some aspects of these routing strategies to increase throughput. The initial computer runs were on a small scale computer (DDP-24) and a highly connected 8-node network on which many combinations of routing strategies were employed (see Appendix B4.1 and B4.2). It was envisioned that after these initial test runs were made, selected algorithms giving the best performance would be tried on a 19-node ARPA network and run on a CDC-3800 computer:

4.3 NO LINK OR NODE FAILURES

Table 4.1 gives a listing of some specially selected routing algorithms as applied to the 8-node network. This sampling was taken to show the strong points and weaknesses of various routing techniques.

With no destruction of the network, it was assumed that the ideal routing tables would give optimal results up to moderate network loading. Thus the no updating (NU) routine was used as a basis to compare all others.

Routing tables were initialized by computing the minimum path matrix on an empty network. These initializing algorithms would be optimum if either the network did not become too congested or there were no net failures.

TABLE 4.1	
PROGRAM ALGORITHMS FOR DDP-24 SIMULATION	
1.	Dynamic Programming (DY)
2.	No Updating (NU)
3.	Backward Learning (BL)
4.	Bi-Adaptive (BA)
5.	Superposition (SP)
6.	ARPA Shortest Que + Bias + Periodic Updating (ARPA)
7.	Last M Nodes Visited (LMNV)
8.	Best Three Outgoing Links (BTL)
9.	Aging Parameter ($X\%$ of MDT_{MAX})

} $\equiv MOD_1$

Fig. (4.2) depicts the data plots of Table (4.2). These were selected from all runs made in Table B4.1 as being the most informative. Routines such as negative reinforcement (NR), bi-adaptive (BA) and no updating (NU), gave identical results for all values of message volume (MV), even up to 100% network loading. The quantity $[2/5 MV]$ is a scaling factor to avoid plotting distortions. For a 20-link network and an MV given as a percentage, the fraction $2/5$ appears in the brackets and $[X] \equiv$ greatest whole integer less than X .

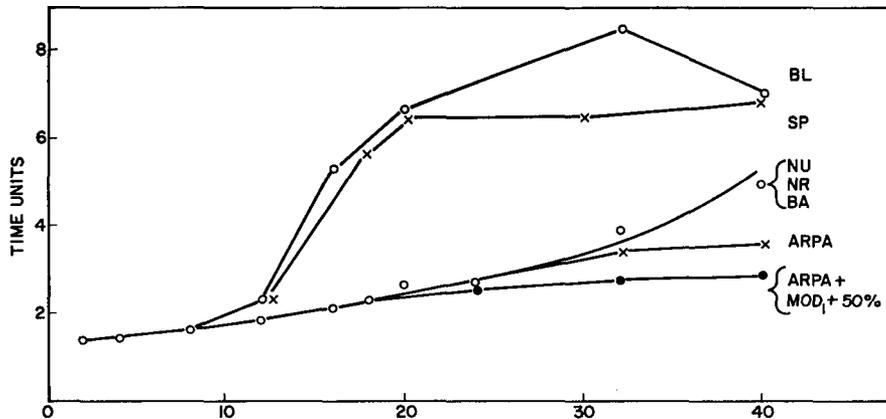


Fig. 4.2 - Message Volume = $[2/5 MV]$ Per Unit Time
Average Message Delay for 8-Node Net

Backward learning appeared to be not too effective. This was largely due to the “ping-pong” effect between successive nodes. Backward learning utilizes no direct information on the progress of messages passing through the system, and its indirect method of computing delay values is often misleading or inadequate or both. The drawback on the BL technique will become more apparent as it tries (unsuccessfully) to adapt to network topology changes.

Superposition (SP) of NR and BL, where BL can increase or decrease table entries, shows only a modest improvement over BL as far as average message delay is concerned. Table (4.3) shows the message throughput for the various routing algorithms and the quantities in parentheses indicate the number of undelivered messages.

It can be noted from Table (4.3) that the SP routine eliminates the number of messages not being delivered, and as a consequence the total number of messages delivered increases. The average message delays for the ARPA and $ARPA + MOD_1 + 50\%$ aging indicate substantial gains in both minimizing average delay and maximizing throughput. Fig. (4.3) in conjunction with Tables (4.2) and (4.3) illustrate the effectiveness of these routines.

The curves in Fig. (4.3) were plotted against the no updating (NU) routine to indicate how an adaptive routine enhances message delays and throughputs when the message volume increases substantially. The ARPA routine was comprised of a shortest queue + bias + periodic updating technique; although this was a sizable increase in programming and bookkeeping, the gains were significant enough to warrant such an investment. The

TABLE 4.2
AVERAGE TIME FOR MESSAGES IN AN 8-NODE HIGHLY CONNECTED NETWORK

[2/5 MV]	NR, BA, NU	BL	SP	NU + MOD ₁ + 50% AGING	ARPA	ARPA + MOD ₁ + 50% AGING
2	1.32	1.32	1.32	1.32	1.32	1.32
4	1.36	1.36	1.36	1.36	1.36	1.36
8	1.50	1.54	1.54	1.50	1.48	1.47
12	1.81	2.27	2.35	1.81		
16	2.15	5.25	5.26	2.18	1.85	1.85
18	2.32	5.15	5.74	2.39		
20	2.62	6.63	6.55	2.54		
24					2.58	2.60
32	3.96	8.56	6.54	3.58	2.89	2.87
40	5.05	7.16	7.14	3.77	3.03	3.00

TABLE 4.3
MESSAGE THROUGHPUT IN AN 8-NODE HIGHLY CONNECTED NETWORK

[2/5 MV]	NR, BA, NU	BL	SP	ARPA	NU + MOD ₁ + 50% AGING	ARPA + MOD ₁ + 50% AGING
2	1000 (0)	1000 (0)	1000 (0)	1000 (0)	1000 (0)	1000 (0)
4	1998 (0)	1998 (0)	1998 (0)	1998 (0)	1998 (0)	1998 (0)
8	3997 (0)	3997 (0)	3997 (0)	3999 (0)	3997 (0)	3999 (0)
12	6001 (0)	5962 (0)	5924 (0)		6001 (0)	
16	7932 (0)	5984 (0)	5957 (0)	7996 (0)	7908 (0)	8002 (0)
18	8779 (0)	6423 (0)	5995 (0)			
20	9307 (0)	5379 (0)	5494 (0)		9385 (0)	
24				11,241 (0)		11,190 (0)
32	9683 (2)	4392 (65)	6038 (0)	12,531 (0)	10,548 (0)	12,599 (0)
40	7848 (37)	5596 (2)	5593 (0)	12,815 (0)	10,483 (0)	12,993 (0)

modification to the ARPA routine (LNV, BTL and 50% aging) indicated some further improvement as regards to message delay and throughput.

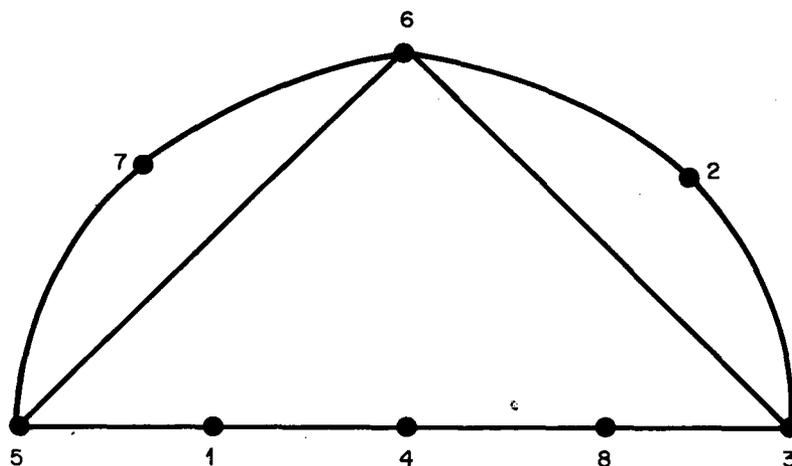


Fig. 4.3 – 8-Node Network With 50% Disabled Links

After all the routines listed in Appendix B4.1 were run, it became apparent as to what routines were the most effective. None of the modifications when applied to the BTL doctrine would significantly increase its effectiveness. The various combinations of last M nodes visited and aging parameters were applied to the ideal routing tables (NU) and the values that “peaked” our performance measures were selected as being optimal. The combination of last node visited (LNV) and best 3 output links (BTL) gave the minimum average message delays and maximum message throughput. This combination of LNV and BTL, henceforth called MOD_1 , was applied to all the routing strategies listed and resulted in some significant improvements.

The aging parameter, whereby at some percentage of MDT_{MAX} a message is given priority on the queues, peaked both message delay and throughput at a value of 50%.³ Since there was no direct method of calculating the optimal values of LNV and aging parameter, it had to be determined via simulation. Once these optimal values were determined they were applied to the better routing strategies such as NU and ARPA and these became the adjusted routines given in Table (4.2) as $MOD_1 + 50\%$ and $ARPA + MOD_1 + 50\%$.

The preliminary conclusion would be that the combination of $LNV + BTL \equiv MOD_1$ when applied with the 50% aging parameter should yield the minimum message delay and maximum throughput for the network under consideration. It was further felt that if this technique were applied to any of the listed routing algorithms of Appendix B, it should enhance their performance. This enhancement has been effectively verified from the tabulated results of Tables (4.2) and (4.3). The next step in confirming the optimal adaptive routing technique would be to see how it fares when applied to a damaged network.

3. 50% aging was close to the predicted value for optimal k given in Chap. III.

4.4 SIMULATION WITH 50% LINK DESTRUCTION

The logical follow through on the 8-node highly connected network would be to see how the adaptive routines adjust to high network destruction. The destruction applied only to the links (50% outage) and was used to simulate a highly disastrous situation where half of the network links (or lines) were inoperative.

Figure (4.3) shows the connectivity with 50% link destruction on the 8-node network. This level of destructive was just short of disconnecting the net. Table (4.4) lists the same set of selected runs as were given for the no link or node failure case. Although there was an extensive set of runs made, as indicated by Table (B4.2) in Appendix B, the algorithms selected represented a good cross section in terms of our performance measurements. The dynamic (DY) algorithm would be the presumed theoretical optimum in this case, in that it would immediately sense link or node failures and recompute the routing tables. The intensity of network traffic was increased to the point where the number of messages finding the queue full (not tabulated) exceeded 10% of the total messages generated and this was found to be for a $MV = 40$; hence, $[1/5(40)] = 8$ became the maximum intensity that was used for this set of runs. The identical set of links were disabled for all iterations of any given routing algorithm, thus all algorithms were compared for identical network topology and message intensities.

It is apparent from Table (4.4) and the curves of Fig. (4.4) that backward learning gets almost hopelessly bogged down and this is reflected in its average message delays and throughputs. Table (4.4) also illustrates how the no updating (NU) algorithm fails to adjust to the net loading, since after initializing its tables they are no longer updated. The ARPA routine appears to be fairly adaptive to the network topology and loading effects. The modifications to the ARPA routine ($LNV + BTL + 50\%$ Aging) improved the performance of the routing strategies. We should take cognizance of ($ARPA + MOD_1 + 50\%$) routine in the area of $MV = 6$ and 7, which represented moderate loading. The $MOD_1 + 50\%$ addition slightly surpassed the DY routine in minimum average message delays; hence, DY is not a theoretical lower bound (see Table (4.5)).

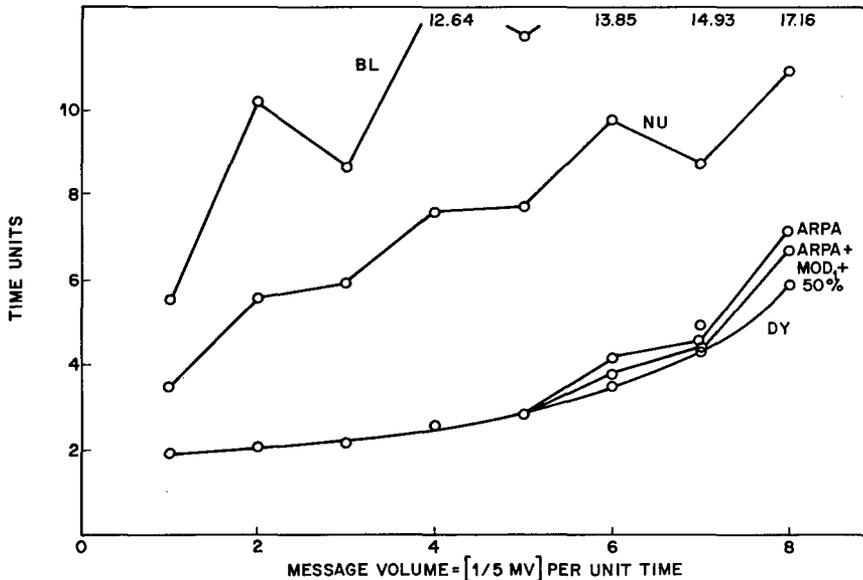


Fig. 4.4 - Message Volume = $[1/5 MV]$ Per Unit Time Average Message Delay for Disabled 8-Node Net

TABLE 4.4

AVERAGE TIME FOR MESSAGES ON AN 8-NODE NETWORK WITH 50% LINK DESTRUCTION

[1/5 MV]	DY	NU	BL	NR	ARPA	ARPA + MOD ₁ + 50%
1	1.85	3.41	5.07	2.32	1.92	1.87
2	2.06	5.67	10.18	2.41	2.09	2.08
3	2.13	5.90	8.59	2.52	2.16	2.15
4	2.55	7.59	12.64	3.08	2.56	2.54
5	2.80	7.67	11.75	3.51	2.72	2.69
6	3.51	9.83	13.85	4.75	4.11	3.70
7	4.30	8.71	14.93	4.96	4.49	4.36
8	5.85	10.96	17.16	8.66	7.10	6.62

TABLE 4.5
MESSAGE THROUGHPUT FOR AN 8-NODE NETWORK WITH 50% LINK DESTRUCTION

[1/5 MV]	DY	BL	ARPA	ARPA + MOD ₁ + 50%
1	500 (0)	471 (227)	500 (0)	500 (0)
2	1000 (0)	790 (388)	1001 (0)	1002 (0)
3	1500 (0)	939 (500)	1505 (0)	1501 (0)
4	2001 (0)	1180 (414)	2002 (0)	1999 (0)
5	2497 (0)	1282 (484)	2497 (0)	2496 (0)
6	2997 (0)	1016 (588)	3003 (0)	3008 (0)
7	3444 (0)	1207 (501)	3460 (0)	3480 (0)
8	3558 (0)	1053 (570)	3499 (4)	3512 (0)

The DY and ARPA routines did not have the congestion control afforded by the 50% aging parameter that gave priority to messages already in the system and was set at $0.5 MDT_{MAX}$. This congestion control, although not directly a routing algorithm, indirectly enhances some of the delay table updates by removing some of the node congestion.

4.5 ROUTING TABLES

A look at some of the selected routing tables may give some insight into the effectiveness of the various routing algorithms. Initially all routing tables will have the same values. When message intensity starts increasing, the various adaptive routines will update the tables and depending on the effectiveness of these routines the new routing tables will or will not yield optimal path assignments. This example also illustrates why the BL routine has such a high percentage of undelivered messages. Tables for higher message intensities continue to show the futility of BL routing. The tables for the ARPA routine along with $MOD_1 + 50\%$ show almost identical routings for the same network loading. We used the extreme cases to make the illustration more informative.

It should be reiterated here that BL was duly noted by its creators [BO66] as being something less than optimum. They introduced NR and SP as a means to counter the undesirable effects of BL. The use of BL in this study was to dramatize the fact that a routing algorithm although adaptive may not be very effective. Hence, our concern is not only for adaptive routines but effective ones, when measured against the performance criterion.

We will define our routing tables with the following notations:

$$A^i(j, k) \equiv \text{routing tables at node } i, \text{ with destination } j, \\ \text{via node } k$$

and

$$A^1(j, k) \equiv A^1$$

$$A^2(j, k) \equiv A^2$$

.

.

.

$$A^n(j, k) \equiv A^n$$

thus

$$A = (A^1, A^2, \dots, A^n) \equiv \text{routing matrix}$$

An example of how the BL routine gets bogged down with very light network loading can be quickly seen by comparing its routing tables to those of the DY routine. The tables are used as follows.

Assume a message is at some node i and has a destination node j , where $j \neq i$. Then go to the appropriate table for $A^i(j, k)$ and select the minimum (j, k) element value, this gives the next intermediate node k . If $k \neq j$, then continue until $k = j$ and this will be the final destination node. If there are more than one minimum (j, k) value, take any one and proceed until destination is reached. The final tally of intermediate nodes will give the delay time for the message in going from i to j .

An example will quickly demonstrate the above procedures. Take node 1 as a source node and node 4 as destination. Fig. (4.3) shows these two nodes to be neighboring nodes, and hence a message should take only *one* time unit to go from 1 to 4. From Table 4.6 the DY routine gives the following routing:

$$\text{for } A^i(j, k), \text{ where } i = 1 \text{ and } j = 4 \\ A^1(4, k) = 1,$$

and this corresponds to a $k = 4$, and $k = j = 4$, hence message is at destination in 1 time unit.

Table 4.7 gives the BL routing for the identical network loading, and its routing tables direct a message from node 1 to 4 as follows:

$$A^1(4, k) = 2 \Rightarrow k = 5 \neq j \text{ (continue until } k = j)$$

$$A^5(4, k) = 3 \Rightarrow k = 6 \text{ or } 7$$

$$A^6(4, k) = 2 \Rightarrow k = 2 \text{ or } 3$$

$$A^2(4, k) = 3 \Rightarrow k = 6$$

$$A^6(4, k) = 2 \Rightarrow k = 2 \text{ or } 3$$

$$A^3(4, k) = 3 \Rightarrow k = 6$$

Message is now looping between nodes 2, 3, and 6 and will not get delivered since there is no alternate route to take. If at node 5, the message had taken 7, then it would have traveled the following route.

$$A^1(4, k) = 2 \Rightarrow k = 5 \neq j$$

$$A^5(4, k) = 3 \Rightarrow k = 6 \text{ or } 7$$

$$A^7(4, k) = 6 \Rightarrow k = 5 \text{ or } 6$$

$$A^6(4, k) = 2 \Rightarrow k = 2 \text{ or } 3$$

and the next set of nodes are the same as the previous set, except this time there is an additional entry at $k = 7$, and the message will still not get delivered.

Thus we can see how some routines although adaptive can result in very poor performance as regards to average message delay and throughput.

TABLE 4.6

ROUTING TABLES FOR DY ROUTINE WITH 50% DISABLED LINKS AND $[1/5 \text{ MV}] = 5$

$A^1(j, k)$	$A^2(j, k)$	$A^3(j, k)$	$A^4(j, k)$																																																																																																																														
<table border="1" style="border-collapse: collapse; width: 100px; height: 100px;"> <tr><td style="border: none;">$j \backslash k$</td><td style="border: none;">4</td><td style="border: none;">5</td></tr> <tr><td style="border: none;">1</td><td>0</td><td>0</td></tr> <tr><td style="border: none;">2</td><td>4</td><td>3</td></tr> <tr><td style="border: none;">2</td><td>3</td><td>3</td></tr> <tr><td style="border: none;">4</td><td>1</td><td>3</td></tr> <tr><td style="border: none;">5</td><td>3</td><td>1</td></tr> <tr><td style="border: none;">6</td><td>4</td><td>2</td></tr> <tr><td style="border: none;">7</td><td>4</td><td>2</td></tr> <tr><td style="border: none;">8</td><td>2</td><td>4</td></tr> </table>	$j \backslash k$	4	5	1	0	0	2	4	3	2	3	3	4	1	3	5	3	1	6	4	2	7	4	2	8	2	4	<table border="1" style="border-collapse: collapse; width: 100px; height: 100px;"> <tr><td style="border: none;">$j \backslash k$</td><td style="border: none;">3</td><td style="border: none;">6</td></tr> <tr><td style="border: none;">1</td><td>4</td><td>3</td></tr> <tr><td style="border: none;">2</td><td>0</td><td>0</td></tr> <tr><td style="border: none;">3</td><td>1</td><td>2</td></tr> <tr><td style="border: none;">4</td><td>3</td><td>4</td></tr> <tr><td style="border: none;">5</td><td>3</td><td>2</td></tr> <tr><td style="border: none;">6</td><td>2</td><td>1</td></tr> <tr><td style="border: none;">7</td><td>3</td><td>2</td></tr> <tr><td style="border: none;">8</td><td>2</td><td>3</td></tr> </table>	$j \backslash k$	3	6	1	4	3	2	0	0	3	1	2	4	3	4	5	3	2	6	2	1	7	3	2	8	2	3	<table border="1" style="border-collapse: collapse; width: 100px; height: 100px;"> <tr><td style="border: none;">$j \backslash k$</td><td style="border: none;">2</td><td style="border: none;">6</td><td style="border: none;">8</td></tr> <tr><td style="border: none;">1</td><td>4</td><td>3</td><td>3</td></tr> <tr><td style="border: none;">2</td><td>1</td><td>2</td><td>3</td></tr> <tr><td style="border: none;">3</td><td>0</td><td>0</td><td>0</td></tr> <tr><td style="border: none;">4</td><td>4</td><td>4</td><td>2</td></tr> <tr><td style="border: none;">5</td><td>3</td><td>2</td><td>4</td></tr> <tr><td style="border: none;">6</td><td>2</td><td>1</td><td>3</td></tr> <tr><td style="border: none;">7</td><td>3</td><td>2</td><td>4</td></tr> <tr><td style="border: none;">8</td><td>3</td><td>3</td><td>1</td></tr> </table>	$j \backslash k$	2	6	8	1	4	3	3	2	1	2	3	3	0	0	0	4	4	4	2	5	3	2	4	6	2	1	3	7	3	2	4	8	3	3	1	<table border="1" style="border-collapse: collapse; width: 100px; height: 100px;"> <tr><td style="border: none;">$j \backslash k$</td><td style="border: none;">1</td><td style="border: none;">8</td></tr> <tr><td style="border: none;">1</td><td>1</td><td>3</td></tr> <tr><td style="border: none;">2</td><td>4</td><td>3</td></tr> <tr><td style="border: none;">3</td><td>4</td><td>2</td></tr> <tr><td style="border: none;">4</td><td>0</td><td>0</td></tr> <tr><td style="border: none;">5</td><td>2</td><td>4</td></tr> <tr><td style="border: none;">6</td><td>3</td><td>3</td></tr> <tr><td style="border: none;">7</td><td>3</td><td>4</td></tr> <tr><td style="border: none;">8</td><td>3</td><td>1</td></tr> </table>	$j \backslash k$	1	8	1	1	3	2	4	3	3	4	2	4	0	0	5	2	4	6	3	3	7	3	4	8	3	1									
$j \backslash k$	4	5																																																																																																																															
1	0	0																																																																																																																															
2	4	3																																																																																																																															
2	3	3																																																																																																																															
4	1	3																																																																																																																															
5	3	1																																																																																																																															
6	4	2																																																																																																																															
7	4	2																																																																																																																															
8	2	4																																																																																																																															
$j \backslash k$	3	6																																																																																																																															
1	4	3																																																																																																																															
2	0	0																																																																																																																															
3	1	2																																																																																																																															
4	3	4																																																																																																																															
5	3	2																																																																																																																															
6	2	1																																																																																																																															
7	3	2																																																																																																																															
8	2	3																																																																																																																															
$j \backslash k$	2	6	8																																																																																																																														
1	4	3	3																																																																																																																														
2	1	2	3																																																																																																																														
3	0	0	0																																																																																																																														
4	4	4	2																																																																																																																														
5	3	2	4																																																																																																																														
6	2	1	3																																																																																																																														
7	3	2	4																																																																																																																														
8	3	3	1																																																																																																																														
$j \backslash k$	1	8																																																																																																																															
1	1	3																																																																																																																															
2	4	3																																																																																																																															
3	4	2																																																																																																																															
4	0	0																																																																																																																															
5	2	4																																																																																																																															
6	3	3																																																																																																																															
7	3	4																																																																																																																															
8	3	1																																																																																																																															
$A^5(j, k)$	$A^6(j, k)$	$A^7(j, k)$	$A^8(j, k)$																																																																																																																														
<table border="1" style="border-collapse: collapse; width: 100px; height: 100px;"> <tr><td style="border: none;">$j \backslash k$</td><td style="border: none;">1</td><td style="border: none;">6</td></tr> <tr><td style="border: none;">1</td><td>1</td><td>3</td></tr> <tr><td style="border: none;">2</td><td>4</td><td>2</td></tr> <tr><td style="border: none;">3</td><td>4</td><td>2</td></tr> <tr><td style="border: none;">4</td><td>2</td><td>4</td></tr> <tr><td style="border: none;">5</td><td>0</td><td>0</td></tr> <tr><td style="border: none;">6</td><td>3</td><td>1</td></tr> <tr><td style="border: none;">7</td><td>3</td><td>2</td></tr> <tr><td style="border: none;">8</td><td>3</td><td>3</td></tr> </table>	$j \backslash k$	1	6	1	1	3	2	4	2	3	4	2	4	2	4	5	0	0	6	3	1	7	3	2	8	3	3	<table border="1" style="border-collapse: collapse; width: 100px; height: 100px;"> <tr><td style="border: none;">$j \backslash k$</td><td style="border: none;">2</td><td style="border: none;">3</td><td style="border: none;">5</td><td style="border: none;">7</td></tr> <tr><td style="border: none;">1</td><td>4</td><td>4</td><td>2</td><td>3</td></tr> <tr><td style="border: none;">2</td><td>1</td><td>2</td><td>3</td><td>3</td></tr> <tr><td style="border: none;">3</td><td>2</td><td>1</td><td>3</td><td>3</td></tr> <tr><td style="border: none;">4</td><td>4</td><td>3</td><td>3</td><td>4</td></tr> <tr><td style="border: none;">5</td><td>3</td><td>3</td><td>1</td><td>2</td></tr> <tr><td style="border: none;">6</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td style="border: none;">7</td><td>3</td><td>3</td><td>2</td><td>1</td></tr> <tr><td style="border: none;">8</td><td>3</td><td>2</td><td>4</td><td>4</td></tr> </table>	$j \backslash k$	2	3	5	7	1	4	4	2	3	2	1	2	3	3	3	2	1	3	3	4	4	3	3	4	5	3	3	1	2	6	0	0	0	0	7	3	3	2	1	8	3	2	4	4	<table border="1" style="border-collapse: collapse; width: 100px; height: 100px;"> <tr><td style="border: none;">$j \backslash k$</td><td style="border: none;">5</td><td style="border: none;">6</td></tr> <tr><td style="border: none;">1</td><td>2</td><td>3</td></tr> <tr><td style="border: none;">2</td><td>3</td><td>2</td></tr> <tr><td style="border: none;">3</td><td>3</td><td>2</td></tr> <tr><td style="border: none;">4</td><td>3</td><td>4</td></tr> <tr><td style="border: none;">5</td><td>1</td><td>2</td></tr> <tr><td style="border: none;">6</td><td>2</td><td>1</td></tr> <tr><td style="border: none;">7</td><td>0</td><td>0</td></tr> <tr><td style="border: none;">8</td><td>4</td><td>3</td></tr> </table>	$j \backslash k$	5	6	1	2	3	2	3	2	3	3	2	4	3	4	5	1	2	6	2	1	7	0	0	8	4	3	<table border="1" style="border-collapse: collapse; width: 100px; height: 100px;"> <tr><td style="border: none;">$j \backslash k$</td><td style="border: none;">3</td><td style="border: none;">4</td></tr> <tr><td style="border: none;">1</td><td>4</td><td>2</td></tr> <tr><td style="border: none;">2</td><td>2</td><td>4</td></tr> <tr><td style="border: none;">3</td><td>1</td><td>3</td></tr> <tr><td style="border: none;">4</td><td>3</td><td>1</td></tr> <tr><td style="border: none;">5</td><td>3</td><td>3</td></tr> <tr><td style="border: none;">6</td><td>2</td><td>4</td></tr> <tr><td style="border: none;">7</td><td>3</td><td>4</td></tr> <tr><td style="border: none;">8</td><td>0</td><td>0</td></tr> </table>	$j \backslash k$	3	4	1	4	2	2	2	4	3	1	3	4	3	1	5	3	3	6	2	4	7	3	4	8	0	0
$j \backslash k$	1	6																																																																																																																															
1	1	3																																																																																																																															
2	4	2																																																																																																																															
3	4	2																																																																																																																															
4	2	4																																																																																																																															
5	0	0																																																																																																																															
6	3	1																																																																																																																															
7	3	2																																																																																																																															
8	3	3																																																																																																																															
$j \backslash k$	2	3	5	7																																																																																																																													
1	4	4	2	3																																																																																																																													
2	1	2	3	3																																																																																																																													
3	2	1	3	3																																																																																																																													
4	4	3	3	4																																																																																																																													
5	3	3	1	2																																																																																																																													
6	0	0	0	0																																																																																																																													
7	3	3	2	1																																																																																																																													
8	3	2	4	4																																																																																																																													
$j \backslash k$	5	6																																																																																																																															
1	2	3																																																																																																																															
2	3	2																																																																																																																															
3	3	2																																																																																																																															
4	3	4																																																																																																																															
5	1	2																																																																																																																															
6	2	1																																																																																																																															
7	0	0																																																																																																																															
8	4	3																																																																																																																															
$j \backslash k$	3	4																																																																																																																															
1	4	2																																																																																																																															
2	2	4																																																																																																																															
3	1	3																																																																																																																															
4	3	1																																																																																																																															
5	3	3																																																																																																																															
6	2	4																																																																																																																															
7	3	4																																																																																																																															
8	0	0																																																																																																																															

TABLE 4.7
 ROUTING TABLES FOR BL ROUTINE WITH 50% DISABLED LINKS AND $[1/5 MV] = 5$

$A^1(j, k)$			$A^2(j, k)$			$A^3(j, k)$				$A^4(j, k)$			
<i>j</i>	<i>k</i>		<i>j</i>	<i>k</i>		<i>j</i>	<i>k</i>			<i>j</i>	<i>k</i>		
	4	5		3	6		2	6	8		1	8	
1	26	23	1	2	43	1	2	37	43	1	22	37	
2	2	23	2	8	24	1	13	14	42	2	2	6	
3	2	25	3	24	15	3	47	14	47	3	2	19	
4	17	2	4	48	3	4	44	3	48	4	37	44	
5	38	2	5	46	18	5	47	18	2	5	44	41	
6	22	15	6	47	43	6	46	45	47	6	27	2	
7	40	7	7	44	43	7	38	12	48	7	45	48	
8	29	36	8	48	8	8	42	9	21	8	27	42	

$A^5(j, k)$				$A^6(j, k)$				$A^7(j, k)$			$A^8(j, k)$				
<i>j</i>	<i>k</i>			<i>j</i>	<i>k</i>			<i>j</i>	<i>k</i>			<i>j</i>	<i>k</i>		
	1	6	7		2	3	5	7		5	6		3	4	
1	1	34	34	1	44	38	5	40	1	40	80	1	44	38	
2	25	15	40	2	6	16	32	46	2	27	9	2	46	7	
3	27	20	39	3	14	8	28	15	3	16	3	3	48	19	
4	4	3	3	4	2	2	5	3	4	6	6	4	47	46	
5	20	23	12	5	20	28	1	19	5	8	2	5	3	45	
6	47	20	46	6	44	41	7	32	6	22	16	6	47	3	
7	44	49	21	7	44	44	3	44	7	43	18	7	44	48	
8	37	14	2	8	9	4	2	20	8	15	10	8	22	46	

4.6 SUMMARY OF 8-NODE SIMULATION

The 8-node simulation on the DDP-24 was essentially employed to check out some of the selected routing algorithms. It should be further noted that not all available adaptive routines were tested, but instead some representative sampling of some known algorithms were used to demonstrate strong points and weaknesses.

The most desirable features sought for in an adaptive routing algorithm would be as follows.

1. Minimum hardware requirement at each node.
2. Minimum software and inter-node traffic.
3. Adaptability to changes in network topology and traffic intensity.
4. Maximum message throughput.
5. Minimum average message delay.
6. Minimum number of undelivered messages.

It is very conceivable with present day technology that with unrestricted use of items 1 and 2, a very effective adaptive routine could and may already be developed. Relaxing hardware and software requirements would certainly add to the initial overhead and general operating cost. Item 3 is really the crux of problem, for were it not for the changing of net topology (either link or node outages) and if traffic intensity were always moderate $\rho \ll 1$, then any effective minimum path algorithm would certainly suffice. The performance measures of items 4 and 5 are indicative to this study and may differ for different networks. While item 6 may be tolerated if very small in some networks, a network dedicated to emergencies or military use, etc., may not tolerate even 1 lost message.

Our initial runs on the small scale computer and 8-node network gives us some insight into the selection of algorithms to be tried out on the 19-node ARPA network using a CDC-3800. It appears that the technique of last node visited (LNV) and best three output links (BLT), when combined (MOD_1) yields some effective results. The additional feature of the aging parameter appears to "peak" the network performance when a value of 50% is employed.

There is a distinction to be made between throughput (total messages handled in the system) and throughput factor $\phi(t, k)$ (fraction of messages handled and delivered). Table (4.3) shows for a $[2/5 MV] = 32$, the throughput was quite different for various routing algorithms. The SP had a throughput of 6038 messages and a throughput factor $\phi(t, k)$ of 1.0. However, the ARPA and $ARPA + MOD_1 + 50\%$ aging had over twice the total throughput as the SP and also a $\phi(t, k) = 1.0$. Thus, one would like to maximize $\phi(t, k)$ for all $\rho < 1$, plus maintain the highest total throughput rate. Generally, when the average message delay is minimized and $\phi(t, k) = 1.0$, for all $\rho < 1$, then the total throughput will be at its peak.

The dynamic (DY) routine, although not a theoretical optimum, but still not possible to employ on a real network, will still be used as a basis for comparing other algorithms. The set of runs for the 19-node

ARPA net that were run on the CDC-3800 are tabulated in Appendix C. Note that runs were made on the ARPANET with a uniform input message distribution and a Poisson input distribution. Many iterations of the selected algorithms were made to see if performance still "peaked" with the same algorithms.

V. CDC-3800 SIMULATION ON A 19 NODE ARPA NETWORK

The same rules and constraints were employed in these simulation runs as were used in the previous chapter. The network was a 19 node ARPA network and the computer a CDC-3800. The algorithms were run with no destruction, then with 10% of the links disabled, and finally with 1 node disabled. Again only a few selected runs will be tallied and discussed, since inclusion of all runs would make for a prohibitive amount of listing. Appendix C lists the runs made on this net configuration.

5.1 19 NODE ARPA SIMULATION

Table (5.1) lists the algorithms chosen for the 19 node ARPA network of Fig. (5.1). This series of runs was made with no link or node failures, with first a uniform input message distribution, and then with a Poisson distribution.

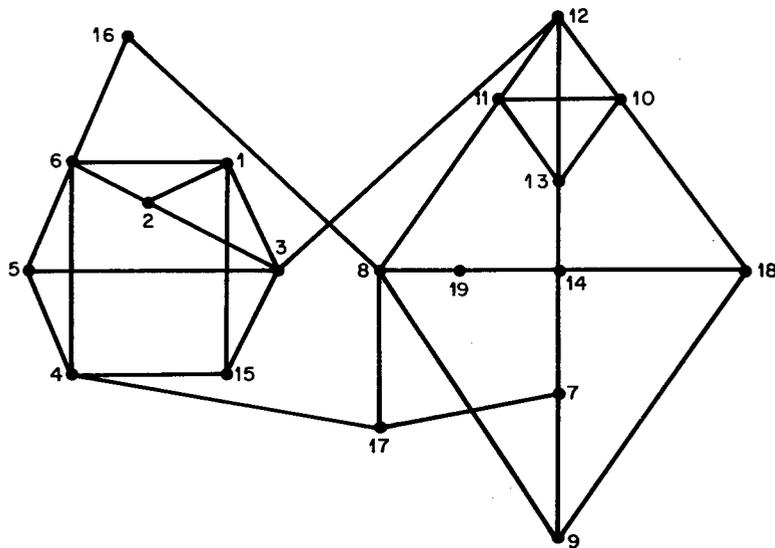


Fig. 5.1 - ARPA Network 19 Nodes

TABLE 5.1	
PROGRAM ALGORITHMS FOR CDC-3800 SIMULATION	
1.	Dynamic Programming or No Updating (<i>NU</i>)
2.	Backward Learning (<i>BL</i>)
3.	No Updating, 50 percent Aging (<i>NU + 50%</i>)
4.	ARPA Shortest <i>Queue + Bias + Periodic Updating (ARPA)</i>
5.	<i>MOD</i> ₁ to ARPA (<i>MOD</i> ₁ + <i>ARP</i>)

Fig. (5.2) shows the curves for the tabulated values of average message delay given in Table (5.2). The NU routine shows a slightly better performance than the *MOD*₁, but the throughput with the *MOD*₁, given in Table (5.3) was slightly better at high traffic intensity. The BL routine still has its difficulties as shown by its average message delays and throughput and also the number of undelivered messages (undelivered messages are in parenthesis).

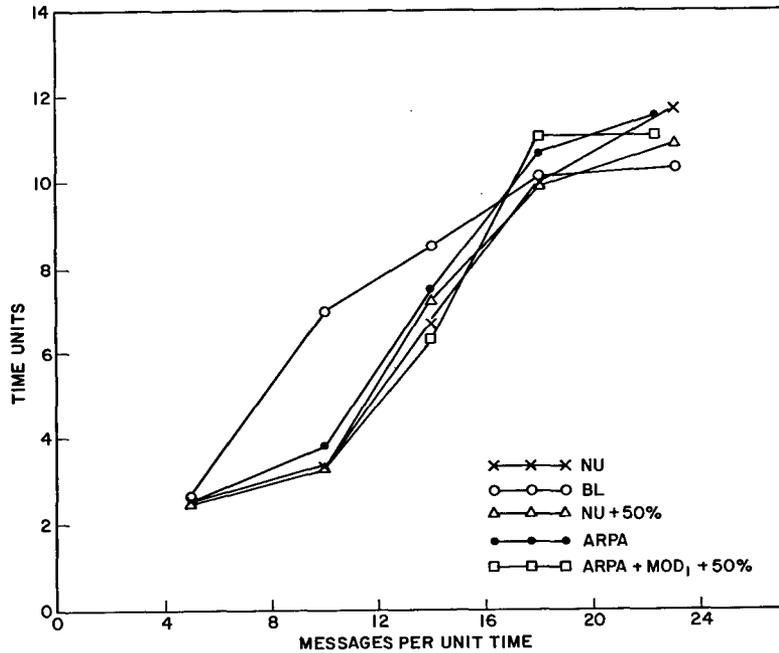


Fig. 5.2 – Average Message Delays for 19 Node Net With Uniform Input Distribution

TABLE 5.2
AVERAGE TIME FOR MESSAGES WITH A UNIFORM INPUT DISTRIBUTION

<i>MV</i>	<i>NU</i>	<i>BL</i>	<i>MOD</i> ₁	<i>NU + ARPA</i>	<i>ARPA</i>	<i>ARPA + MOD</i> ₁
3	2.50	2.52	2.50	2.52	2.56	2.55
6	2.96	5.19	2.96	3.77	3.44	3.31
9	4.57	9.67	4.57	7.79	6.30	6.35
12	8.44	9.65	8.41	9.93	10.25	10.89
15	11.73	11.22	11.07	10.94	11.33	12.22

TABLE 5.3
MESSAGE THROUGHPUT FOR MESSAGES WITH A UNIFORM INPUT DISTRIBUTION

<i>MV</i>	<i>NU</i>	<i>BL</i>	<i>NU + MOD</i> ₁ + 50%	<i>NU + ARPA</i>	<i>ARPA</i>	<i>ARPA + MOD</i> ₁ + 50%
3	750(0)	750(0)	750(0)	750(0)	750(0)	750(0)
6	1502(0)	1036(0)	1502(0)	1455(41)	1499(1)	1499(2)
9	2203(0)	860(679)	2203(0)	1660(274)	2023(94)	1987(102)
12	2367(227)	1010(671)	2365(278)	1815(476)	1608(428)	1657(420)
15	2255(521)	1124(619)	2328(495)	1826(669)	1692(575)	1610(597)

The ARPA data simulation plotted in Fig. (5.2) was slightly aided by the *MOD*₁ + 50% aging addition to it. The periodic updated ARPA (*ARPA + PUD*) is the algorithm of interest, since it is used on an existing network.

Table (5.3) shows that a small advantage at moderate network loading (*MV* = 12) is obtained when the *MOD*₁ + 50% is used with the ARPA routine. Table (5.4) shows the extent of network loading. At a *MV* = 12, over 1/3 of the total messages generated for any algorithm were not able to get into the system. At a *MV* = 15, almost 50% of the total messages generated found the queue full, this is in addition to the number of undelivered messages. Hence, it can be seen that with a *MV* = 12, the network has a high message intensity.

TABLE 5.4
NUMBER OF MESSAGES FINDING THE QUEUES FULL

<i>MV</i>	<i>NU</i>	<i>BL</i>	<i>NU + MOD₁ + 50%</i>	<i>NU-ARPA</i>	<i>ARPA</i>	<i>ARPA + MOD₁ + 50%</i>
3	0	0	0	0	0	0
6	0	96	0	19	1	0
9	42	610	42	257	139	168
12	594	1213	608	670	844	878
15	1465	1941	1401	669	1464	1522

Fig. (5.3) shows the plots associated with Tables (5.5) and (5.6) when the message input process was Poisson. In the Poisson case messages were identically distributed with an average message rate at each node i of λ_i and the total message rate into the system per unit time was $\sum_{i=1}^{19} \lambda_i$.

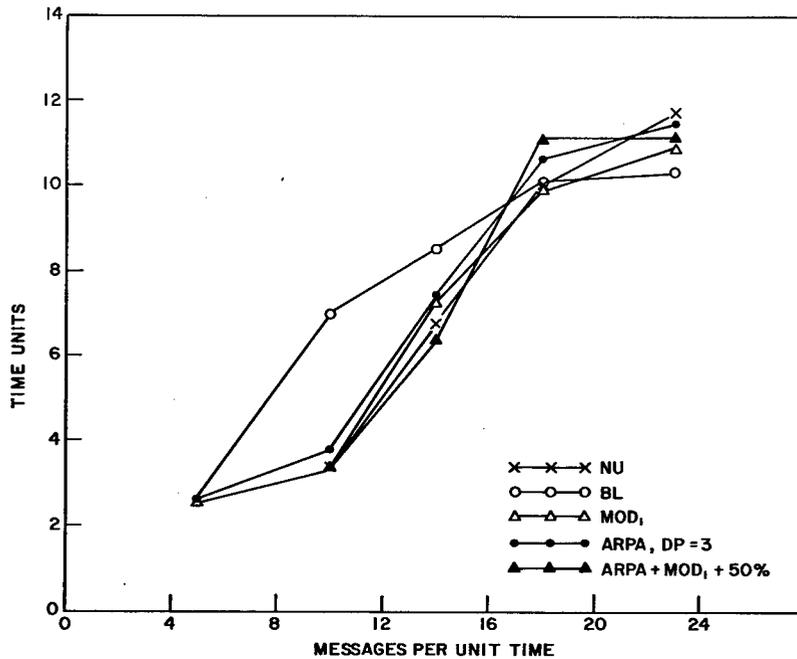


Fig. 5.3 - Average Message Delay for 19 Node Net With Poisson Input Distribution

TABLE 5.5
AVERAGE TIME FOR MESSAGES WITH A POISSON INPUT DISTRIBUTION

λ_i	<i>MV</i>	<i>NU</i>	<i>BL</i>	<i>BL + 50%</i>	<i>BL</i> <i>MOD₁ + 50%</i>
0.158	5	2.57	2.58	2.57	2.57
0.316	10	3.39	6.99	3.39	3.39
0.474	14	6.73	8.48	7.20	7.20
0.632	18	10.04	10.14	9.93	9.93
0.790	23	11.71	10.36	10.91	10.91

TABLE 5.6
AVERAGE TIME FOR MESSAGES WITH A POISSON INPUT DISTRIBUTION

λ_i	<i>MV</i>	<i>ARPA</i> <i>NU</i>	<i>ARPA</i> <i>DP = 3</i>	<i>ARPA, NU +</i> <i>MOD₁</i>	<i>ARPA, NU +</i> <i>MOD₁ + 50%</i>	<i>ARPA +</i> <i>MOD₁ + 50%</i>
0.158	5	2.57	2.61	2.61	2.62	2.57
0.316	10	4.90	3.82	4.48	4.00	3.29
0.474	14	8.76	7.50	7.69	7.65	6.28
0.632	18	9.79	10.69	10.98	10.61	11.04
0.790	23	10.96	11.50	11.93	11.84	11.19

When the input distribution was Poisson the average number of messages at an arbitrary node $i = \lambda_i \approx E(x_i)$. When all nodes were identically distributed $E(x) = 19\lambda_i$ for the 19 node ARPA net and since there were 68 duplexed links in the ARPA net, we have

$$\left[\frac{68 MV}{100} \right] \approx 19\lambda_i$$

Thus, we can get an equivalent value of *MV* to go with the selected λ_i .

The plots of Table (5.7) was an attempt to improve on the BL routine with 50% aging. Noteworthy, was the increase in throughput and resulting decrease in undelivered messages as tabulated in Table (5.7). The NU routine was still used as a basis for comparison.

TABLE 5.7
MESSAGE THROUGHPUT FOR MESSAGES WITH POISSON INPUT DISTRIBUTION

λ_i	<i>MV</i>	<i>NU</i>	<i>BL</i>	<i>BL+50%</i>	<i>NU ARPA</i>	<i>ARPA</i>
0.158	5	738(0)	738(0)	738(0)	738(0)	738(0)
0.316	10	1528(0)	965(320)	1528(0)	1413(65)	1518(7)
0.474	14	1974(77)	713(779)	1919(106)	1634(317)	1999(116)
0.632	18	1913(337)	657(998)	2132(184)	1876(462)	1745(451)
0.790	23	1767(484)	565(1152)	1844(470)	1854(626)	1651(630)

The updated ARPA routine was further adjusted by a $DP = 2$ factor. The DP factor is a constant bias that is applied to reduce looping. A more extensive treatment of the effectiveness of the DP bias is given in Fultz [FU72]. Fultz further suggested the optimum routine as far as his study was concerned was the shortest *queue + bias + periodic updating (SQ + BIAS + PUD)* and this algorithm was the one extensively used on the 19 node network as a basis for comparing the various $MOD_1 + X\%$ aging modifications. Fig. (5.3), along with Tables (5.6) and (5.8), indicate there is some slight improvement in average message delay and throughput respectively when $MOD_1 + 50\%$ aging was applied to the ARPA algorithms.

TABLE 5.8
MESSAGE THROUGHPUT FOR MESSAGES WITH A POISSON INPUT DISTRIBUTION

λ_i	<i>MV</i>	<i>NU ARPA</i> + MOD_1	<i>NU ARPA</i> + $MOD_1 + 50\%$	<i>ARPA +</i> $MOD_1 + 50\%$
0.158	5	738(0)	738(0)	738(0)
0.316	10	1493(25)	1527(3)	1528(0)
0.474	14	1917(155)	1940(137)	2199(1)
0.632	18	1833(374)	1729(395)	1639(462)
0.790	23	1771(564)	1693(582)	1796(545)

5.2 ARPA NETWORK WITH 3 LINKS DISABLED

Table (5.9) lists the average message delays (see Fig. 5.4) associated with 3 links (10%) disabled and these delays are plotted in Fig. (5.5). The dynamic (*DY*) routine was used as a standard in this case since

with link outages there would have to be some adjustments made on the routing tables. The ideal routing tables used with no updating (*NU*) would not be an unbiased comparison, for it would not adjust its tables for the disabled links. Tables (5.10) and (5.11) indicate that the total throughput was lowered considerably for all the routing algorithms.

TABLE 5.9

AVERAGE TIME FOR MESSAGES ON 19 NODE ARPA NETWORK WITH THREE OF LINKS DISABLED

λ_i	<i>NMSG</i>	<i>DYN</i>	<i>BA</i>	<i>DYN + MOD₁</i>	<i>ARPA</i>	<i>ARPA + MOD₁</i>	<i>ARPA + MOD₁ + 50%</i>
0.053	1.0	3.20	4.37	3.20	3.26	3.26	3.23
0.105	2.0	4.06	4.65	4.06	4.39	4.58	4.55
0.125	2.4	5.00	6.12	5.01	5.29	5.35	5.28
0.145	2.8	6.27	6.97	6.54	6.73	6.61	6.45
0.165	3.1	7.18	9.00	7.64	7.74	8.08	8.59
0.200	3.9	8.37	8.26	8.49	8.28	8.55	8.93

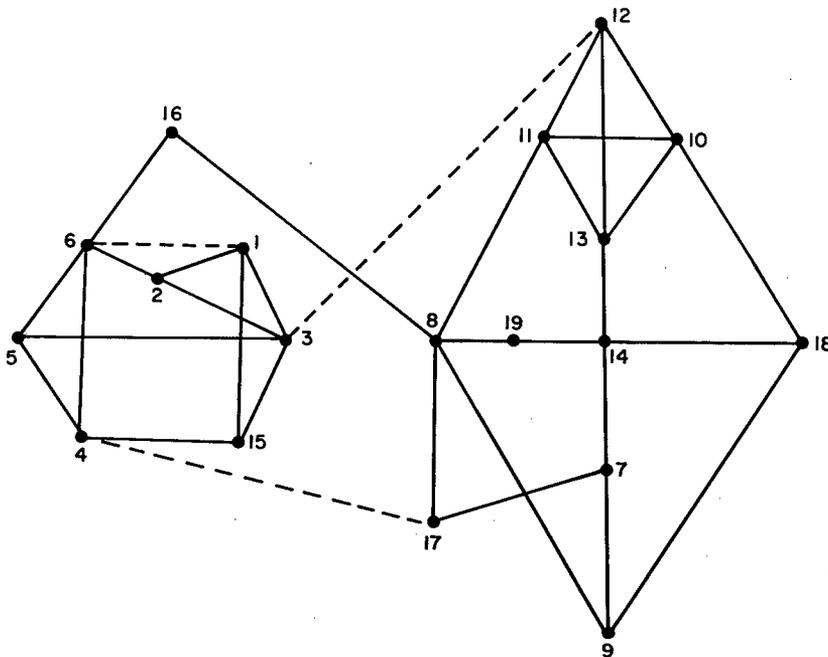


Fig. 5.4 – ARPA Network With Three Links Disabled

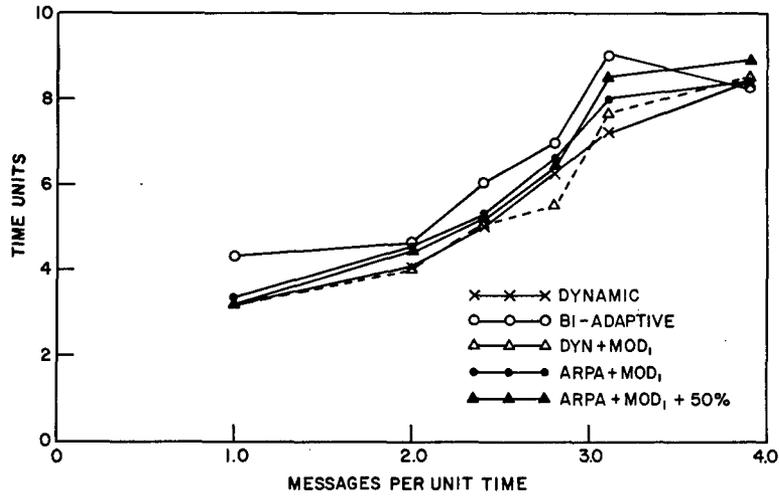


Fig. 5.5 – Average Message Delay on 19 Node Net With Three Links Disabled

TABLE 5.10

MESSAGE THROUGHPUT FOR 19 NODE ARPA NETWORK WITH THREE OF LINKS DISABLED

λ_i	<i>NMSG</i>	<i>ARPA</i>	<i>MOD₁ + ARPA</i>	<i>MOD₁ + 50% + ARPA</i>
0.053	1.0	266(0)	266(0)	266(0)
0.105	2.0	482(5)	487(1)	488(2)
0.125	2.4	592(4)	594(2)	592(1)
0.145	2.8	667(6)	665(8)	671(3)
0.165	3.1	619(128)	625(115)	625(114)
0.200	3.9	595(257)	564(291)	676(203)

TABLE 5.11

MESSAGE THROUGHPUT FOR 19 NOTE ARPA NETWORK WITH THREE OF LINKS DISABLED

λ_i	<i>NMSG</i>	<i>DYN</i>	<i>BA</i>	<i>DYN + MOD₁</i>
0.053	1.0	265(0)	261(10)	265(0)
0.105	2.0	479(0)	464(28)	479(0)
0.125	2.4	587(0)	561(35)	587(0)
0.145	2.8	670(4)	562(35)	667(2)
0.165	3.1	673(71)	673(90)	686(56)
0.200	3.9	626(225)	227(237)	633(228)

Fig. (5.4) shows which links were disabled and indicates that the network had its cross link capacity reduced by $2/3$. Since all messages have a random source and destination, then the probability of message being in the 8 node group with destination for the 11 node group using link (16, 8) becomes $(8/19) (11/18) = .257$. The same probability holds when the source group is in the 11 node set and sends messages via link (8, 16) to the 8 node group.

When link (16, 6) is used, then the probability of an arbitrary message coming into the net and uses link (16, 6) in either direction becomes $(7/19) (12/18) = .246$. Since link (16, 8) has the greatest probability of being used, it establishes the limiting value of message flow that can be tolerated. Hence, link capacity = $0.257 \times$ (number of messages) and since the link capacity is unity, the number of messages per unit time becomes $1/0.257 = 3.89$ messages/unit time as the upper limit. For an identically distributed 19 node network, $\lambda_i = 3.89/19 = .205 \Rightarrow \rho = 1$, or saturation level. The value of $\lambda_i = .200$ was the maximum value selected for this set of runs to avoid overloading the network and Table (5.9) reflects the selection of λ_i used on this network with three links disabled.

The $DY + MOD_1 + 50\%$ aging showed some improvement as compared to $ARPA$ in average message delay and throughput, but as previously stated the DY routine requires instant knowledge of node or link failures, hence could not be considered as a practical routine in a real network. The $ARPA + MOD_1 + 50\%$ aging showed some slight improvements in throughput and also a fewer number of undelivered messages.

5.3 ARPA NETWORK WITH ONE NODE DISABLED

Fig. (5.6) shows the node disabled for the runs tabulated in Tables (5.12) and (5.13) and plotted in Figs. (5.7) and (5.8). The $ARPA$ routine still faired quite well and the $ARPA + MOD_1 + 50\%$ aging helped increase throughput (Table 5.15) slightly at high message intensities, but the $ARPA + MOD_1$ performed better in this case. The disabling of one node had a noticeable effect on message throughput, since the disabled node would service half of the cross network traffic. Also, by disabling node 17, it took 3 links with it, resulting in a $1/3$ reduction of available links for cross net traffic. All algorithms were adjusted such that node 17 would not send or receive messages, but routings that previously routed thru node 17 would have to be adaptive. Applying 50% aging to BA, NR, and SP, aided them in their throughput, Table (5.14), but only slightly in message delay (Table (5.12) and Fig. (5.8)).

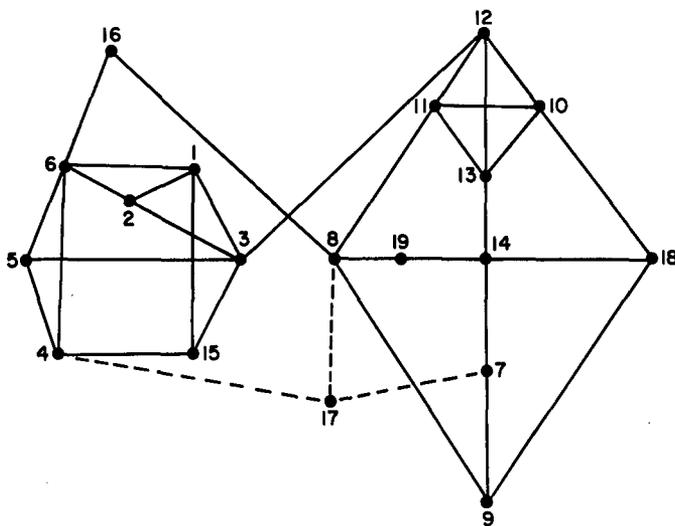


Fig. 5.6 - ARPA Network With One Node Disabled

Fig. 5.7 - Average Message Delay on 19 Node Net With One Node Disabled

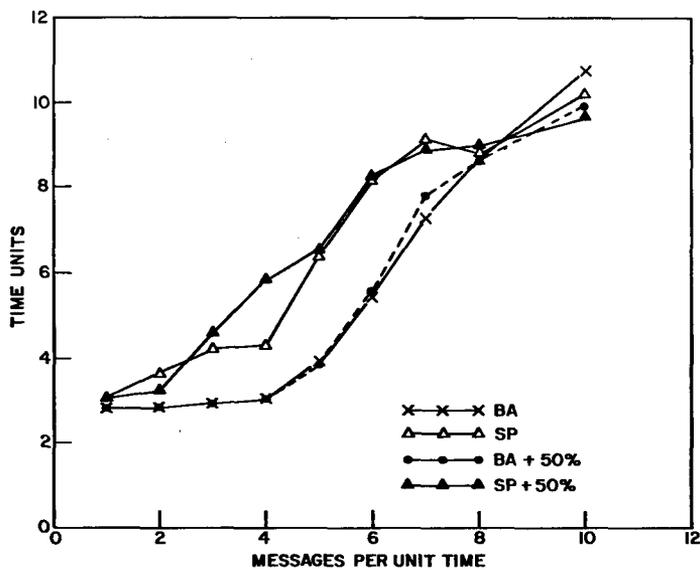
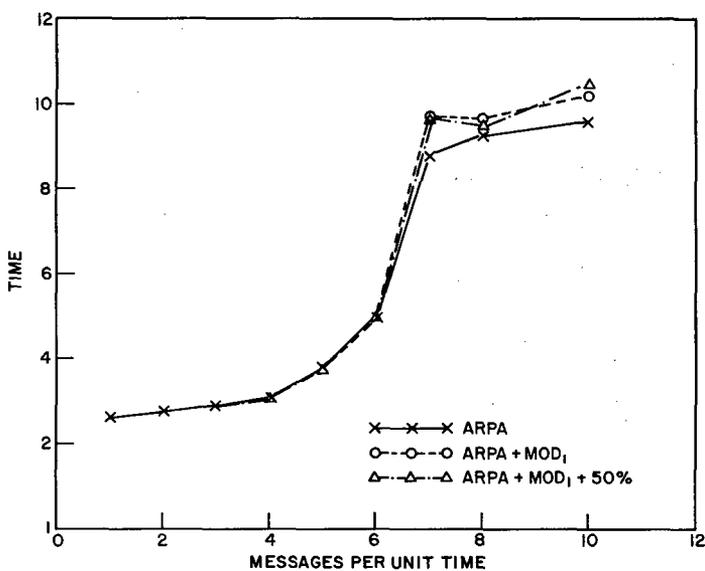


Fig. 5.8 - Average Message Delay on 19 Node Net With One Node Disabled

TABLE 5.12

AVERAGE TIME FOR MESSAGES ON A 19 NODE ARPA NETWORK WITH ONE NODE DISABLED

λ_i	<i>NMSG</i>	<i>BA</i>	<i>NR</i>	<i>SP</i>	<i>BA + 50%</i>	<i>NR + 50%</i>	<i>SP + 50%</i>
0.053	1	2.85	2.85	3.06	2.85	2.85	3.07
0.105	2	2.85	2.85	3.60	2.85	2.85	3.25
0.165	3	2.95	2.95	4.24	2.95	2.95	4.61
0.211	4	3.15	3.16	4.32	3.15	3.16	5.84
0.263	5	3.93	3.93	6.40	3.89	3.77	8.26
0.316	6	5.43	5.43	8.16	5.51	5.51	8.17
0.368	7	7.30	7.43	9.12	7.84	7.84	8.96
0.421	8	8.70	8.74	8.75	8.68	8.68	9.18
0.526	10	10.77	10.77	10.18	9.96	9.96	9.76

TABLE 5.13

AVERAGE TIME FOR MESSAGES ON A 19 NODE ARPA NETWORK WITH ONE NODE DISABLED

λ_i	<i>NMSG</i>	<i>ARPA</i>	<i>MOD₁ + ARPA</i>	<i>MOD₁ 50% + ARPA</i>
0.053	1	2.60	2.60	2.60
0.105	2	2.78	2.78	2.78
0.165	3	2.88	2.88	2.88
0.211	4	3.11	3.11	3.09
0.263	5	3.80	3.80	3.69
0.316	6	5.07	5.05	5.19
0.368	7	8.86	9.79	9.77
0.421	8	9.33	9.68	9.50
0.526	10	9.61	10.22	10.45

TABLE 5.14
MESSAGE THROUGHPUT FOR A 19 NODE ARPA NETWORK WITH ONE NODE DISABLED

λ_i	<i>NMSG</i>	<i>BA</i>	<i>BA + 50%</i>	<i>NR</i>	<i>NR + 50%</i>	<i>SP</i>	<i>SP + 50%</i>
0.053	1	251(0)	251(0)	251(0)	251(0)	250(1)	251(0)
0.105	2	443(1)	443(1)	443(0)	443(1)	436(2)	443(2)
0.165	3	729(0)	729(0)	729(0)	729(0)	686(19)	659(47)
0.211	4	942(1)	942(1)	942(1)	942(1)	710(152)	705(151)
0.263	5	1216(0)	1216(0)	1216(0)	1216(0)	716(313)	631(433)
0.316	6	1418(4)	1425(4)	1418(4)	1425(4)	749(494)	778(460)
0.368	7	1575(50)	1591(36)	1595(35)	1591(36)	629(793)	594(827)
0.421	8	1530(203)	1507(191)	1530(203)	1507(191)	626(905)	740(820)
0.526	10	1364(570)	1411(553)	1364(570)	1411(553)	708(1205)	669(1150)

TABLE 5.15
MESSAGE THROUGHPUT FOR A 19 NODE ARPA NETWORK WITH ONE NODE DISABLED

λ_i	<i>NMSG</i>	<i>ARPA</i>	<i>ARPA MOD₁</i>	<i>ARPA MOD₁ + 50%</i>
0.053	1	250(0)	250(0)	250(0)
0.105	2	443(0)	443(0)	443(0)
0.165	3	730(0)	730(0)	730(0)
0.211	4	942(0)	942(0)	944(0)
0.263	5	1215(0)	1215(0)	1212(0)
0.316	6	1432(0)	1430(1)	1425(5)
0.368	7	1490(123)	1468(110)	1427(137)
0.421	8	1475(250)	1472(247)	1420(284)
0.526	10	1277(724)	1320(670)	1345(679)

SUMMARY OF 19 NODE ARPA NETWORK

The results of the 19 node ARPA network do not demonstrate the dramatic effects of program modification as in the 8 node network. The 8-node network was a highly connected one, in addition to being very symmetrical in its topology. As a consequence of network design, it may be stated with some reservation that the high connectivity would lend itself to higher throughput, smaller message delays, and fewer undelivered messages, even with some small percentage of its links disabled.

The 19 node ARPA network is not highly connected and appears vulnerable to any disabling of nodes and links. Higher connectivity results in greater overhead cost, hence the ideal topology of N nodes with $N(N - 1)$ links fully duplexed would be very costly. With the 19 node net structuring, the ARPA routine works fairly well and any modifications to it improve it only slightly. However, the ARPA routine takes a lot of updating and bookkeeping, hence, from the standpoint of minimum computer hardware and software it could not be viewed as optimum.

A preliminary conclusion at this juncture would be to state that MOD_1 with or without the aging aspect tends to help any routine thus far studied. It is further envisioned that a more simpler algorithm than the ARPA used in conjunction with MOD_1 and the aging parameter could yield effective results for most net topologies, as demonstrated by $BL + 50\%$. Backward learning is a rather simplified concept as well as easy to implement in software. If we allow that simplicity in software implies minimum hardware, then we could assert that some routine similar in implementation to BL might have a significant effect on the ARPA network.

It is noteworthy that a value of 50% for the aging factor, proved to be optimum on both net configurations, with and without destruction. Further, this value of 50% is in keeping with our preliminary predictions in Chapter III, that throughput peaked in the vicinity of a priority threshold setting of $k = 50\%$.

Chapter VI will compare some numerical computations against some selected simulation data of this Chapter. The comparison will show how the predicted throughput factor agrees with the simulated data when using the optimal value of priority threshold k .

VI. THEORETICAL MODEL

Communication networks do not lend themselves to tractable mathematical models, particularly when the networks are inter-connected. The store-and-forward procedures of computer communication further complicates the analysis. When adaptive routing vs. fixed routing is incorporated, the mathematical modeling is virtually impossible. The problem becomes obfuscated even further when finite buffers (queues) are incorporated, along with finite limits on channel capacities. There is in fact no known mathematical analysis to describe a physical computer communication network under all loading conditions.

The only approach to this seemingly insoluble dilemma is by some appropriate mathematical approximations. The technique is generally one of analyzing a single node for the parameters of interest and extending the analysis over the entire set of nodes comprising the network. The theoretical model then becomes a close approximation to actual operating conditions within the network.

6.1 AVERAGE MESSAGE DELAY

It has been stated in prior chapters that one of the measurement parameters of interest is the average message delay. The average message delay gives an excellent over-view on network performance when viewed by the user. High message delays would discourage users who are attempting to use the system and would tend to reduce subscribers in such a system. For a fixed investment in initial overhead for setting up a computer network, fewer subscribers would tend to increase the cost per subscriber to the point of further reducing network usage. Hence, in a real system, the average message delay plays a very important role and could directly effect the network utility, or lack of it.

The average message delay has to be calculated on the basis of whether or not there is some priority classification and on the average number of nodes visited.

6.1.1 Average message delay with no priorities in the system

The delay experienced by a message has been developed from queueing theory and can be developed by two different theoretical approaches: one through differential difference equations due to Erlang, and the other through integral equations studied by [Lindley 52]. There is an excellent resumé of queueing theory by Saaty [SA57] giving most of the formulations generally used in queueing analysis.

In a single server queue with Poisson arrivals and exponential servicing the average delay in the steady state is given by:

$$W = \int_0^{\infty} \tau P(<\tau) = \frac{\rho}{\mu(1-\rho)}, \rho < 1, \quad (6.1)$$

where

$$P(<\tau) = \sum_{n=0}^{\infty} p_n w_n(\tau) = 1 - \rho e^{-\mu\tau(1-\rho)}, \quad (6.2)$$

and

$$W_n(\tau) = 1 - \sum_{i=0}^{n-1} \left[\frac{(\mu\tau)^i}{i!} \right] e^{-\mu\tau}. \quad (6.3)$$

When the service time is arbitrary and the input process is still Poisson, the steady state delay becomes:

$$W = \frac{\rho}{2\mu(1-\rho)} [1 + (s\mu)^2]. \quad (6.4)$$

Where s is the standard deviation of the service time, hence for constant servicing, $s=0$ and the average delay becomes:

$$W = \frac{\rho}{2\mu(1-\rho)}. \quad (6.5)$$

It should be noted here that the assumption on service discipline was FCFS, however, when the queue discipline is changed, the distribution of waiting times change but not the average waiting time. One can further note from equations (6.1) and (6.5) that the average waiting time is directly effected by the service distribution. The average delay for constant servicing is 1/2 that for exponential servicing for the same utilization factor ρ .

The average delay given by the above formulations are associated with the delay at a single-server or node. If we knew the average number \bar{N} of nodes visited by a given message, then the total average delay in the system of an arbitrary message would be:

$$D_T = \bar{N} W. \quad (6.6)$$

Equation (6.6) has been further simplified where it is assumed that the delay is the same at all nodes visited and are independent random variables identically distributed.

6.1.2 Average delay for a two class priority system

Priority queueing assumes that the class of messages arriving at a given service facility, need special handling either within the queue or with their service requirements. The priority classification can be established with some parameter $p(1, 2, \dots, k)$, where 1 denotes the class with highest priority and k the lowest. The service discipline will be non-preemptive and FCFS within the priority groups.

Cobham [COB54] formulated a generalized equation for finding the expected waiting time for messages of each priority level in a single-server (unsaturated) waiting line system. His equation takes the form:

$$W_p = \frac{1}{2}\lambda \int_0^{\infty} t^2 dB(t)/(1-\sigma_{p-1})(1-\sigma_p), \quad (6.7)$$

where

$$\rho_i = \lambda_i/\mu_i \text{ and } \lambda = \sum_1^N \lambda_i, \quad (6.8)$$

$$\sigma_p = \sum_1^k \rho_i < 1, \sigma_0 \equiv 0,$$

$$B(t) = (1/\lambda) \sum_1^N \lambda_i B_i(t). \quad (6.9)$$

$B_i(t)$ is the cumulative service time distribution function for the i^{th} priority. Thus for a two class priority system ($p=1,2$), the average message delay for the priority message becomes:

$$W_1 = W_0/(1-\rho_1), \quad (6.10)$$

where

$$W_0 = \frac{1}{2}\lambda \int_0^{\infty} t^2 dB(t) \equiv \text{average time to complete existing service}.$$

For the non-priority ($p=2$) messages, we obtain

$$W_2 = W_0/(1-\rho_1)(1-\rho_1-\rho_2). \quad (6.11)$$

When priority assignments are established within the system, such that

$$p(k) = P_r(D_T \geq k m), \quad (6.12)$$

where $p(k)$, k , and m are as defined in Chapter III, then

$$\lambda p(k) \equiv \lambda_1,$$

and

$$\lambda(1-p(k)) \equiv \lambda_2. \quad (6.13)$$

The unconditional delay becomes:

$$\begin{aligned}
 W &= p(k)W_1 + (1-p(k))W_2 \quad (6.14) \\
 &= \frac{W_o}{1-\rho} \left[\frac{1-p(k)\rho}{1-\rho_1} \right] \quad \rho < 1 ,
 \end{aligned}$$

where

$$1-\rho_1-\rho_2 = 1-\rho.$$

Cole's formulation [(0 71)] has essentially the same form and is given by:

$$W = W(FCFS) \left[\frac{1-\alpha \lambda \bar{x}}{1-\alpha \lambda x_1} \right] , \quad (6.15)$$

where

$$\bar{x} = \alpha x_1 + (1-\alpha) \bar{x}_2 ,$$

and

$$W(FCFS) = W_o/(1-\rho).$$

For the case where servicing is the same for both classes, i.e., $\mu_1=\mu_2$, then equations (6.14) and (6.15) reduce to

$$W = W(FCFS) , \quad (6.16)$$

since $\bar{x} = \bar{x}_1$, when $\mu_1=\mu_2$ in equation (6.15).

The average message delay D_τ cannot be expressed like that for the single class system given by equation (6.6), but instead must be given by

$$D_\tau = \bar{n} W_1 + \bar{m} W_2 , \quad (6.17)$$

where \bar{n} = average number of nodes visited as a priority message

and

\bar{m} = average number of nodes visited as a non-priority message.

Equation (6.17) assumes messages are independent random variables, identically distributed for a given message class. As with equation (6.6) the average number of nodes visited is also a random variable, whose distribution has to be determined. The determination of the distribution function for the number of nodes visited by a message within a communication network has not yet been solved.

6.2 EFFECT OF A HIGH UTILIZATION FACTOR ρ

The discussion and analysis up to this point has dealt with steady state formulations ($\rho < 1$). When the utilization factor $\rho \rightarrow 1$ or $\rho > 1$, then the system has no steady state solution and the average waiting time approaches infinity. P.M. Morse [M058] has studied the transient case where $\rho > 1$ and develops some formulations for the time-dependent probabilities. Although there exists a mathematical model for over saturated systems ($\rho > 1$), these formulations are more esoteric than of practical concern.

In a real physical communication network the conditions when $\rho > 1$ are intolerable from the standpoint of prohibitive queueing delays. Although a real system can become over saturated this is generally a short term phenomena since users will tend to subside when networks are over-loaded resulting in the utilization becoming unsaturated again, i.e., $\rho < 1$. Thus in the final analysis the transient case is just that, a condition that will not persist very long and steady state will be the rule, rather than the exception.

6.3 EFFECT OF PRIORITY ASSIGNMENT

The justification for priority assignment generally stems from desiring to minimize the average message delay and maximize the system throughput. The majority of priority disciplines give high priority to those classes with low mean servicing time. The proof that such a rule leads to optimality requires the a priori arrival and service distributions of each class are known and have stationary statistics.

The technique employed in this research suggests that the overall performance of the system may be improved if a message, after reaching some queueing delay, is given in a non-preemptive priority. The idea was to minimize the queueing delay by an optimal selection of priority threshold $p(k)$. Where $p(k)$ has been defined as

$$p(k) = P_r(km \leq t < m), 0 \leq k \leq 1, \quad (6.18)$$

where

m = maximum allowable message delay.

However, it appears at first glance from equation (6.14) that the average message delay

$$W = \frac{W_0}{1-\rho} \left[\frac{1-p(k)\rho}{1-\rho_1} \right], \rho < 1 \quad (6.19)$$

is invariant to the selection of $p(k)$, equation (6.19) is equation (6.14) repeated for convenience. Since we defined the priority rate as

$$\lambda_1 = p(k)\lambda, \quad (6.20)$$

and when the servicing for both priority classes are equal

$$\mu_1 = \mu_2 = \mu.$$

Then equation (6.19) reduces to

$$W = \frac{W_0}{1-\rho} = W(FCFS),$$

since

$$\rho_1 = \frac{\lambda_1}{\mu} = p(k)\lambda/\mu = p(k)\rho.$$

There appears to be some contradiction with (6.22) and the results of the simulation given in Chapters IV and V that show average message delay is reduced when priority assignments are initiated. The simulation gives the average message delay over the network as opposed to an arbitrary node given by (6.17) and (6.19). The missing parameter that makes up for this apparent discrepancy, is the average number of nodes visited. Note in equation (6.17) the random variables \bar{n} and \bar{m} representing the number of nodes visited by a priority and non-priority message respectively. These two random variables are the quantities that can not be readily assessed by analytic means and are directly effected by the updating routines that in turn are effected by priority assignment.

The priority assignment helps to relieve congestion at the nodes, hence reducing over all queueing delay. The reduction in queueing delay at previously congested nodes change the routing of messages and consequently the number of nodes visited. The final effect results in a smaller value of average message delay D_T when viewed over the entire network. The adaptive routing process negates mathematical modeling, therefore one can only obtain a heuristic analysis.

6.3.1 Balking and renegeing

Balking is the condition where an arriving message may not join the queue because of the length of the existing queue. In this study the queues are of finite size, hence balking is attributed to those messages finding the queue full. The utilization factor directly effects balking by filling the queues faster than servicing can dispatch the messages. For a finite queue of size N , with Poisson input and exponential servicing, P.M. Morse gives the following probability of finding n messages in the queue

$$P_n = \left[\frac{1-\rho}{1-\rho^{N+1}} \right] \rho^n, \rho < 1, \quad (6.23)$$

where

N = queue size .

The mean number in the system (in queue and in service) is

$$L = \sum_{n=0}^N nP_n = \rho \frac{1(N+1)\rho^N + N\rho^{N+1}}{(1-\rho)(1-\rho^{N+1})}$$

$$\rightarrow \begin{cases} \rho + \rho^2 & (\rho \ll 1) \\ \frac{1}{2} N + \frac{1}{12} N(N+2)(\rho-1) & (\rho \rightarrow 1) \\ N-(1/\rho) & (\rho \gg 1) \end{cases} \quad (6.24)$$

The probability of balking becomes:

$$P_N = \left[\frac{1-\rho}{1-\rho^{N+1}} \right] \rho^N \quad (6.25)$$

The number of messages finding the queue full is heavily dependent on ρ . If for a fixed service rate μ , the message arrival rate λ increases, then the fraction of messages balking given by P_N increases rapidly, approaching $1-(1/\rho)$ when ρ is much larger than unity. Thus in an over saturated system ($\rho \gg 1$) only $(1/\rho)$ of the messages get into the system. Conversely when $\rho \ll 1$, the system is being under-utilized and the probability of balking approaches zero. Balking can also be viewed from the probability that the queue is empty given by

$$P_o = \left[\frac{1-\rho}{1-\rho^{N+1}} \right] \quad (6.26)$$

From (6.26) one can see that for $\rho \ll 1$ $P_o \rightarrow 1$, hence no balking and for $\rho \gg 1$, $P_o = 1/\rho^N$ and balking is highly probable. The number of units balking cannot be readily calculated.

A renege is when a message due to its aging may expire from the system. The probability that a message will renege is given by

$$P_r(D_T > MDT_{max}) \equiv \text{renewing}, \quad (6.27)$$

where

$$D_T = \bar{n} W_1 + \bar{m} W_2 \text{ (see 6.17),}$$

and

MDT_{max} is the maximum allowable message delay.

When $\rho \rightarrow 1$, the equations for average message delay given by (6.10) and (6.11) give very large values for W_1 and W_2 respectively (these equations are not defined for $\rho \geq 1$). The probability that a message will expire, hence renege, becomes very probable as $\rho \rightarrow 1$. The numerical number of messages that have expired can be approximated from the total messages generated per unit time, times $P_r(D_T > MDT_{max})$. When the input is Poisson the number of messages renegeing per unit time becomes:

$$\gamma_n = P_r(D_T > MDT_{max}) \sum_{i=1}^N \lambda_i. \quad (6.28)$$

Then γ_n will be the expiring messages per unit time in a N node network.

6.3.2 Throughput factor $\phi(t,k)$

The utilization factor also directly influences the throughput factor $\phi(t,k)$ as defined in section (3.5). The defining equation for the throughput factor given by equation (3.43) show the implicit dependence of $\phi(t,k)$ on ρ . Although the equations are for values of $\rho < 1$, it can be readily shown that the throughput factor decreases with increasing ρ . From the basic definition of the throughput factor it is clear that an increasing ρ , increases the percent of undelivered messages $\nu(t,k)$, hence decreases $\phi(t,k)$. The influence of ρ on $\phi(t,k)$ is further demonstrated in the next section.

6.4 COMPARISON OF PREDICTED TO SIMULATION RESULTS

The computer runs of Chapters (IV) and (V) showed some tendencies to bear out analytic predictions. Numerical calculations of average queuing delay, number of messages balking and renegeing cannot be readily assessed due to the inter-connectivity of computer networks, and to alternate (adaptive) routing procedures.

There appears to be an area where the predicted values were to within fractions of a percentage to the simulation results for the throughput factor $\phi(t,k)$. Table (6.1) shows the computed $\phi(t,k)$ versus the simulated. The agreement was very good from almost an unmeasurable difference to within 1.1% for the worst case. The comparison was made with the exact values of ρ and priority k for both cases. The curves of Fig. (6.1) are a plot of the tabulated values of Table (6.1).

TABLE 6.1
COMPUTED VS. SIMULATED THROUGHPUT FACTOR $\phi(t,k)$

	CASE NUMBER	UTILIZATION FACTOR ρ	PRIORITY k FACTOR	% UNDEL. MESSAGES $\nu(t,k)$	THROUGHPUT FACTOR $\phi(t,k)$	% DIFFERENCE BETWEEN COMPUTED & SIMULA.
SIMULATED	1	.158	.50	0.0	1.0	No Significant Difference
COMPUTED	1	.158	.50	0.26×10^{-6}	.9999	
SIMULATED	2	.316	.50	0.0	1.0	No Significant Difference
COMPUTED	2	.316	.50	0.42×10^{-6}	.9999	
SIMULATED	3	.474	.50	.052	.948	.0007
COMPUTED	3	.474	.50	.0513	.9487	
SIMULATED	4	.632	.50	.079	.921	.011
COMPUTED	4	.632	.50	.068	.932	
SIMULATED	5	.790	.50	.203	.797	.008
COMPUTED	5	.790	.50	.195	.805	

The simulated data comes from the ARPA run using Backward Learning plus 50% aging (BL+50%), and these are tabulated in Table 5.7 of Chapter V. The computed data refers to theoretical calculations obtained from Eqs. (3.42), (3.43), and (3.44) and is associated with the 3-node network of Chapter III. A judicious choice of λ_1' and λ_2' in Eq. (3.44) created the same loading effects in the computed (theoretical) 3-node net as the 19 node simulated net, even though the two networks had different message volumes for the same value of ρ .

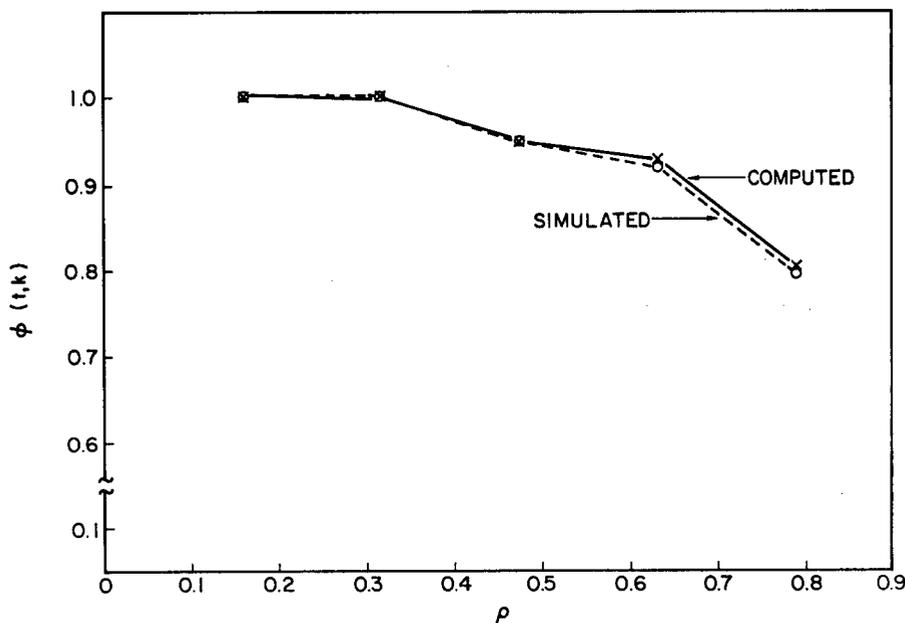


Figure 6.1 – Computed Vs. Simulated Throughput Factor $\phi(t,k)$

It is also instructive that the percent of undelivered messages $\nu(t,k)$ is analogous to the percent of messages renegeing in the system. Thus there is a way to get a reasonable prediction of the percentage of messages that will be undelivered for some selected values of ρ . The actual predicted throughput $\psi(t)$ for any network loading can be roughly approximated by the following (where the input is Poisson).

$$\psi(t) = \sum_{i=1}^N \lambda_i T , \quad (6.29)$$

where

λ_i = rate per node/unit time ,

T = total time increment ,

N = total number of nodes in network.

In an actual network one could not readily find the various values of λ_i and these would have to be approximated by appropriate assumptions regarding network loading and average servicing rate.

VII. SUMMARY AND CONCLUSIONS

7.1 SUMMARY

Several techniques for adaptive routing algorithms in a specific class of networks have been examined. In particular, it has been demonstrated by simulation how some relatively simple add-ons to already existing adaptive routines could decrease the average message delay and increase throughput in the selected networks. A further objective of this study was to assess the effects of priority assignment to messages that had reached some specified aging threshold and note the effects of such priority assignments on network performance. The performance measures considered were average message delay, throughput, and number of messages undelivered.

A historical review of communication networks was given in Chapter I. Chapter II dealt with deterministic and stochastic routing strategies and a brief overview of how they operate. It was asserted in Chapter III that there was a general lack of knowledge of system specifications that renders extensive analysis of store-and-forward networks to be essentially intractable. However, a measurement parameter called the throughput factor $\phi(t,k)$ was introduced. This parameter could be optimized in terms of the selection of the priority threshold k . The closed form solution for $\phi(t,k)$ was made on a 3-node network and it was envisioned that this analysis could be extended to predict performance on some specific interconnected network. The problem of finite buffers, network interconnections, and adaptive routing makes exact analysis intractable.

Chapters IV and V dealt with the simulation study on an 8-node and 19-node network respectively. The simulation was to test the performance of some selected algorithms along with the modifications suggested by this study, and note how the algorithms fared when there was no network damage and then with link and node failures. The 8-node highly connected network was a preliminary run with the routing algorithms and the 19-node ARPA net was an attempt to verify the consistency of the algorithms on a larger net topology.

Finally, Chapter VI was a comparison of predicted performance with actual simulated results. This comparison of the analytic with the simulated can be only viewed as cursory due to the limitations of the mathematical modeling.

7.2 CONCLUSIONS

The technique of using last node visited (LNV) and an aging parameter for priority selection can be viewed as an asynchronous, stochastic routing algorithm. When the network is lightly loaded and there are no link or node outages, then the minimum path algorithms would work rather well. When the network gets congested at some node or nodes and traffic intensity becomes heavy, the adaptive routing algorithms suggested in this paper attempt to unblock the system and keep average message delay at a minimum and throughput at a maximum. Hence, there appeared to be a significant improvement over all routing algorithms when MOD_1 and 50% aging was applied to them. It is noteworthy that the 50% threshold value was the optimum setting in the simulation and corresponded closely to the predicted value.

There are a couple of caveats that should be stressed at this point. First, although Chapter VI showed a very close comparison (within 1%) of the predicted throughput factor $\phi(t,k)$ with that of the simulated data of Chapter V, it cannot be considered conclusive. The predicted $\phi(t,k)$ was made on a net topology where the average path length did not correspond to that of the simulated networks. However, by a judicious choice of input message rates λ_2' and λ_1' it was possible to create the same net loading effects thus giving some very close comparisons between predicted and simulated throughput factor.

The second area of caution deals with the appearance that backward learning (BL) can outperform the ARPA routine, as demonstrated in Chapter V. Both ARPA and BL inquire about nodes and links via nearest neighbor. BL does this inquiry by incoming messages; hence, under heavy loading it gets current information. ARPA works better at lightly loaded nets using nearest neighbor information, and at higher traffic intensities its updating is slightly delayed; also, it adds to the net loading with its inquiries. However, the BL routine cannot adjust effectively to nodes or links being restored and its updating techniques generally yield inadequate or even misleading table delay values. The tendency to "ping-pong" is very probable when using BL techniques. The ARPA routine is both synchronous and asynchronous in that it updates every $\frac{1}{2}$ second, but also if there have been link or node changes.

In either case, it appears that adding the asynchronous technique of MOD_1 and/or 50% aging, all the selected algorithms performed better and in some cases (Chapter IV) the effects were dramatic.

7.3 FUTURE WORK

It appears that additional simulation techniques are desirable to further evaluate routing algorithms on store-and-forward systems. The technique of asynchronous updating or adaptation should be further explored, since such techniques minimize network loading by minimizing traffic inquiries about the state of the system.

The problem of analytic modeling still needs to be addressed to obtain a closer approximation of predicted performance to simulation results. Not only is there the apparent limitation of the mathematics but also the normal lack of complete knowledge of the system specifications on which to use such analysis.

The area of an adjustable threshold might also prove enlightening. The technique in this study was to select an optimal value of k for all $\rho < 1$. It would be interesting to see how selecting k based on the actual message intensity would effect average message delay and throughput.

APPENDIX A

QUEUEING THEORY AND FORMULATIONS

A.1 QUEUEING THEORY

A queue, or a waiting line, involves items arriving at some service facility and their subsequent wait to be served. The items in the queue may be people, vehicles, commodities, or almost anything that joins a waiting line for the purpose of being served; there are almost unlimited specifications for the server.

The principal elements of a queue are the inputs, waiting line, and the service facility; thus the statistics of a queue are associated with the fluctuations of these three elements. The ratio of the rate of input (λ) and the departure rate (μ) has been generally defined as utilization factor $\rho = \lambda/\mu$, and it's this factor that directly affects the size (length, total number, etc.) of a given queue. Optimizing any system of queues depends directly on minimizing congestion at the node points where queues develop.

In a store-and-forward communication network the subject of queueing must be addressed to handle congestion at the nodes within the network. Messages in communication nets are sent from node i to some node j ($i \neq j$) via some intermediate nodes k ($k = 0, 1, 2, \dots, N$) and it's the congestion at the intermediate nodes k , where queueing occurs and adds to the average message delay. The source nodes i , destination nodes j , and intermediate nodes k are all random variables generally from a Poisson distribution. The problem of adequate waiting room within the queue is important, especially when the queue size is finite.

The queueing problem for nets is usually concerned with knowing the input distribution, the queue discipline (e.g., random, ordered, or priority selection for servicing), and the service-time distribution to determine the desired measures of effectiveness. For problems dealing with finite queue length, one must also determine the probability of balking (not joining the queue). In addition, due to time limitations imposed on the message's stay in the system, one must determine the probability of abandoning the queue (reneging) after joining it due to exceeding the imposed aging limit.

The literature on queues is very extensive and Saaty [SA61] lists over 900 references and develops many other specifications for describing various forms of the queueing process; Syski [SY60] has over 400 references on queueing and congestion theory. Morse [MO58] and Cox [COX61] are other texts useful for the engineering development, and Jaiswal [JA68] treats exclusively the subject of priority queues. The above mentioned references are more than adequate to find a very diversified treatment of the queueing process.

The queueing model generally used for a network is the single exponential channel. Thus one can get the characteristic behavior of a large class of more complicated systems when the following are assumed.

1. The interarrival times are Poisson-distributed with an average arrival rate of λ messages per unit time.

2. The message lengths are exponentially distributed with an average length of $1/\mu$ bits per message.
3. Queue discipline is first-come first-served with infinite storage capacity.
4. There is one channel available for transmission.

With the above assumptions, and with the abundance of available literature on the single exponential channel ([MO58] and [SA61]), the following steady state statistics can be readily found:

- a. $P_r [n \text{ messages in the system}] = (1 - \rho) \rho^n, \rho < 1$. (A.1)
- b. $E(n) = \text{expected value of number of messages in the system} = \frac{\rho}{1 - \rho}$.
- c. $T = \text{average time message spends in the system} = \frac{1}{\mu(1 - \rho)}$.
- d. $Pr [\text{more than } n \text{ messages in the system}] = \rho^{n+1}$.
- e. $Pr [\text{total time message spent in system} > t] = e^{-(1-\rho)\mu t}$.

The above quantities are derived with the aid of the birth-death process and resulting forward and backward equations, and assuming the existence of a limiting distribution for $P_n(t)$, such that

$$\lim_{t \rightarrow \infty} P_n(t) = P_n \quad , \quad (\text{A.2})$$

and further that the limiting distribution is such that

$$\sum_{n=0}^{\infty} P_n = 1 \quad . \quad (\text{A.3})$$

The above mentioned process is Markovian and the techniques for solving them are well covered in the literature [Feller 57 and Kendall 51]; the use of Little's [LI61] equation allows for a quick calculation of the expected time a message spends in a queueing system (in a conservative system). Little proved that in a conservative system (i.e., messages are neither created or destroyed) the following is always true:

$$\bar{n} = \lambda T, \quad (\text{A.4})$$

where

$\bar{n} \equiv$ expected number of messages in a queueing system

$T \equiv$ expected time they spend in the system

and

$\lambda \equiv$ average arrival rate.

Thus from Eq. (A.1), item b,

$$E(n) = \bar{n} = \sum_{n=0}^{\infty} n P_n = \frac{\rho}{1-\rho}, \rho < 1, \quad (\text{A.5})$$

and from the definition of utilization factor ρ

$$\rho \equiv \lambda/\mu, \quad (\text{A.6})$$

we can find the average time T from Little's equation

$$T = \bar{n}/\lambda = \frac{\rho}{\lambda} \frac{1}{(1-\rho)} = \frac{1}{\mu} \frac{1}{(1-\rho)}, \quad (\text{A.7})$$

which is given by item c. The formulations just presented were for a Markovian M/M/1 system with infinite queueing room, where both arrival and service are Markovian.

A.2 IMBEDDED MARKOV CHAIN

The system under consideration in this study is a M/G/1 system. This type of system employs a general service time distribution and the memoryless property no longer exists; hence, the employment of the birth-death process is no longer valid. The notion of an imbedded Markov chain is due to D. G. Kendall [KE51&53] and was introduced by him so that non-Markovian processes could be studied by extracting a set of *regeneration* points for which the Markov property holds. In a M/G/1 system, the set of *departure* instants from service is an extremely convenient set of regeneration points. Thus, if we specify the number of messages left behind (in the queue) by a departing message (one already served), we can calculate the same quantity at some point in the future given only additional inputs to the system. The concept or technique of the imbedded Markov chain will allow analysis for our priority queueing system.

A.3 POLLACZEK-KHINTCHINE FORMULA

The average number of queueing customers (those customers waiting in the queue) left behind by a departing customer is given by the Pollaczek-Khintchine formula

$$E[q] = \rho + \frac{\rho^2 + \lambda^2 \text{var}(t)}{2(1-\rho)} \quad (\text{A.8})$$

Once we know the variance of the service time t from its given distribution, the average number of messages in the queue can be determined. We must note that the above average has been taken over instants just following departures and thus does not include what's presently being serviced.

To obtain the average waiting time we can follow the argument presented by Saaty [SA61] who argued as follows: If we write $E[W]$ for the average waiting time in the queue (not including service), $\lambda [E[W] + 1/\mu]$ is the expected number of arrivals during the total waiting plus service of one message; i.e., its stay in the system. But this must be just the number in the system immediately after its departure; namely $E[q]$. Hence

$$W_q \equiv E[W] = \frac{\rho^2 + \lambda^2 \text{var}(t)}{2\lambda(1-\rho)} = \frac{L_q}{\lambda} \quad (\text{A.9})$$

We can obtain equation (A.9) by direct application of Little's equation and substituting λ/μ for ρ , then

$$W_q = T = \frac{E[q]}{\lambda} = \frac{1}{\mu} + \frac{\rho + \lambda \mu \text{var}(t)}{2\mu(1-\rho)} \quad (\text{A.10})$$

where

$$\bar{n} = E[q] = L_q \quad ,$$

and

$$W_q = T \quad .$$

Both equation (A.9) and (A.10) have the same reduced form and state that the *average* total time spent in the system is the average time spent in the service plus the average time spent in the queue. There are numerous other statistical parameters that can be assessed in a queueing problem such as length of a busy period, unfinished work, probability of being idle, etc. The list can get quite lengthy, but for the purpose of this study we have limited our attention to what has been previously stated. Our main concern is minimum average message delay and maximum throughput with minimum hardware and software investment. We should further state that this minimum average message delay should be realized from a real-time system and thus some further constraints must be employed when analytical modeling is employed. The problem of priority assignment to messages in the system must be dealt with, along with some type of parameterization to optimize the queueing assignment. The aggregate collection of nodes, links, buffer size, queueing (with and without priorities) and network topology must somehow be systematically developed and assessed to get maximum utilization out of a communication network with minimum overhead. The interplay of the network components and lack of some systematic analytical analysis makes for a very formidable problem.

A.4 WAITING TIME DISTRIBUTION

The waiting time distribution for the non-priority, single-server queue will be used to establish the priority assignment for all messages; i.e., priorities are established endogenously. Messages are given a priority if their wait in the system (queue) is longer than some preselected time T , and this priority threshold T is the same for

all messages. Thus, the probability of $\{\text{wait in } Q \geq T\}$ establishes priority, and

$$Pr \{\text{wait in } Q \geq T\} = 1 - \int_0^T dW(t) \equiv \text{priority assignment} ,$$

where

$$\int_0^T dW(t) \equiv \text{the Stieltjes integral} , \quad (\text{A.11})$$

and

$W(t) \equiv$ waiting time distribution for non-priority messages .

The waiting-time distribution $W(t)$ for the non-priority, single-server queue can be obtained from the Pollaczek-Khintchine formula

$$W(s) = \frac{1 - \rho}{1 - \lambda/s (1 - \beta(s))} , \quad (\text{A.12})$$

where

$$\beta(s) = \int_0^{\infty} e^{-st} dB(t) \equiv \text{the Laplace-Stieltjes transform of the service-time distribution} ,$$

and

$B(t) \equiv$ service-time distribution.

Equation (A.12) has been derived by many authors (e.g., R. G. Miller [MI60], L. Takacs [TA55], and Riordan [RI62] as being the steady state solution for a single class queue with Poisson arrivals (λ) and general service distribution $B(t)$. Thus for a constant service-time distribution

$$B(t) = \begin{cases} 1, & t \geq 1/\mu \\ 0, & t < 1/\mu \end{cases} , \quad (\text{A.13})$$

the derivative of $B(t)$ becomes a Dirac delta function of the form

$$dB(t) = \delta(t - 1/\mu) dt , \quad (\text{A.14})$$

hence

$$\beta(s) = \int_0^{\infty} e^{-st} \delta(t - 1/\mu) dt = e^{-s/\mu} ,$$

and

$$W(s) = \frac{1 - \rho}{1 - \lambda/s [1 - e^{-s/\mu}]} . \quad (\text{A.15})$$

The inverse of $W(s)$ as shown by Riordan [RI] and to have been calculated by A. K. Erland in 1909 (Ref. E. Brockmeyer et al, 1948) is given by

$$W(t) = (1 - \rho) \sum_{n=0}^{[\mu t]} (-1)^n \frac{(\lambda t - \lambda n)^n}{n!} e^{\lambda t - n\lambda}, \quad (\text{A.16})$$

where

$$[\mu t] \equiv \text{least whole integer,}$$

and

$$\mu = 1$$

in this investigation.

A.5 CALCULATION OF THROUGHPUT FACTOR $\phi(t, k)$

The throughput factor is expressed by

$$\phi(t, k) = 1 - v(t, k), \quad (\text{A.17})$$

where

$$v(t, k) = [((\lambda_2'/2) r(t, k) + \lambda_1' S(t, k)) / (\lambda_2'/2 + \lambda_1')].$$

The solution of $v(t, k)$ is given in terms of $r(t, k)$ and $S(t, k)$. From Fig. (3.2) we defined $r(t, k)$ as the probability that a message with source S and destination D is undelivered and $S(t, k)$ is the probability that a message with source M and destination D is undelivered. Then $r(t, k)$ is the sum of the following mutually exclusive independent events:

$$r(t, k) = g(m) + (1 - \lambda_2')P(A) + p(k)P(B) + u(k)P(C), \quad (\text{A.18})$$

and

$$S(t, k) = P(A),$$

where

$g(m) = \Pr(t > m)$ = probability that a message with source S and destination D , expires before it reaches node M .

$(1 - \lambda_2')P(A)$ \equiv the joint probability that a message finds the queue empty at S and expires at D .

$p(k)P(B)$ \equiv the joint probability that a message belongs to a priority class and expires at D .

$u(k)P(C)$ \equiv the joint probability that a non-priority message with time on it expires at D .

We should note that when dealing with a single class system, $H_1(t) = H_2(t)$ for $\rho_2 = 0$ and $\rho_1 = 0$ respectively. The distribution functions $H_1(t)$ and $H_2(t)$ are given by Eqs. (3.1) and (3.3), denoting priority and non-priority distributions respectively.

Unless otherwise noted, it has been assumed that the average service rates are unity and given by

$$\mu = \mu_1 = \mu_2 = 1 \quad (\text{A.19})$$

and hence

$$\rho = \lambda, \rho_1 = \lambda_1, \text{ and } \rho_2 = \lambda_2 .$$

The three probabilities $P(A)$, $P(B)$, and $P(C)$ have been defined as the conditional probability of some event Y , given the event that the observed value of X is equal to x , denoted in symbols by $P(Y|X = x)$. Where the event Y and the random variable X are both defined on the same probability space [PARZEN 62].

From a knowledge of $P(Y|X = x)$ one may obtain $P(Y)$ by the following Stieljes integral formulas:

$$P(Y) = \begin{cases} \int_{-\infty}^{\infty} P(Y|X = x) dF_x(x) \\ \int_{-\infty}^{\infty} P(Y|X = x) f_x(x) dx \\ \sum P(A|X = x) \rho_x(x), \end{cases} \quad (\text{A.20})$$

over all x such that $\rho_x(x) > 0$

in which the last two equations hold if X is respectively continuous or discrete. The solution to $P(A)$ and $P(B)$ are as given in the text; however, $P(C)$ needs further clarification. $P(C)$ is the probability that a message that had to wait at node S (Fig. 3.2) is undelivered at D .

$$P(C) = \Pr(T > m) = \int_{-\infty}^{\infty} \rho_3(t) dF_{T_3}(t) , \quad (\text{A.21})$$

where

$T = T_1 + T_2 =$ total time spent in system

$\left. \begin{array}{l} T_1 = \text{time spent at node } S \\ T_2 = \text{time spent at node } D \end{array} \right\} \equiv \text{random variables assumed to be independent .}$

$$\begin{aligned}
P_3(t) &= \Pr(T > m \mid T_1 = t) \\
&= \Pr(T_1 + T_2 > m \mid T_1 = t) \\
&= \Pr(T_2 > m - t \mid T_1 = t)
\end{aligned}$$

and due to the independence of T_1 and T_2

$$P_3(t) = \Pr(T_2 > m - t), \quad (\text{A.22})$$

also

$$f_{T_3}(t) = \begin{cases} 0, & t > km \\ (1 - \lambda_2')e^{-(1-\lambda_2')t} / [\lambda_2'(1-e^{-(1-\lambda_2')km})], & \text{otherwise} \\ 0, & t \leq 0. \end{cases}$$

Then by application of the distribution function for non-priority messages given by $H_2(t)$ in Eq. (3.3) we obtain:

$$P_3(t) = \frac{\rho^2 - \rho_1}{\rho - \rho_1} e^{-\alpha(m-t)} + \frac{1-\rho}{2\pi} \int_{\alpha_1}^{\alpha_2} e^{-rt} \frac{\sqrt{(\alpha_2 - r)(r - \alpha_1)}}{r(r - \alpha)} dr, \text{ for } \rho^2 \geq \rho_1,$$

where

$$\alpha = (\rho - \rho_1)(1 - \rho)/\rho,$$

$$\alpha_1 = (1 - \sqrt{\rho_1})^2, \quad (\text{A.23})$$

$$\alpha_2 = (1 + \sqrt{\rho_1})^2.$$

Then

$$\begin{aligned}
P(C) &= \int_0^{km} P_3(t) f_{T_3}(t) dt \\
&= \frac{(1 - \lambda_2')}{(1 - \rho^{-(1-\lambda_2')km})} \int_0^{km} e^{-(1-\lambda_2')t} \left[\frac{\rho^2 - \rho_1}{\rho - \rho_1} e^{-\alpha(m-t)} + \frac{1-\rho}{2\pi} \int_{\alpha_1}^{\alpha_2} f(r) dr \right] dt,
\end{aligned}$$

where

$$f(r) = e^{-rt} \frac{\sqrt{(\alpha_2 - r)(r - \alpha_1)}}{r(r - \alpha)}. \quad (\text{A.24})$$

Let

$$K_1 = (1 - \lambda_2') / (1 - e^{-(1-\lambda_2')km}), \quad (\text{A.25})$$

then

$$P(C) = K_1 \left(\frac{\rho^2 - \rho_1}{\rho - \rho_1} \right) e^{-\alpha m} \int_0^{km} e^{-(1-\lambda_2' - \alpha)t} dt \quad (\text{A.26a})$$

$$+ K_1 \left(\frac{1 - \rho}{2\pi} \right) \int_0^{km} e^{-(1-\lambda_2')t} \int_{\alpha_1}^{\alpha_2} f(r) dr dt$$

for $\rho^2 \geq \rho_1$,

and

$$P(C) = K_1 \left(\frac{1 - \rho}{2\pi} \right) \int_0^{km} e^{-(1-\lambda_2')t} \int_{\alpha_1}^{\alpha_2} f(r) dr dt \quad (\text{A.26b})$$

for $\rho^2 < \rho_1$.

We define I_1 and I_2 as follows:

$$I_1 = K_1 \left(\frac{\rho^2 - \rho_1}{\rho - \rho_1} \right) e^{-\alpha m} \int_0^{km} e^{-(1-\lambda_2' - \alpha)t} dt \quad (\text{A.27})$$

$$= K_1 \left(\frac{\rho^2 - \rho_1}{\rho - \rho_1} \right) \frac{e^{-\alpha m}}{(1 - \lambda_2' - \alpha)} [1 - e^{-(1-\lambda_2' - \alpha)km}],$$

$$I_2 = K_1 \left(\frac{1 - \rho}{2\pi} \right) \int_0^{km} e^{-(1-\lambda_2')t} \int_{\alpha_1}^{\alpha_2} f(r) dr dt \quad (\text{A.28})$$

$$= K_1 \left(\frac{1 - \rho}{2\pi} \right) \int_0^{km} e^{-(1-\lambda_2')t} \int_{\alpha_1}^{\alpha_2} e^{-r(m-t)} \sqrt{(\alpha_2 - r)(r - \alpha_1)} / [r(r - \alpha)] dr dt,$$

and reversing the order of integration I_2 becomes:

$$I_2 = K_1 \left(\frac{1 - \rho}{2\pi} \right) \int_{\alpha_1}^{\alpha_2} \frac{e^{-mr} \sqrt{(\alpha_2 - r)(r - \alpha_1)}}{(1 - \lambda_2' - r)(r - \alpha)} [1 - e^{-(1-\lambda_2' - r)km}] dr. \quad (\text{A.29})$$

The solution to Eq. (A.29) can best be handled by numerical techniques using Simpson's rule (see [Krylov 62], pp 94). Thus we can express $P(C)$ as:

$$P(C) = \begin{cases} I_1 + I_2, & \text{for } \rho^2 \geq \rho_1 \\ I_2, & \text{for } \rho^2 < \rho_1, \end{cases} \quad 0 \leq k \leq 1. \quad (\text{A.30})$$

When $k = 0$, $I_1 = I_2 = 0$, hence $P(C) = 0$. From Eq. (3.33) when $k = 1$, $P(B) = 0$, since the limits on the integral become equal.

APPENDIX B

ADAPTIVE ROUTING PROGRAM FOR COMPUTER SIMULATION

The computer simulation for adaptive routing was initially run on a DDP-24 small general purpose computer built by Computer Control Company (3C's). A CDC-3800 was used for a more expanded look at the effectiveness of the selected algorithms.

The framework for the program was set up from the RAND program of Boehm and Mobley [BO66] and was extensively modified to allow for all the adaptive routing techniques addressed in this study. The RAND program originally had no queueing (buffers) or PAL (Preassigned Links), hence messages were generated such that the network had a constant loading. Without buffers, the messages generated had a dependency on the current status of the system; therefore message volume (MV) would not generate the same total number of messages. Incorporating the buffers at each node helped to establish a common basis for comparing the selected routing algorithms. Finally, the RAND program had to use the "Hot-Potato" technique since there was no queueing.

B.1 FLOW DIAGRAM OF ADAPTIVE ROUTINE

The flow diagram for the simulation routine is given in Figs. (B.1a) and (B.1b). The program assumes that the network topology and minimum delay tables are already on magnetic tape. There were two programs used in this study, RANSIM and ARPSIM. Their difference was only in the manner of updating and type of doctrine used (NDOCT). The Fortran listing of the ARPSIM program is given in (B.2). After some prescribed running time T , the program prints out the following:

1. Total messages
2. Undelivered messages, numbers that are too old
3. Number of messages finding queue full
4. Delivered messages
5. Average time for message in network
6. Total number of messages handled by network, and
7. Tabulation of routing tables.

The RANSIM program had enough flexibility to try various updating strategies such as backwards learning, negative reinforcement, etc., with different combinations of routing procedures. Both programs could incorporate options for priority assignment of messages, servicing of queues, and a tag on the message to indicate last node visited, last two nodes visited, etc.

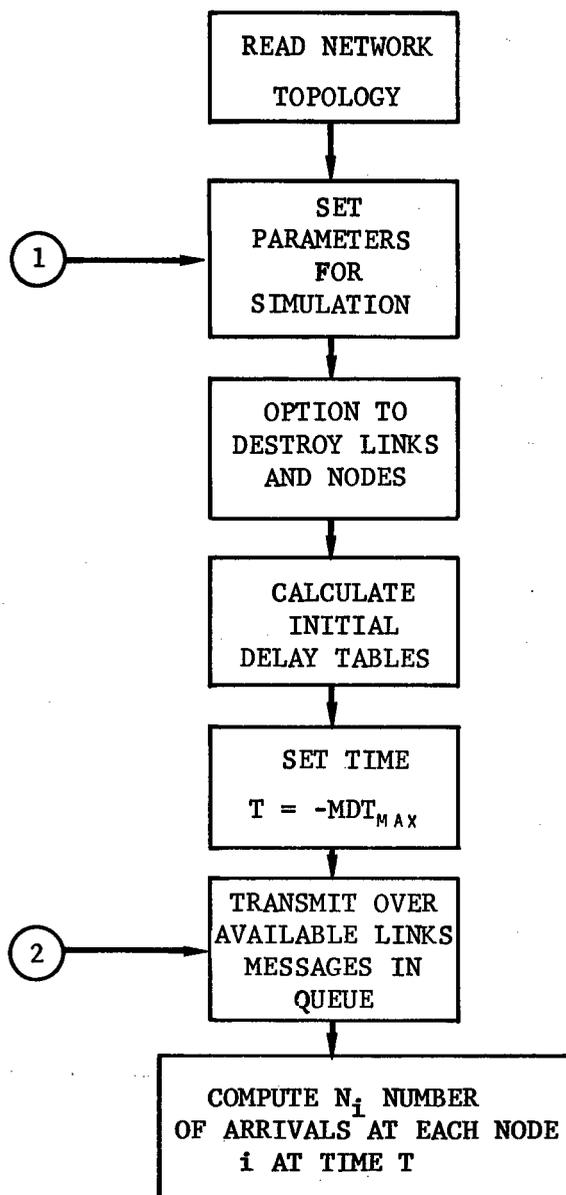


Fig. B1a - Flow Diagram of Program Routine

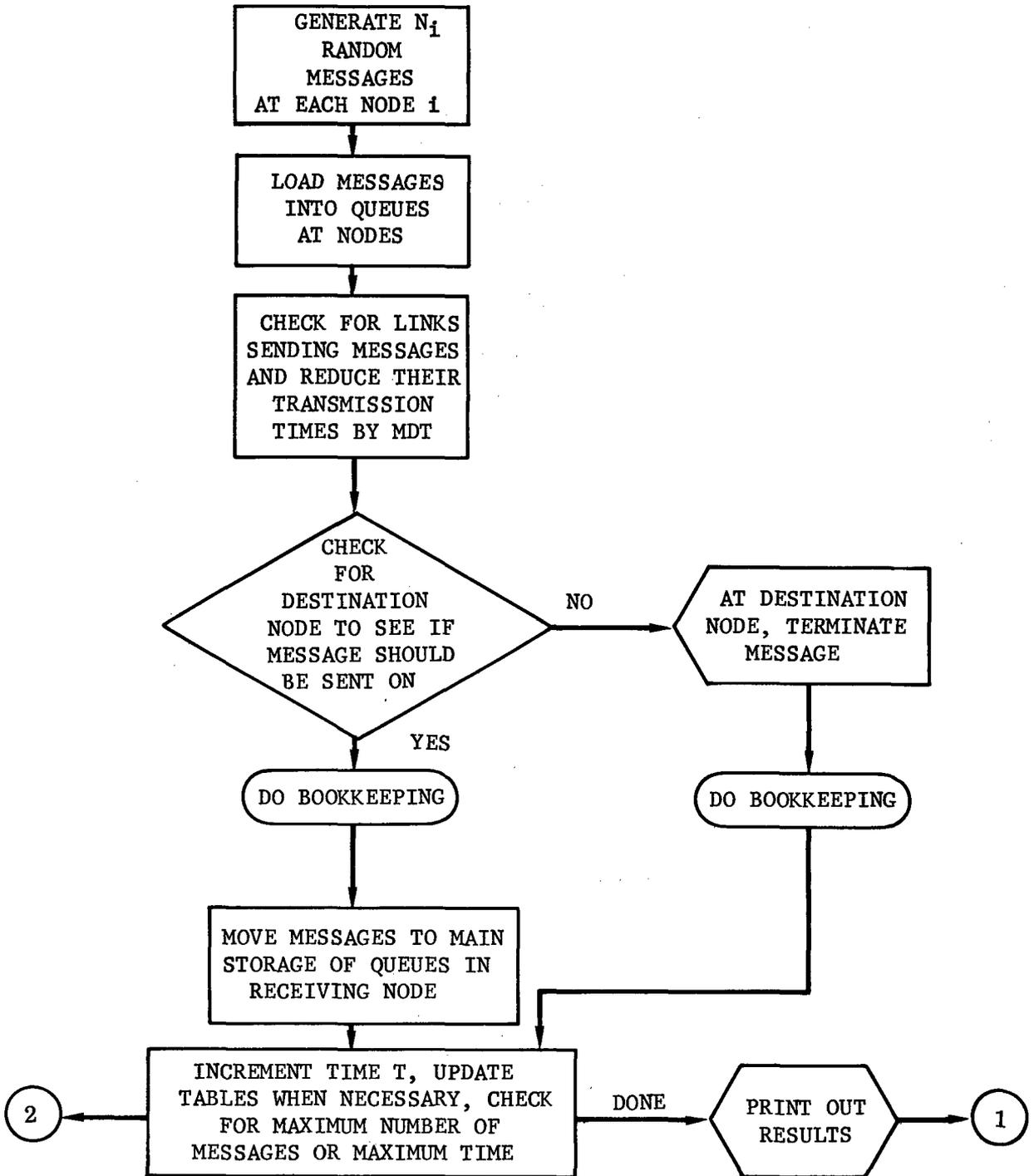


Fig. B1b - Flow Diagram of Program Routine

B.2 PROGRAM ARPSIM

```

COMMON IWORK(19,19),NL(19),LN(19,6)
COMMON LDT(19,6),LID(19,6),IROUTE(4),NREM(19)
COMMON IWRITE(21,30),IRTI(19,2,19)
COMMON IWARK(19,19),IQ(19,50),NLREM(19)
COMMON MESGI(4),NM(19),IPATH(19,50),NQ(19,6)
COMMON/BLK1/AA(19,10),A(19)
NMAX=19
NLMAX=6
MAXMSG=50
REWIND 2
CALL RANFGET(S)
2 CONTINUE
READ (60,1902) (A(I),I=1, NMAX)
CALL RSTART
CALL RANFSET(S)
READ(60,1901) MV,MTF,MDTMAX,MDT,KPLF,KPCL,MIND,NDELTA,INDOCT,INDPRM,
1 IMIX,MMAX
IF (MDT.EQ.0) GO TO 170
READ(60,1901) NTRY,LOOP,MTG,MESPER,IDP
MESPER=MESPER*MDTMAX/100
READ (60,1901) NMM
WRITE(61,1950) LINC,INDOCT
WRITE(61,1962) MV,MTF,MDTMAX,MDT,KPLF,KPCL,MIND,NDELTA,IMIX
WRITE (61,1902) (A(I), I=1, NMAX)
3 IF (UNIT,2) 3,150
150 READ (2) LINC,NL,LN,LDT,LID,IWARK
4 IF (UNIT,2) 4,160
160 REWIND 2
NLF=0
NCL=0
DO 430 I=1,NMAX
IF (NL(I).EQ.0) GO TO 430
NLF=NLF+1
DO 420 J=1,NLMAX
IF (LID(I,J).EQ.1) NCL=NCL+1
420 CONTINUE
430 CONTINUE
NCL=NCL/2
R=-ANF(-1)
ARN=R
AARN=ARN
MLF=(NLF*(100-KPLF)+50)/100
MCL=(NCL*(100-KPCL)+50)/100
NPOS=0
NPOSA=0
MRATE=0
NDELTA=NDELTA
MREC=0
MIXR=0
MX=0
MY=0
MZ=0
NZ=0
DO 505 I=1, NMAX
NM(I)=0
DO 506 J=1, NLMAX
506 NQ(I,J)=0
DO 505 J=1, MAXMSG

```

```

      IQ(I,J)=0
505  IPATH(I,J)=0
      KLF=NLf-MLF
      IF (KLF.LE.0) GO TO 530
      DO 520 I=1, KLF
      N=NLf+1-1
      R=RANF(-1)
      KILL=MINO(N,INT(R*FLOAT(N)+1.))
      N=0
      DO 510 J=1, NMAX
      IF (NL(J).LE.0) GO TO 510
      N=N+1
      IF (N.LT.KILL) GO TO 510
      NL(J)=-NL(J)
      GO TO 520
510  CONTINUE
520  CONTINUE
530  DO 580 I=1, NMAX
      IF (NL(I).GE.0) GO TO 580
      N=-NL(I)
      DO 570 J=1, N
      L=LN(I,J)/100000
      K=MOD(LN(I,J),100)
      LID(I,J)=-LID(I,J)
570  LID(L,K)=-LID(L,K)
580  CONTINUE
      KCL=NCL-MCL
      IF (KCL.LE.0) GO TO 711
      DO 710 I=1, KCL
      KCLI=2*(NCL-I+1)
      R=RANF(-1)
      KILL=MINO(KCLI,INT(R*FLOAT(KCLI)+1.))
      DO 690 J=1, NMAX
      NLN=IABS(NL(J))
      IF (NLN.EQ.0) GO TO 690
      DO 680 K=1, NLN
      IF (LID(J,K).NE.1) GO TO 680
      KILL=KILL-1
      IF (KILL.GT.0) GO TO 680
      LID(J,K)=-LID(J,K)
      L=LN(J,K)/100000
      N=MOD(LN(J,K),100)
      LID(L,N)=-LID(L,N)
      GO TO 695
680  CONTINUE
690  CONTINUE
695  CONTINUE
710  CONTINUE
711  DO 720 I=1, NMAX
      NLREM(I)=0
      IF (NL(I).LE.0) GO TO 720
      DO 718 J=1,NLMAX
      IF (LID(I,J).EQ.1) NLREM(I)=NLREM(I)+1
718  CONTINUE
      IF (NLREM(I).EQ.0) NL(I)=-NL(I)
720  CONTINUE
      MCL=0
      MLF=0
      DO 725 I=1,NMAX

```

```

IF (NL(I).LE.0) GO TO 725
MLF=MLF+1
DO 724 J=1,NLMAX
IF (LID(I,J).EQ.1) MCL=MCL+1
724 CONTINUE
725 CONTINUE
DO 729 I=1,NMAX
IF (NL(I).LE.0) GO TO 729
DO 727 J=1,NMAX
IRTI(J,1,I)=0
IRTI(J,2,I)=0
IF (J.EQ.1.OR.NL(J).LE.0) GO TO 727
DO 726 K=1,NLMAX
IR=(K+2)/3
L=LN(I,K)/100000
IF (L.EQ.0) GO TO 9000
IRTI(J,IR,I)=MINO(IWARK(J,L)+LDT(I,K),999)+1000*IRII(J,IR,I)
GO TO 726
9000 IRTI(J,IR,I)=999+1000*IRTI(J,IR,I)
726 CONTINUE
727 CONTINUE
729 CONTINUE
CALL IDEAL
WRITE(61,1800)
DO 7115 I=1,NMAX
IF (NL(I).LE.0) GO TO 7115
K=0
DO 7110 J=1,NLMAX
IF (LID(I,J).NE.1) GO TO 7110
K=K+1
IWRITE(K,1)=LN(I,J)/100000
7110 CONTINUE
WRITE(61,1801) I,(IWRITE(J,1), J=1, K)
7115 CONTINUE
1800 FORMAT (12H NODE LINKS)
1801 FORMAT (1H ,I3,4X,30I3)
MT=-MDTMAX
7295 DO 810 I=1, NMAX
NLN=NL(I)
DO 810 J=1, NLN
IF (IQ(I,J)/100.EQ.0) GO TO 810
IQ(I,J)=IQ(I,J)-MDT*100
810 CONTINUE
LRN=MDTMAX+MT
DO 880 II=1, NMAX
NLN=NL(II)
IF (NLN.LE.0) GO TO 880
DO 870 JRN=1, NLN
J=MOD(LRN+JRN,NLN)+1
IF (IQ(II,J).EQ.0) GO TO 870
IF (IQ(II,J)/100.NE.0) GO TO 870
IQ(II,J)=0
KK=LN(II,J)/100000
JJ=MAXMSG+MOD(LN(II,J),100)-NL(KK)
L=MOD(IQ(KK,JJ)/1000000000,100)+1
MESGI(3)=MOD(IQ(KK,JJ)/10000,1000)
I=KK
IF (L.EQ.1) GO TO 828
IF (MESGI(3).LT.MDTMAX) GO TO 850

```

```

826 IF (MT.LE.0) GO TO 827
    MY=MY+1
    GO TO 829
828 IF (MT.LE.0) GO TO 827
    MREC=MREC+1
    MIXR=MIXR+1
    MRATE=MRATE+1
    MZ=MZ+MESGI(3)
829 MX=MX+1
    IF (IQ(KK,JJ)/1000000000000.EQ.2) NPOS=NPOS+1
827 CONTINUE
    NQ(II,J)=NQ(II,J)-1
    IQ(KK,JJ)=0
    IF (MT.LE.0) GO TO 840
    IF (MOD(MX,IMIX).NE.0) GO TO 840
830 NPOSA=NPOSA+NPOS
    WRITE(61,1961) MX,MREC,NPOSA,IMIX,MIXR,NPOS,MI
    NPOS=0
    MIXR=0
    IF (MX.GE.MMAX) GO TO 930
840 CONTINUE
    GO TO 870
850 NLN=NL(I)
    DO 754 III=1, NTRY
        NN=32768
        KOUT=0
        DO 753 K=1, NLN
            IR=(K+2)/3
            IRX=2-MOD(K-1,3)
            AARN=AMOD(AARN+.3,1.)
            IROUTE(1)=MOD(IRT(I,L,IR,I)/1000**IRX,1000)+NQ(I,K)
            IF (NN.LT.IROUTE(1)) GO TO 753
            IF (LID(I,K).NE.1) GO TO 753
            IF (III.EQ.1) GO TO 755
            IF (III.GE.2.AND.LN(I,K)/100000.EQ.KL) GO TO 753
            IF (III.GE.3.AND.LN(I,K)/100000.EQ.KLL) GO TO 753
755 CONTINUE
            IF (NN.EQ.IROUTE(1).AND.AARN.LI..5) GO TO 753
            NN=IROUTE(1)
            KOUT=K
753 CONTINUE
            IF (KOUT.EQ.0) GO TO 826
            IF (LOOP.EQ.0) GO TO 766
            DO 765 IK=1, LOOP
                KLL=LN(I,KOUT)/100000
                IF (KL.EQ.KLL) GO TO 765
                IF (KLL.EQ.MOD(IPATH(I,JJ)/101**((IK-1),101)) GO TO 756
765 CONTINUE
                GO TO 766
756 CONTINUE
                IF (III.EQ.1) KL=KLL
                IF (III.EQ.NTRY) GO TO 826
754 CONTINUE
766 DO 873 K=1, NMM
            KL=K+NLN
            IF (IQ(KK,KL).NE.0) GO TO 873
            IQ(KK,KL)=IQ(KK,JJ)/100*100+KOUT
            IQ(KK,JJ)=0
            NQ(KK,KOUT)=NQ(KK,KOUT)+1

```

```

      NQ(II,J)=NQ(II,J)-1
      IPATH(KK,KL)=IPATH(KK,JJ)
      GO TO 870
873  CONTINUE
      IQ(II,J)=100*MDT+1
      MESGI(3)=MIN0(MOD(IQ(KK,JJ)/10000,1000)+MDI,999)
      IQ(KK,JJ)=IQ(KK,JJ)+10000*(MESGI(3)-MOD(IQ(KK,JJ)/10000,1000))
870  CONTINUE
880  CONTINUE
      DO 750 JJ=1, NMAX
      IF (NL(JJ).LE.0) GO TO 750
      R=RANF(-1)
      CALL POIS(R,JJ,NMSG)
      ARN=R
      AARN=R
      IF (NMSG.EQ.0) GO TO 750
      MESGI(2)=JJ
      DO 741 I=1, NMSG
      R=RANF(-1)
      N=MLF-1
      IF (MIND.NE.0) NLF=NLF-1
      N=MIN0(N,INT(R*FLOAT(N)+1.))
      DO 745 J=1, NMAX
      IF (J.EQ.MESGI(2)) GO TO 745
      IF (NL(J).GT.0) N=N-1
      IF (NL(J).LT.0.AND.MIND.NE.0) N=N-1
      IF (N.LE.0) GO TO 747
745  CONTINUE
747  MESGI(1)=J
      N=MESGI(2)
748  NLN=NL(N)
      DO 751 II=1,NMM
      M=NLN+II
      IF (IQ(N,M).NE.0) GO TO 751
      IPATH(N,M)=MESGI(2)
      IQ(N,M)=1000000000*J+10000101*MESGI(2)-1010000101
      IF (IWORK(N,J).NE.32768) IQ(N,M)=IQ(N,M)+200000000000
      NLN=NL(N)
      NN=32768
      DO 752 K=1, NLN
      IR=(K+2)/3
      IRX=2-MOD(K-1,3)
      AARN=AMOD(AARN+.3,1.)
      IROUTE(1)=MOD(IRTI(J,IR,N)/1000**IRX,1000)+NQ(N,K)
      IF (NN.LT.IROUTE(1)) GO TO 752
      IF (LID(N,K).NE.1) GO TO 752
      IF (NN.EQ.IROUTE(1).AND.AARN.LI..5) GO TO 752
      NN=IROUTE(1)
      KOUT=K
752  CONTINUE
      IQ(N,M)=IQ(N,M)/100*100+KOUT
      NQ(N,KOUT)=NQ(N,KOUT)+1
      GO TO 741
751  CONTINUE
      IF (MT.GT.0) NZ=NZ+1+NMSG-I
      GO TO 750
741  CONTINUE
750  CONTINUE
      AARN=AMOD(AARN+R,1.)

```

```

DO 881 I=1, NMAX
  IF (NLREM(I).LE.0) GO TO 881
  NLN=NL(I)
  DO 882 K=1, NLN
    IF (IQ(I,K).NE.0) GO TO 882
  DO 883 J=1, NMM
    JJ=J+NLN
    IF (MOD(IQ(I,JJ),100).EQ.K) GO TO 884
883 CONTINUE
    GO TO 882
884 IF (MTC.EQ.0) GO TO 892
    NN=NLN+NMM
    DO 891 KL=JJ,NN
      IF (MOD(IQ(I,KL),100).NE.K) GO TO 891
      IF (MOD(IQ(I,KL)/10000,1000).LT.MESPER) GO TO 891
      JJ=KL
      GO TO 892
891 CONTINUE
892 KOUT=K
    IQ(I,K)=100*LDT(I,K)+1
    MM=LN(I,K)/100000
    LL=MOD(LN(I,K),100)
    NN=MAXMSG-NL(MM)+LL
    MESGI(3)=MINO(MOD(IQ(I,JJ)/10000,1000)+LDI(MM,LL),999)
    IPATH(MM,NN)=101*MOD(IPATH(I,JJ),104060401)+I
    IQ(MM,NN)=IQ(I,JJ)/10000000*10000000+100*I+MM-101+10000*MESGI(3)
    IQ(I,JJ)=0
865 CONTINUE
882 CONTINUE
886 K=NLN
    NN=NMM+NLN
    JJ=NLN+1
    DO 887 J=JJ,NN
      IF (IQ(I,J).EQ.0) GO TO 887
      MESGI(3)=MINO(MOD(IQ(I,J)/10000,1000)+MDT,999)
888 K=K+1
      IPATH(I,K)=IPATH(I,J)
      IQ(I,K)=IQ(I,J)/10000000*10000000+MESGI(3)*10000+MOD(IQ(I,J),10000
1)
      IF (J.EQ.K) GO TO 887
      IQ(I,J)=0
887 CONTINUE
881 CONTINUE
    IF (MT.LT.0) GO TO 925
    IF (NDELTI.GT.0) GO TO 922
    NDELTI=NDELTI
921 WRITE (61,1955) MT,MRATE
    MRATE=0
922 CONTINUE
    NDELTI=NDELTI-MDT
925 CONTINUE
    IF (MT.GE.MTF) GO TO 930
    MT=MT+MDT
    IF (MOD(MT,MDTMAX/4).NE.0) GO TO 7295
    IF (NDOCT.EQ.0) GO TO 7295
    DO 113 I=1, NMAX
      DO 112 J=1, NMAX
        IWARK(I,J)=32768
      DO 112 K=1, NLMAX

```

```

IF (LID(I,K).NE.1) GO TO 112
IR=(K+2)/3
IRX=2-MOD(K-1,3)
IROUTE(1)=MOD(IRTI(J,IR,I)/1000**IRX,1000)+NQ(I,K)
IF (IROUTE(1).LT.IWARK(I,J)) IWARK(I,J)=IROUTE(1)
112 CONTINUE
113 IWARK(I,I)=0
DO 124 I=1, NMAX
DO 124 J=1, NMAX
IRTI(J,1,I)=0
IRTI(J,2,I)=0
DO 124 K=1, NLMAX
IR=(K+2)/3
IF (LID(I,K).NE.1) GO TO 125
L=LN(I,K)/100000
IRTI(J,IR,I)=1000*IRTI(J,IR,I)+MINO(LDT(I,K)+IWARK(L,J)+IDP,999)
GO TO 124
125 IRTI(J,IR,I)=1000*IRTI(J,IR,I)+999
124 CONTINUE
GO TO 7295
930 CONTINUE
AMZ=FLOAT(MZ)/FLOAT(MREC)
950 M=MOD(MX,IMIX)
NPOSA=NPOSA+NPOS
WRITE(61,1961) MX,MREC,NPOSA,M,MIXR,NPOS,MT
WRITE(61,1952) MX,MY,MREC,AMZ
WRITE(61,1953) NZ
IF (INDPRM.EQ.0) GO TO 995
NROW=NMAX+2
DO 955 I=1, NROW
DO 955 J=1,30
955 IWRITE(I,J)=0
N=0
DO 990 I=1,NMAX
IF (NLREM(I).LE.0) GO TO 972
N=N+2
IWRITE(1,N-1)=I
M=2
DO 957 J=1,NMAX
IF (NLREM(J).LE.0) GO TO 957
M=M+1
IWRITE(M,N)=J
957 CONTINUE
DO 970 J=1,NLMAX
IF (LID(I,J).NE.1) GO TO 970
N=N+1
KK=2
IWRITE(2,N)=LN(I,J)/100000
IR=(J+2)/3
IRX=2-MOD(J-1,3)
DO 960 K=1, NMAX
IF (NLREM(K).LE.0) GO TO 960
KK=KK+1
IWRITE(KK,N)=MOD(IRTI(K,IR,I)/1000**IRX,1000)
960 CONTINUE
970 CONTINUE
972 IF (I.LT.NMAX.AND.N+NLREM(I+1)+2.LE.30) GO TO 990
WRITE(61,1960) ((IWRITE(J,K),K=1, 30),J=1, KK)
N=0

```

```

      DO 980 J=1, NROW
      DO 980 K=1,30
980  IWRITE(J,K)=0
990  CONTINUE
995  CONTINUE
      GO TO 2
170  STOP
1901 FORMAT (12I6)
1902 FORMAT(12F6.3)
1802 FORMAT (10I12)
1950 FORMAT (1H1,10X,18HLINK CONFIGURATION,18,5X,8HDOCTRINE,18)
1952 FORMAT (17HOTOTAL MESSAGES =18/17HOUNDELIVERED      =18/17HODELIVERED
1D      =18/17HOAVERAGE TIME      =F8.2)
1953 FORMAT (1X,I6,30H MESSAGES FOUND THE QUEUE FULL )
1955 FORMAT (3HOT=,I6,5X,12HMESSAGES/DT=,I6/)
1960 FORMAT (1H0,30X,14HROUTING TABLES/(30I4))
1961 FORMAT (21HOTOTAL MSGS EXPIRED =15,5X,16HTOTAL MSGS DLV =15,5X,10H
1POSSIBLE =15/21H OF LAST      15,5X,16H      MSGS DLV =15,5
2X,10HPOSSIBLE =15/4H T =15)
1962 FORMAT (7HOMV      I4,5X,6HMTF      I4,5X,6HMDTMAXI4,5X,6HMDT      I4,5X,6
1HKPLF I4,5X,6HKPCL I4/7H MIND I4,5X,6HNDELT I4,5X,6HIMIX I4)
      END

```

```

SUBROUTINE RSTART
COMMON/BLK1/AA(19,10),A(19)
NMAX=19
DO 200 I=1, NMAX
AA(I,1)=EXP(-A(I))
T=AA(I,1)
DO 200 J=2,10
TJ=J-1
T=T*A(I)/TJ
200 AA(I,J)=AA(I,J-1)+T
      RETURN
      END

```

```

SUBROUTINE POIS (R,I,NMSG)
COMMON/BLK1/AA(19,10),A(19)
DO 200 J=1,10
IF (R.LT.AA(I,J)) GO TO 100
200 CONTINUE
NMSG=10
      RETURN
100 NMSG=J-1
      RETURN
      END

```

```

SUBROUTINE IDEAL
COMMON IWORK(19,19),NL(19),LN(19,6)
COMMON LDT(19,6),LID(19,6),IROUTE(4),NREM(19)
DIMENSION NC(19)
NMAX=19
DO 500 I=1,NMAX

```

```
DO 20 J=1,NMAX
IWORK(I,J)=32768
NC(J)=0
20 CONTINUE
IF (NL(I).LE.0) GO TO 500
NODE=I
IDT=0
50 CONTINUE
N=NL(NODE)
DO 100 J=1,N
IF (LID(NODE,J).NE.1) GO TO 100
L=LN(NODE,J)/100000
IWORK(I,L)=MIN0(IWORK(I,L),LDT(NODE,J)+IDT)
100 CONTINUE
NC(NODE)=1
IDT=32768
DO 200 J=1, NMAX
IF ((NC(J).NE.0).OR.(IWORK(I,J).EQ.32768))GO TO 200
IF (IWORK(I,J).GT.IDT) GO TO 200
NODE=J
IDT=IWORK(I,J)
200 CONTINUE
IF (IDT.NE.32768) GO TO 50
IWORK(I,I)=0
500 CONTINUE
RETURN
END
```

B.3 EXPLANATION OF VARIABLES

Suppose we are given a communication network η with nodes $\{i\}$, links $\{l_{ik}\}$ where l_{ik} is the k^{th} link out of node i and t_{ik} is the time to transmit along the k^{th} link out of node i . Each node is assumed to have a storage area for messages and the assumption is made that if there is a link of length t_{ij} from node i to node j , then there is also a similar link from node j to node i .

The program ARPSIM was designed to run a simulation of the above communication network, with and without damage under several methods of routing procedures. To aid in the understanding of this program, the following list of variables is given.

<i>NMAX</i>	– maximum number of nodes in the undamaged network
<i>NLMAX</i>	– maximum number of links which can come out of any node
<i>IWORK(I,J)</i>	– minimum path length from node I to node J
<i>NL(I)</i>	– number of links out of node I in undamaged network
<i>LN(I, K)</i>	= 100000 * J + L where J is the end node of the K^{th} link out of node I and the backward link is the L^{th} link out of node J
<i>LDT(I, K)</i>	– the length of the K^{th} link out of node I
<i>NREM(I)</i>	– number of links out of node I in the undamaged network
<i>NLREM(I)</i>	– number of links out of node I in the damaged network, in particular if $NLREM(I) \leq 0$, node I is isolated
<i>IWRITE(I, J)</i>	– is used for outputting data only
<i>NMM</i>	– number of storage locations in each queue available for messages waiting to be transmitted
<i>MAXMSG</i>	– $NMM + 2 * NLMAX$
<i>MV</i>	– percent of network loading. The number of messages generated in each time interval is equal to $MV * (\text{number of links in network}) / 100$ (used only for constant input)
<i>MZ</i>	– number of delivered messages
<i>NZ</i>	– number of generated messages finding the storage queue full
<i>AMZ</i>	– number of messages it was possible to deliver
<i>MT</i>	– accumulated simulation time
<i>IRTI(J, IR, I)</i>	– contains the routing tables from node I to node J , in particular the time to go from node I to node J using the K^{th} link out of node I is given by $(IRTI(J, IR, I)) / 100 \cdot \cdot IRX$, modulo 100 where $IRX = 2 - K \text{ mod } 3$ and $IR = (K + 2) / 3$

- $IQ(I, 50)$ – the storage queue for node I . $IQ(I, 50)$ contains three parts
- (Part I) for $K = 1, \dots, NLMAX$, $IQ(I, K) = 100 * J + 1$ indicates the status of the K^{th} link out of node I . If $IQ(I, K) \neq 0$ the link is busy. J is the remaining time required to complete transmission of the message.
- (Part II) for $K = NLMAX + 1, NLMAX + NMM$, $IQ(I, K)$ is the storage area for messages waiting to be transmitted; if $IQ(I, K) = 0$ this space is empty.
- (Part III) for $K = 50 - NLMAX + 1, \dots, 50$, $IQ(I, K)$ contains the message being transmitted from the node $L = LN(I, K) / 100000$ along link number $LN(I, K) \bmod 100$ to node I . If $IQ(I, K) = 0$, there is no message being transmitted along this link.
- $IPATH(I, K)$ – contains the past history of the message presently at $IQ(I, K)$. $IQ(I, K) / 11^{**} IRX$ modulo 10 is the $(IRX + 1)$ previous node visited.
- $A(I)$ – the average number of messages arriving at node I during a time interval

The actual message M being transmitted is the following:

- $M \bmod 10^2 + 1$ – is the present node of the message M if the message is in Part III of the queue or has not been placed in the queue
- $M \bmod 10^2 + 1$ – contains the number link over which the message will be transmitted if the message is on Part II of the queue
- $M / 10^2 \bmod 10^2 + 1$ – is the last node visited by the message
- $M / 10^3 \bmod 10^3$ – is the time the message has been in the network
- $M / 10^7 \bmod 10^2 + 1$ – is the source node of the message
- $M / 10^9 \bmod 10^2 + 1$ – is the destination node of the message
- $M / 10^{11} \bmod 10$ $\left\{ \begin{array}{l} \text{– if this is even, message is deliverable; if odd, undeliverable} \\ \text{– if this is greater than 1 the next node, the message is going to is the} \\ \text{same as the last node visited} \end{array} \right.$
- MTF – maximum simulation time
- $MDTMAX$ – warm-up time for the system; maximum time a message can remain in the network before being considered undeliverable
- MDT – length of simulation time unit; must divide $LDT(I, K)$ evenly for all nodes I and links K
- $KPLF$ – percent of nodes to be disabled
- $KPCL$ – percent of links to be disabled

- MIND* – an indicator to determine whether or not generated messages can have disabled nodes as destinations. If *MIND* = 0, we can use inoperable nodes as destinations, if *MIND* = 1, we cannot.
- NDEL* – time increment between summary prints
- NDOCT* – indicates the method of routing table updating to be used. If *NDOCT*
- | <u><i>RANSIM</i></u> | <u><i>ARPSIM</i></u> |
|----------------------------------|------------------------------|
| = 1 no updating | = 0 no updating |
| = 2 backwards learning | ≠ 0 update every MDT_{MAX} |
| = 3 negative reinforcement | |
| = 4 superposition of (2) and (3) | |
| = 5 bi-adaptive | |
| = 6 dynamic programming | |
- INDPRM* – indicator for printing routing matrices. If *INDPRM* = 1 we print matrices; if *INDPRM* = 0 we do not.
- MMAX* – maximum number of messages to be delivered before simulation is stopped
- LOOP* – number of previous nodes that a message will have remembered visiting
- NTRY* – number of links out of a node that a message will try in order to find a node different from a previous node; if the search is unsuccessful the message is considered undeliverable
- MTC* – indicator as to whether or not old messages should be given a priority; if *MTC* = 0 no priority is given, if *MTC* = 1 a priority is given
- MESPER* – on input percent of MDT_{MAX} which determines when a message is old; later it is changed to the age when a message is considered old
- MX* – total number of messages handled by network
- MY* – number of undelivered messages

SUBROUTINES

- RSTART* – initializes random number generator
- RANDOM* – generates a random number in [0., .999]
- IDEAL* – calculates for a given communication network the minimum path lengths between nodes

B4.1 SIMULATION RUNS FOR THE DDP-24 (No Destruction of Networks)

The following list comprises the various routing algorithms used on the DDP-24 computer. All runs were made on an 8-node network with 40 links fully duplexed and no network damage. Runs were made for identical net loadings under the following constraints:

1. Messages were generated from a uniform distribution
2. Single destination for each message; all destinations equally likely
3. Finite storage capacity at each node
4. Defections from the system were allowed as network loading increased
5. All messages were of equal length
6. Servicing was on a first-come-first-served basis (FCFS)
7. Service distribution was a constant
8. Perfect transmission and reception assumed

The program algorithms are defined along with their abbreviations as follows:

- a. Dynamic programming (DY)
- b. Basic program, no updating (NU)
- c. Backward learning (BL)
- d. Negative reinforcement (NR)
- e. Superposition of BL and NR, where BL can increase or decrease table entries (SP)
- f. Last M nodes visited (LMNV)
- g. Best three (output) links (BTL)
- h. The combination of LMNV and BTL will be further defined as (MOD_M) where M = number of past nodes visited
- i. Aging parameter ($X\%$ aging on queue)

Table B1 lists the computer runs for the 8-node network configuration (see Fig. 4.1).

TABLE B1
COMPUTER RUNS ON AN 8-NODE 40 LINK NETWORK WITH NO DESTRUCTION

1. <i>NU</i>	12. <i>NU, LNV, BTL, 10% aging</i>
2. <i>BL</i>	13. <i>NU, LNV, BTL, 50% aging</i>
3. <i>NR</i>	14. <i>NU, LNV, BTL, 75% aging</i>
4. <i>SP</i>	15. <i>BL, LNV, BTL, 10% aging</i>
5. <i>BA</i>	16. <i>BL, LNV, BTL, 50% aging</i>
6. <i>NU, LNV, BTL</i>	17. <i>BL, LNV, BTL, 75% aging</i>
7. <i>NU, L3NV, BTL</i>	18. <i>BL, L3NV, BTL, 10% aging</i>
8. <i>NU, L5NV, BTL</i>	19. <i>BL, L3NV, BTL, 50% aging</i>
9. <i>BL, LNV, BTL</i>	20. <i>BL, L3NV, BTL, 75% aging</i>
10. <i>BL, L3NV, BTL</i>	21. <i>ARPA, DP = 0</i>
11. <i>BL, L5NV, BTL</i>	22. <i>ARPA, DP = 0, LNV, BTL, 50%</i>

B4.2 SIMULATION RUNS FOR THE DDP-24 (With 50% Destruction on Links, No Node Damage)

The algorithms listed in Table B2 were run on the 8 network of section B1 with 50% link destruction (see Fig. 4.2). The definitions of the algorithms and constraints were identical with section B4.1.

TABLE B2
COMPUTER RUNS ON AN 8-NODE NETWORK WITH 50% LINK DESTRUCTION

1. <i>DY</i>	16. <i>NU, LNV, BTL, 10% (aging on que)</i>
2. <i>NU</i>	17. <i>NU, LNV, BTL, 50%</i>
3. <i>BL</i>	18. <i>NU, LNV, BTL, 75%</i>
4. <i>NR</i>	19. <i>BL, LNV, BTL, 10%</i>
5. <i>SP</i>	20. <i>BL, LNV, BTL, 50%</i>
6. <i>BA</i>	21. <i>BL, LNV, BTL, 75%</i>
7. <i>NU, LNV, BTL</i>	22. <i>BA, BTL, 10%</i>
8. <i>NU, L3NV, BTL</i>	23. <i>BA, BTL, 50%</i>
9. <i>NU, L5NV, BTL</i>	24. <i>BA, BTL, 75%</i>
10. <i>BL, LNV, BTL</i>	25. <i>ARPA, NU</i>
11. <i>BL, L3NV, BTL</i>	26. <i>ARPA (standard with updating)</i>
12. <i>BL, L5NV, BTL</i>	27. <i>ARPA, LNV, BTL, 50%</i>
13. <i>BA, LNV, BTL</i>	28. <i>ARPA, DP* = 1</i>
14. <i>BA, L3NV, BTL</i>	29. <i>ARPA, DP = 1, LNV, BTL, 50%</i>
15. <i>BA, L5NV, BTL</i>	

* *DP* = a bias term that is added to the routing delays to prevent looping

APPENDIX C
SIMULATION ON THE CDC 3800

C.1 SIMULATION RUNS FOR THE CDC 3800 (No Destruction of Network)

All runs were made for identical network message intensities on a 19 Node ARPA Network. The algorithm abbreviations and program constraints are the same as given in Appendix B.

Runs C2.1 and C2.2 were made with messages generated from a uniform distribution. Runs C2.3 and C2.4 were made with Poisson input distributions, such that the average message rate per node was λ_i . The total message rate into the network per time was $19 \lambda_i$ (when all 19 nodes were operable).

The periodic updating (PUD) was employed on some of the runs to make the ARPA routine conform more closely to its present configuration and a bias term of DP=2 was used.

After some preliminary runs were made to see the extent of link or node destruction that could be used without disconnecting the network, these percentages were used for setting up Tables C2.3 and C2.4. The message intensities were also adjusted to prevent net saturation for any given set of runs.

TABLE C2.1
COMPUTER RUNS ON THE 19 NODE ARPA NET WITH UNIFORM
MESSAGE INPUT DISTRIBUTION

1. NU
2. BL
3. NU, 50% aging on queue
4. NU, LNV, BTL, 50% aging
5. ARPA, NU
6. ARPA, DP=3
7. ARPA, DP=0, 50% aging
8. ARPA, DP=3, 50% aging
9. ARPR, LNV, DP=3, 50% aging

TABLE C2.2
COMPUTER RUNS ON THE 19 NODE ARPA NET WITH POISSON
MESSAGE INPUT DISTRIBUTION

1. NU
2. BL
3. NU, 50% aging
4. NU, LNV, BTL, 50% aging
5. ARPA, NU, DP=0
6. ARPA, DP=3
7. ARPA, DP=0, 50% aging
8. ARPA, DP=3, 50% aging
9. ARPA, LNV, DP=3, 50% aging
10. ARPA, LNV, BTL, DP=3, 40% aging
11. ARPA, LNV, BTL, DP=3, 60% aging
12. ARPA, PUD, LNV, BTL, DP=3, 50% aging

TABLE C2.3
COMPUTER RUNS ON 19 NODE ARPA NET WITH 10% LINK DESTRUCTION
(POISSON INPUT DISTRIBUTION)

1. DY
2. BA
3. BA, LNV, BTL, 50% aging
4. DY, LNV, BTL, 50% aging
5. ARPA
6. ARPA, 50% aging
7. ARPA, LNV, BTL, 50% aging

TABLE C2.4
COMPUTER RUNS ON 19 NODE ARPA NET WITH 1 NODE DISABLED
(POISSON INPUT DISTRIBUTION)

1. ARPA, DP=2, PUD
2. ARPA, DP=2, PUD, 50% aging
3. ARPA, DP=2, PUD, LNV, BTL, 50% aging

REFERENCES

- BA 64A Baran, P., "On Distributed Communications: Introduction to Distributed Communication Networks," Rand Corporation, Memorandum, RM-3420-PR, August 1964.
- BA 64B Baran, P., "On Distributed Communications: Priority, Precedence, and Overload," Rand Corporation, Memorandum, RM-3638-PR, August 1964.
- BA 64C Baran, P., "On Distributed Communications: History, Alternative Approaches and Comparisons," Rand Corporation, Memorandum, RM-3097-PR, August 1964.
- BA 64D Baran, P., "On Distributed Communication: Mini-Cost Microwave," Rand Corporation, Memorandum, RM-3762-PR, August 1964.
- BA 64E Baran, P., "Tentative Engineering Specifications and Preliminary Design for a High Data Rate Distributed Network Switching Node," Rand Corporation, Memorandum, RM-3763-PR, August 1964.
- BA 64F Baran, P., "On Distributed Communications: The Multiplexing Station," Rand Corporation, Memorandum, RM-3764-PR, August 1964.
- BA 64G Baran, P., "On Distributed Communications: Security, Secrecy, and Tamper-Free Considerations," Rand Corporation, Memorandum, RM-3765-PR, August 1964.
- BA 64H Baran, P., "On Distributed Communications: Cost Analysis," Rand Corporation, Memorandum, RM-3766-PR, August 1964.
- BA 64I Baran, P., "On Distributed Communications: Summary Overview," Rand Corporation, Memorandum, RM-3767-PR, August 1964.
- BAR 57 Barrer, D. Y., "Queueing With Impatient Customers and Ordered Service," *Operations Research*, Vol. 5, pp. 650-656 (1957).
- BEL 62 Bellman, R. E., and Dreyfus, S. E., "Applied Dynamic Programming," Rand Corporation, Memorandum, R-352-PR, May 1962.

- BO 64 Boehm, S., and Baran, P., "Digital Simulation of Hot Potato Routing in a Broadband Distributed Communication Network," Rand Corporation, Memorandum, RM-3103-PR, August 1964.
- BO 66 Boehm, B. W., and Mobley, R. L., "Adaptive Routing Techniques for Distributed Communications Systems," Rand Corporation, Memorandum, RM-4781-PR, February 1966.
- BO 69 Boehm, B. W., and Mobley, R. L., "Adaptive Routing Techniques for Distributed Communications Systems," *IEEE Trans. on Communication Technology*, Vol. Com-17, No. 3, pp. 340-349 (1969).
- BRO 48 Brockmeyer, E., Halstrom, H. L., and Jensen, A., *The Life and Works of A. K. Erlang*, Danish Acad. Tech. Sci., No. 2 (1948).
- BU 56 Burke, P. J., "The Output of a Queueing System," *Operations Research*, Vol. 4, pp. 699-704 (1956).
- COB 54 Cobham, A., "Priority Assignments in Waiting Line Problems," *Operations Research*, Vol. 2, pp. 70-76 (1954); "A Correction," *Operations Research*, Vol. 3, pp. 547 (1955).
- CO 71 Cole, G. D., "Computer Network Measurements: Techniques and Experiments," Ph.D. Thesis, Dept. of Computer Science, University of California, L. A., (1971).
- CON 67 Conway, R. W., Maxwell, W. L., and Miller, L. W., *Theory of Scheduling*, Addison-Wesley, Reading, Mass. (1967).
- COX 61 Cox, D. R., and Smith, W. L., *Queues*, Methuen and Company, Ltd. (1961).
- DA 66 Davis, R. H., "Waiting-Time Distribution of a Multi-Server, Priority Queueing System," *Operations Research*, Vol. 14, pp. 133-136 (1966).
- DA 71 Davies, D'W., "The Control of Congestion in Packet Switching Networks." Proceedings of the Second ACM IEEE Symposium in the Optimization of Data Communications Systems, Palo Alto, Calif., October 1971.

- EV 57 Everett, R. R., Zraket, C. A., and Bannington, H. D., "SAGE: A Data Processing System for Air Defense," *EJCC*, pp. 148-155 (1957).
- FE 57 Feller, W., *An Introduction to Probability and Its Applications*, John Wiley & Sons, Inc., New York (1957) (2nd Edition).
- FLO 62 Floyd, R. W., "Algorithm 97, Shortest Paths," *Communication of ACM* 5, p. 345 (1962).
- FO 62 Ford, L. R., Jr., and Fulkerson, D. R., *Flows in Networks*, Princeton Univ. Press, Princeton (1962).
- FR 71 Frank, H., "Research in Store and Forward Computer Networks," Semi-annual Report No. 4, (Contract No. DAHC15-70-C-0120), December 1971.
- FR 71 Frank, H., and Frisch, I. T., *Communication, Transmission, and Transportation Networks*, Addison-Wesley Publishing Co., (1971).
- FR 72 Frank, H., Kahn, R. E., and Kleinrock, L., "Computer-Communication Network Design -- Experience With Theory and Practice," *AFIPS Conference Proceedings, SJCC*, pp. 255-270 (1972).
- FU 71 Fultz, G. L., and Kleinrock, L., "Adaptive Routing Techniques for Store-and-Forward Computer-Communication Networks," *Proceedings of the 1971 International Conference on Communications*, June 14-16, Montreal, Canada, pp. 39-1 to 39-8.
- FU 72 Fultz, G. L., "Adaptive Routing Techniques for Message Switching Computer-Communication Networks," *Engineering Report No. UCLA-ENG-7252, Ph.D. Thesis, University of California, Los Angeles, California, July 1972.*
- HE 70 Heart, F. E., Kahn, R. E., Arnstein, S. M., Crowther, W. R., and Walden, D. C., "The Interface Message Process for ARPA Computer Network," *AFIPS Conference Proceedings*, 36: 551-567, *AJCC* (1970).
- HS 71 Hsu, J., "Analysis of a Continuum of Processor-Sharing Models for Time-Shared Computer Systems," *Ph.D. Thesis, Dept. of Computer Science, University of California, L. A. (1971).*
- JAC 60 Jackson, J. R., "Some Problems in Queueing With Dynamic Priorities," *Nav. Res. Logistics Quart.*, Vol. 7, pp. 235-249 (1960).

- JAC 62 Jackson, J. R., "Waiting-Time Distributions for Queues With Dynamic Priorities," *Nav. Res. Logistics Quart.*, Vol. 9, pp. 31-36 (1962).
- JA 68 Jaiswal, N. K., *Priority Queues*, Academic Press, New York and London (1968).
- KA 71 Kahn, R. E., Crowther, W. R., "A Study of the ARPA Computer Network Design and Performance," Bolt, Beranek and Newman, Inc., Cambridge, Mass. Report No. 2161, August 1971.
- KE 51 Kendall, D. G., "Some Problems in the Theory of Queues," *J. Roy. Statist. Soc., Series B*, 13: 151-185 (1951).
- KE 53 Kendall, D. G., "Stochastic Processes Occurring in the Theory of Queues and Their Analysis by the Method of the Imbedded Markov Chain," *Ann. Math. Stat.*, Vol. 24, pp. 338-354 (1953).
- KES 57 Kesten, H., and Runnenburg, J. T., "Priority in Waiting Line Problems," *Proc. Akad. Wet. Amst. A*, Vol. 60, pp. 312-336 (1957).
- KL 64 Kleinrock, L., *Communication Nets: Stochastic Message Flow and Delay*, McGraw-Hill, New York, N.Y. (1964).
- KR 62 Krylov, V. I., *Approximate Calculation of Integrals*, MacMillan Co., New York, London (1962).
- LI 52 Lindley, D. V., "The Theory of Queues With a Single Server," *Proceedings of the Cambridge Philosophical Society* 48, pp. 277-289 (1952).
- LI 61 Little, J. D. C., "A Proof for the Queueing Formula $L = \lambda W$," *Operations Research*, Vol. 9, pp. 383-387 (1961).
- MA 72 Martin, J., *System Analysis for Data Transmission*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey (1972).
- MIL 68 Millar, J. Z., "DCS Autodin Trunking Transmission Between Switching Centers," Invitational Workshop on Networks of Computers, Proceedings, National Security Agency, Fort Meade, Maryland, NOC-68: 221, 224, October 14-18, 1968.
- MI 60 Miller, R. G., "Priority Queues," *Ann. Math. Statist.*, 31, pp. 86-103 (1960).

- MO 22 Molina, E. C., "The Theory of Probabilities Applied to Telephone Trunking Problems," *Bell System Tech. J.*, 1(2): 69-81 (1922).
- MO 27 Molina, E. C., "Application of the Theory of Probability to Telephone Trunking Problems," *Bell System Tech. J.*, 6: 461-497 (1927).
- MO 58 Morse, P. M., *Queues, Inventories, and Maintenance*, John Wiley & Sons, Inc., New York (1958).
- NBS 73 NBS Special Publication 384, "Annotated Bibliography of the Literature on Resource Sharing Computer Networks," September 1973.
- NEE 65 Nee, D., "Application of Queueing Theory to Information System Design," (Final Report) SRI Project 5187 (1965).
- O'D 20 O'Dell, G. F., "Theoretical Principles of the Traffic Capability of Automatic Switches," *P. O. Elec. Engrs., J.*, 13: 209-223 (1920).
- O'D 27 O'Dell, G. F., "An Outline of the Trunking Aspect of Automatic Telephone," *J. Inst. Elec. Engrs.* (London), 65: 185-122 (1927).
- PA 62 Parzen, E., *Stochastic Processes*, Holden-Day, Inc., San Francisco, Calif. (1962).
- PH 56 Phipps, T. E., Jr., "Machine Repair as a Priority Waiting Line Problem," *Operations Research*, Vol. 4, pp. 76-85 (1956). (Comments by W. R. VanVoorhis, *ibid.*, p. 86.)
- PL 61 Plugge, W. R., and Perry, M. H., "American Airlines SABRE Electronic Reservations System," Proceedings WJCC, pp. 593-602, May 1961.
- PR 62 Prosser, R. J., "Routing Procedures in Communication Networks, Part I: Random Procedures," IRE Transactions on Communication Systems, CS-10, pp. 329-335 (1962).
- RI 62 Riordan, J., *Stochastic Service Systems*, John Wiley and Sons, Inc., New York, London (1962).
- RO 67 Roberts, L. G., "Multiple Computer Networks and Inter-Computer Communications," ACM Symposium on Operating Systems Principles, Gatlinburg, Tennessee, October 1967.

- SA 57 Saaty, T. L., "Resumé of Useful Formulas in Queueing Theory," *Operations Research*, Vol. 5, pp. 161-200 (1957).
- SA 61 Saaty, T. L., *Elements of Queueing Theory With Applications*, McGraw-Hill Book Company, New York (1961).
- SC 72 Schwartz, M., Boorstyn, R. R., and Pikholtz, R. L., "Terminal-Oriented Computer-Communication Networks," *Proceedings IEEE*, Vol. 60, No. 11, pp. 1408-1423, November 1972.
- SM 64 Smith, J. W., "Determination of Path Lengths in a Distributed Network," Rand Corporation, Memorandum, RM-3578-PR, August 1964.
- SY 60 Syski, R., *Introduction to Congestion Theory in Telephone Systems*, Oliver and Boyd, Edinburgh and London (1960).
- TA 55 Takács, L., "Investigation of Waiting Time Problems by Reduction to Markov Process," *Acta Math., Acad. Sci. Hung.*, Vol. 6, pp. 101-128 (1955).
- TA 63 Takács, L., "Priority Queues," *Operations Research*, Vol. 12, pp. 63-74 (1964).
- TA 62 Takács, L., "A Single-Server Queue With Poisson Input," *Operations Research*, Vol. 10, pp. 388-394 (1962).
- VER 58 Vernam, G. S., "Automatic Telegraph Switching System Plan 55A," *Western Union Tech. Rev.*, 12(2): 37-50 (1958).
- WO 73 Wolf, E. C., "An Advanced Computer Communication Network," *Proceedings AIAA*, Computer Network System Conference, Huntsville, Alabama, April 16-18, 1973.
- ZE 71 Zeigler, J. F., "Nodal Blocking in Large Networks," Ph.D. Thesis, Dept. of Computer Science University of California, Los Angeles, Calif. (1971).