UCLA-ENG-7165
October 1971

ARPA CONTRACT NO. DAHC-15-69-C-0285

COMPUTER NETWORK MEASUREMENTS: TECHNIQUES AND EXPERIMENTS

Gerald D. Cole

COMPUTER SYSTEMS
MODELING AND ANALYSIS GROUP

COMPUTER SCIENCE DEPARTMENT

School of Engineering and Applied Science
University of California
Los Angeles

COMPUTER SYSTEMS MODELING AND ANALYSIS GROUP
REPORT SERIES

1. Cole, G.D., "Computer Network Measurements: Techniques and Experiments," October 1971, UCLA-ENG-7165 (ARPA).

published by

REPORTS GROUP
SCHOOL OF ENGINEERING AND APPLIED SCIENCE
UNIVERSITY OF CALIFORNIA, LOS ANGELES 90024

UNCLASSIFIED
Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author)
University of California
School of Engineering and Applied Science
Los Angeles, California 90024

2a. REPORT SECURITY CLASSIFICATION
UNCLASSIFIED

2b. GROUP

3. REPORT TITLE
COMPUTER NETWORK MEASUREMENTS: TECHNIQUES AND EXPERIMENTS

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)
Scientific

5. AUTHOR(S) (First name, middle initial, last name)
Gerald D. Cole

6. REPORT DATE
October 1971

7a. TOTAL NO. OF PAGES
364

7b. NO. OF REFS
89

8a. CONTRACT OR GRANT NO
DAHC-15-69-C-0285

b. PROJECT NO

c.

d.

9a. ORIGINATOR'S REPORT NUMBER(S)
UCLA-ENG-7165

9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)

10. DISTRIBUTION STATEMENT

11. SUPPLEMENTARY NOTES

12. SPONSORING MILITARY ACTIVITY

13. ABSTRACT

The ARPA (Advanced Research Projects Agency) computer network involves the interconnection of about twenty (as of 1971) different research computers across the country by means of a store-and-forward message switching net. The development of such a network is an expensive and complicated undertaking which involves a variety of engineering trade-offs and decisions. Since there has been little prior experience which directly relates to the design of such a network, an extensive measurement and evaluation capability was included in the message switching computers (the Interface Message Processors or IMP's). UCLA was designated to be the Network Measurement Center with the responsibility of defining the measurements that were necessary for the support of the analytic and simulation model activities, and to determine the performance of the network by the use of these measurement facilities. The primary concern of this report has been with the development of such a measurement capability and the utilization of this capability to create (and iteratively improve) analytic models of the network behavior as well as the true system parameters.

DD FORM 1 NOV 65 1473 (PAGE 1)
0102-014-6600

UNCLASSIFIED
Security Classification

14 KEY WORDS	LINK A ROLE	LINK A WT	LINK B ROLE	LINK B WT	LINK C ROLE	LINK C WT
Measurement of Computer Network Performance						
Computer Networks						
Networks						
Queueing						
Priority Queueing						
ARPA Computer Network						

UCLA-ENG-7165

COMPUTER NETWORK MEASUREMENTS: TECHNIQUES AND EXPERIMENTS

by

Gerald D. Cole

Supported by

ADVANCED RESEARCH PROJECTS AGENCY
Computer Network Research
ARPA Order No. 1380, Program Code No. 9D30
Contract No. DAHC-15-69-C-0285

Leonard Kleinrock, Principal Investigator

Computer Systems Modeling and Analysis Group

Computer Science Department
School of Engineering and Applied Science
University of California
Los Angeles, California 90024

October 1971

PREFACE

The research described in this report, "Computer Network Measurements: Techniques and Experiments," by Gerald D. Cole, is part of a continuing investigation of Computer Network Research, sponsored by the Advanced Research Projects Agency (ARPA), Department of Defense Contract DAHC-15-69-C-0285, under the direction of L. Kleinrock, Principal Investigator, and G. Estrin, M. Melkanoff, and R. Muntz, Co-Principal Investigators, in the Computer Science Department of the School of Engineering and Applied Science, University of California, Los Angeles.

This report was the basis of a Ph.D. dissertation (June 1971) submitted by the author under the chairmanship of Leonard Kleinrock.

ACKNOWLEDGEMENTS

The author would like to express his appreciation to his committee consisting of Professors Leonard Kleinrock, David Cantor, Gerald Estrin, Glenn Graves, and Richard Muntz, with a special debt of gratitude to his chairman, Dr. Leonard Kleinrock for his continued encouragement and guidance, for his many hours of consultation, and for numerous suggestions and improvements in the research efforts. Special thanks are also due to Dr. Gerald Estrin for help in the selection of the research topic, and for his counsel and encouragement throughout the investigation.

This work was supported by the Advanced Research Projects Agency of the Department of Defense, Contract Number DAHC-15-69-C-0285, under the direction of Dr. Lawrence Roberts, whose pioneering work in resource sharing computer networks provided the impetus for the development of such a network. I would also like to acknowledge the contribution of Dr. Robert Kahn of Bolt Beranek and Newman, Inc., who was responsible for the detailed design and implementation of the network measurement routines. Numerous colleagues at UCLA also contributed to the measurement efforts; in particular Ari Ollikainen, Vint Cerf, and Steve Crocker. I would also like to acknowledge the support and encouragement of Clark Weissman and others at the System Development Corporation during the past three years of concurrent employment and dissertation research.

Thanks are also due for the assistance in the typing and preparation of the dissertation; in particular to my wife, for typing the draft version, and to Charlotte La Roche and Barbara Warren for the preparation of the final copy.

ABSTRACT

The ARPA (Advanced Research Projects Agency) computer network involves the interconnection of about twenty (as of 1971) different research computers across the country by means of a store-and-forward message switching net. The development of such a network is an expensive and complicated undertaking which involves a variety of engineering trade-offs and decisions. Since there has been little prior experience which directly relates to the design of such a network, an extensive measurement and evaluation capability was included in the message switching computers (the Interface Message Processors or IMP's). UCLA was designated to be the Network Measurement Center with the responsibility of defining the measurements that were necessary for the support of the analytic and simulation model activities, and to determine the performance of the network by the use of these measurement facilities. The primary concern of this report has been with the development of such a measurement capability and the utilization of this capability to create (and iteratively improve) analytic models of the network behavior as well as the true system parameters.

The measurement facilities which were designed into the network included: accumulated data such as histograms and totals; snap-shot data relating to queue lengths and routing information; and traces of message flow through the network. Any of these measurement routines can be selectively enabled at one or more of the network IMP's to avoid an excessive data collection and artifact problem. In addition to the selective control over the measurements, artifact was further reduced by the careful selection of the variables to be measured, and by the development of measurement techniques such as the use of non-uniform (logarithmic) scale histograms for data which was expected to have an exponential-like distribution.

An extensive artificial traffic generation capability was also developed at UCLA and was utilized with the measurement facilities to verify and improve analytic models of the system behavior. In some instances, these models were initially developed based on a priori expectations of the system behavior, while in other cases, the models were devised from observed system performance data. In both situations, the iterative usage of model-building and experimental verification was found to provide valuable feedback regarding needed improvements in the models, and resulted in good agreement between the final models and the observed system behavior.

The models developed were chosen to represent a reasonably broad spectrum of interest, and included priority traffic analysis, message segmentation, buffering consideration, message delay considerations, and network thru-put. The models consisted of both original contributions, and extensions to previous analytic results, although the most significant aspect of the research is felt to be the demonstration that an extensive set of measurement facilities can be developed utilizing the existing network resources, and that such measurements can provide valuable insight into the network behavior which can be utilized as feedback in an iterative model-building, evaluation and design effort.

CONTENTS

	Page
FIGURES	xiii
TABLES	xvii
CHAPTER 1 - INTRODUCTION	1
1.1 Evolution of Computer Networks	3
1.2 Other Computer Networks	6
1.3 The ARPA Computer Network	10
1.4 Analysis and Modeling of Networks	15
1.5 The Measurement Program Associated With the ARPA Network	17
1.6 A Statement of the Problem Being Considered	18
CHAPTER 2 - DEVELOPMENT OF A SET OF NETWORK MEASUREMENTS	21
2.1 A Survey of Related Measurement Efforts	21
2.2 Some Fundamental Concepts of Measurement	25
2.3 Identification of the Measurement Needs	29
2.4 The Design of a Set of Network Measurements	32
2.5 The Set of Measurement Tools	38
2.5.1 Accumulated Statistics	39
2.5.2 Snap-Shot Statistics	46
2.5.3 Trace Statistics	50
2.5.4 Arrival Time Data	53
2.5.5 Status Reports	54
2.5.6 Pseudo-Message Generator	54
2.5.7 Network Measurement Center Control Programs	55
2.5.8 NMC Artificial Traffic Generation	56
2.5.9 HOST Cooperation in Measurements	58
CHAPTER 3 - DATA GATHERING AND REDUCTION TECHNIQUES	61
3.1 Time Sequence Data	61
3.1.1 Queue Lengths as a Function of Time	62
3.1.2 Routing Table Measurements	62
3.1.3 Blocking Effects	63
3.1.4 Inferences of User Behavior	64
3.2 The Use of Histograms to Estimate Density Functions	65
3.2.1 The Rationale for the Use of Log-Histograms	66
3.2.2 Consideration of the Log-Histogram as a Transformation of Variable	67
3.2.3 Approximating Density Functions	81
3.3 Estimation of Moments	88
3.3.1 A Comparison of Moment Estimation Techniques	90

CONTENTS (Continued)

Page

3.3.2 Methods of Estimating Moments from Log-Histogram Data 91

3.3.2.1 Moment Transformations 93
3.3.2.2 Superposition of Transformed Histogram Segments 99
3.3.2.3 Moment Estimates from the Approximated f(x) Function 107
3.3.2.4 A Comparison of Moment Estimates from Uniform and Log-Histograms 109

3.3.3 Some Guidelines fro the Measurement of Moments . 110

3.4 Artifact Considerations and Corrections 116

3.4.1 The Use of IMP Resources to Generate the Measurements 117
3.4.2 The Use of the Network Transmission Resources for the Measurement Data 119

3.5 Data Analysis 122

CHAPTER 4 - MEASUREMENTS OF NETWORK PERFORMANCE 125

4.1 Message Delay Measurements 126

4.1.1 Delays Due to Store-and-Forward Handling 128
4.1.2 Delay as a Function of Message Length 130
4.1.3 The Queueing Component of Message Delay 131

4.2 Measurements Related to Network Applications 137

4.2.1 Measurements of Network Usage 138
4.2.2 Estimating User Behavior 143
4.2.3 Estimating Traffic Statistics 143

4.3 Interference Between Users 147

4.3.1 Estimates of the Number of Users that a Communications Channel Can Support 148
4.3.2 Interference Tests Utilizing Artificial Traffic 153
4.3.3 Queueing Behavior in a Closed System 156

4.4 Thru-Put Tests (RFNM Driven) 160
4.5 Alternate Routing Tests 167

4.5.1 Detection of Loops in the Routing Procedure . . 167
4.5.2 Delay Reduction Due to Alternate Routing 170

CHAPTER 5 - MEASUREMENTS AS RELATED TO ANALYTIC MODELS 173

5.1 A Priority Traffic Analysis Model 174

5.1.1 The General Analysis of the Two Priority Class Queueing System 182

CONTENTS (Continued)

Page

5.1.2 The Two Priority Class Case With Priority Based on Message Length 188

5.1.2.1 Examples of the Two Priority Class Case 190

5.1.2.2 Further Analysis of the Two Priority Class Case 203

5.1.2.3 Introduction of a Delay Cost Function . 209

5.1.2.4 Reduction in Buffering Needs Due to the Priority Handling of Messages . . . 216

5.1.2.5 Experimental Verification of the Model 220

5.1.3 The Introduction of Additional Priority Classes 226

5.1.3.1 An Example With Three Priority Classes 226

5.1.3.2 The Analysis for P Priority Classes . 231

5.1.3.3 The Limiting Case of a Continuum of Priority Classes 233

5.1.3.4 Considerations of the Cost to Implement Additional Priority Classes . 236

5.1.3.5 Design of an Experiment to Verify the Multiple Priority Model 241

5.2 Models for the Segmented Message Case 245

5.2.1 Rationale and Effects of Segmentation 245

5.2.2 The Separation Between Packets 249

5.2.3 Experimental Verification of the Packet Separation Model 264

5.2.4 The Special Case of Exponential Message Lengths 270

5.2.5 The Optimal Segment Size 279

5.2.5.1 The Optimal Segmentation for Exponential Message Lengths 280

5.2.5.2 Experimental Verification of the Model 285

CHAPTER 6 - CONCLUSIONS AND SUGGESTED FUTURE RESEARCH 289

BIBLIOGRAPHY 293

APPENDIX A The Moments of the Truncated Exponential Density . . 301

B The Truncated 1/x Density Function 305

C Test Cases for the Moment Transformations 307

D Moment Corrections for Grouped Data 313

E Moment Estimation Capabilities of Uniform and Log-Histograms 331

GLOSSARY 345

FIGURES

1.3.1	The ARPA Network Configuration as of Mid-1971	12
2.2.1	A Hypothetical System Activity Diagram Showing the System State Vector and the Measurement Extension	27
2.5.1	A Typical Accumulated Statistic Print-Out	41
2.5.2	A Simplified Model of the IMP Queue Structure	47
2.5.3	A Typical Snap-Shot Print-Out	49
2.5.4	A Sample Trace Print-Out.	52
3.2.1	A Comparison of Conventional and Logarithmic Histograms.	68
3.2.2	Examples of the Exponential Density and the Transformed Equivalents	71
3.2.3	The Two Component Hyperexponential Density and the Equivalent Transformed Components	76
3.2.4	A Comparison of the Hyperexponential and 1/x Densities and the Resulting Histograms.	83
3.2.5	The 1/x Density Function and its Approximation From the Log-Histogram Data	84
3.2.6	The Hyperexponential Density (μ= 1/15 and γ= 1/3) and its Approximation From the Log-Histogram Data . .	84
3.2.7	A Comparison of the Erlang (2) Density ($\bar{x}$ = 20) and its Log-Histogram Approximation	86
3.2.8	A Log-Histogram for the Erlang (2) Density Function.	86
3.3.1	A First-Order Approximation to the Probability Distribution Across a Histogram Interval.	108
4.1.1	A Detailed Picture of the Delays Involved in a UCLA-RAND Message Transmission.	128
4.1.2	Queueing Delay as a Function of Message Length for a Fixed Average Interarrival Time	135
4.1.3	Theoretical and Measured Values of ρ for the Artificial Traffic Generator.	136

FIGURES (Continued)

4.2.1 Round-Trip Delay Measurements of the SRI-Utah Traffic 142

4.2.2 HOST-IMP and IMP-IMP Traffic Histograms 145

4.2.3 A Typical Log-Histogram From an Accumulated Statistics Message. 146

4.3.1 Average Round-Trip Delay as a Function of the Number of Message Generators Being Utilized 155

4.3.2 Arrival and Unfinished Work Diagrams for a Set of RFNM Driven Message Generators 157

4.3.3 Determination of the Limiting Queue Level for a Set of N RFNM Driven Message Generators 158

4.4.1 Thru-Put Measurements Between UCLA and the Neighboring UCSB IMP. 161

4.5.1 Loops in the Routing of Packets During a Special Test Condition. 168

4.5.2 Average Round-Trip Delay as a Function of the Number of Active Message Generators for Full Packet Messages 171

5.1.1 Plots of the Queueing Delays for Priority and Non-Priority Messages as a Function of the Priority Threshold, X_T. 192

5.1.2 The Locus of Optimal Value Points for Various Service Time Density Functions. 198

5.1.3 The Relationship Between the Values of ρ and the Normalized Priority Threshold for a Spectrum of Density Functions 200

5.1.4 The Three Density Functions 202

5.1.5 A Comparison of Queueing Delay Versus Priority Threshold for the Hyperexponential, Discrete, and 1/x Service Time Distributions. 203

5.1.6 Examples of the Exponential Family of Cost Functions 213

FIGURES (Continued)

5.1.7 Optimal Priority Threshold Curves for Several Parameter Values of the Exponential Cost Function. . . 215

5.1.8 The Effect of Priorities on the Buffer Requirements. . 219

5.1.9 The Composite Service Time Density 221

5.1.10 A Comparison of the Theoretical and Experimental Values of Queueing Delay Versus the Priority Threshold Location 224

5.1.11 A Comparison of the Delay Curves for Two and Three Priority Classes (Exponential Service) 229

5.1.12 The Effect of a Second Priority Threshold, $X_{T2} = \beta X_{T1}$. 230

5.1.13 A Comparison of the Queueing Delays for the Cases of No Priorities, Two Priority Classes, Three Priority Classes, and a Continuum of Priority Classes. 234

5.1.14 Typical Plots of Queueing Delay Versus Priority Threshold. 235

5.1.15 The Effect of the Number of Priority Classes on the Expected Queueing Delay. 238

5.1.16 The Expected Storage Requirement Including the Priority Implementation Costs. 242

5.2.1 Typical Unfinished Work Diagrams for Contiguous and Segmented Message Flows. 247

5.2.2 The Effect of Interferring Traffic to Create a Separation Between Arrivals of the Packets of a Message. 251

5.2.3 The Expected Inter-Packet Separation After Going Through n Nodes. 255

5.2.4 Arrivals and Departures for the Queue and Server in the Priority Queueing Case. 257

5.2.5 The Expected Inter-Packet Gap for the Priority System After Going Through n Nodes 263

FIGURES (Continued)

5.2.6 A Comparison of Measured and Theoretical Inter-Packet Gap Times Due to Interference Traffic. 266

5.2.7 The Effect of Segmentation on the Exponential Density of Message Lengths. 281

5.2.8 Buffer Utilization Efficiency as a Function of the Normalized Packet Length and Packet Storage Overhead. . 283

5.2.9 Buffer Utilization Efficiency as a Function of Average Message Length (Exponential Density). 286

5.2.10 Buffer Utilization Efficiency as a Function of the Normalized Packet Length. 288

A.1 The Exponential Density and Several Variations of its Truncated Form 302

C.1 The Density Functions and Convergence of the Estimate of the Mean for the Example Considered 309

C.2 An Example in Which the Moment Transformations Do Not Converge 311

D.1 A First Order Approximation to a Generalized Histogram Format. 315

D.2 An Exponential Density Shown With the Theoretical Log-Histogram Values and the First-Order Slope Approximation 328

D.3 An Enlarged Portion of the First-Order Approximation of Figure D.2 329

E.1 A Comparison of the Errors in the Moment Estimates From Uniform and Log-Histograms 341

TABLES

3.3.1	Components of the Density and First and Second Moments for Log and Uniform Histogram Intervals (Exponential Density)	92
3.3.2	A Summarization of the Monte Carlo Test Results	111
3.4.1	Transmission Requirements for Measurement Data.	119
4.1.1	Measured Round-Trip Transmission Times From UCLA to Several Different Sites	129
4.1.2	Expected Service Times for Various Length Parameter Values of the Random Artificial Traffic Generator	131
4.2.1	A Sub-Set of the Accumulated Statistics Measurement of the SRI-Utah Traffic	139
5.1.1	Nomenclature for the Three Priority Class Case.	227
5.2.1	Theoretical Values of the Inter-Packet Gap, τ	270
E.1	Theoretical Values of the Moments of the Truncated Density Functions Being Considered.	333
E.2	A Comparison of Estimates of the Mean From Uniform and Log-Histograms.	336
E.3	A Comparison of Estimates of the Variance From Uniform and Log-Histograms.	337
E.4	The Effect of the First-Order Moment Corrections for an Exponential Density.	338

CHAPTER 1

INTRODUCTION

In the quarter-century since the first electronic digital computer was developed, the computer industry has grown at a rapid rate with hundreds of different computers being marketed and scores of programming languages being written. This relatively unrestricted proliferation of hardware and software systems has contributed to the rapid development of computer technology, but has also created a tremendous incompatability problem. As a consequence, many programs have been rewritten at each instance in which they were to be utilized on hardware unlike that of the originating facility, while in other cases, programs have been so dependent on special hardware or operating system characteristics that they have not been exportable in any practical manner.

Standardization of hardware and software systems has been advocated as a solution to these incompatibility problems, but has not been viable in the past, nor does it appear to be in the near future. Some progress has been made in the areas of ASCII character code standardization, data communication interface standards, and some language definitions, but acceptable industry-wide standards on critical factors such as data structures and access methods have not been established. Indeed, not even the definition of many of these terms are standardized! The computing community is then faced with the conflicting desires of continuing the development of hardware-software systems in an

unfettered technological environment, while at the same time, being able to utilize each other's developments without extensive reprogramming efforts.

Computer networks offer a solution to this dilemma to the extent that they offer a viable method for interconnecting incompatible systems, and thereby allow resources to be shared over a wider range of users without having to standardize beyond the acceptance of a common set of interface conventions for hardware, programs, and data. Such a network also allows the user community to share other resources such as data bases, large computing power facilities, or specialized hardware systems, and in some cases to provide load-sharing and back up reliability. For more research oriented facilities, networks may provide the mechanism for utilizing and building upon the work of others, and eventually to allow us to correct the following situation as quoted from Hamming (HA69) in regard to the lack of "science" in computer science,

> "Indeed, one of my major complaints about the computer field is that whereas Newton could say 'If I have seen a little further than others it is because I have stood on the shoulders of giants', I am forced to say, 'Today we stand on each other's feet.' Perhaps the central problem we face in all of computer science is how we are to get to the situation where we build on top of the work of others rather than redoing so much of it in a trivially different way. Science is supposed to be cumulative, not almost endless duplication of the same kind of things."

The ARPA network community* has already shown a refreshing willingness to work together to solve the problems of networking and to work

*The ARPA (Advanced Research Projects Agency) network community consists of about twenty different research centers across the country. A description of the facilities in the network is given in Section 1.3.

towards the cumulative development effort to which Hamming referred. This cooperative spirit and improved communications may provide an environment suitable to bring together what Roberts (RO67) calls a "critical mass" of interdisciplinary talents and resources to solve problems which heretofore were beyond the realm of solution. In addition, the entire network is being considered as an experiment, and is being modeled and evaluated (both analytically and by direct measurement) to determine how well it meets its design goals, and to investigate areas of future improvement.

This dissertation is primarily concerned with the measurement aspects of the network evaluation, and considers the design of a set of measurement facilities, the development of data gathering and reduction techniques, and the usage of these measurements to evaluate the network performance. The latter will include comparisons of measured results with the predictions of analytic models which were developed for this study. However, before discussing the measurement considerations, we will review some earlier efforts at network-related analytic and simulation model developments.

1.1 Evolution of Computer Networks

Computer networks are a natural part of the evolution of telecommunication oriented computer systems. These systems originated with the utilization of telephone circuits for batch-oriented data transfers, and blossomed due to the need for remote computer access for time-sharing and remote-job-entry applications. These remote terminals were often made more sophisticated by the introduction of

pre-processing hardware, terminal multiplexing functions, display refresh requirements, or off-line data collection and minor processing functions, and hence began to distribute the processing functions across such a network. However, it was not until the ARPA network experiments that autonomously operating processors were interconnected for resource sharing functions. The first APPA experiment involved the experimental interconnection of a small Control Data Corporation 160A computer at SRI (Stanford Research Institute) to the large Q-32 computer at SDC (System Development Corporation) in 1964. These tests led to the interconnection of two large processors, the Q-32 at SDC and the TX-2 computer at the Lincoln Laboratories in 1966 and 1967. The experimental "network" was described by Marill and Roberts (MA66), and demonstrated the utilization of two large autonomously operating processors in a network configuration, and also provided a test bed for developing an intercommunication protocol, for evaluation of telephone communication facilities, for integration of networking functions into the two time-sharing monitors, and for demonstrating the usage of programs at one facility as subroutines to the other computer.*

The Q-32/TX-2 "network" utilized conventional direct-dial telephone facilities for the inter-processor communications, but these communication facilities were found to be unsatisfactory due to the

*One example usage of the experimental TX-2 to Q-32 net involved the use of a Q-32 LISP program at SDC to perform an infix-to-prefix translation of a source program sent from the TX-2 at Lincoln Laboratories. The Q-32 would then return the translated form back to the TX-2 where the program would be executed. In this manner, the Q-32 program was utilized as a subroutine to the TX-2 although they were different machines, separated by several thousand miles, ran under different operating systems, and utilized different programming languages.

time required to form the connection at each data transfer. The only alternative was to maintain the connection for the duration of the experiment, and therefore to utilize the line very inefficiently. These problems lead to the development of a more cost-effective store-and-forward communication system for the more recent ARPA network. Such a system is referred to as a message switching system as opposed to the conventional line-switching system of the existing telephone network, and required additional considerations of store-and-forward delays, alternate routings, and error control. However, the message switching system provides the benefit of multiplexing the usage of higher bandwidth lines so that the bandwidth can be more efficiently utilized, and dynamically selects the transmission paths based on line quality and congestion. In contrast, line switching facilities dedicate a fixed bandwidth to each user, and select the transmission path via an electronic switch mechanism at the time the connection is first made.

The choice between line and message switching depends on both technological advances and tariff changes, such that the optimal selection between the two techniques may vary with time. The communication networks being proposed by MCI (Microwave Communications Incorporated) and DATRAN may also impact these considerations as may the use of PCM (pulse code modulation) techniques by the existing common carriers, since the higher bandwidth of the PCM system could be available for direct-dial data transmission.* However, at the present time, the use

*Further information on telecommunications systems can be found in Martin (MA69), Hamsher (HA68), and Hersch (HE71).

of message switching is very cost effective, with a cost reduction of two orders of magnitude being shown by Roberts and Wessler (RO70) for the case of sending a megabit of information between two average network sites.

1.2 Other Computer Networks

Before proceeding with a discussion of other networking attempts, it is appropriate to define, or at least delimit, our meaning of "computer networks". A precise definition is of little consequence since we hope that many of our measurement and modeling techniques will be useful to a wide variety of telecommunication and computer systems. However, in the interest of delimiting the scope of the study, we offer the following definition of 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." With this description serving more as a guideline than definition, we will now consider some other network systems and experiments.

One of the earliest attempts at integrating a network of computers was the SAGE (Semi-Automatic Ground Environment) system which was developed to collect, analyze, and display radar data from sensors scattered over the continent (MA69). The system became operational in 1958, and has subsequently been updated and improved. About that same time, the American Airlines SABRE reservation system was being developed on a commercial basis (EV67), and its success has lead to the use of similar systems by other airlines, car rental agencies, hotels, etc.

The need for improved data transmission facilities led the military to development of the AUTODIN (Automatic Digital Network) data communications system in 1963 (HA68). The system includes both message and line switching facilities, and was designed with considerable thought being given to the network survivability and vulnerability. In contract to this military need for ultra-reliability, commercial and experimental networks have utilized relatively simple interconnection patterns or have relied on the direct-dial telephone network for communications. Examples of such systems include the Control Data Corporation Cybernet system (BE69) and the experimental CMU-Princeton-IBM network (RU69B) respectively. The latter net consisted of three IBM 360/67 computers, each operating under the TSS/67 time sharing monitor. Both networks are typical of most earlier networking attempts in the similarity of machine types and operating systems within any given net.

Several networks have been designed using a central store-and-forward message switch, which can reduce the network cost, but of course increases the vulnerability of the network to the loss of the central node. Such a topology has been utilized for the COINS (Community On-Line Intelligence Network System), the Lawrence Radiation Laboratory OCTOPUS system, and a proposed IBM network. The COINS system was described at the 1968 Computer Network Workshop (GE69), although little detail was given due to the military intelligence nature of the system. It appeared to be intended as a shared data base system, and utilized a standardized ASCII character code and a MIL-STD-188B hardware interface to simplify the interconnections.

The Lawrence Radiation Laboratory network was called OCTOPUS due to its star-like topology, and was described by Fletcher at this same workshop. The central computer is a PDP-6 which serves as a store-and-forward switch between the large processors such as CDC 6600, 7600, and STAR, as well as the IBM Stretch and 360/91 computers. The central switch also provides access to the huge photo-store mass memory by any of the other machines, and allows an evolutionary growth of the multi-computer complex since new computers can be connected to the system resources and can gradually be brought up to operational status.

The third star network is the IBM computer network* which has several unusual features. The central node was initially to be a medium size 360/50 computer, but was later changed to be a partition in the large 360/91, which serves not only as a store-and-forward switch, but also as a master operating system. The network will consist of several IBM 360 computers and a Control Data 6600 computer, the latter being connected via a small Honeywell DDP-516 preprocessor. The non-IBM machine will introduce a degree of generality into the network due to the considerable difference in the CDC and IBM architecture and data structures.

Several other networks or proposed networks utilize multiply interconnected topologies, much like that utilized for the ARPA network. The most similar was a network proposed by the National Physical Laboratory in England and described by Davies, et al. (DA67 & 68),

---

*The IBM network will be described in a future ACM conference paper by D. McKay and D. Karp, entitled, "Network/440 - IBM Research Computer Sciences Department Computer Network".

although it has apparently not developed beyond the proposal stage. It was first described in this country at the Gatlinburg conference on operating systems, concurrent with Roberts' description of the proposed ARPA network (RO67), which was an extension of his previously described pioneering work. The NPL network was to be a store-and-forward network using interface computers and 1.5 Mb/sec. transmission lines for the message switching net, with an expected network response time (the time from the receipt of a packet to the beginning of the output at the destination) of less than 100 msec. Packets were defined as any multiple of 128 bit segments up to a maximum of 1024 bits. The NPL network design differed from that of the ARPA net in the choice of acknowledgements (negative acknowledgements and "stop sending" messages in the NPL design compared to positive acknowledgements in the ARPA net), in the use of a handover number in the NPL design to discard messages in the case of excessive looping, and in the use of a check sum with each 128 bit segment while the ARPA net utilizes a 24 bit cyclic check code with each packet. Other features of the ARPA net will be described in Section 1.3.

A small experimental computer network is being developed at Carnegie Mellon University (BE70A), consisting of two DEC PDP-10 computers, a pair of PDP-8 minicomputers, and a hybrid computer. All five computers will be located together, and since the communications costs will be insignificant, they plan to experiment with completely connected nets as well as with more typical network interconnection topologies. They also plan to investigate various aspects of operating system control. Since CMU will also be a node in the ARPA network,

this may be one of several local sub nets within the overall network.

A network which is considerably different from those discussed above is the National Crime Information Center (NCIC) network which connects law enforcement data retrieval systems across the nation into the Federal Bureau of Investigation data bank (GE69). Many states have expanded these facilities to include other data bases such as department of motor vehicles records and local police file systems. The resulting network is quite complex, but has been reasonably successful due to the ability of each system to simulate the terminal characteristics and inputs of any other system being accessed. This was possible due to rigid format definitions and a reasonably small number of differing terminal types.

Numerous other networks are presently in the design stage, and when implemented, will include several additional large universities. New network structures are also being investigated with the goal of providing more cost-effective data transmission facilities; one such example being the ring structure proposed by Pierce (PI71). The virtual explosion of interest in networking should produce significant changes in the way in which computers are utilized, and will add a valuable new set of resources to each computer center involved in such a network environment.

1.3 The ARPA Computer Network

The ARPA network of computers is by far the most ambitious network effort ever undertaken, both in terms of the variety of facilities and in the degree of sophistication of the resources being

integrated. The basic network community consists of about twenty ARPA sponsored research sites, many of which have areas of specialization such as the graphics work at the University of Utah, the man-machine interaction work at SDC, the text editing and information retrieval work at SRI and the network measurement and modeling work at UCLA. Some other nodes have special hardware including the ILLIAC IV computer, the trillion bit laser memory, and perhaps later, special microprogramming and list processing equipment. The combination of these hardware and software resources and the cooperating interdisciplinary human resources at the various sites, creates a viable environment for new research in computer related fields.

The various sites are interconnected via a distributed store-and-forward communication net consisting of IMP's (Interface Message Processors) and 50 Kbit/sec. full duplex communication lines. Each site typically consists of one or more HOST computers operating in a time-sharing environment, but range in size from a terminal IMP (a partition within the IMP itself) to the very large ILLIAC IV. More typical computer facilities are in the Digital Equipment Corporation PDP-10 or the IBM 360/67 class, but often include special terminal or graphics equipment as well.

Each IMP connects to its local HOST(s) and typically to three or four communication lines as shown in Figure 1.3.1. The IMP is itself a small computer, a Honeywell DDP-516, with 12 K 16-bit words of memory, special communication interface circuits, and other features as needed to perform the message switching task. However, it does not include any disk or tape storage facilities nor any redundant

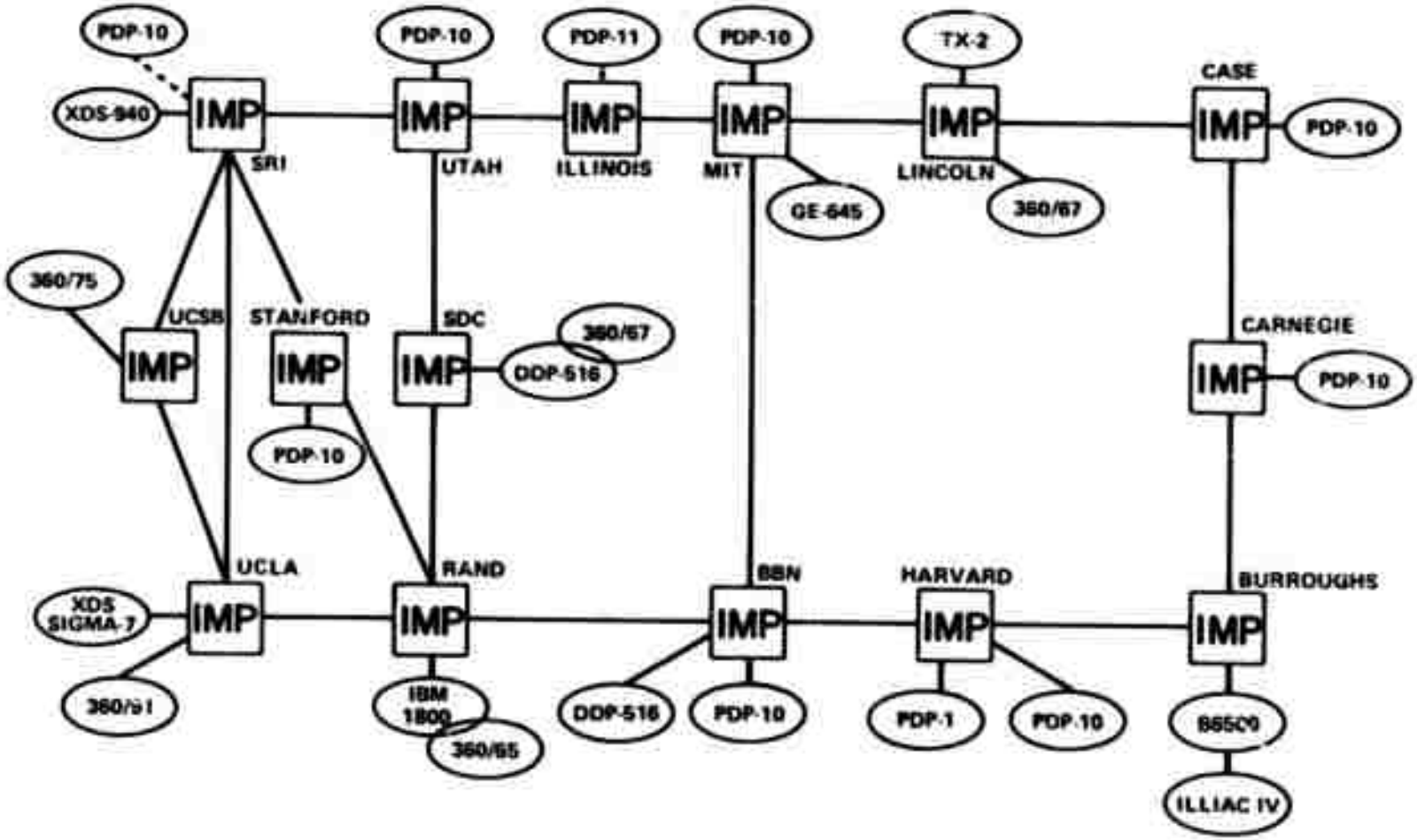


Figure 1.3.1. The ARPA Network Configuration as of Mid-1971

capability for back-up operation. The software for the IMP functions consists of a set of interrupt driven routires which handle the real time I/O functions and create tasks which are then queued and scheduled according to their relative priorities and need for output channels or other resources. A number of low priority background programs are also included in the IMP functions, and are considered to be special "Fake HOST's" since they can be the source and/or destination of messages. These functions include the IMP console Teletype, the debug package and the measurement facilities.*

HOST-IMP transmission are in terms of physical records called "messages", which consist of a 32-bit leader followed by up to 8064 bits of data. These messages are segmented into smaller "packets" for the store-and-forward transmission through the net, with each packet consisting of a 64-bit header and up to 1008 bits of data. The packets are reassembled at the destination IMP with the IMP-to-HOST transmission being in terms of a message length record. However, within the network, the flow is in terms of packets, with each packet being saved until a successful acknowledgement has been received from the next IMP along the store-and-forward path. The next step in the path is selected at each IMP from its own routing table which is updated about every half-second based on nearest-neighbor routing information as described by Fultz and Kleinrock (FU71). The successful receipt of a packet is based on the agreement of a 24-bit cyclic check code and the availability of storage space, and upon acceptance, it becomes the

*More detailed information on the Interface Message Processors can be found in the paper by Heart, et al. (HE70).

responsibility of that IMP to ensure the successful store-and-forward transmission along the next step of the path. The combination of error control and alternate routing makes the network a very reliable transmission media.

Congestion is a potential problem in any store-and-forward net because the peak data rates can easily exceed the instantaneous channel capacity. However, several congestion control features were built into the ARPA network to minimize the consequences of overload conditions including; (1) the use of logical links for communication between programs at separate HOST's, with the links becoming blocked until a RFNM (Request For Next Message) has been received, (2) the use of adaptive alternate routing, (3) present limits on IMP buffer allocation maxima for reassembly and store-and-forward packets, and (4) the use of time-outs to discard undeliverable messages or message fragments.

The problems of congestion and message delays are some of the more interesting aspects of such a network from a measurement and modeling viewpoint, and will be pursued at length in subsequent portions of this study. Such models typically involve queues in which messages or packets are delayed, and in the more general case involve networks of queues, both within the store-and-forward switches and across the global aspect of the communication net. Since much of the measurement and modeling work involves queueing theory and queueing related variables such as the arrival times, service times, waiting time, number in the system, etc., it is appropriate to review the state of development of this portion of queueing theory.

1.4 Analysis and Modeling of Networks*

The analysis of queueing systems has reached a high degree of applicability and mathematical sophistication in recent years. However, the problems of tandem queues or networks of queues have been found to be untractable except for certain special cases, due to the interactions between the various queues, and the resulting dependencies which complicate the analysis.

Burke (BU56) made an initial contribution to the analysis of networks of queues by proving that for a simple Markovian queueing system, (i.e., a system with infinite queueing space, exponentially distributed service times, and Poisson arrivals with an arrival rate of λ), the output process is also Poisson with parameter λ. Thus, for such queueing subsystems, there are no dependencies and the analysis can be handled in a reasonably straightforward manner. J. R. Jackson (JA57) utilized this approach for the analysis of a network of queues, in which each node is an independent multiserver subsystem with exponential service, and with arrivals from both within the net and from the outside, (but with all inputs forming a composite Poisson stream).

---

*This summary of developments in the analysis of networks of queues is rather cursory, but is intended to give an appreciation for the problems involved in the analysis of interconnected queueing systems. For more information on these topics, the reader is referred to Reich (RE57) and Finch (FI59) which define necessary and sufficient conditions for Burke's result, and for the analysis of sequential arrays of queues with finite storage, to Hunt (HU56), Kleinrock (KL65) and Cox and Smith (CO61). Simulation results have been described by Kleinrock (KL64 and KL70) and by Wallace (WA66), and have included performance evaluation as well as topological, routing, and channel capacity assignment considerations. Recent work in the analysis of closed queueing systems using the results of Gordon and Newell (GO67), is also relevant to network analysis.

However, this analysis did not consider the case of message transmission through a network which requires that the service, i.e., serial transmission, be a fixed value at each subsequent server. Kleinrock (KL64) recognized this problem and concluded that the only tractable analytic approach was to assume that the message length was an exponential random variable, to be selected independently at each server, i.e., the "independence assumption". His original model also assumed noise free channels, reliable nodes, single destination messages, no defections, infinite buffering, stationarity of the stochastic processes, fixed routing, negligible nodal delays, and the lack of induced traffic such as response messages, but his more recent results (KL69B) have included the latter two effects and have approximated the effect of non-exponential service. Other work at UCLA and other research sites have utilized simulation methods to gain insights into the queueing behavior of such complicated networks, with the combination of analytic and simulation techniques being complementary in many cases.

There have also been several recent developments in the design of optimal (or "sub-optimal") network topologies and channel capacity assignments. These problems typically defy a precise optimal solution but good selection techniques have been described by Frank, et al. (FR70) and Kleinrock (KL70) respectively.

Implicit in any optimal network design are one or more performance measures which have been minimized (or maximized), and the careful selection of these performance criteria is quite important since any subsequent optimization is based on satisfying these primary goals. Kleinrock (KL64) chose the average message delay as his

performance measure for a store-and-forward net, and analogous response time performance measures have been utilized in time-sharing analysis and other job turn-around time studies. These measures meet the general requirements of a performance metric, namely they have a physical meaning or interpretation, are quantitative, and are an indicator of the system effectiveness.*

The close tie between (1) performance measures, (2) analytic and simulation models, and (3) measurements was brought out by Estrin and Kleinrock (ES67B) in a paper titled "Measures, Models and Measurements for Time-Shared Computer Utilities". The combination of these factors is felt to be basic to the scientific method of investigation, and will be stressed throughout this study, although our main focus will be on the measurement aspects of the network evaluation.

1.5 The Measurement Program Associated With the ARPA Network

Performance measurement of the ARPA network has been considered to be an integral part of the design requirements from the early stages of the network development. This concern for measurements is due to several factors including the desire to gain insights into network behavior, to aid in modeling and analysis, to evaluate the network design, and to indicate areas where redesign may be needed. In some cases, the measurements may even indicate design parameters which can not be adequately selected as a constant value, but rather should dynamically vary depending on the network state or traffic load. Such parameters would then become functions of the measured values, similar

*Adapted from Goode, H. and R. Machol, System Engineering, McGraw-Hill, 1957, pp. 125-126.

to the present routing table dynamic updating. However, our initial concerns are primarily those of performance evaluation and gaining insight into critical performance parameters.

1.6 A Statement of the Problem Being Considered

UCLA is actively involved in the measurement and analysis of the ARPA network behavior, and as the Network Measurement Center (NMC), has the responsibility of defining the measurement techniques, and performing appropriate network measurements and experiments. These efforts closely follow Hamming's philosophy* that "the purpose of measurements is insight, not numbers!", with the measurement results being utilized in support of the analytic and simulation modeling efforts, the determination of network performance, and to determine any areas in which network improvements need to be made.

The overall measurement problem can be described in several somewhat overlapping tasks, including:

1. Identification of the measurement needs for modeling, simulation, performance evaluation, user behavior measurement, and possible areas of dynamic control.

2. Development of new data gathering techniques as required by the network environment and constraints.

3. Design of a set of measurement routines of sufficient power and generality to be useful for an open-ended set of experiments.

---

*An often paraphrased quote attributed to R. Hamming.

4. Provide test cases by developing several analytical models for experimental evaluation.

5. Empirical investigation of the usefulness of the measurement facilities in both model verification and in exploratory data gathering experiments, which might then produce new insights and results in an iterative sequence of experiments.

These tasks are felt to be necessary to provide an extensive measurement capability of the ARPA network performance, and to overcome the limitations observed by Estrin and Kleinrock (ES67B), that "...present methods of measurement do not allow sufficient freedom in design of experiments on complex systems". And "Most of the weaknesses in the relevancy of measurements to models arises, at present, from the unavailability of tools for making desired observations of dynamic systems". Hopefully, the network measurement techniques have avoided these past limitations, and will be sufficient to meet the needs of future experiments, and will provide the necessary feedback to evaluate the continuing usage of the ARPA network.

CHAPTER 2

DEVELOPMENT OF A SET OF NETWORK MEASUREMENTS

The desire to measure and evaluate the network performance has been one of the goals of the ARPA network since early in its conception, but because there has been little precedence in this area of measurement, we found it necessary to consider a number of aspects of the measurement needs and related costs in the network environment. This investigation included a survey of measurement techniques in related computer applications, considered some of the more fundamental measurement concepts, attempted to identify the areas in which measurement data were needed, and ended in the design of a set of measurement tools for the network study.

2.1 A Survey of Related Measurement Efforts

Although few previous network measurements have been made, there were some data gathering facilities included in the earlier experimental links between the Q-32 computer at SDC, the TX-2 computer at Lincoln Laboratories, and the DEC 338 display computer at ARPA.* These results were quite dependent upon the particular programs being utilized in the experimental net, and were limited to measurements of the number of calls, the call durations, the number of messages sent, and the number of characters sent. Some data was also taken on the COINS network (GE69) by use of a program which periodically started a pseudo-user job

*As reported in "An Experimental Computer Network", Computer Corporation of America Report No. ESD-TR69-74 (AD694-055), March 1969.

Preceding page blank

and measured the response time. Earlier work in computer system measurements has developed a variety of hardware and software recording techniques. These efforts can be categorized into three broad classes; (1) measurements of time-sharing system behavior, (2) measurements of user-computer interactions, and (3) measurements of program behavior. We shall summarize some of these results which relate to the network measurements in the following paragraphs.

Measurements were taken on the original time-sharing systems and were reported by Scherr (SC67A) and Totschek (TO65) on the MIT Project MAC Compatible Time-Shared System (CTSS) and on the System Development Corporation Q-32 system respectively. Scherr's measurements were related to analytic models which he developed and included the effects of differing time quanta and number of users, as well as measuring distributions of user think time, program sizes, and processing requirements. Totscheck measured the distributions of service and interarrival times, the effect of the number of simultaneous users, and the system overhead, and the utilization of the CPU, drum and core memory. Many of the empirical distributions were found to have similar characteristics including a range that covered several orders of magnitude, with density functions that displayed long slowly decreasing tails, and resulted in a standard deviation exceeding the mean value. He also observed that other data appeared to be from a non-homogeneous population, and assumed that the underlying population was actually a set of different sub-groups.

In both of these measurement efforts, the data were obtained as a result of modifications to the operating system programs to record

appropriate times and events for later reduction, although Scherr discussed the feasibility of statistical reduction of the data as it was gathered. In contrast to these software techniques of measurement, one can utilize a hardware data gathering approach as described by Schulman (SC67B). This device, called a Time-Sharing System Performance Activity Recorder (TS/SPAR), was able to record up to 256 internal signals of the IBM 360/67 CPU, registers, and channels which were being physically monitored by electrical probes. The data was then written on a separate magnetic tape which was part of the hardware monitor. Similar monitors have become commercially available in recent years, as have special software monitoring packages. Both have strengths and limitations, and as usual, a suitable mixture of such hardware and software techniques is often found to be efficient. A proposed experiment which utilized such a combination of techniques was the SNUPER computer (ES67A) which was designed to monitor the operation of a larger processor by the use of hardware probes, but with data selection instants being determined by special calls that had been imbedded in the program being executed. However, a second mode of operation was included in the design to eliminate the need to such program modifications by having the monitor computer sample the state of the larger machine at appropriate times. This latter mode of operation is complicated by difficulty in recognizing the desired states in the object machine.

The location of the measurement "probes" and the sample timing is a non-trivial task in both the hardware and software approaches, but can be simplified in both cases if performance monitoring is considered in the early system design phases. For example, the

software measurements on the American Airlines SABRE reservation system (EV67) were reported to have been complicated by the need to modify many of the programs to obtain suitable measurement points. The author strongly recommended that monitoring be included in the design of any subsequent system programs.

Several measurements of the user-computer interface have been made to determine the data flow, response time, and communication needs for interactive system users, and include the measurements of the JOSS system at RAND (BR67), a comparison of three systems by Fuchs, Jackson, and Stubbs (FU70 and JA69), as well as the previously described papers by Scherr and Totschek. The techniques used to measure and record the interactions included both hardware and software methods.

Computer program measurements have been made by a number of investigators, and have primarily utilized software monitoring techniques. The degradation of the program running time is not nearly as important in many of these cases since no real-time or interactive applications are involved, and therefore the degree of artifact ranges from a few percent to an order of magnitude reduction in operating speed. The latter cases are those in which a more detailed examination of each program step is required, and are fairly infrequent due to the large overhead cost involved in obtaining such data.

A sampling of measurements of this category includes the program segment size study of Batson, et al. (BA70), the measurements on programs under the GECOS II operating system by Cantrell and Ellison (CA68A), which were extended by Campbell and Heffner (CA68B), the measurements on FORTRAN programs by Russell and Estrin (RU69A), the design

of translator programs which automatically introduce suitable measurement points into object code as suggested by Presser and Melkanoff (PR69), and the survey of measurements described by Uzgalis (UZ70).

Several observations can be made about general similarities in the measurement efforts described above. First, in almost all cases, significant system performance degradations were discovered and corrected. A second common feature was the frequent occurrence of distributions with exponential-like shapes, but often with a higher coefficient of variation than the exponential density, and thirdly, the general agreement that measurement facilities should be built into the original system design rather than being an add-on or later system modification.

2.2 Some Fundamental Concepts of Measurement

A given digital system can be defined by describing either its detailed structure, i.e., the hardware and software component parts and their interconnection, or by describing the system activity, i.e., the state vector transitions for all possible system states and input stimuli. Neither of these can be completely measured or defined for systems of any significant complexity, so that measurements typically record some subset of the system behavior, and the experimenter must then draw conclusions or inferences from this limited set of observations. The art of measurement is then to obtain the necessary information about the system behavior from a relatively small set of data points.

The type of measurement information which is desired will

depend on the intended purpose of the measurements, and may vary considerably depending on whether we wish to determine if the system performs as specified, or to determine the limiting factors in the system performance. In either case, the observed system behavior will be only a subset of the state vector information, which will be sampled at some subset of the state transitions. This notion can be seen in Figure 2.2.1 which shows a hypothetical system activity diagram in terms of the state vector at sequential setps in time.* If we include the input stimuli in the state vector, then such an activity diagram would uniquely define the system behavior if the observations were made over a sufficiently long time interval.

Measurements can be conveniently related to such an activity diagram, with the "complete set" of measurement data being the entire activity record. More typically, one samples the activity data in "space" and time, (where "space" refers to the state vector space). For example, the snap-shot measurements which we shall consider in Section 2.5.2 are a set of selected state vector components which are recorded at 800 msec. time intervals; and therefore establishes a given space-time resolution level for the resulting sampled data. Some of the other measurements can be considered to generate additional state vector components, such as the accumulated counts of messages handled or histograms of message size. These new state vector components would be periodically read-out and reset to zero, and might include counts, sums, max or min values, most recent value, or inter-event times. In

*This approach is based on the system activity theory of Klir (KL67).

ADDITION TO THE STATE VECTOR DUE TO MEAS'MTS	NO. OF UNITS SERVICED	0	0	0	0	1	1	2	2
	TOTAL WORK DONE	0	0	1	2	3	4	5	5
NORMAL SYSTEM STATE VECTOR COMPONENTS	NO. IN THE SYSTEM	0	1	1	2	1	1	0	0
	STATE OF THE SERVER	IDLE	BUSY	BUSY	BUSY	BUSY	BUSY	IDLE	IDLE
	TOTAL UNFINISHED WORK	0	3	2	3	2	1	0	0
	TIME TO COMPLETE PRES. SERV.	0	3	2	1	2	1	0	0
	CLOCK VALUE	T_o	$T_o + 1$	$T_o + 2$	$T_o + 3$	$T_o + 4$	$T_o + 5$	$T_o + 6$	$T_o + 7$

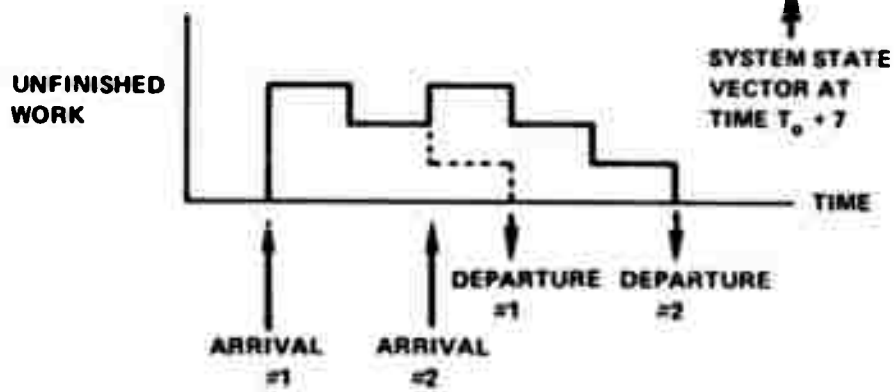


Figure 2.2.1. A Hypothetical System Activity Diagram Showing the System State Vector and the Measurement Extension

some other instances, the data would be read-out based on the occurrence of some particular system state (or event), with the two situations being generalized if clock values are considered to be part of the state vector. In fact, this assumption that the clock value is part of the state vector allows us to generalize all of our measurement techniques into one common activity representation, namely that some subset of the state vector is to be recorded at each occurrence of some other state vector component. Since we will only be recording components of the state vector at the time of the sample, any previous state information must be carried along as an extension to the basic state vector. For example, if we desire a count of the number of message inputs, we would include an N-bit component in the state vector, and would increment this field at each message input. Other records of past history such as histograms, or running sums would be handled similarly.

A more subtle measurement concept is that of observability of the desired state components. Since the data is to be taken upon the occurrence of a predefined state vector component, we may or may not be able to find meaningful data in some other component of interest depending on possible dependencies of the two components. For example, if component A can occur only when component B is equal to zero, and if the measurement is triggered on the occurrence of component A, then the measured value of B would always be zero. Such a situation actually occurs in the network measurements due to the relative priorities of the task queue and the background routines.

Other aspects in the selection of a suitable set of state vector component measurements are those of necessary sets of

measurements and sufficient sets of measurements. For example, if we are interested in the behavior of three random variables, x, y and z where we know that $z = x + y$, it is sufficient to measure the mean value of any two to compute the mean of the third, but we need statistics on all three (or exact dependence information) to obtain variance data on the three variables. An increasing amount of information is therefore necessary if we have dependent random variables. The ideal set of measurements is then the minimal set of values which is both necessary and sufficient for the desired measurement objective. However, such a mathematically precise approach is not generally feasible due to imprecise descriptions, and the constraints and trade-offs involved in the design of the set of measurements. These constraints involve the costs of implementing the measurements, including the manpower effort and the usage of system resources, as well as the allowable degradation of the system performance due to the presence of the measurement facilities. The trade-offs involve many of these same factors, but consider the possible alternatives with the constraint boundaries. We will consider such trade-offs in the design of the ARPA network measurements in subsequent sections of this study.

2.3 Identification of the Measurement Needs

The network measurement efforts have been directed towards serving a variety of measurement needs. In one aspect, the entire network is considered to be an experiment, and the measurements are to evaluate its usage and degree of success. In a second role, the measurements are to determine the performance level of the existing network

and to ensure that it meets the original design goals, while another aspect is to determine the areas in which these goals should be changed. In this effort, the measurement functions are closely related to the analytic and simulation modeling work, since each can complement the other in gaining insight into network behavior and in developing new techniques of network flow control to improve performance.

Since the delay which a message encounters in going through the network is a primary performance measure, we are particularly concerned with delay-related measurements. These include measurements of the overall end-to-end message delays, the components of the delay due to serial transmission, queueing, retransmission, etc., and the effect of system parameters such as channel bandwidth, priority classification, and packet lengths on the message delay. We are also concerned with measurements of the message length distributions and the site-to-site traffic requirements since these data are needed for analytic and simulation modeling of delay-related network functions such as routing, channel capacity, and network topology designs. Other measurements which will provide needed data for such designs include time-of-day load variations, the peak buffer storage requirements, the allocation of buffers between store-and-forward and reassembly functions, and the routing effectiveness in avoiding loops and in bifurcating flow in heavy traffic conditions.

The above comment of using message delay as the primary performance measure needs further clarification since the message delay is actually a random variable. Therefore, we must utilize some attribute of the underlying distribution as the performance measure if we are to

have meaningful and quantitative measure of performance. The use of the average delay is most common in the literature, but is not necessarily sufficient for the total design. For example, we are also interested in some measure of the variation in the delay (such as the variance or the 90th percentile) and therefore, need more than just average delay information. In many cases, we shall attempt to measure actual distributions to obtain this additional information, especially since such information is of value in the modeling efforts as well. These measurements should include distributions of message lengths, packet lengths, queueing delays, buffer requirements, and of course, the message delay itself.

Many of these measurements are associated with the queueing phenomena since it can cause a significant portion of the total delay, and can be modified by appropriate strategies in selecting the queue and service disciplines. Such changes are less expensive than brute force increases in communication line bandwidth or processor speed, and may accomplish similar reductions in the effective message delays. There are several ways of measuring queueing congestion including measurements of the time in the queueing system, the number in the system, and the busy period of the server. The first two factors are of concern to us since they directly relate to the message delays and the buffer requirements, and will therefore be part of our measurement requirements. The busy period is difficult to interpret in such a context, so it was not considered to be of particular interest.

The desire to utilize the measurement facilities and data to improve the network design can be considered to be a feedback process

in which design deficiencies are detected and subsequently corrected. Such deficiencies would not necessarily mean that the original design was poorly done, but could be due to changing demands on the network, or in other cases could be a matter of fine-tuning the network parameters.* Of course, gross-tuning would also be accomplished if imbalances existed in the overall system, e.g., if the store-and-forward processor could not support the transmission speeds of the communication lines. The occurrence of such "bottlenecks" will be the subject of certain exploratory measurements which will search for exceptional conditions, in addition to gathering information about the overall network behavior.

2.4 The Design of a Set of Network Measurements

The general design goal of the network measurement effort was to provide a flexible set of measurement tools which would allow an experimenter to evaluate various aspects of the network behavior, without having to solicit the assistance of other network sites, and without introducing an excessive burden on the IMP's and communication facilities. These measurements were to be designed to support the previously described needs of network performance evaluation and providing insight and feedback for the development of analytic and simulation models of network behavior, and several of these design considerations are discussed in the following paragraphs.

---

*Parameters which vary with loading might be made self-adaptive by introducing a continual measurement and feedback loop in the design.

a. A priori expectations: Due to the fact that the network consists primarily of time-shared computer facilities, we took advantage of previous measurement results (regarding terminal-to-computer message lengths and response times) in the design of the data gathering techniques.

b. Time and space distribution of measurements: A spectrum of measurement techniques is possible ranging from taking very detailed measurements at a single node, to taking more gross measurements across the entire network. The measurement facilities of the ARPA network cover a large portion of this spectrum by means of selective control over the data gathering routines which are enabled at the various IMP's, but are limited by several factors including (1) the rate at which the destination HOST can accept the data, (2) the bandwidth required for the data transmission, and (3) the resulting bias in the measurement data due to the use of the network for transmission.

Our experience has indicated that extensive data can be accumulated simultaneously from our three nearest neighbor IMP's, or from an arbitrary pair of IMP's, but that severe performance degradation occurs when eight or ten IMP's are monitored simultaneously. Of course, these figures should be considered to be guidelines rather than well defined limits, since the measurement effects are quite dependent upon the type of measurements being made and the other network utilization at the time.

c. Hardware versus software measurement techniques: Hardware measurement techniques have the advantage of introducing little or no

degradation to the system being monitored. However, the techniques have limited versatility and do not have ready access to data that are kept in software formats. For example, packet lengths can be determined from a pair of buffer pointers, but these pointers are kept in core locations, and therefore, the hardware probes could not assess the values without the aid of special software support. The need for such software support in either case, as well as the cost and inflexibility of the hardware monitoring techniques, led to the adoption of an entire software monitoring approach.

d. Transmission of the measurement data: The data must be transmitted to the experimenter for analysis, and preferably this transmission should be in a near real-time manner to allow interactive experimentation. The logical choice of transmission media is to use the network itself, and indeed, this was the selected mode of operation, although it introduced artifact considerations which will be discussed later. The alternative of storing the data for transmission during off hours would require a bulk store facility at each IMP, or HOST assistance in such storage, and neither was felt to be a desirable alternative. However, the interactive experimentation which one can perform with a real-time feedback of the resulting data has been found to be extremely beneficial, if not absolutely necessary, and other alternatives which do not provide such feedback are of questionable utility.

e. Artifact considerations: Transmitting the measurement data in the regular message stream introduces perturbations in several portions of the data. Any message statistics, such as length

distributions and site-to-site traffic are biased due to these data messages, as are the statitstics on round trip delays, buffer utilization, and modem channel traffic. A more subtle artifact is introduced in the input handling characteristics of the IMP, since any full buffer must be serviced and an empty buffer linked in its place before the next message starts to arrive on that channel. This condition can occur simultaneously on all input channels, so the code involved in the input servicing must be kept to a minimum. For this reason, statistics are taken at the modem outputs rather than the inputs whenever possible.

A third form of artifact of concern is the creation of disruptive conditions in a situation when they would not otherwise occur. For example, if statistics messages create a shortage of buffers at the Network Measurement Center IMP, that IMP becomes blocked for subsequent messages to its HOST. Neighboring IMP's may then have their store-and-forward buffer supply exhausted as they fail to receive acknowledgements on packets sent to the NMC.* Fortunately, this condition does not persist due to time-out periods which allow such messages to be discarded, but the transient effect could easily occur if the experiments are not carefully designed to avoid such conditions. These and other artifact considerations will be discussed in more detail in Section 3.4.

f. Data compaction and selection techniques: The concern for minimizing the data transmission requirements for statistics

*The analysis of such blocking and congestion effects has been considered as another part of the UCLA network research, and has been reported by Zeigler and Kleinrock (ZE71).

messages leads one to utilize data compacting techniques such as total counts, averages, histograms, etc. instead of transmitting the raw data samples. The need for processing in such data compaction schemes results in a trade-off between data compression and data transmission, with the optimal balance of the two being determined by their costs and the value of the resulting data to the experimenter. The problem is complicated by the fact that the measurement tools are designed for an open set of experiments, such that the designer must make decisions based on an incomplete knowledge of the total requirements. No guidelines can be given for these decisions other than common sense, intuition, and close coordination with the expected users of the measurement facilities. In the ARPA network measurements, we were only concerned with determining the general shapes of density functions, and with estimates of average values to the order of ± 10%. However, these same facilities did allow us to precisely verify the measurement capabilities on known traffic experiments, due to the combination of histograms, total word counts, etc.

Our primary application of data compacting techniques, in the sense of reducing a set of data values to some considerably smaller set of numbers, was in the usage of histograms for message and packet lengths, and total counts of the words transmitted, the number of acknowledgements, RFNM's, retransmissions, etc., and averages of round trip times. However, an even more important aspect of minimizing the data transmission is the careful selection of what data is to be collected. These considerations enter into both the design of the measurement package itself, and the design of the individual experiments.

Our measurement tools were separated into several types, each of which can be selectively enabled or disabled at any IMP. Therfore, only a subset of the measurement facilities would be enabled at a selected set of nodes, resulting in a considerable reduction in the data transmission artifact and also minimizing the data retrieval and reduction problem at the NMC.

An additional technique to minimize the data transmission requirement is to use a "report by exception" method. This technique results in data being sent only when a certain change has been noted from some stored set of values, and in one sense, only sends data when something interesting has happened. The method requires additional storage for the set of comparison values, and additional processing for the comparison operations, but of more concern, it requires a precise definition of what is of interest, and may result in a huge number of "exception reports" when unusual circumstances such as line errors or buffer congestion occur. These are situations in which one certainly does not want to send a multitude of statistics messages, and although logic could be built in to send no more often than every N seconds, the added complexity detracts from the desirability of the method. Our usage of the technique was limited to reporting arrival times (only sent when arrivals occur) and traces (only sent when certain flags are set), but in general the measurement philosophy was more one of "take the data every N seconds, and discard it at the NMC if it is not of interest". This approach also has the advantage of placing the determination of what is "of interest" in a local facility where it can be modified depending on the particular experiment being conducted.

g. Data reduction and analysis: The design of the data reduction and analysis methods sould be considered in an overall design to avoid incompatabilities with the data acquisition methods. In the case of the ARPA network statistics messages, each message has tag information which identifies its source, its type of measurement data and the IMP clock value. Information can be retrieved by selecting messages based on these tag fields and then indexing to the appropriate data field within the message. Our present technique is to format the desired information, and print it out with suitable annotation to provide a readable summary of the measurement data. Most extensive data reduction and analysis techniques could be utilized, particularly when such resources become available via the network.

2.5 The Set of Measurement Tools

The measurement facilities can be categorized into three classes; (1) those within the IMP, (2) those that have been developed at the NMC (UCLA), and (3) those available in cooperating HOST sites. The IMP generated statistics consist of accumulated statistics, snapshots, trace data, arrival times, and status reports and the IMP also has the capability of artificial traffic generation. The NMC facilities include a more extensive artificial traffic generation capability, having either deterministic or pseudo-random message lengths and intermessage times, as well as the ability to enable to disable the various IMP statistics generators, to scan the network for interesting traffic activity, and to perform reduction and print-out of annotated data summaries. The cooperating HOST facilities are primarily a future

expectation rather than a present capability, but some cooperative work has been done, particularly with SRI. Future cooperative experiments may include the measurement of HOST induced delays and total user command-to-response round trip delays as well as the usage of remote site data reduction and display techniques for data analysis.

Each of the measurement facilities is described in some detail in the following sections starting with the IMP generated statistics. The IMP generates these data messages in a set of background (low priority) programs which are selectively enabled as needed. The statistics are sent to the NMC at periodic time intervals from the "fake HOST" mechanism in the IMP's, although being low priority programs, their generation can be delayed considerably if the IMP becomes congested. The individual IMP statistics programs are summarized in Sections 2.5.1 to 2.5.6, and the reader is referred to the BBN description of their mechanization if more detail is desired.*

2.5.1 Accumulated Statistics

The accumulated statistics routine has been utilized more frequently than any of the other measurement tools, since it provides a summary report of the activity at each node where such statistics have been enabled. These data reports include histograms of the lengths of any HOST-to-IMP or IMP-to-HOST messages, and of packets which were transmitted on the modem output lines, as well as counts of ACK's, RFNM's, input errors, retransmissions, total words sent, and other data

*BBN Report No. 1822, "Specification for the Connection of a HOST and an IMP, "Bolt, Beranek and Newman, Inc., Cambridge, Mass., Feb. 1970.

of concern. Each such data message represents the activity over a 12.8 second interval, this period being determined primarily by the concern for counter overflow. The data is sent to the Network Measurement Center as a 390 byte message, which is then inspected and either discarded, printed, or put in a file for further data compacting. When printed, the data is annotated, reformatted, and in some instances additional values are computed, with a resulting print-out as shown in Figure 2.5.1. The various fields of this data format are described below.

Part 1 of the data gives message size statistics for both HOST-to-IMP and IMP-to-HOST data transfers. (In cases where more than one HOST share the same IMP, these data are the composite of all such transfers.) The statistics on single packet messages are accumulated in a logarithmic scale histogram, while multipacket traffic is recorded as a uniform interval histogram. The average number of words in the last packet of a message is also given, which in many cases, allows one to more precisely compute the actual message lengths. In the example shown, there were a total of 220 messages received from the HOSTS, of which 208 were single packet messages and twelve were two packets in length. A single message was sent to the HOST, namely this statistics message which can be seen to consist of four packets with only six words in the fourth packet for a total message length of:

$$\text{length} = 3 \cdot 63 + 6 = 195 \text{ IMP words.}$$

The second part of Figure 2.5.1 shows that these 220 messages were split fairly evenly between six destinations. These sites were selected as destinations of the artificial traffic to show the

ACCUMULATED STATISTICS SOURCE • UCLA TIME • 29146 DATE • 04-17-71

1. MESSAGE SIZE STATISTICS

1.1 SINGLE PACKET MESSAGES

LENGTH (WORDS)	FROM HOSTS		TO HOSTS
0 - 1	0		0
2 - 3	23	••••••	0
4 - 7	29	••••••••	0
8 - 15	53	••••••••••••••••	0
16 - 31	72	••••••••••••••••••••••	0
32 - 63	31	••••••••	0

1.2 ALL MESSAGES

LENGTH (PKTS)	FROM HOSTS		TO HOSTS
1	208	••••••••••••••••••••••	0
2	12	•	0
3	0		0
4	0		1
5	0		0
6	0		0
7	0		0
8	0		0

1.3 AVE. # WORDS IN LAST PACKETS : FROM HOSTS •17.7 TO HOSTS • 6.0

2. ROUND TRIP STATISTICS

DESTINATION	TOTAL R.T. TIME	# R.T.	AVE. R.T. TIME (MSEC)
UCLA	25	1	20.0
SRI	0	0	0.0
UCSB	1028	34	26.0
UTAH	2072	34	58.4
BBN	196	2	78.4
MIT	5211	35	119.2
RAND	0	0	0.0
SDC	0	0	0.0
HARV	0	0	0.0
LL	5704	33	138.4
STAN	0	0	0.0
ILL	5428	46	94.4
CASE	6565	36	145.6
CMU	0	0	0.0

3. MESSAGE TOTALS

	HOST 0	HOST 1	HOST 2	GHOST 0	GHOST 1	GHOST 2	GHOST 3
• INPUT MESSAGES FROM HOST	220	0	0	0	0	0	2
• OUTPUT MESSAGES TO HOST	218	0	1	0	0	0	0

Figure 2.5.1, A Typical Accumulated Statistic Print-Out

4. CHANNEL ACTIVITY

	# HELLO SENT	# IHY RECVD	# ACK SENT	# ACK RECVD	# PKTS RECVD	# MODEM ERRORS
CHANNEL 1 (TO SRI)	24	23	156	156	335	0
CHANNEL 2 (TO UCSB)	23	24	34	36	94	0
CHANNEL 3 (TO RAND)	23	23	65	39	127	0
CHANNEL 4 (TO)	0	0	0	0	0	0
CHANNEL 5 (TO)	0	0	0	0	0	0
TOTALS	70	70	255	231	556	0

	# RE-TRANS.	# RFNM SENT	# WORDS SENT	FREE LIST EMPTY	# BUFFERS REMOVED
CHANNEL 1 (TO SRI)	0	0	3159	0	0
CHANNEL 2 (TO UCSB)	0	0	604	0	0
CHANNEL 3 (TO RAND)	0	0	690	0	0
CHANNEL 4 (TO)	0	0	0	0	0
CHANNEL 5 (TO)	0	0	0	0	0
TOTALS	0	0	4453	0	0

5. PACKET SIZE STATISTICS (LEAVING THE IMP)

LENGTH (WORDS)	CHANNEL 1 (TO SRI)		CHANNEL 2 (TO UCSB)		CHANNEL 3 (TO RAND)	
0 - 1	14	••••••	4	•	5	••
2 - 3	9	••••	6	••	2	
4 - 7	15	••••••	4	•	7	•••
8 - 15	44	••••••••••••••••••••	8	•••	7	•••
16 - 31	44	••••••••••••••••••••	9	••••	15	••••••
32 - 63	31	••••••••••••••	5	••	4	•
TOTAL # PKTS	157		36		40	

LENGTH (WORDS)	CHANNEL 4 (TO)	CHANNEL 5 (TO)
0 - 1	0	0
2 - 3	0	0
4 - 7	0	0
8 - 15	0	0
16 - 31	0	0
32 - 63	0	0
TOTAL # PKTS	0	0

Figure 2.5.1, Continued

difference in delays when messages are sent through several store-and-forward nodes. The locations of these sites relative to UCLA are shown in Figure 1.3.1, and the round trip times can be reasonably well correlated with the topological locations, even though subsequent investigation showed that the packet routing was not as expected.

The round trip times are measured from the receipt of the first packet of the message at the originating IMP, until the receipt of the RFNM by this IMP. (The RFNM is generated by the destination IMP when the first packet has been accepted by the HOST.) The data values are accumulated as a sum of round trip times and a count of the number of round trips, with the average being computed at the NMC.

The selection of the clock resolution for the round trip times was found to be quite important. Initially the 25.6 msec. IMP clock was utilized, but this was too coarse since many round trip times are of the order of 20 msec. The alternative 100 μsec. clock was then utilized, and while the resulting resolution was very good, the 16-bit total count frequently overflowed. On some tests, this overflow could be detected and corrected by reasonableness checks, but for experiments involving hundreds of round trips, such correction was not possible. The clock resolution was then changed to 0.8 msec. by an IMP code change which shifted the 100 μsec. clock value by three bit positions prior to recording the time. The 0.8 msec. resolution is quite adequate for the round trip times, and avoids any overflow until a total round trip time of 52.5 seconds has been exceeded. Even this range is subject to overflow in some cases, e.g., for 1000 round trips with an average round trip time of more than 52.5 msec., but such overflow

should be correctable in most cases.

Part 3 lists the activity for each HOST, both real and fake, the latter being referred to as GHOST's. The table helps to determine which HOST's contributed to the composite HOST-to-IMP behavior as recorded in Parts 1 and 2. The "fake HOST's" are those listed previously in Section 1.3.

The activity on the individual modem channels is recorded in Part 4. HELLO and IHY (I Heard You) messages are sent periodically, and proper counts indicate that the lines are active. (The HELLO message is combined with the routing table update message.) The "#PKTS RECVD" total includes these HELLO messages, and any ACK's and RFNM's that have been received, as well as the number of actual message packets that have been received (both those received successfully and any having check sum errors.) Other data include the number of RFNM's sent, the total number of data words sent, the number of times an arrival found the free storage list empty, and in such cases, the number of times that an unacknowledged store-and-forward packet was discarded to free an input buffer. Each of these items are recorded separately for the various modem channels connected to the IMP, except for the number of retransmissions. No information is kept as to the channel on which a packet was originally sent, so when retransmissions are required, only a composite total is kept.

If we consider the data from channel 1 (to SRI) we see that a total of 335 packets were received, consisting of the 23 IHY's, 156 RFNM's (for which the 156 ACK's were sent), and 156 ACK's (from the packets sent via SRI). These values agree within one count to that of

Part 5, but the data for channel 3 (to RAND) do not, because of the 65 ACK's sent, when about 40 would have been expected. This effect can not be resolved from the accumulated data at one site due to the lack of additional input data. Such data is not available because of the input time constraints described in Section 2.4 (e).

The final section of Figure 2.5.1 shows logarithmic scale histograms of packet lengths on each modem channel. Only the message content of the transmissions are recorded in these histograms, i.e., they do not include the HELLO, IHY, ACK, or RFNM traffic. The choice of the logarithmic scale was due to the a priori expectations on the length distribution and will be discussed in detail in Chapter 3.

The various portions of the accumulated data can be utilized together to develop a reasonably good picture of the activity over the 12.8 second interval. In the example of Figure 2.5.1 we know the total number of messages from the HOST, their final destinations, and their average round trip times. The HOST-IMP histograms give us information of the message size distribution, and in the modem channel histograms give similar data for the packet size distribution as well as channel utilization information. In this example, the modem channel statistics show that more packets were sent via SRI than was expected from the routing table at the UCLA IMP, which would normally send about half of these packets via RAND. Such data can lead the experimenter to probe further into the cause of such behavior, but to do so, he needs the routing table information which is contained in another set of measurements called snap-shots.

2.5.2 Snap-Shot Statistics

Each snap-shot statistics message contains the queue lengths and the routing table information for that particular IMP and that particular instant of time. These values are recorded and sent at 0.82 second intervals, this time period being a reasonable compromise between the desire to see a time-sequence of state changes, and the conflicting desire to reduce the artifact caused by sending the statistics too frequently. Like all of the other measurement tools, the snap-shots can be enabled or disabled at each individual IMP.

Before describing the snap-shot measurements of the queue lengths, some background is needed on the internal operation of the IMP and the interconnections of the various queues within the IMP. Figure 2.5.2 shows a simplified picture of this queue structure, which includes the high priority task queues,* and the hierarchy of queues for the modem and HOST output channels. Each modem output channel also has a SENT queue which holds the packets which have been transmitted, but not yet acknowledged. When an ACK is received, the appropriate packet buffer is returned to the free storage list, but if no ACK is received by the end of the time-out period (of about 100 msec.), the packet is retransmitted.

The buffer handling statistics are divided into two classes; (1) store-and-forward buffers for the modem output functions, and (2)

---

*The task queue is a linked list of service requests for routing of packets, routing updates, acknowledgement handling, etc. Its priority is such that the background snap-shot routine will be deferred any time that a task requires service, and therefore it is not measurable by this method.

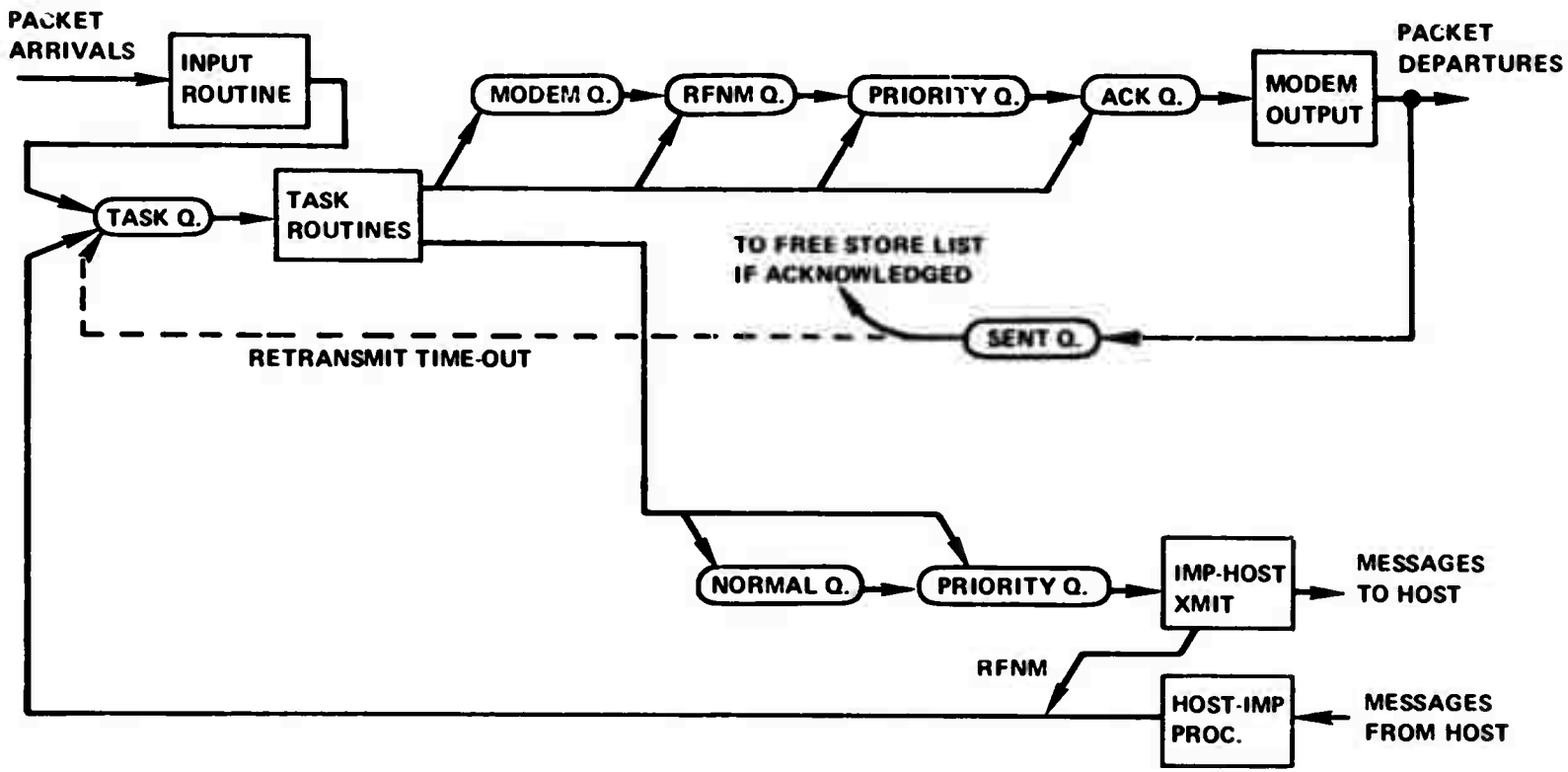


Figure 2.5.2. A Simplified Model of the IMP Queue Structure

reassembly buffers for the messages which are to be sent to the local HOST. Upper limits are set on each of these two functions, and their total is presently limited to 41 buffers.*

A typical formatted and annotated snap-shot print-out is shown in Figure 2.5.3, with this particular message being the state of the SRI IMP during a test in which a moderately heavy traffic load was being sustained between UCLA and Utah. Part 4 of the data shows that six store-and-forward buffers are in use, and the utilization of these buffers is shown in Part 2, where one buffer is on the regular output queue to Utah, another is on the priority queue, three are on the SENT queues, and the sixth buffer is the one being transmitted.†

The third part of the print-out shows the routing table at the instant the snap-shot was taken, which includes; (1) the HOP#, i.e., the number of store-and-forward "links" to get to the various destinations, (2) the estimated delay to get to the other sites (in arbitrary units of delay), (3) the best modem channel to take to get to the various sites, and (4) the status of each HOST. It should be noted that although the routing tables are based on the output queue lengths, the routing updates and snap-shots occur at different times, so one can not necessarily correlate the two sets of values.

---

*There are actually a total of about 70 buffers in the IMP, but several are dedicated to high priority functions such as input handling, and others form a contingency pool to avoid deadlock conditions. More detailed information on this and other IMP functions can be found in Reference HE70.

†These two accountings of the buffers do not always agree since the snap-shot data values are gathered as a low priority background routine, and therefore may be preempted. Such preemption can result in differing values when the routine is continued at a later time.

SNAPSHOT STATISTICS SOURCE = SRI TIME = 19499 DATE = 04-17-71 PRINT FREQUENCY = 1

1. HOST QUEUE LENGTHS

	HOST 0	HOST 1	HOST 2	GHOST 0	GHOST 1	GHOST 2	GHOST 3
NORMAL MESSAGE QUEUE	0	0	0	0	0	0	0
PRIORITY MESSAGE QUEUE	0	0	0	0	0	0	0

2. CHANNEL QUEUE LENGTHS

	OUTPUT Q	RFNM Q	PRIORITY	ACK Q	SENT Q
CHANNEL 1 (TO UCLA)	0	0	0	0	2
CHANNEL 2 (TO UCSB)	0	0	0	0	0
CHANNEL 3 (TO UTAH)	1	0	1	0	1
CHANNEL 4 (TO STAN)	0	0	0	0	0
CHANNEL 5 (TO)	0	0	0	0	0

3. ROUTING TABLE INFORMATION

DESTINATION	HOP #	DELAY	BEST CHAN.	HOST ALIVE
UCLA	1	4	1	YES
SRI	0	0	0	NO
UCSB	1	4	2	YES
UTAH	1	4	3	YES
BBN	3	12	4	YES
MIT	3	12	3	NO
RAND	2	8	4	NO
SDC	2	8	3	YES
HARV	4	16	4	NO
LL	4	16	3	NO
STAN	1	4	4	NO
ILL	2	8	3	NO
CASE	5	20	3	NO
CMU	6	24	4	NO

4. BUFFER STORAGE

FREE STORAGE LIST LENGTH = 35
NO. OF STORE & FWD BUFFERS = 6
NO. OF MES. REASSY BUFFERS = 0
TOTAL NO. OF BUFFERS = 41

Figure 2.5.3, A Typical Snap-Shot Print-Out

The snap-shot measurements were developed primarily with routing measurements in mind, but have also been found to be useful for other experiments involving the queueing behavior of the IMP's. Some examples of these experiments will be given in Chapters 4 and 5.

2.5.3 Trace Statistics

The trace capability allows one to literally follow a message through the network, and to learn of the route which it takes and the delays which it encounters. However, it is one of the more complicated of the measurement data formats to describe because the interpretation of the data depends on the function of the IMP handling the message, i.e., either source, store-and-forward, or destination, as well as depending on the possibility of retransmissions or multiple copies of packets.

A typical store and forward situation will, when traced, result in a record of (1) the time of arrival, (2) the time at which the message is processed, i.e., put on an output queue, (3) the time at which transmission is initiated, and (4) the time when the ACK is received. In the case of a retransmission, this latter time is the time of retransmission, and is so indicated by a tag bit in the data block. At the destination IMP, this fourth word is the time at which the IMP-to-HOST transmission was completed. All times are recorded in terms of a clock with a 100 μsec. resolution, and with a full scale range of about 6.5 seconds (for a 16-bit word).

The other information in the trace data message includes the output channel (if on a modem channel), the HOST number, or the "fake

HOST" number plus three, as well as the entire header of the packet. The header contains the source, the destination, the link number, the message number, the packet number, and the priority/non-priority status.

A typical trace data message is shown in Figure 2.5.4, which records the times of the various events at the SDC IMP, which also happens to the destination of the message. The message was four packets in length, and was sent from HOST #0 at UCLA on link number one. (The GHOST 3 destination is the "fake HOST' which is used to discard artificial traffic. Such discarding is necessary to generate the RFNM for the logic link.)

The interarrival times can be obtained from the T(IN) data, and indicate the presence of some interference traffic since the packets are otherwise contiguous at 23.0 msec. intervals. Since this data message is from the destination IMP, the values of T(QUEUE) are the times at which the packets were placed on the reassembly queue, and a comparison of the clock readings shows that the transmission to the discard "fake HOST" did not begin until after the last packet of the message was received, i.e., the entire message had been reassembled. The value of T(*) is the time at which the transmission to the HOST was completed, with the discard "fake HOST" accepting a full packet of data in about 2.5 msec. Similar trace data could have been obtained at each IMP which handled the message, although the data content would have differed in some aspects as described earlier.

An IMP will generate trace data if it handles a packet with its trace bit set <u>and</u> if the trace function has been enabled at that particular IMP. The trace bit in the packet header can be set in

TRACE DATA SOURCE = SDC DATE = 04-17-71 OVERFLOW : NONE PRINT FREQUENCY = 1

MSG. #	TYPE	SOURCE	DESTINATION	FUNCTION	T(IN)	T(QUEUE)	T(XMIT)	T(*)
58	REGULAR	HOST 0,UCLA	GHOST 3,SDC	DESTIN	2097.3	2098.0	2107.1	2107.6
58	REGULAR	HOST 0,UCLA	GHOST 3,SDC	DESTIN	2091.3	2092.0	2104.8	2107.0
58	REGULAR	HOST 0,UCLA	GHOST 3,SDC	DESTIN	2065.7	2066.4	2102.0	2104.7
58	REGULAR	HOST 0,UCLA	GHOST 3,SDC	DESTIN	2039.8	2040.6	2098.3	2101.9

MSG. #	LINK #	PKT #	LAST PKT	PRIORITY	CHANNEL	STATUS
58	1	3	YES	NO	6	-
58	1	2	NO	NO	6	-
58	1	1	NO	NO	6	-
58	1	0	NO	NO	6	-

T(*) IS TIME TRANSMISSION TO HOST COMPLETED FOR IMP TO HOST MESSAGES - OTHERWISE TIME ACKNOWLEDGMENT RECEIVED

Figure 2.5.4, A Sample Trace Print-Out

either of two ways; (1) if the trace bit was set in the leader of the message as received from the HOST, or (2) by means of the autotrace feature of the measurements. The latter function allows one to trace every Nth message originating at one or more IMP's.

It is important to note that the various statistics messages are indeed "messages", and therefore are included in the autotrace count, and will be traced if they are so chosen. The frequent status report messages* have therefore lessened the value of the autotrace feature for use in monitoring the early network usage on a "report by exception" basis. That is, one could have otherwise obtained data on network message transmissions without a continual polling of the network nodes.

2.5.4 Arrival Time Data

The times at which packets arrive on a specified channel can be recorded by selectively turning on this measurement routine. The arrival times are in terms of a clock with 100 μsec. resolution and will be sent to the preselected destination at 1.6 second intervals. The allocated buffering is such that no more than sixty such arrivals can be recorded in the 1.6 second period, but this is not a problem since the statistic of interest is the interarrival time, and losing some arrivals merely reduces the sample size somewhat.

---

*The status report messages are described in Section 2.5.5.

2.5.5 Status Reports

Reports are sent to a specified HOST (most often the IMP Teletype at BBN) to indicate the status of the IMP's, the HOST's, and the data transmission lines. These reports are sent at 52 second intervals if the status has changed, or otherwise at 14 minute intervals.

2.5.6 Pseudo-Message Generator

Each IMP can be utilized as a source of periodic messages referred to as pseudo-messages since they have no actual information content. Such messages are useful as artificial traffic in network loading tests, for use as periodic traced messages for delay sampling, and for any other tests in which one desires known selectable message characteristics.

The control parameters allow the experimenter to select the length of the message from 0 to 777 16-bit words, the period at which the message is sent in 25.6 increments of time (subject to having received any previous RFNM), the destination of the messages, the link number, and to set the trace bit if traces are desired. These parameters are single valued, e.g., the message generator cannot send to more than one destination at a time nor can it send on more than one link at a time.

The pseudo-messages originate from "fake HOST" #3, and typically are sent to the discard "fake HOST" at some other side. This discard HOST accepts messages at a rate of about one 16-bit word every 40 μsec., although being a low priority bakcground routine, the discard

routine is subject to preemption, and therefore the effective discard rate may be considerably slower when the IMP is very active.

2.5.7 Network Measurement Center Control Programs

Several routines have been written at UCLA to control the data gathering and reduction processes, and have been integrated into an overall measurement control program of considerable flexibility and generality. This program consists of routines which (1) accept measurement data from the network, (2) control its processing and print out, and (3) handle the initialization, control and termination of tests. The data handling routine is a FORTRAN program which formats and prints out the measurement statistics generated by the IMP's. The program uses a number of assembly language subroutines which interface with the system monitor and control the interrupts, the clock and the IMP interface.

The measurement program is controlled from the operator's console in a manner similar to that used to set up experiments at the IMP console. For example, the experimenter would type SNAP (NL) followed by 5 (NL) if he wished to have only every fifth snap-shot data message printed out. He has similar control capabilities for the accumulated statistics and trace data messages, as well as having control over the destination of the data (the line printer or magnetic tape), the source of the data (from the IMP or magnetic tape), and other optional features such as a decimal "dump" print-out of the received data.

The measurement program can be utilized in a time-sharing

environment, but for extensive data gathering and traffic generation experiments, it is operated as a dedicated system function. Most of the tests which have been run as part of this research have required the latter type of operation, although the use of the time-sharing is also a very pwoerful mode of operation since it provides auxiliary services to the measurement activity. These capabilities, both within the UCLA HOST and at other network facilities, will be utilized extensively in future network monitoring and data analysis.

2.5.8 NMC Artificial Traffic Generation

One of the routines which the NMC control program can call is the artificial traffic generator. This routine allows the user to specify which of the 63 possible generators that he wishes to utilize, and to define the destination, the initial message length, the final message length, the increments for successive runs, the duration of the run in milliseconds, and the number of messages sent on each link. If a non-zero length increment has been specified, the experiment will terminate after the end message length has been run, but for a zero increment value, it will create continuous traffic until stopped from the operator's console. Statistics on the artificial traffic can also be printed out on the console at the operator's request.

The artificial traffic can be either deterministic, (predefined message lengths and transmission intervals) or can be stochastic with pseudo-random message lengths and/or intertransmission times. In the NMC implementation, these two variables are selected from tables of random digits having the desired distribution shape and

parameters. (Run-time generation of such random variables was too time consuming due to the lack of floating point hardware on the UCLA Sigma 7.)

The operation of the artificial traffic generator involves additional logic to ensure that transmission are not attempted on any link that is blocked, i.e., that is waiting for a RFNM from a previous transmission. If no link is free for use, the next time interval, Δt, is selected from the random number table and the program waits Δt seconds to retry the transmission. A new message length is selected for each transmission from a table of random lengths, thereby providing the desired distribution of message lengths. The two tables are independent and can be initialized separately to have different distributions for the intertransmission times and message lengths. Both tables are circular, i.e., the pointers wrap around from the bottom of the list back to the top, but are of differing lengths to avoid a cyclic repetition each time through the lists.

Artificial traffic can be sent to any destination since the leader information can be supplied separately for subsets of links. However, the network traffic loading patterns that can be obtained by such traffic are limited by the topology of the network, and by the proximity of the nodes and lines to the NMC. For example, the interarrival times of messages can be reasonably well controlled at the UCLA IMP to observe the queueing and alternate routing behavior, but little control can be maintained over these factors at other nodes due to the approximately equal input and output data rates. (Alternate routing effects can provide transient conditions in which the input

and output rates are not equal, but these effects can not be well controlled as part of an experiment.)

The serial transmission time of the HOST-to-IMP interface also tends to limit the degree of control over the interarrival times of messages at the IMP, but since the transmission rate is twice that of the IMP-to-IMP modem channels, the effect has not been a serious problem in the establishment of high traffic levels. However, the queue which may form on the HOST side of this interface will result in a tandem queue system which is particularly difficult to analyze. This effect will be discussed further in Section 5.3.1.

Since the modem channel requires about 5 to 20 msec. to service a typical short message, the artificial traffic generator must be able to provide new arrivals with at least this frequency. The need to provide some control over the arrivals, e.g., to have approximately exponential interarrival times, requires a higher resolution timer and the 2 msec. clock of the Sigma 7 was used for this purpose. Such artificial traffic generation required the dedicated use of the machine, since these real-time functions could not be performed in a time-sharing environment.

2.5.9 HOST Cooperation in Measurements

Although a measurement ground rule was that we would not require the assistance of the various HOST computers to obtain data, HOST cooperation can add significantly to the confidence in the measurements. An early example of such cooperation is described in the experiment of Section 4.2.1 in which John Melvin of SRI modified his

program to record the console activity. The data correlated quite well with the measurement results. Further discussions of their mode of operation clarified other aspects of the data such as the predominance of 5 word packets and 5 packet messages. Such cooperation will hopefully lead to more extensive measurement of HOST related network activities.

CHAPTER 3

DATA GATHERING AND REDUCTION TECHNIQUES

The network measurement package includes a variety of monitoring techniques such as obtaining counts, averages, histograms, event times, and traces of store-and-forward flow. The use of a spectrum of measurement methods was required to meet the wide range of interests of the measurement project, including design verification tests, performance evaluation, support of the analytic and simulation modeling efforts, and gaining insights into potential network improvements. Some of the results of these tests and evaluations will be presented in Chapter 4, but first we shall describe the techniques that were developed for gathering and reducing the data.

3.1 Time Sequence Data

Many of the network functions of interest can be considered to be stochastic processes (i.e., time sequences of random variables). These functions include queueing processes, routing effects, blocking effects, and the random nature of the user-computer communications. The time element is an important factor in each of these areas, since the sequence in which events occur can alter the resulting behavior to a considerable degree. Therefore in many cases we will be more concerned with following the sequence of such events rather than measuring averages or distributions.

Preceding page blank

3.1.1 Queue Lengths as a Function of Time

The detailed time behavior of queue lengths is not of particular concern, but we are interested in a more gross time scale for the effects on routing and possible blocking conditions. Indeed, the detailed queue behavior is not measurable by our methods due to the relatively short period in which the queue lengths can change. A full packet of 1008 bits can be transmitted over the 50 Kbit/sec. line in about 20 msec., while our snap-shot measurements of queue lengths occur only every 820 msec. which is as frequently as they can occur without introducing an excessive transmission artifact. However, the snap-shot timing is satisfactory for investigating longer term phenomena such as congestion periods and their effect on routing and blocking, as well as the effectiveness of time-outs and alternate routing in circumventing blocked or defective lines or nodes.

3.1.2 Routing Table Measurements

Each IMP maintains its own routing table which it updates about every half second using nearest neighbor estimates of delay to each destination. This procedure is adaptive, although it is not optimal or even necessarily stable. An example of an instability in routing is the simple case in which node A routes messages to node B to be forwarded to their destination C, and node B thinks that the best path to get to C is via node A. A ping-ponging effect would then occur between A and B. Such loops can occur as transient cases between updates, but should not persist indefinitely. Measurements of the IMP routing tables are included in the snap-shot data which can be taken

at 820 msec. intervals. The snap-shot timing is reasonably synchronized across the network to help detect such loops and to provide an instantaneous picture of the entire net.

The delay estimates which are utilized in the routing updates are formed in a matrix which includes all of the destinations and all of the communication channels to the nearest neighbors. For example, the estimated delay to a given destination, if the channel to node A is taken, is the delay from A to the destination, plus the queue length on the channel to A, plus four. The delay is expressed in arbitrary units since the intent is merely to find the best channel to take. (The number four is a threshold value for the alternate routing.) Many variations of this routing update scheme are possible, and part of the measurement effort is to provide some insight into viable alternative procedures which may improve the network performance (FU71B).

3.1.3 Blocking Effects

If a store-and-forward node exhausts its buffer supply, it becomes blocked in the sense that neighboring nodes can not successfully transmit messages to it. This massive congestion can then propogate throughout a surrounding region until such time as the congestion is cleared by successful transmissions, alternate routing, or discarding of messages due to time outs. We are interested in such blocking effects, including the time and place at which they occur, the number of nodes affected, and the duration of the blockage (ZE71). Unfortunately, the present measurement data messages tend to suffer the same congestion as other messages in such a blocked condition, and do not

necessarily reach the NMC. The fact that the measurements are a background task also limits their effectiveness in such instances when the IMP becomes very "busy" with foreground tasks. However, the loss or skew in the data message timing is some indication in itself of the blocking effect, so a first-order measure of the blocking can be obtained.

Blocking can occur in either store-and-forward and/or reassembly functions because each of these functions has a separate upper limit on buffer allocation. Therefore, blocking might occur in the reassembly function, which would not allow subsequent "for HOST" inputs, but the IMP could continue store-and-forward message handling including the sending of snap-shot data. The opposite effect could also allow measurement data to be obtained, but only if the store-and-forward blockage occurred at the NMC.

3.1.4 Inferences of User Behavior

The observed network traffic can sometimes be used to infer the type of computer applications which are being utilized via the network. For example, in one early test, the traffic data clearly showed a pattern of large file transmissions followed by a period of interactive traffic, with the interactive traffic consisting of character-by-character commands in one direction and line-by-line transmission for the replies. (This test is described in some detail in Section 4.2.1).

In other instances, the use of the network for interactive applications versus batch work and file transfers should be apparent,

particularly when the source-destination pairs are considered.

3.2 The Use of Histograms to Estimate Density Functions

A common practice in the analysis of test data is to generate a histogram or relative frequency bar chart. Such a histogram gives an estimate of the underlying probability density function, and if the sample size and the histogram interval size are chosen properly, the histogram can model the density function quite closely. Typically the interval size is selected after an inspection of the test data, so that the estimate of the density function has an acceptable quantization, and as many intervals would be used as were felt to be necessary. However, in experiments where there is a limited ability to store and/or transmit the raw data values, some compaction of the data is necessary, and therefore, one can not use a heuristic approach. Such an experiment would have to be designed with a predetermined number of histogram intervals covering a particular range of the variable. The success of such an allocation of the histogram intervals would depend on the amount of knowledge which the experimenter had about the density function being measured.

Conventional histograms are composed of a number of equal size intervals covering the range of the variable. Therefore, for a given range and number of intervals, the interval size is fixed. For skewed densities such is the exponential, the resulting histograms are often very poorly quantized since most of the occurrences fall in a relatively small fraction of the intervals. A more satisfactory approach to the measurement in such cases is to use non-uniform interval

sizes, so that the intervals are smaller in those regions which are most likely to occur. For exponentially skewed densities, a logarithmic variation has been found to be a viable solution to this problem, and the rest of this section is devoted to the analysis of such logarithmic histograms.

3.2.1 The Rationale for the Use of Log-Histograms

Logarithmic histograms are of particular interest because; (1) there is an expectation that certain measurements will be of an exponential shape, e.g., more short packets than long packets, and (2) the logarithmic intervals are computationally easy to obtain in the case of base two logarithms. A binary computer data word can be readily categorized into one of the logarithmic intervals by merely finding the most significant ONE in the word. In this manner the first interval is for a value equal to one, the next for two or three, the next for four through seven, etc. Zero can also be included if it is a meaningful value.

An intuitive measure of the capability of a histogram to model a density was mentioned in the above example of the exponential density function. We concluded that non-uniform interval sizes should be used, because most of the data would otherwise occur in a few intervals. This observation leads to the conclusion that an optimal choice of non-uniform interval sizes would be to have an equal probability of occurrence for each interval. Such a histogram would appear to be a uniform density, and in effect, would be the result of a special

transformation of the independent variable. The logarithmic interval histogram is one such transformation, and as shown in Figure 3.2.1, does produce a more uniform distribution of values in the histogram intervals. Of course the original density shape is lost in the transformation, but as we shall see later, it may actually be easier to test "goodness of fit" for analytic functions on the transformed variable than on the original variable.*

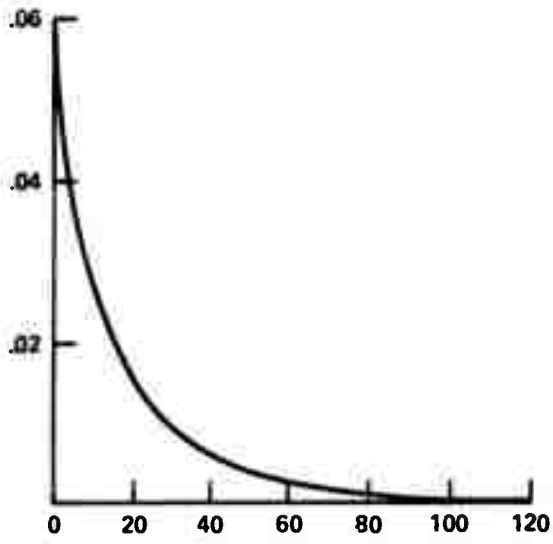
The data in any realistic experiment is of finite resolution, and therefore our random process would be a discrete rather than a continuous process. This discrete nature is even further emphasized when we consider the data in histogram form. However, there is a considerable analytic value in considering the continuous case since it represents the envelope of the discrete function, and is in some cases more easily handled in the mathematical formulation. We will actually consider both cases, since each provides a certain insight into the problem.

3.2.2 Consideration of the Log-Histogram as a Transformation Variable

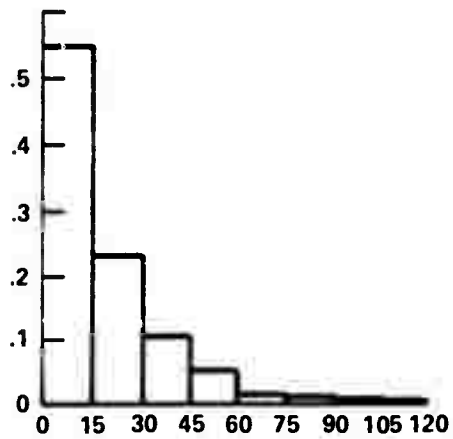
A comparison of the conventional and logarithmic histograms in Figure 3.2.1 lead us to the observation that, what, in effect, had happened was a transformation of the independent variable. The original exponential density function:

---

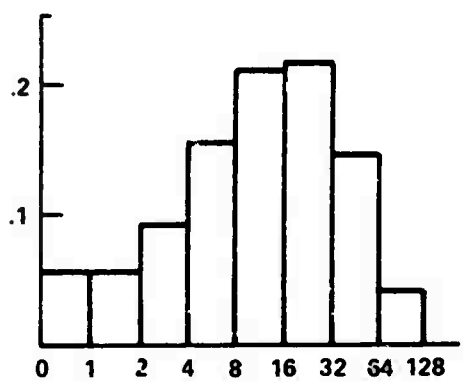
*A classical goodness of fit test does not apply outside of the domain for which the test was made. However, the transformed equivalent of exponential-like densities have more significant differences in appearance in the log-histogram domain than in the conventional scale domain.



(a) THE ORIGINAL DENSITY FUNCTION



(b) A CONVENTIONAL HISTOGRAM



(c) A LOGARITHMIC HISTOGRAM

Figure 3.2.1. A Comparison of Conventional and Logarithmic Histograms

$$e(x;a) \equiv a \exp(-ax) \qquad 0 \leq x \tag{3.1}$$

was transformed into a new function, $\varepsilon(\xi;a)$ by the transformation of variables:

$$\xi = \log_2 x \tag{3.2}$$

The new density function can be obtained by:

$$e(x;a)dx = \varepsilon(\xi;a)d\xi \qquad \forall\ x,\xi \tag{3.3}$$

which yields:

$$e(\xi;a) = e[x = \exp(\xi \ln 2)] \cdot \frac{dx}{d\xi}$$

or:

$$e(\xi;a) = a \exp(-a \exp(\xi \ln 2)) \cdot (\ln 2 \exp(\xi \ln 2)$$

If we make use of the identity:

$$a \equiv \exp(\ln a) \tag{3.4}$$

we then obtain:

$$e(\xi;a) = \ln 2 \exp[(\xi \ln 2 + \ln a) - \exp(\xi \ln 2 + \ln a)] \tag{3.5a}$$

We note in passing that this density is the "doubly exponential" or Fisher-Tippett type 1 density (SA61), which is also known as the distribution of extreme values. A form which more closely represents the physical probelm being studied can be obtained by converting to base two exponentials.

$$e(\xi;a) = \ln 2 \exp[(\xi + \frac{\ln a}{\ln 2})\ln 2 - \exp((\xi + \frac{\ln a}{\ln 2})\ln 2)]$$

or

$$\varepsilon(\xi;a) = \ln 2 \exp_2[(\xi + \log_2 a) - \frac{1}{\ln 2} \exp_2(\xi + \log_2 a)] \quad (3.5b)$$

This equation can be written as a function of a standardized variable ψ,

$$\psi = \xi + \log_2 a \quad (3.6)$$

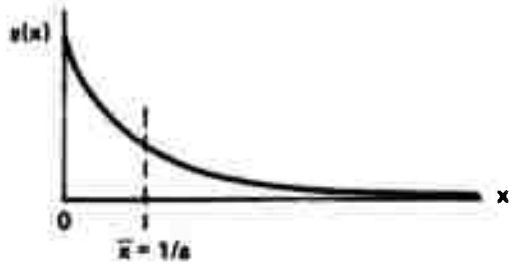
resulting in:

$$\varepsilon(\psi) = \ln 2 \exp_2[\psi - \frac{1}{\ln 2} \exp_2(\psi)] \quad (3.7)$$

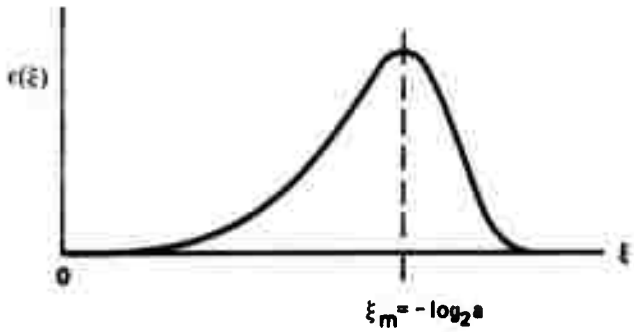
The definition of ψ shows that it is merely the variable ξ plus the constant $\log_2 a$, i.e., assuming a fixed value of the parameter a. Therefore, the function $\varepsilon(\xi;a)$ is of the same shape as $\varepsilon(\psi)$, but is shifted along the ξ axis by an amount equal to $\log_2 a$. This observation has some interesting and important consequences.

1. The transformed exponential will have exactly the same shape regardless of its mean value.
2. The effect of the mean is merely to shift the function along the ξ axis by $\log_2 a$.
3. From Theorem 3.2.1 we shall see that the mean value, 1/a, will coincide with the mode of the transformed density.
4. The standardized density $\varepsilon(\psi)$ will have its modal value at $\psi = 0$.

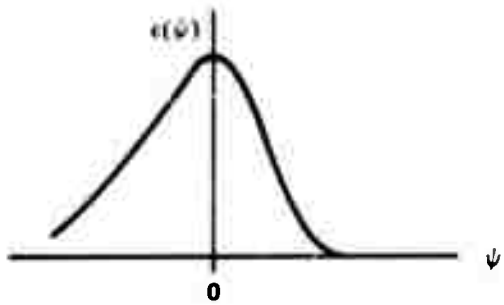
These observations are shown pictorially in Figure 3.2.2 which shows the original exponential density and both the transformed density and



(a) THE EXPONENTIAL P.D.F.



(b) THE LOGARITHMIC TRANSFORMATION



(c) THE NORMALIZED TRANSFORMATION

Figure 3.2.2. Examples of the Exponential Density and the Transformed Equivalents

the standardized version of the transformed function.

Theorem 3.2.1

The modal value of the function $\varepsilon(\xi;a)$, which is the log-histogram transform of the exponential density function, $e(x;a)$, occurs at a value, ξ_m, where ξ_m is related to the mean value of the original density function by:

$$\xi_m = -\log_2 a$$

Proof:

This relationship can be shown by setting the derivative of $\varepsilon(\xi;a)$ to zero,

$$\frac{d}{d\xi}\,\varepsilon(\xi;a) = \frac{d}{d\xi}\left\{a \ln 2 \exp[(\xi \ln 2) - a \exp(\xi \ln 2)]\right\}\Bigg|_{\xi = \xi_m} = 0$$

such that:

$$a \ln 2 \,\{\exp[\xi_m \ln 2 - a \exp(\xi_m \ln 2)]\}$$

$$\cdot\ [\ln 2 - a \ln 2 \exp(\xi_m \ln 2)] = 0$$

Solving for ξ_m, we have:

$$a \exp(\xi_m \ln 2) = 1$$

So that:

$$\xi_m = -\log_2 a \tag{3.8}$$

which completes the proof of the theorem.

Other analytic density functions could be transformed in a manner similar to that for the exponential. However, we prefer to consider only the hyperexponential and Erlang(K) families at this time, since they allow a mathematical simplification in some of the subsequent analysis. These two families of functions are also called the Erlang parallel and series functions respectively, i.e, they are equivalent to a parallel or cascade group of exponential functions. In the case of interest, the packet lengths may be modeled by such density functions, and therefore, so would the resulting service time density for a serial transmission facility.

a. <u>The Logarithmic Transformation for Hyperexponential Densities</u>

If we consider the two component hyperexponential density:

$$h(x;a_1,a_2) = w_1[a_1\exp(-a_1x)] + w_2[a_2\exp(-a_2x)], \quad 0 \leq x \tag{3.9}$$

where:

$$w_1 + w_2 = 1 \qquad \text{and} \qquad w_1,w_2 \geq 0,$$

and the logarithmic transformation:

$$\xi = \log_2 x$$

we obtain the transformed density:

$$\eta(\xi) = h(x = \exp(\xi \ln 2)) \cdot \frac{dx}{d\xi}$$

so that:

$$\eta(\xi) = \{w_1 a_1 \exp[-a_1 \exp(\xi \ln 2)] + w_2 a_2 \exp[-a_2 \exp(\xi \ln 2)]\}$$

$$\cdot \{\ln 2 \exp(\xi \ln 2)\}$$

or:

$$\eta(\xi) = w_1\{a_1 \ln 2 \exp[\xi \ln 2 - a_1 \exp(\xi \ln 2)]\}$$

$$+ w_2\{a_2 \ln 2 \exp[\xi \ln 2 - a_2 \exp(\xi \ln 2)]\} \tag{3.10}$$

We can put these expressions in the doubly exponential form by a manipulation similar to that of equation (3.5a) and obtain:

$$\eta(\xi) = w_1\{\ln 2 \exp[(\ln a_1 + \xi \ln 2) - \exp(\ln a_1 + \xi \ln 2)]\}$$

$$+ w_2\{\ln 2 \exp[(\ln a_2 + \xi \ln 2) - \exp(\ln a_2 + \xi \ln 2)]\} \tag{3.11}$$

We can recognize this form as the weighted superposition of the two exponential components,

$$\eta(\xi) = w_1 \varepsilon(\xi; a_1) + w_2 \varepsilon(\xi; a_2) \tag{3.12}$$

This form can be generalized to the n component case, namely:

$$\eta(\xi) = \sum_{i=1}^{n} w_i \varepsilon(\xi; a_i) \tag{3.13}$$

where:

$$\sum_{i=1}^{n} w_i = 1 \quad \text{and} \quad w_1, w_2, \ldots, w_n \geq 0$$

The weighted superposition can be visualized as the density resulting from the sum of n of the $\varepsilon(\psi)$ functions, each of which has been scaled in amplitude by its w_i weighting and shifted by the $\log_2 a_i$ value. Some examples of such hyperexponential densities and their transformed functions are shown in Figure 3.2.3.

b. <u>The Logarithmic Transformation for the Erlang(K) Family</u>

A second family of "generalized" density functions is the set of Erlang(K) functions which can be used to model densities which have a variance to mean-square ratio of less than unity, i.e., whose standard deviation is less than the mean.

The general form for this family of density function is:

$$g(x;a,k) = \frac{1}{(k-1)!}\, ka(kax)^{k-1} \exp(-kax), \quad \begin{matrix} 0 \leq x \\ 1 \leq k \end{matrix} \tag{3.14}$$

where, in all cases, the mean is 1/a.

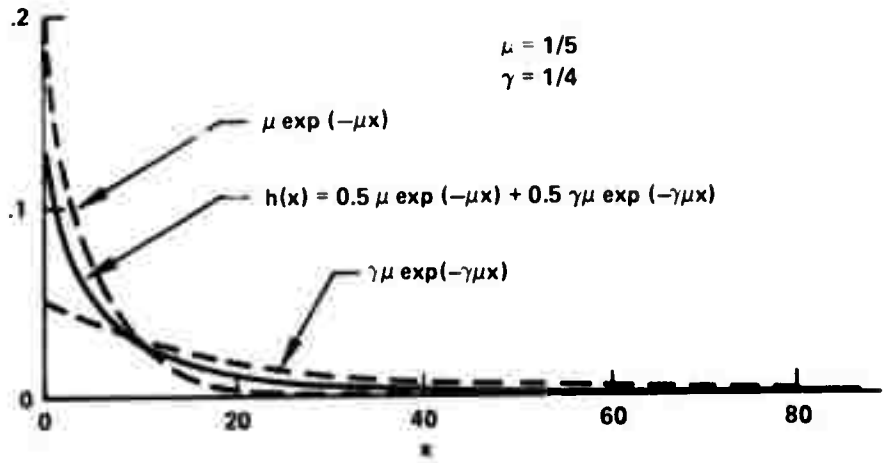
For $k = 1$, this equation reverts back to the exponential For $k = 2$, it is:

$$g(x;a,2) = 4a^2 x \exp(-2ax) \qquad 0 \leq x \tag{3.15}$$

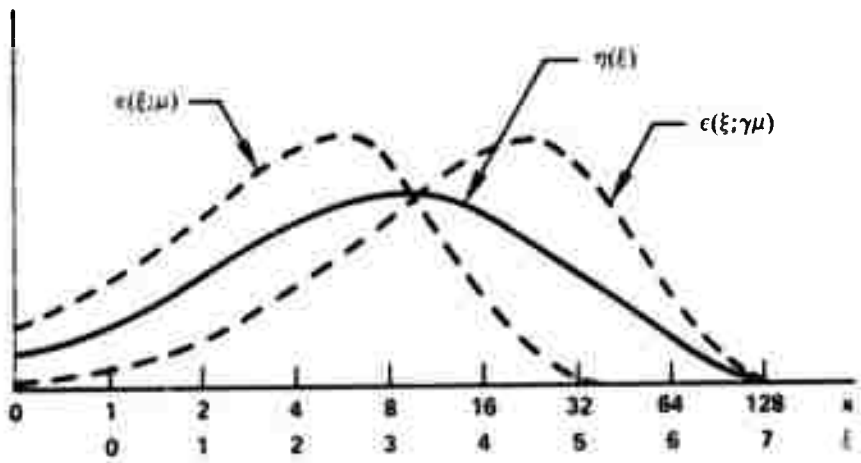
The logarithmic transformation of variables of Equation (3.2), with $k = 2$, produces:

$$\gamma(\xi;a,2) = 4a^2 \ln 2 \exp[2\xi \ln 2] \cdot \exp[-2a \exp(\xi \ln 2)] \tag{3.16}$$

Some additional algebraic manipulation and the use of the identity of Equation (3.4) result in:



(a) THE ORIGINAL TWO COMPONENT HYPEREXPONENTIAL DENSITY



(b) THE TRANSFORMED DENSITY COMPONENTS

Figure 3.2.3. The Two Component Hyperexponential Density and the Equivalent Transformed Components

$$\gamma(\xi;a,2) = 4 \ln 2 \exp[2(\ln a + \xi \ln 2) - 2 \exp(\ln a + \xi \ln 2)]$$

which can also be written as:

$$\gamma(\xi;a,2) = 4 \ln 2 \exp[2 \ln 2(\xi + \log_2 a) - 2 \exp(\ln 2(\xi + \log_2 a))]$$

If we make the substitution of variables of Equation (3.6), and convert to base two exponentials, we obtain:

$$\gamma(\psi;2) = 4 \ln 2 \left| \exp_2[\psi - \frac{1}{\ln 2} \exp_2(\psi)] \right|^2 \tag{3.17}$$

for:

$$\exp_2(y) = 2^y = \exp(y \ln 2)$$

and:

$$\psi = \xi + \log_2 a$$

We can compare this result with the exponential case of Equation (3.7), which is also a member of the Erlang family,

$$\gamma(\psi;1) = \varepsilon(\psi) = \ln 2 \exp_2[\psi - \frac{1}{\ln 2} \exp_2(\psi)]$$

Two observations can be made from the comparison. First, a generalization seems to be readily available, and secondly, the functions both have their modal points at the transformed values of the mean. The latter observation is apparent because only the exponent portion of the expression can be equal to zero, and this portion is the same in both equations.

The general case for the transformed function can be shown to be:

$$\gamma(\xi;a,k) = \frac{(k)^k \ln 2}{(k-1)!} \left| \exp[(\ln a + \xi \ln 2) - \exp(\ln a + \xi \ln 2)] \right|^k$$

or in normalized form:

$$\gamma(\psi;k) = \frac{(k)^k \ln 2}{(k-1)!} \left| \exp_2[\psi - \frac{1}{\ln 2} \exp_2(\psi)] \right|^k \tag{3.18}$$

We can find the modal point by taking the derivative:

$$\frac{d}{d\psi} \gamma(\psi;k) = k\left[\frac{(k)^k \ln 2}{(k-1)!}\right] \cdot \left\{\ldots\right\}^{k-1} \cdot \frac{d}{d\psi} \left\{ \exp_2[\psi - \frac{1}{\ln 2} \exp_2(\psi)] \right\}$$

The only part of the resulting derivative which can be equal to zero is:

$$\frac{d}{d\psi} [\psi - \frac{1}{\ln 2} \exp_2(\psi)] = 0$$

which results in:

$$\psi_m = \xi_m + \log_2 a = 0$$

or:

$$\xi_m - -\log_2 a \tag{3.19}$$

This result is a particularly interesting generalization for the entire Erlang(K) family. The mode of the transformed density will occur at the transformed equivalent of the mean, 1/a, and as seen for the exponential case, changes in the mean will merely shift the function along the ξ axis.

c. <u>The Logarithmic Transformation of the Geometric p.m.f.</u>

The geometric probability mass function is of special interest since it is the discrete analog of the exponential density, and thereby has the well known memoryless property which is so useful

and simplifying in analysis. Fortunately, these two functions are reasonable models for many real-world processes, e.g., processes in which we are interested in the time (or number of bernoulli trials) before an event will occur. In our particular application, that "event" is the occurrence of the "end of packet" terminator.

The geometric p.m.f. can be considered as the probabilities associated with a series of bernoulli trials. The trials are repeated until a success is achieved, with a probability of success equal to q on each trial. The probability of a success, preceded by n failures is the classical geometric p.m.f.;

$$p(n;q) = q(1 - q)^n \qquad n = 0,1,2,\ldots \tag{3.20}$$

In our application, the "success" is the occurrence of the special terminator symbols, which will, in all cases, be preceded by at least one word of message content. Therefore, we require that:

$$p(0;q) \equiv 0$$

so that we must renormalize the p.m.f. by:

$$p(n;q) = \frac{1}{1 - p(0;q)} [q(1 - q)^n] \qquad n = 1,2,3,\ldots$$

Since we know that $p(0;q)$ is equal to q we have:

$$p(n;q) = q(1 - q)^{n-1} \qquad n = 1,2,3,\ldots \tag{3.21}$$

This new function is merely the original geometric p.m.f. shifted to the right one integer value in n, and serves as an example of the memoryless property. That is, the conditional p.m.f., given that the length is greater than zero, is still a geometric p.m.f., as would be

the case for the conditional p.m.f. given that the length is greater than any arbitrary value, n.

The transformation which we wish to consider causes the data to be grouped into a logarithmic histogram according to the most significant bit of the binary data representation. Therefore, one interval will contain the occurrences of $n = 1$, the next will contain the occurrence of $n = 2$ or 3, the next the occurrences of $n = 4,5,6$, or 7, etc. The probabilities of each of these intervals is therefore:

$$\pi(0;q) = p(1) = q$$

$$\pi(1;q) = p(2) + p(3) = q(1 - q) + q(1 - q)^2$$

$$\pi(2;q) = p(4) + p(5) + p(6) + p(7) = q(1 - q)^3 + \ldots + q(1-q)^6$$

Each of these expressions can be considered to be a truncated geometric series, resulting in the set of equations:

$$\pi(0;q) = q$$

$$\pi(1 - q)[1 - (1 - q)^2]$$

$$\pi(2;q) = (1 - q)^3[1 - (1-q)^4]$$

or in general:

$$\pi(\nu;q) = (1 - q)^{2^{\nu}-1}[1 - (1 - q)^{2^{\nu}}] \tag{3.22}$$

For ease of computation, a recursive form of the equations is more convenient, namely:

$$\pi(\nu;q) = \pi(\nu - 1,q) \cdot (1 - q)^{2^{\nu-1}}\left[1 + (1 - q)^{2^{\nu-1}}\right] \tag{3.23}$$

The discrete function $\pi(\nu;q)$ is analogous to the continuous function,

$\varepsilon(\xi;a)$, in the previous analysis. However, in the $\pi(\nu;q)$ case, the mode is not so precisely locatable, and hence is not such a good indicator of the average value. Fortunately, there is another indicator for the discrete case, as follows:

$$\pi(0;q) = q,$$

and since we know that,

$$E[n] = \bar{n} = 1/q,$$

we therefore have:

$$\bar{n} = 1/\pi(0;q). \tag{3.24}$$

3.2.3 Approximating Density Functions

The preceding section considered log-histogram representations of density functions, and discussed a number of their properties and characteristic shapes. These histogram shape differences are both an asset and a liability, as shown in Figure 3.2.4. In this example the hyperexponential and truncated $1/x$ densities are shown to have significantly different log-histogram shapes, but differ only slightly in their uniform interval histograms,* due to the poor resolution near the origin. However, the uniform histogram has the intuitive advantage of approximately resembling the general shape of the density being approximated, while the log-histograms have shapes which are initially somewhat peculiar. However, with some experience in using them, these log-histogram shapes become readily recognizable, and are very useful

*Histograms with uniform, i.e., equally spaced, intervals will be referred to as "uniform histograms" in the sequel.

for distinguishing between different density functions such as the two shown in Figure 3.2.4. Part (c) of that figure shows two scales on the log-histogram, and leads to an alternative representation which has the best features of both types of histograms as shown in Figures 3.2.5 and 3.2.6, where these same two log-histograms are shown with a rescaled probability axis.

The rescaling is based on the probability area preserving property of the transformation of variables,

$$f(x)dx = \phi(\xi)d\xi \tag{3.25}$$

which makes quantiles of the density invariant to the transformation, and therefore allows one to transform a density function in a piece-wise fashion.* That is, the component of the $\phi(\xi)$ density function between ξ_i and ξ_{i+1} can be transformed into the equivalent component of $f(x)$ between x_i and x_{i+1}, where:

$$x_i = g(\xi_i) \tag{3.26}$$

is the appropriate transformation of variables.

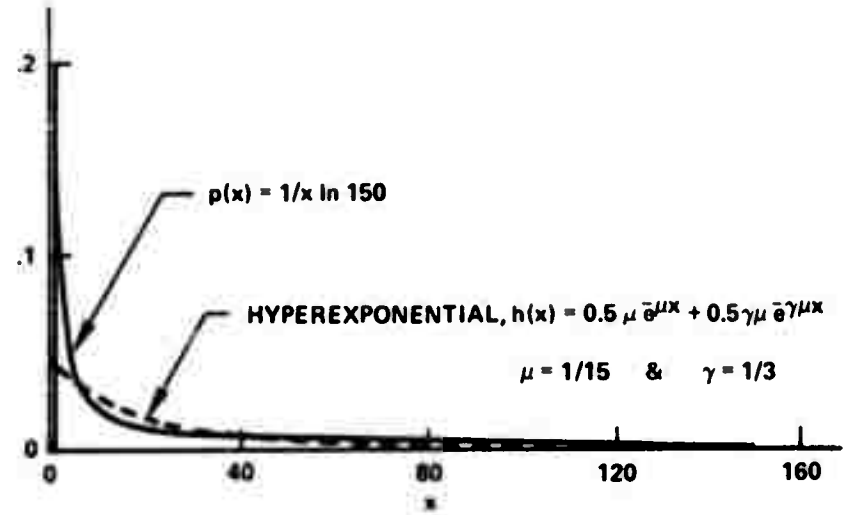
Because such a transformation must preserve the probability area, we have the discrete equivalent of Equation (3.25),

$$\pi_i(\xi_{i+1} - \xi_i) = p_i(x_{i+1} - x_i) \tag{3.27}$$

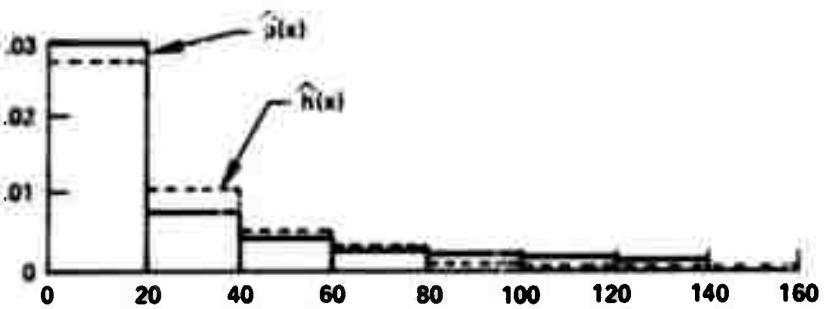
---

*The kth quantile of an arbitrary density function, $p(y)$, is the value, Y_k, for which:

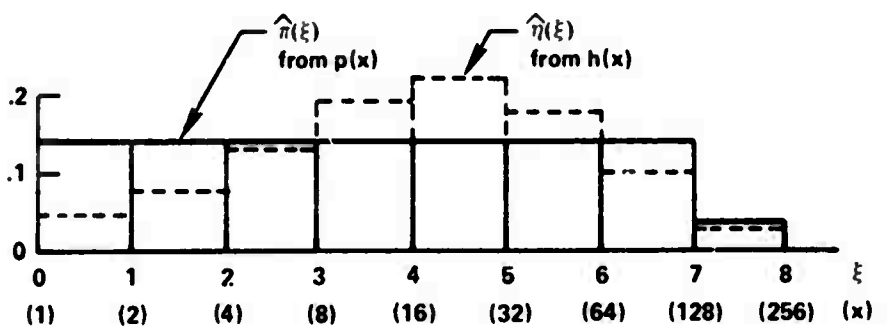
$$\int_{-\infty}^{Y_k} p(y)dy = k \qquad 0 \leq k \leq 1$$



(a) TWO DENSITY FUNCTIONS



(b) CONVENTIONAL UNIFORM INTERVAL HISTOGRAMS



(c) LOGARITHMIC HISTOGRAMS

Figure 3.2.4. A Comparison of the Hyperexponential and 1/x Densities and the Resulting Histograms

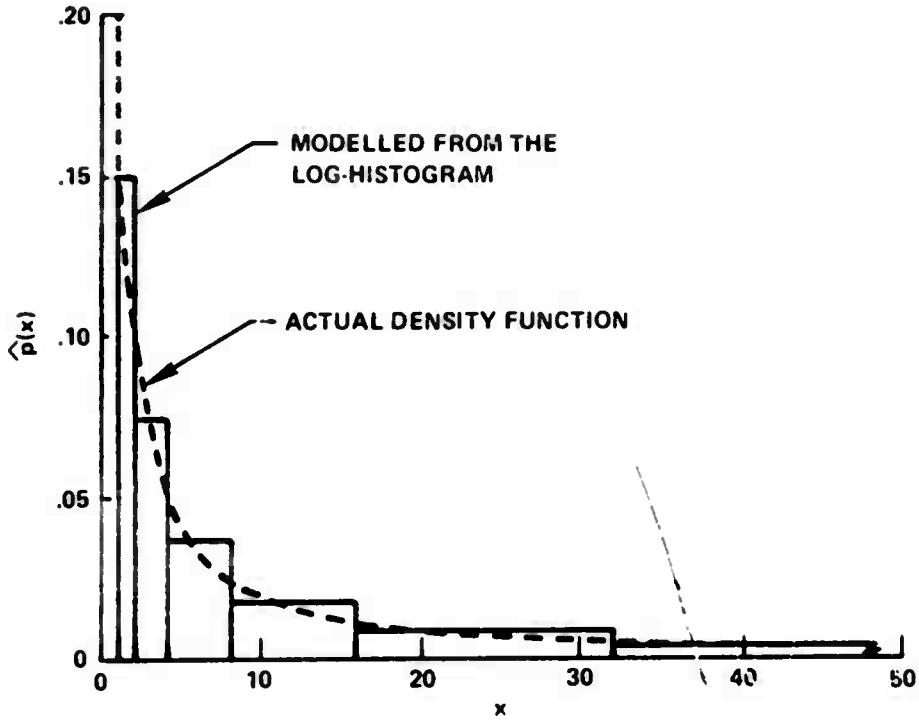


Figure 3.2.5. The 1/x Density Function and its Approximation from the Log-Histogram Data

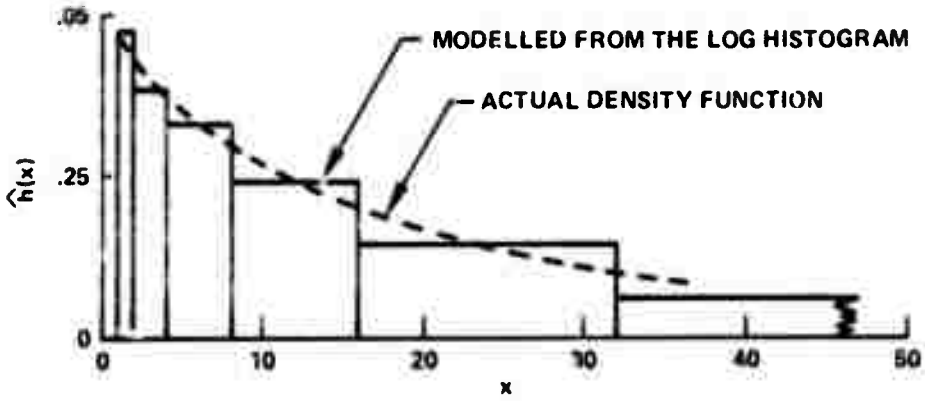


Figure 3.2.6. The Hyperexponential Density (μ = 1/15 and γ = 1/3) and its Approximation from the Log-Histogram Data

where π_i and p_i and the density values in the ith intervals of the ξ and x domains respectively. For the general log-histogram transformation of variables,

$$x_i = (b)^{\xi_i} \tag{3.28}$$

and:

$$\xi_i = i, \quad i = 0,1,2,\ldots,N-1,$$

which results in the simplification:

$$\xi_{i+1} - \xi_i = 1 \tag{3.29}$$

and the density rescaling becomes:

$$p_i = \pi_i/[(b-1)(b)^i] \tag{3.30}$$

Figures 3.2.5 and 3.2.6 show that such area transformations do indeed produce estimates of the density function which closely model the true density for the hyperexponential and truncated 1/x densities, and in Figure 3.2.7 a similar fit is noted for the Erlang(2) density. Figure 3.2.8 shows the log-histogram form of the Erlang(2) density, and one can see that the function is somewhat more peaked than that for the exponential, and that the mode corresponds to the transformed value of the mean as shown in Section 3.2.2 (b).

These examples show that for the exponential-like densities of interest, i.e., densities with most of the probability area near the origin and with a long "tail" extending over a wide range, that log-histograms provide an effective measurement technique, and require a relatively small number of histogram segments. Although no claim is

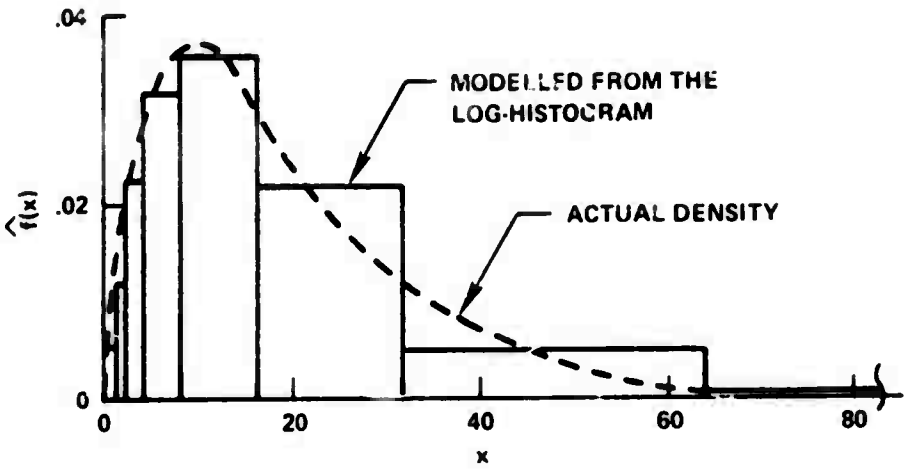


Figure 3.2.7. A Comparison of the Erlang(2) Density, ($\bar{x}$ = 20) and its Log–Histogram Approximation

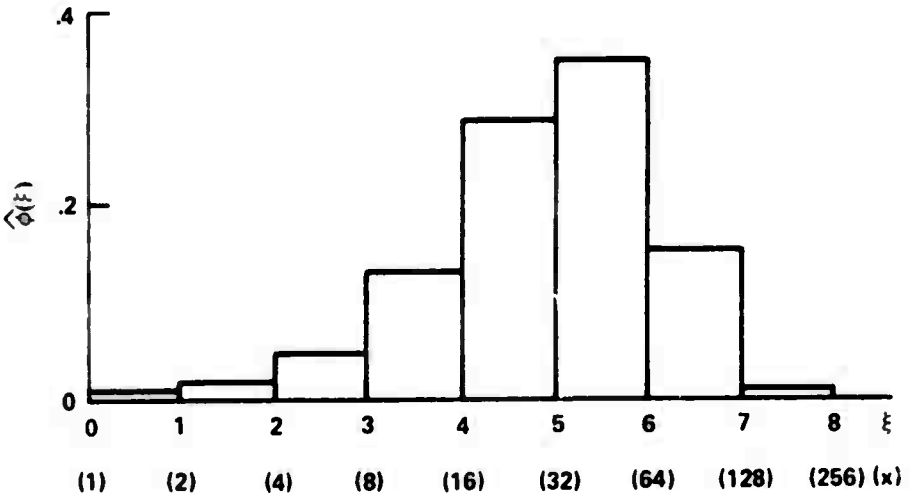


Figure 3.2.8. A Log-Histogram for the Erlang(2) Density Function

made that log-histograms are appropriate for arbitrary density functions, these examples do cover a reasonable range of skewed densities, and in the Erlang(2) case, differs considerably from the "equal interval area" criterion expressed in Section 3.2.1. For densities with some other general shape, another transformation of variable may give similar improvements, and much of the resulting analysis would be similar to that presented here and in the moment estimation aspect considered in Section 3.3.

A variation of the component-by-component transformation can be obtained if we consider each histogram interval in the ξ domain to be a uniform density segment, and transform these uniform segments into their equivalent f(x) components, as indicated in the following theorem.

Theorem 3.2.2

The uniform density segment,

$$\phi_i(\xi) = \pi_i \ , \qquad i \leq \xi < i + 1$$

is the transform equivalent of the segment:

$$f_i(x) = \pi_i / x \ln b \tag{3.31}$$

for the transformation of variables,

$$x = (b)^{\xi}$$

Proof:

The ith segment of a log-histogram has a functional form of:

$$\phi_i(\xi) = \pi_i \ , \qquad i \leq \xi < i + 1$$

such that for the transformation of variables,

$$x = (b)^{\xi}$$

and:

$$f(x)dx = \phi(\xi)d\xi$$

we have the approximating function:

$$\hat{f}_i(x) = \phi_i(\xi)\,\frac{d\xi}{dx}$$

or:

$$\hat{f}_i(x) = \pi_i / x \text{ in } b \qquad (b)^i \leq x < (b)^{i+1} \tag{3.32}$$

Thus, the density f(x) is approximated by a set of truncated 1/x density segments which are weighted by the π_i probabilities. This result proves the theorem, but unfortunately proves an asthetically poor approximation to the f(x) function due to the sharp cusps at the beginning of each density segment. The technique would be increasingly worse as the density departed further from the 1/x shape, and therefore was not seriously considered as a density modeling technique. However, a variation of the method is pursued in Section 3.3.2.2 for estimating moments.

3.3 Estimation of Moments

Theoretically all of the information about a simple random variable is given by either its density function, the cumulative distribution function, the characteristic function, or the set of all moments. The latter is infinite in number, but fortunately the higher

order moments are of rapidly decreasing interest. Indeed, for much analysis work, only the first one or two moments are of concern. For example, in some processes the expected performance of the system can be obtained from given average values of the variables, while in others, such as the queueing system considered in the Pollaczek-Khinchine equation (SA61),

$$E[n] = \lambda\bar{x} + \frac{\rho^2 + \lambda^2\sigma_s^2}{2(1 - \rho)} \tag{3.33}$$

both the first and second moments of the dependent variable are necessary to predict the average system behavior. In both cases, this analytical tractability is the basic reason that moments play such a prominent role in analysis work, rather than other sets of parameters such as quantiles. In the latter case, there is no analogous basis for combining random variables in even such a simple case as:

$$E[x + y] = E[x] + E[y]$$

due to the mathematically awkward definition of quantiles; where as previously defined, the kth quantile is at the value Y_k for which:

$$\int_{-\infty}^{Y_k} p(y)dy = k \qquad 0 \leq k \leq 1 \tag{3.34}$$

In contrast, the nth moment is defined as:

$$E[x^n] = \int_{-\infty}^{\infty} x^n \, p(x)dx \tag{3.35}$$

for a continuous density function, p(x), or in general:

$$E[x^n] = \int_{-\infty}^{\infty} x^n \, d\,P(x) \tag{3.36}$$

This latter form is the Stieltjes integral form which also includes discontinuous and discrete functions. A similar definition applies to moments about a point, a,

$$E[(x - a)^n] = \int_{-\infty}^{\infty} (x - a)^n \, d\,P(x) \tag{3.37}$$

which are the central moments for the usual case of $a = E[x]$.

3.3.1 A Comparison of Moment Estimation Techniques

The most straightforward method of estimating the moments of a population is the classical approach,

$$E[x^n] = \frac{1}{N} \sum_{N} x^n$$

This method is quite satisfactory for estimates of the mean because only the summation is necessary in that case. However, the requirement to raise each measured data value to a power for the higher moments may be infeasible in some applications. In the network measurement example, the operation of the IMP would have been degraded by the calculation necessary for even the second moment due to the lack of multiplication hardware, and due to the possibility of register overflow unless double-precision arithmetic were used. These lengthy processing requirements were not felt to be consistent with the need for real time message handling capabilities, so other techniques were investigated.

A viable alternative method to estimate the moments is the use of grouped data accumulation procedures, the best known of which is the simple uniform histogram. Other variations are the log-histogram and the quantile-gram which were intended to have more optimal interval sizes based on the equal probability area criterion. However, a better criteria for the moment estimates would be to have intervals with approximately equal $x \cdot p(x)$ or $x^2 \cdot p(x)$ values, which would typically require narrower intervals at higher values of x; which are just the opposite of the interval variations which were desired for density estimates. The desire for optimal interval sizes based on not only equal probability areas, but also equal $x\,p(x)$ and $x^2 p(x)$ components would require three separate sets of grouped data. However, we will find that for our purposes, histograms provide an appropriate compromise between these needs. The contribution of each such histogram component to the probability area and to the first and second moments of an exponential density is shown in Tables 3.3.1(a) and (b) for both uniform and log-histograms. While none of the estimation criteria are optimal, the histograms do provide a reasonable balance of the three estimation needs.*

3.3.2 Methods of Estimating Moments from Log-Histogram Data

Several different methods are possible for estimating the moments of the actual density function, $f(x)$, from the log-histogram data. For example, if we consider the transformation of variable

*The area and moment totals for the two tables are exactly the same since the component values are calculated from the moment integrals.

TABLE 3.3.1

COMPONENTS OF THE DENSITY AND FIRST AND SECOND MOMENTS FOR LOG AND UNIFORM HISTOGRAM INTERVALS (EXPONENTIAL DENSITY)

a. Log-Histogram Intervals

Interval I	$Pr[x \varepsilon I]$	$\int_I x\, p(x)dx$	$\int_I x^2 p(x)dx$
0-1	.061	.05	.5
1-2	.056	.08	.5
2-4	.104	.29	1.5
4-8	.173	1.05	5.1
8-16	.238	2.76	33.2
16-32	.233	5.31	125.7
32-64	.117	5.01	225.5
64-128	.018	1.42	114.0
Totals	1.000	15.97	506.0

b. Uniform Histogram Intervals

Interval I	$Pr[x \varepsilon I]$	$\int_I x\, p(x)dx$	$\int_I x^2 p(x)dx$
0-16	.632	4.23	40.8
16-32	.233	5.31	125.7
32-48	.085	3.26	127.5
48-64	.032	1.75	198.0
64-80	.011	.77	55.0
80-96	.004	.38	33.0
96-112	.002	.20	17.0
112-128	.001	.07	9.0
Totals	1.000	15.97	506.0

approach, we can compute the moments of the $\phi(\xi)$ function and relate the two sets of moments. Alternatively, we can transform each element of the log-histogram into an equivalent $f(x)$ elemental component, or finally, we can make a simple area transformation. In the latter case, a considerable refinement of the estimates can be made by introducing a first-order approximation to the slope of the density function over each interval. Each of these techniques shall be considered in detail in the following sections.

3.3.2.1 Moment Transformations

If data samples are accumulated in a log-histogram, we can consider the resulting information to be in the transformed or ξ domain, and can therefore estimate the moments of $\phi(\xi)$. Of course, we are actually concerned with the moments of $f(x)$, and therefore need some relation or transformation between the two sets of moments. Such a relation is shown in the corollaries to the following theorem.

Theorem 3.3.1

The moments of $f(x)$ can be related to those of $\phi(\xi)$ by:

$$E_f[x^n] = \Phi_\phi(jw = n \ln b) = E_\phi[e^{n\xi \ln b}] \tag{3.39}$$

where $\Phi_\phi(jw)$ is the characteristic function of the density, $\phi(\xi)$,

$$\Phi_\phi(jw) = \int_{-\infty}^{\infty} \exp[jw\xi]\phi(\xi)d\xi \tag{3.40}$$

and the transformation of variables has been generalized to:

$$\xi = \log_b x$$

Proof:

The nth moment of $f(x)$ is:

$$\overline{x^n} = \int_0^{\infty} x^n f(x)dx \qquad 0 \leq x$$

which for the logarithmic transformation of variables,

$$\xi = \log_b x \qquad x = \exp(\xi \ln b)$$

becomes:

$$\overline{x^n} = \int_{\xi=-\infty}^{\xi=+\infty} \{\exp(\xi \ln b)\}^n [f(x = \exp(\xi \ln b))\frac{dx}{d\xi}]d\xi$$

We can recognize the term in the brackets as $\phi(\xi)$, so we have:

$$\overline{x^n} = \int_{-\infty}^{\infty} \exp(n\ \xi \ln b)\phi(\xi)d\xi$$

The integral expression is identical to the definition of the characteristic function if we replace jw by $n \ln 2$. Therefore, we have:

$$\overline{x^n} = \Phi_{\phi}(jw = n \ln 2)$$

which completes the proof of the theorem.

The characteristic function contains all the moment information; indeed a similar development using a real variable exponent is called the moment generating function. We can modify the result of the theorem to make the moments explicit by expanding the exponential term

as shown in the following corollary.

Corollary 1

The relationship between the moments of $f(x)$ and those of $\phi(\xi)$ is:

$$E_f[x^n] = \sum_{i=0}^{\infty} \frac{(n \ln b)^i}{i!} E_\phi[\xi^i] \tag{3.42}$$

Proof:

The exponential term can be expanded in a power series,

$$\exp[n \xi \ln b] = 1 + n \xi \ln b + \frac{(n \xi \ln b)^2}{2!} + \ldots \tag{3.43}$$

which when substituted into Equation (3.39), results in:

$$\overline{x^n} = \int_{-\infty}^{\infty} [1 + (n \xi \ln 2) + \frac{(n \xi \ln 2)^2}{2!} + \ldots] \phi(\xi) d\xi$$

from which we obtain:

$$\overline{x^n} = \int_{-\infty}^{\infty} \phi(\xi) d\xi + n \ln 2 \int_{-\infty}^{\infty} \xi \phi(\xi) d\xi + \ldots$$

or

$$\overline{x^n} = 1 + n \ln 2 \, \overline{\xi} + \frac{(n \ln 2)^2}{2!} \overline{\xi^2} + \frac{(n \ln 2)^3}{3!} \overline{\xi^3} + \ldots \tag{3.44}$$

which completes the proof of the corollary.

Equation (3.44) literally says that we need to know all of the moments of $\phi(\xi)$ to calculate any one (or all) of the moments of $f(x)$. In practice, we would expect that the first few moments of $\phi(\xi)$

would be adequate to estimate the mean and variance of $f(x)$. Unfortunately, the series does not converge very rapidly, and is therefore of little general value. An example usage of the equation is shown in Appendix C, in which $\phi(\xi)$ is the uniform density function corresponding to $f(x) = 1/x \ln M$. The example demonstrates the poor convergence since even after considering the first five moments of $\phi(\xi)$ the estimate of $\bar{x}$ is still in error by 26%.

A variation of the moment expansion which results in improved convergence properties involves the central moments of $\phi(\xi)$, and is developed in the following corollary.

Corollary 2

The moments of $f(x)$ can be found in terms of the central moments of $\phi(\xi)$ by:

$$E_f[x^n] = \exp[n\,\bar{\xi}\,\ln b]\left|1 + \sum_{i=2}^{\infty} \frac{(n \ln b)^i}{i!} E_\phi[\xi - \bar{\xi})^n]\right| \qquad (3.45)$$

Proof:

If we consider the transformation of variables to be the function,

$$x = g(\xi) = \exp[\xi \ln b] \qquad (3.46)$$

and expand the function in a Taylor series about the mean,*

$$g(\xi) = g(\xi) + (\xi - \bar{\xi})g'(\bar{\xi}) + \frac{(\xi - \bar{\xi})^2}{2!} g''(\bar{\xi}) + \ldots \qquad (3.47)$$

*This approach is similar to that considered by Papoulis (PA658), pp. 151-152, in which he approximates the expected value of $q(z)$ in terms of the first few moments of the density function for z.

so that when we take expected values, we obtain:

$$\bar{x} = g(\bar{\xi}) + E_\phi[\xi - \bar{\xi}]\, g'(\bar{\xi}) + \frac{1}{2!} E_\phi[\xi - \bar{\xi})^2]\, g''(\bar{\xi}) + \ldots$$

We then have an expression for $\bar{x}$ in terms of the central moments of $\phi(\xi)$, which when we make the substitution,

$$g^{(k)}(\bar{\xi}) = (\ln 2)^k \exp(\bar{\xi} \ln 2)$$

we obtain:

$$\bar{x} = \exp(\bar{\xi} \ln 2)\{1 + \mu_2 \frac{(\ln 2)^2}{2!} + \mu_3 \frac{(\ln 2)^3}{3!} + \ldots\} \qquad (3.48)$$

where μ_k is the kth central moment of $\phi(\xi)$, e.g.,

$$\mu_1 = 0$$

$$\mu_2 = \sigma_\xi^2$$

For the exponential transformation of interest, the method can be extended to higher moments of $f(x)$ by considering:

$$x^n = [g(\xi)]^n = \exp[n\, \xi \ln 2] = g(n\, \xi) \qquad (3.49)$$

so that the Taylor series expansion about the mean becomes,

$$g(n\, \xi) = g(n\, \bar{\xi}) + n(\xi - \bar{\xi})\, g'(n\, \bar{\xi}) + \ldots$$

$$+ \frac{n^k(\xi - \bar{\xi})^k}{k!} g^{(k)}(n\, \bar{\xi}) + \ldots$$

We need to consider the derivatives a bit more carefully at this point;

$$g^{(k)}(n\, \bar{\xi}) = \frac{d^k}{d(n\, \xi)^k} \exp(n\, \xi \ln 2)\Big|_{\xi = \bar{\xi}}$$

or:

$$g^{(k)}(n\,\bar{\xi}) = (\ln 2)^k \exp(n\,\bar{\xi}\,\ln 2)$$

Therefore, the expansion becomes:

$$g(n\,\xi) = \exp(n\,\bar{\xi}\,\ln 2) + n(\xi - \bar{\xi})\,\ln 2\,\exp(n\,\bar{\xi}\,\ln 2) + \ldots$$

$$g(n\,\xi) = \exp(n\,\bar{\xi}\,\ln 2)\{1 + n\,\ln 2(\xi - \bar{\xi}) + \frac{(n\,\ln 2)^2}{2!}(\xi - \bar{\xi})^2 + \ldots\}$$

When we take expected values, we obtain:

$$\overline{x^n} = \exp(n\,\bar{\xi}\,\ln 2)\{1 + \frac{(n\,\ln 2)^2}{2!}\mu_2 + \frac{(n\,\ln 2)^3}{3!}\mu_3 + \ldots\} \qquad (3.50)$$

which completes the proof of the corollary.

The analysis also considered other forms of moment-like parameters including cumulants, which are related to the central moments by:

$$k_1 = \mu_1$$

$$k_2 = \mu_2 - (\mu_1)^2 = \sigma^2$$

$$k_3 = \mu_3 - 3\mu_2\mu_1 + 2(\mu_1)^2$$

etc.,

or in general by their respective generating functions:

$$K(t) = \ln \Phi(t) \qquad (3.51)$$

The cumulants are also known as the semi-invariants since, with the exception of k_1, they are invariant to the translation of the density along the axis. The use of cumulants in selected test case examples

resulted in convergence properties similar to that observed for the central moments, and since more viable moment estimation approaches were found, no further investigation of the moment transformation approach was pursued. These other techniques will be described in the following sections.

3.3.2.2 Superposition of Transformed Histogram Segments

We can find an alternative method of estimating the moments by an extension of the results of Theorem 3.2.2. This theorem showed that a uniformly shaped $\phi(\xi)$ corresponds to a truncated $1/x$ density function for $f(x)$, and that translating a uniform density segment along the ξ axis was equivalent to selecting a new segment of the $1/x$ density, and weighting it by the probability π_i.

If we consider a histogram in the ξ domain to be a set of uniform density components, we can then apply the theorem result to compute a corresponding set of properly scaled $1/x$ components. The composite function in the x domain would not be aesthetically pleasing due to the discontinuities and sharply peaked cusps at the beginning of each $1/x$ density segment. However, these spikes do not necessarily contain sufficient area to affect the moment estimates which are the real objective of the method. We can find the moment estimates by the following theorem.

Theorem 3.3.2

The moments of the distribution in the x domain are related to the log-histogram interval probabilities, π_i by the

relationship:

$$E_f[x^n] = \left[\frac{(b)^n - 1}{n \ln b}\right] \sum_{i=0}^{N} \pi_i (b)^{ni} \tag{3.52}$$

where b is the base of the exponential transformation, such that:

$$x = (b)^{\xi} \tag{3.53a}$$

and:

$$\pi_i = \Pr[i \leq \xi < i+1] \tag{3.53b}$$

<u>Proof</u>:

We know from Theorem 3.2.2 that for the component:

$$\hat{\phi}_i(\xi) = \pi_i \qquad i \leq \xi < i+1$$

the corresponding segment in the x domain is:

$$\hat{f}_i(x) = \pi_i / x \ln b \qquad (b)^i \leq x < (b)^{i+1} \tag{3.54}$$

This component will have a contribution to the nth moment of $f(x)$ of:

$$E_f[x^n \mid \text{the } i^{th} \text{ component}] = \int_{(b)^i}^{(b)^{i+1}} \frac{x^n \pi_i}{x \ln b} dx$$

or:

$$E_f[x^n \mid \text{the } i^{th} \text{ component}] = \frac{\pi_i}{n \ln b} \left[(b)^{ni}((b)^n - 1)\right]$$

which when all such components are considered, becomes:

$$E_f[x^n] = \sum_{i=0}^{N} \frac{\pi_i}{n \ln b} \left[(b)^{ni}((b)^n - 1)\right]$$

or:

$$E_f[x^n] = \left[\frac{(b)^n - 1}{n \ln b}\right] \sum_{i=0}^{N} \pi_i (b)^{ni}$$

which is equivalent to assuming that all of the probability mass, π_i, is located at $\xi = 0,1,2,\ldots,\xi_i,\ldots$, and correcting for this offset by the coefficient outside the summation. This completes the proof of Theorem 3.3.2, and we shall later consider some examples of this method of estimating moments, for the special case of b equal to two, i.e., the base two histograms which are most convenient with binary computers. The above theorem considered the general case since it requires little additional effort, and because it will be of value in subsequent analysis.

Theorem 3.3.2 is valid for $1 \leq \xi$ which corresponds to $1 \leq x$. Since the log-histogram method is also viable for $0 \leq x$, we need to extend the moment estimate to include this interval to be mathematically complete. (In the case of integer values of x, the only value to be added is zero, which has no contribution to the moments. However, for some of our test cases, we consider a continuum of values of x, and such values might also be found in some other application of the technique.)

Corollary 1

The interval $0 \leq x < 1$ has a contribution to the nth moment of approximately:

$$E[x^n \mid 0 \leq x < 1] \cong \frac{(b)^n - 1}{(b)^{n+1} - 1} \cdot \frac{\pi_-}{n \ln b} \tag{3.56}$$

where π_- is the probability for the corresponding interval in the ξ domain.

Proof:

We can not utilize a $1/x$ density in the region $0 \leq x < 1$ because the resulting area does not remain finite. Instead, we will assume a uniform density in the x domain for that region, and knowing that the probability components in the ξ domain are (from Equations (3.53a) and (3.53b)):

$$\pi_- = \pi_{-1} + \pi_{-2} + \pi_{-3} + \cdots$$

which to maintain a uniform density in x, must be:

$$\pi_- = \pi_{-1} + \frac{\pi_{-1}}{b} + \frac{\pi_{-1}}{b^2} + \cdots$$

or

$$\pi_- = \pi_{-1} \frac{b}{b - 1} \tag{3.57}$$

such that:

$$\pi_{-1} = \frac{b - 1}{b} \pi_- , \quad \pi_{-2} = \frac{b - 1}{b^2} \pi_- , \quad \pi_{-k} = \frac{b - 1}{b^k} \pi_- , \ \ldots$$

Since the theorem results are applicable for both positive and negative values of the index, we have:

$$E[x^n \mid \text{due to } 0 \leq x < 1] = \left[\frac{(b)^n - 1}{n \ln b}\right] \sum_{i=1}^{\infty} \pi_{-i} (b)^{-ni}$$

$$= \frac{(b)^n - 1}{n \ln b} \sum_{i=1}^{\infty} \left(\frac{b - 1}{(b)^i}\right) \pi_{-} (b)^{-ni}$$

which simplifies to:

$$E[x \mid \text{due to } 0 \leq x < 1] = \frac{(b)^n - 1}{(b)^{n+1} - 1} \cdot \frac{\pi_{-}}{n \ln b} \tag{3.58}$$

and therefore proves the corollary.

Corollary 2

The theorem result is the discrete analog of that of Theorem 3.3.1, such that:

$$\lim_{\Delta\xi \to 0} \left[\frac{(b)^n - 1}{n \ln b}\right] \sum_{i=0}^{N} \pi_i (b)^{ni} = \int_0^{\infty} \phi(\xi) e^{n\xi \ln b} d\xi \tag{3.59}$$

where $\Delta\xi$ is the histogram interval.

Proof:

The resolution of a log-histogram is inversely proportional to the base of the logarithms, since the intervals in the x domain are bounded by the values $b, b^2, b^3, \ldots$ etc. Therefore, as one allows the value of b to approach unity, the histogram intervals become arbitrarily small, and the sum approaches an integral form, with the probability element

π_i being replaced by the differential element $\phi(\xi)d\xi$. The limit then becomes:

$$\lim_{\Delta\xi\to 0} E_f[x^n] = \lim_{\substack{b\to 1\\ N\to\infty}} \left|\frac{(b)^n - 1}{n \ln b}\right| \sum_{i=0}^{N} \pi_i (b)^{ni}$$

and since one can show by L'Hospital's rule that:

$$\lim_{b\to 1}\left|\frac{(b)^n - 1}{n \ln b}\right| = 1$$

we are left with the limit:

$$\lim_{\Delta\xi\to 0} E_f[x^n] = \lim_{\substack{b\to 1\\ N\to\infty}} \sum_{\xi=0}^{N} \pi_\xi (b)^{n\xi} \tag{3.60}$$

where we have taken advantage of the result of Theorem 3.3.2 that allows one to consider all of the probability mass to be located at $\xi = i$. We can then write the exponential term as:

$$(b)^{n\xi} = e^{n\xi \ln b}$$

which we can show to be invariant to the limit process since for any given value of the variable x the value of ξ is:

$$\xi = \log_b x = \ln x/\ln b$$

such that:

$$\xi \ln b = \ln x \tag{3.61}$$

and therefore, for a given value of x, the product $\xi \ln b$ is fixed. The overall limit then becomes the integral,

$$\lim_{\Delta\xi \to 0} E_f[x^n] = \int_0^\infty \phi(\xi) e^{n\xi \ln b} d\xi \tag{3.62}$$

which is the result from Theorem 3.3.1. The analogy is further indicated by noting that the original form of the sum can be considered to be a probability generating function;

$$E[x^n] = \left[\frac{(b)^n - 1}{n \ln b}\right] \Pi(z = b^n) \tag{3.63}$$

where:

$$\Pi(z) = \sum_{k=0}^{N} \pi_k z^k \tag{3.64}$$

and this probability generating function can be considered to be the discrete equivalent of the characteristic function (or its more similarly named variation, the moment generating function).

Corollary 3

The result of Theorem 3.3.2 produces estimates of the moments which are asymptotically unbiased.

Proof:

The moment estimate from Theorem 3.3.2 was:

$$E[x^n \mid \text{log histogram}] = \left[\frac{(b)^n - 1}{n \ln b}\right] \sum_{i=0}^{N} \pi_i (b)^{ni}$$

which was shown in the limit to be equivalent to the continuous result of Theorem 3.3.1. Therefore, the limiting value of the estimate is

unbiased as the interval size decreases toward zero since:

$$E[x^n \mid \text{log-histogram}] - E[x^n]$$

Corollary 4

The result of Theorem 3.3.2 produces a consistent estimator of the mean.

Proof:

The factors, π_i, should actually be considered to be estimates of the probabilities, and will be therefore indicated as $\hat{\pi}_i$, the observed relative frequency of events occurring in the interval $i \leq \xi < i+1$. Since the coefficient outside the summation is a deterministic factor, as are the $(b)^{ni}$ terms, we can consider the estimate of the mean to be a weighted average of other estimated values, namely the $\hat{\pi}_i$'s. Each of the $\hat{\pi}_i$ estimates is known to be a consistent estimator of π_i, so we must show that the weighted average of a set of consistent estimators, is itself consistent. This can be easily shown if the components of the set are independent, since if we have the random variables u, v, and w such that:

$$w = a\,u + b\,v$$

then:

$$E[w] = a\,E[u] + b\,E[v]$$

and assuming independence of u and v,

$$\sigma_w^2 = a\,\sigma_u^2 + b\,\sigma_v^2$$

Of a total of η samples, η_u will contribute to $\hat{u}$ and the

remaining η_v will contribute to $\hat{v}$, in such a manner that as $\eta \to \infty$ both η_u and η_v will also increase without bound, so that:

$$\lim_{\eta \to \infty} \frac{\sigma_w^2}{\eta} = 0$$

which proves that the estimator is not only unbiased, but also approaches the true value of the population mean with unity probability, and is therefore a consistent estimator.

3.3.2.3 Moment Estimates from the Approximated $f(x)$ Function

The density function approximations of Section 3.2.3, which converted rectangular segments of $\hat{\phi}(\xi)$ into rescaled rectangular segments of $\hat{f}(x)$, can also be utilized for moment estimates. From Equation (3.30) we know that this rescaling function is,

$$P_i = \pi_i / [(b-1)(b)^i]$$

and therefore, moment estimates can be obtained by:

$$\hat{E}[x^n] = \sum_{i=0}^{N-1} \left\{ \int_{(b)^i}^{(b)^{i+1}} x^n p_i dx \right\} \tag{3.65}$$

where the caret indicates that it is an estimate of the expected value. Note that this estimate is not in general the same as assuming that all of the probability mass is concentrated at the center of the interval, which generates estimates of:

$$\hat{E}_2[x^n] = \sum_{i=0}^{N-1} \frac{(b+1)(b)^i}{2}^n (b-1)(b)^i p_i \tag{3.66}$$

The assumption that the probability is uniformly spread across each interval is typically closer to the actual density function being modeled than is the concentrated mass assumption. However, an even better density approximation can be formed if additional information can be obtained concerning the distribution of the probability within each interval. Such information is often available in the relative probabilities of the adjacent intervals as shown in Figure 3.3.1, and

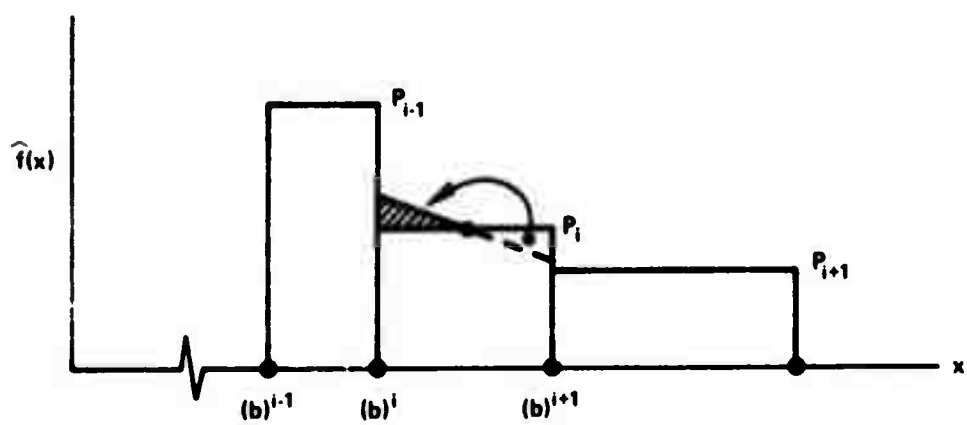


Figure 3.3.1 A First-Order Approximation to the Probability Distribution Across a Histogram Interval

can be utilized to improve the moment estimates by creating a first-order approximation to the slope of the density function over each interval. Instead of assuming that the density is uniform over the ith interval, it would then be assumed to be linearly distributed according

to the function:

$$\hat{f}_i(x) = m_i x + B_i, \qquad (b)^i \leq x < (b)^{i+1} \tag{3.67}$$

with

$$\hat{f}_i(\text{interval midpoint}) = p_i \tag{3.68}$$

The probability area of the interval is maintained automatically for the first-order approximation, as indicated in Figure 3.3.1.

The analysis of such first-order corrections is fairly lengthy, and therefore has been relegated to Appendix D. This analysis includes; (1) a general approach to the problem, (2) the selection of appropriate adjacent slope weighting functions, and (3) the application of the method to both uniform and log-histograms. Some example applications will be shown in Section 3.3.2.4 to indicate the effect of such a first-order correction, and to compare moment estimates from uniform and log-histograms.

3.3.2.4 A Comparison of Moment Estimates from Uniform and Log-Histograms

A series of Monte Carlo tests were made to evaluate the moment estimation capabilities of uniform and log-histograms, and indicated surprisingly comparable results from the two techniques. The log-histograms were expected to fare rather poorly in such a comparison due to the fact that the log-scale intervals are relatively large at the upper end of the scale, and the moment estimates are quite sensitive to this region. However, the first-order correction techniques, as developed in Section 3.3.2.3, were found to be quite effective in

reducing the errors in the estimates, and were applicable to a variety of density function shapes.

The tests considered five different densities; the uniform, Erlang(2), exponential, hyperexponential, and truncated 1/x density functions. Random variables in the range of 0 to 1 were generated by a computer subroutine and were transformed to cover the range of 0 to 500 with the desired density by a transformation of variable. The resulting random variables were sorted into uniform and log-histograms, with 10 intervals each, and estimates of the mean, meansquare, and variance were then calculated from the histogram data. The actual sample moments were also accumulated during the tests and were utilized to calculate the estimate errors for each test. These results have been summarized in Table 3.3.2, but are presented in detail in Appendix E along with a more complete description of the test procedures.

3.3.3 Some Guidelines for the Measurement of Moments

The only moments of interest are typically the mean, meansquare, and variance, so we shall only consider these three moments, although many of the considerations and limitations could apply to higher moments as well. There are several possible alternatives for gathering moment estimates including:

- collecting raw data samples
- calculating the moments at the data source
- compacting the data into histograms or other grouped data formats.

TABLE 3.3.2

A SUMMARIZATION OF THE MONTE CARLO TEST RESULTS

Density Function	Sample Mean/ Variance	Uniform Histogram Error		Log-Histogram Error	
		Mean	Variance	Mean	Variance
Uniform	247 21,300	+0.3%	+ 7.0%	+1.0%	+ 2.8%
Erlang (2)	82.0 3,280	-1.4%	+20.3%	-0.7%	-35.0%
Exponential	80.7 6,090	+2.3%	+ 8.0%	-0.6%	- 5.4%
Hyper-exponential	80.8 7,310	+2.5%	+ 4.3%	+0.3%	- 4.8%
Truncated 1/x	80.9 14,100	+9.0%	- 4.0%	+1.6%	+ 4.1%

These techniques are listed in order of their preference from a statistical viewpoint, but unfortunately several practical constraints limit the applicability of the first two. For example, to calculate a mean value to within 2% to 3% accuracy with a 90% confidence level requires about 1000 samples, and such large samples can require an excessive amount of the buffering and transmission facilities of a network. In contrast, we obtained about this same accuracy in the Monte Carlo tests of Section 3.3.2.4, but only transmitted the ten histogram slot values. This reduction was possible due to the compacting of the data at the source. Such compaction could further reduce the transmission requirements by directly computing the moments, but for other than the mean, the squaring and multiple precision operations (to avoid

overflow) are too time consuming in most small communications control computers.

The accumulated statistics measurement facility as described in Section 2.5.1 provides some practical examples of the trade-offs between these moment estimating techniques. A primary performance measure of the net is message delay, and such delays are recorded by running totals of messages sent, and time for the round trip (message out and RFNM back). Higher moment data would have been very desirable rather than having only the average values, but the multiplicity of destinations made it impractical to keep separate histograms on each, and a composite histogram would be virtually meaningless. The selection of an appropriate set of histogram intervals would have been interesting due to the open ended range of delay values, and the lack of any round trip times of less than about eight milliseconds. A suitable set of histogram intervals might have been obtained by dividing all delay values by eight (a simple shift operation) and considering the intervals to be 0 to 8, 8 to 16, 16 to 32, etc. up to a "catch-all" interval such as "> 256". However, the need for seven such intervals for each of 32 destinations would increase the measurement load considerably.

No precise statement can be made regarding the accuracy requirement for an open ended set of experiments, although some reasonableness guidelines can be stated based on the natural units of the system such as the packet length, the time to service an average message, and the fixed service time due to the packet overhead. Queueing delays will be of the order of one to five times the average service time (under test conditions such as described in Chapter 4), and

should be measured with a resolution of at least 10% to 20% of this average service time. Higher resolution is, of course, very desirable but must be balanced against the possibility of overflow (for short word lengths) or the need for excessive measurement data transmission.

The allowable error in the second moment will depend on the experiment being conducted, but in the queueing delay example, its effect can be seen in the Pollaczek-Khinchine formula (SA61):

$$E[\text{time in the system}] = \bar{x} + \frac{\rho}{1-\rho} \cdot \frac{\overline{x^2}}{2\bar{x}} \tag{3.69}$$

where:

$$\rho = \lambda\bar{x}$$

For a traffic intensity of about ρ equal to 0.5 and for exponential-like service times, the two terms will be of comparable magnitude, and therefore should have similar accuracies. If we consider only the $\bar{x}$ and $\overline{x^2}$ error effects, and write the estimates as the actual value plus an error term, we have:

$$\hat{E}[x] = (1 + \varepsilon_1)\bar{x} \tag{3.70a}$$

$$\hat{E}[x^2] = (1 + \varepsilon_2)\overline{x^2} \tag{3.70b}$$

$$\hat{\rho} = \lambda(1 + \varepsilon_1)\bar{x} \tag{3.70c}$$

The queueing delay term of Equation (3.69) then becomes:

$$\frac{\hat{\rho}}{1-\hat{\rho}} \cdot \frac{\hat{E}[x^2]}{2\hat{E}[x]} = \frac{\lambda(1+\varepsilon_1)\bar{x}}{1-\lambda(1+\varepsilon_1)\bar{x}} \cdot \frac{(1+\varepsilon_2)\overline{x^2}}{2(1+\varepsilon_1)\bar{x}} \tag{3.71}$$

or:

$$\hat{E}[\text{queue delay}] = \frac{(1 + \varepsilon_2)\lambda\, \overline{x^2}}{2[1 - \lambda(1 + \varepsilon_1)\overline{x}]} \tag{3.72}$$

For small values of ρ this estimate is approximately,

$$\hat{E}[\text{queue delay} \mid \text{small } \rho] \cong (1 + \varepsilon_2)\, \frac{\lambda\, \overline{x^2}}{2} \tag{3.73}$$

while for values of ρ near 0.5, the estimate becomes:

$$\hat{E}[\text{queue delay} \mid \rho \approx 0.5] \cong \frac{1 + \varepsilon_2}{1 - \varepsilon_1}\, (\lambda\, \overline{x^2})$$

or approximately:

$$\hat{E}[\text{queue delay} \mid \rho \approx 0.5] \cong (1 + \varepsilon_1 + \varepsilon_2)\lambda\, \overline{x^2} \tag{3.74}$$

Therefore, the fractional, or percentage errors, in the estimates of $\overline{x}$ and $\overline{x^2}$ are of comparable significance in determining the queueing delay estimate accuracies.

The variance can be calculated from these moments about the origin by:

$$\sigma^2 = E[x^2] - (E[x])^2 \tag{3.75}$$

and similarly, an estimate of the variance can be obtained from estimates of these moments by:

$$\hat{\sigma}^2 = \hat{E}[x^2] - (\hat{E}[x])^2 \tag{3.76}$$

If we subtract Equation (3.75) from Equation (3.76) we obtain an expression for the error in the variance estimate,

$$er(\sigma^2) = \hat{\sigma}^2 - \sigma^2$$

or:

$$er(\sigma^2) = \{\hat{E}[x^2] - E[x^2]\} - \{(\hat{E}[x])^2 - (E[x])^2\} \tag{3.77}$$

which can be written as:

$$er(\sigma^2) = er(\overline{x^2}) - \{(\hat{E}[x] + E[x]) \cdot er(\bar{x})\}$$

For most purposes, this expression can be approximated by,

$$er(\sigma^2) \cong er(\overline{x^2}) - 2\,\bar{x}\,er(\bar{x}) \tag{3.78}$$

which relates the error in the variance estimate to the errors in the estimates of the mean and meansquare values. Note that if the latter two errors are of the same sign, they will tend to cancel in the variance estimate, and this is precisely the situation that occurred in the exponential density example of the Monte Carlo test described in Section 3.3.2.4 and Appendix E.

The square root of the variance is the standard deviation, which is dimensionally the same as the mean and indicates the spread or dispersion of the density function. Intuitively, the accuracy of the standard deviation should be comparable to that of the mean, and could be represented by,

$$\hat{\sigma} \cong (1 + \varepsilon_1)\sigma \tag{3.79}$$

for which the corresponding error in the variance becomes:

$$\hat{\sigma}^2 = (\hat{\sigma})^2 \cong (1 + 2\varepsilon_1)\sigma^2 \tag{3.80}$$

Therefore, while the errors in the meansquare and mean values should be comparable, the allowable error in the variance can be twice this value. This relationship can also be seen in the Pollaczek-Khinchine formula of Equation (3.69) which can be written as:

$$E[\text{time in system}] = \bar{x} + \frac{\lambda}{2(1-\rho)} [(\bar{x})^2 + \sigma^2] \tag{3.81}$$

and clearly shows that the percentage errors in the mean and standard deviation are of comparable significance. The moment estimates from the Monte Carlo tests of Section 3.3.2.4 indicate the second moment estimates from histograms will be somewhat less accurate than the estimates of the mean, and will therefore be the more significant source of error in such calculations.

3.4 Artifact Considerations and Corrections

Artifact in measurement data refers to any effect that results in a difference between the actual system variables (as they would exist if the measurements had not been made), and the observed system variables obtained from the measurement. Such artifact can occur either by measurement effects which change the actual system behavior, such as introducing additional delays, or from effects which distort the measured results without actually changing the system behavior. An example of the latter type of "apparent artifact" would occur if, in a hardware monitor, the sampling period is inadvertently made in synchronism with some time varying behavior, and thereby does not record a true picture of the behavior. Both types of artifact will be found to occur in the network measurements as discussed in the following paragraphs.

There are two basic aspects of the network artifact; (1) that introduced in the IMP to generate the statistics data, and (2) that introduced in the transmission facilities to get the measurement results to the NMC. We shall consider each of these areas with regard to the identification of the artifact conditions, the possible ways of correcting for the artifact, the ways in which the artifact was reduced, and some suggestions on how subsequent implementation might further reduce the artifact.

3.4.1 The Use of IMP Resources to Generate the Measurements

The measurement programs reside within the IMP core memory and compete for the IMP processing resources to generate the statistics data. Since many of the IMP processing functions require real-time input data handling (e.g., setting up a new buffer before the next packet starts to arrive) the measurement functions have been designed to minimize their interference with the input operations. Except for the trace and arrival statistics, the measurements concentrate on the modem channel output functions to record the packet size data, etc. (The HOST-IMP data transfers do not have such time constraints, since they are bit asynchronous, and therefore measurement data are taken in both directions between the IMP and HOST.) The modem output test data also includes the effects of the measurement message lengths, which will be considered as part of transmission artifact.

The snap-shot statistics are intended to capture the state vector of the IMP queues and routing tables, but have been assigned a priority below the interrupt handling functions to also avoid

interference with the real-time routines. Therefore, the higher priority task queues will always be empty when the background statistics program is executed, and similarly the background program may be interrupted to handle subsequent higher priority functions. As a result, the snap-shot data may be skewed in time, and will not necessarily be completely consistent. For example, the total number of store-and-forward buffers may differ from that reported in the queue length measurements, if a lengthy interruption has occurred between the time that these two items are recorded. Such skewed results have been observed, but typically occur only when very heavy traffic is introduced into the IMP.

The need for core memory space for the measurement routines introduces an artifact in any blocking effect measurements, since if that same space had been utilized for buffers, the blocking might not have occurred. However, this effect would only be of concern if blocking occurred frequently, and otherwise the information gained on the system performance is well worth the cost. In the case of the IMP, approximately 750 16-bit words are utilized for the measurement routines, which would correspond to about ten additional buffers beyond the present total of seventy (HE70).

The combination of the above mentioned decisions and design compromises, effectively reduced the IMP resource artifact to a very small effect. Unfortunately, the transmission artifact is not nearly as negligible, and requires considerable care in defining experiments and in correcting the artifact distortions of the data.

3.4.2 The Use of the Network Transmission Resources for the Measurement Data

The measurement data messages tend to be fairly long and occur at frequencies which can utilize considerable transmission bandwidth as shown in Table 3.4.1. These data messages have been compreseed

TABLE 3.4.1

TRANSMISSION REQUIREMENTS FOR MEASUREMENT DATA

Measurement	Approximate Data Message Length	Period	Typ. BW[6]
Traces	80 + 160 N bits[1]	as generated	0.4 Kb/sec.[2]
Snap-Shots	2200 bits	0.82 sec.	2.7 Kb/sec.
Arrival Times	64 + 16 M bits[3]	see note [4]	0.34 Kb/sec.[5]
Accum. Data	3120 bits	12.8 sec.	0.25 Kb/sec.

1. N is the number of trace blocks in a trace message.
2. Based on N = 2 and sent once per second.
3. M is the number of arrival times recorded per message.
4. Sent either every 1.6 sec., when 60 arrivals occur, or when another statistics program must be run, whichever occurs first.
5. Based on M = 30 and sent every 1.6 seconds.
6. Peak bandwidth requirements are much larger due to the burst nature of the measurement traffic.

in size as much as possible by utilizing the IMP CPU to compact the data into counts, averages, histograms, etc., but there are a large variety of IMP functions to record and the resulting test data messages are up to four packets in length. These data messages can affect the network performance by the extra traffic load, longer queues, etc., and in some cases, require corrections to the recorded values. For example, each accumulated statistics message includes the previously sent statistics message in the modem channel traffic, but being of known length, it can be subtracted out to correct the measurements. However, if this was one of several messages sent to the NMC, the round trip times could only be partially corrected based on known average data message delays.

The snap-shot statistics are particularly expensive in terms of transmission requirements due to their length and frequency, but are taken at rather gross time increments relative to the "time constant" of the network. Some further data compaction within the IMP might be possible, but is not readily definable due to the changing measurement needs, e.g., for peak queue lengths in one experiment versus average queue lengths in another. An additional artifact involved with the snap-shots is due to the fact that they are taken in approximate synchronism across the net, and therefore introduce traffic in bursts. Of course, the snap-shots are only generated at those IMP's which have had the appropriate flags set for the experiment. (This is true for each of the different measurement facilities as described in Section 2.5.)

The methods of artifact reduction which Estrin, et al. described (ES67A) are applicable candidate techniques if one were to

reduce the transmission artifact of the network measurements. These methods are paraphrased below:

1. make measurements on gross events rather than detailed operations.
2. selective injection of measurement "probes" at key points of interest.
3. use of external measurement devices such as a hardware monitor.

The first technique was somewhat orthogonal to the measurement intent of obtaining detailed measurement data to provide insight into the network behavior, but was utilized in the sense that compacted data was taken when appropriate. The selectivity of the second technique has varying degrees, and in the case of the network involved enabling any subset of the five basic measurement routines, (i.e., the accumulated statistics, snap-shots, traces, arrival times, and status reports) at the IMP's involved in an experiment. Further selectivity could have reduced the transmission requirements somewhat, but only if a clear-cut subdivision could be defined within the data sets. The choice of having fixed or variable data message formats was decided in favor of fixed formats due to the complications in the IMP measurement code if it were made variable. However, this choice would have to be reconsidered if the net changed radically in the number of nodes or their interconnectivity.

The third method, i.e., the introduction of special measurement related devices, would have considerable significance in the

resulting measurement capabilities. For example, the addition of a magnetic tape recorder at each IMP could provide a storage facility for measurement data that would allow more thorough measurements and would avoid the transmission artifact during the experiment. The data would be transmitted after the test, at some off hour, or by mail if the quantity of data was very large. Several interesting tests could be run if such equipment were available,* such as blocking effect measurements which are presently thwarted by the blockage itself. Simultaneous measurements could also be taken at more nodes of the network, where presently the bandwidth limitations require fairly austere experiment planning.

No attempt has been made in these discussions of artifact to identify each and every source of artifact, but merely to point out several typical cases in which such artifact occurs, and how it might be corrected, or in other designs, eliminated. Each experiment must carefully consider the artifact which is introduced for that combination of traffic and measurement data, and attempt to correct, or at least identify, such sources of error.

3.5 Data Analysis

While our basic measurement philosophy was in complete agreement with Hamming's observation that "the purpose of measurement is insight, not numbers"; one must also admit that the direct output of measurement is numbers, and insight must come from analysis and

*HOST cooperation in the measurement activity could provide an alternative source of local storage.

reduction of these numeric results. Unfortunately, it is difficult to develop algorithms for providing insight from a given set of data values, and therefore much of the data analysis requires human intervention, at least until a hypothesis has been formulated on some aspect of the network behavior of interest. Due to this need for human observation of the data, our experiments have been run utilizing line printer records (at the UCLA Network Measurement Center HOST), for the various measurement data from each experiment.

These records were initially printed in a hexadecimal output form, and shortly thereafter in a more readable decimal form. However, these records were very difficult to scan for particular items of interest, so special print formats were developed which provided several advantages including appropriate annotation, a more convenient arrangement of the data, and some simple calculations and bar charts. Such print-outs were shown in the figures of Section 2.5, and have been found to be quite adequate for both the rapid scanning of the results during the data collection phase of an experiment, and for subsequent data analysis and reduction.

The data analysis activities will certainly evolve from this initial manual mode of operation to more sophisticated computer-aided techniques. The first step of this change is presently being implemented as the data collection routines are being integrated into the UCLA time-sharing system. This change allows the experimenter to analyze his data as it is obtained, and to take advantage of the other system facilities for this purpose; a significant improvement over the stand-alone operation in which the Sigma 7 computer was dedicated to

the data collection phase of experiments. Future epochs of this evolution will include the utilization of other network resources for data reduction retrieval, and display, and perhaps may also result in machine-aided perception or insight through the use of artificial intelligence programs in the network.

CHAPTER 4

MEASUREMENTS OF NETWORK PERFORMANCE

The set of measurement tools described in Chapter 2 and the analysis techniques of Chapter 3 can best be illustrated by actual network measurement situations. In this chapter we will consider some example measurements to further demonstrate the utility of the measurements (e.g., to simulate a prescribed level of traffic and measure the resulting delays) and for exploratory tests to answer selected "what if" questions such as, "What happens to the routing mechanism if the traffic load becomes extremely heavy?" These tests will therefore be oriented toward determining what questions should be asked regarding the network behavior. In contrast, the measurements which will be considered in Chapter 5 will be based on the verification of analytic models of the system behavior. Actually these two measurement functions operate together in an iterative manner, with the exploratory measurements resulting in the formulation of models, and subsequent tests being designed to test the validity of the models. These latter tests will typically provide some additional insights which can be utilized to improve the model, and then additional verification tests would be run, etc.

The motivation for these test and model development efforts is to gain insight into the system behavior, and in particular, to determine the "sensitive" areas, i.e., those areas in which small parameter variation can result in large performance changes. The intent

of Chapters 4 and 5 is to provide some examples of this measurement-modeling technique to a variety of problem areas, some of which will be seen to have definite areas of parameter sensitivity, while others will be relatively insensitive to change. Both types of conclusions provide useful information regarding the system behavior, although the former is certainly more gratifying.

4.1 Message Delay Measurements

Message delay is a primary performance measure in a computer network due to its contribution to the overall response time of an interactive message, although like "response time", it is subject to a variety of definitions. For example, in the NPL network described in Section 1.2, the component of the delay due to the network was defined as being the time from the receipt of the last bit of a packet at the source node, to the start of the output transmission at the destination. Variations of the definition are possible, such as by considering the latter time to be the end of transmission at the destination, or alternatively, to consider definitions based on message delays rather than packet delays. Most such definitions differ by reasonably constant terms, and therefore can be compared in a relatively simple manner if a precise definition of each measurement is given in terms of its starting and ending points in time. A potential problem arises because these starting and ending events occur at different sites, i.e., the source and destination nodes. Unless each site has an accurate and synchronized clock, the time difference will be meaningless, and while such synchronization techngiues are feasible, the ARPA network delay measurements were based on time values at a single node by utilizing

the notion of round trip times of messages and their associated RFNM's (Request For Next Message). The component delays involved in this definition are shown in Figure 4.1.1 for the case of two adjacent nodes, with more separated nodes requiring additional store-and-forward queueing delays and transmission times. The delays of the RFNM return will be small due to their short length, and reasonably predictable due to their high priority, so that the return delay can be subtracted from the total round trip time to give a good estimate of the message transmission delay. Unless otherwise specified, all of the following delay measurements will be based on the total round trip delay.

4.1.1 Delays Due to Store-and-Forward Handling

The example utilized in Section 2.5.1 to describe the accumulated statistics functions, demonstrates the effect of additional store-and-forward operations between the source and destination sites. However, that particular example was not felt to reflect the normal topology because of the unexpected ratio in which the traffic was split between the paths via the SRI and RAND nodes. The test was repeated at a later date and resulted in the data shown in Table 4.1.1, which met the expected traffic distribution and round trip delays. Closer inspection of the previous test data indicated that the BBN-MIT channel was inoperative and therefore resulted in an extra store-and-forward delay for the MIT, Lincoln Laboratories, and Case destinations.

The round trip delays consist of two major components; (1) the serial transmission requirements of the message, and (2) the propagation delays which occur twice in the round trip measurement. The artificial traffic messages being sent had an average length of

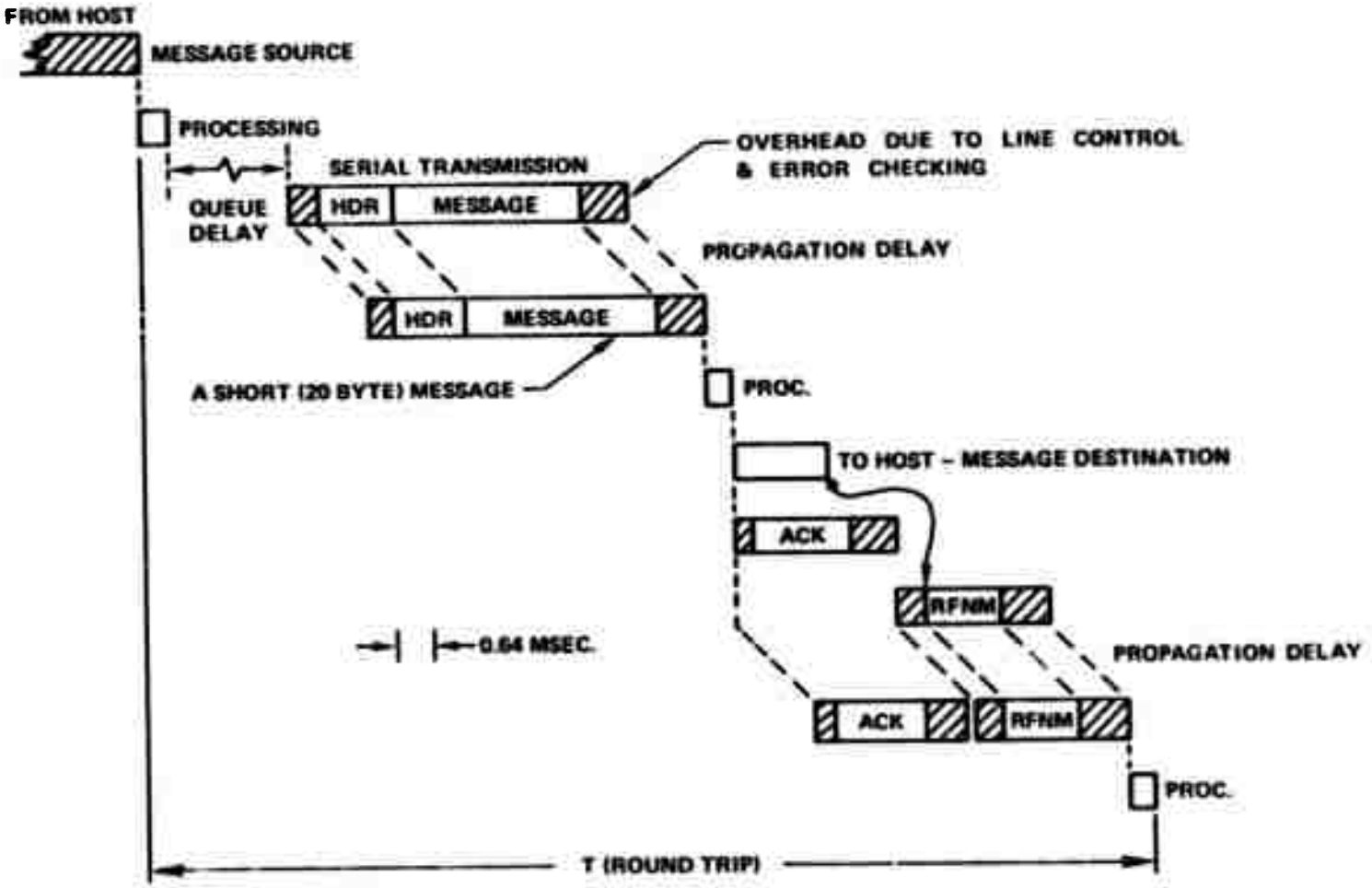


Figure 4.1.1 A Detailed Picture of the Delays Involved in a UCLA-RAND Message Transmission

TABLE 4.1.1

MEASURED ROUND TRIP TRANSMISSION TIMES
FROM UCLA TO SEVERAL DIFFERENT SITES

Destination	No. of S & F Transmissions	Measured Average Round Trip Time msec.	S & F Path from UCLA
UCSB	1	22.4 msec.	direct
Utah	2	53.2 msec.	via SRI
MIT	3	94.0 msec.	via RAND
Lincoln Labs	4	114.0 msec.	via RAND
Case	5	137.5 msec.	via RAND

320 bits, which require a total of about 12.0 msec. for serial transmission, when one includes the overhead characters and the RFNM. The propagation delay is about 10 μsec./mile (HE70),* resulting in a cross-country delay of approximately 30 msec. Therefore, the round trip time from UCLA to MIT would be expected to require 36 msec. for the three store-and-forward transmissions and 60 msec. for the propagation delays (both ways), for a total of about 96 msec. The observed round trip delay is shown in Table 4.1.1 to be 94 msec. The round trip time to Case Institute of Technology is the largest of the values shown due to the five store-and-forward delays and the fact that the path from UCLA to Case goes via Boston, which adds even more propagation delay. (Such seemingly peculiar topological aspects are due to the evolving nature of the network, with only a subset of the communication net being presently installed.) In each instance the delays were quite acceptable

*As compared to 5.4 μsec./mile for the speed of light in free space.

from a user interaction point of view, although of course no queueing delays were involved in these transmissions (due to the lack of any competing traffic).

4.1.2 Delay as a Function of Message Length

Message length variations influence the round trip times in two ways; (1) by the serial transmission delays, and (2) by the effect of the message service requirement on the queueing delays. This latter component of delay will be considered in Section 4.1.3, and for now we will only be concerned with the serial transmission delay. In the description of the previous experiment this component of delay was approximately 12.0 msec. for a 320 bit message, of which 6.4 msec. is required for the transmission of the average size 320 bit message (at 20 μsec./bit or 50 Kbits/sec.), 2.7 msec. is required for the header and line control characters, and an additional 2.7 msec. is for the RFNM. These values do not add to 12.0 msec., since this latter figure includes the fact that the message lengths are random variables, and some messages will exceed one packet in length, and therefore will require additional header and line control overhead, with the expected number of packets per message (for the exponential message length case) being:

$$E[\#\text{ pkts. per mes.}] = 1/[1 - e^{-L_s/\bar{\ell}_m}] \qquad (4.1)$$

where L_s is the segment or packet length, and $\bar{\ell}_m$ is the average message length. (This equation will be developed later in Section 5.2.3.1). When this effect is considered, we obtain the expected

message service times shown in Table 4.1.2.* These values do not include the RFNM delay, since it is not appropriate for some of our later use of these delay components, such as the following queueing delay analysis.

TABLE 4.1.2

EXPECTED SERVICE TIMES FOR VARIOUS LENGTH PARAMETER VALUES OF RANDOM ARTIFICIAL TRAFFIC GENERATOR

Avg. Mes. Length from Sigma 7 Gen. (32-bit words)	Avg. Mes. Length in the IMP (bits)	E[# pkts] per message	E[# bits per mes. incl. pkt. overhead]	E[mes. service time] (msec.)
1	32	1.00	168	3.4
2	64	1.00	200	4.0
5	160	1.00	296	5.9
10	320	1.05	463	9.3
15	480	1.14	635	12.7
20	640	1.27	813	16.3
25	800	1.40	990	19.7
30	960	1.54	1170	23.4

4.1.3 The Queueing Component of Message Delay

If several message sources are in contention for the service facility, queues may form with resulting increases in the average message delay. Such queueing behavior is a function of the average message service requirement, $\bar{x}$, and the average arrival rate of

*For example, Table 4.1.2 indicates an average message service time of 9.3 msec. for a message with an expected length of 320 bits. If we add the 2.7 msec. service for the RFNM, we obtain the 12.0 msec. delay value used in the above discussion.

messages, λ, with these two factors determining the server utilization parameter, ρ, by the relationship:

$$\rho = \lambda \bar{x} \tag{4.2}$$

The message service requirement is a function of the message length as described earlier, and the average arrival rate can be controlled for experimental purposes by the selection of the two other parameters of the artificial traffic generator, i.e., the "transmission attempt interval", T_a, and the number of message generators being utilized. (The RFNM mechanism for congestion control is the reason for considering T_a to be the time interval between "attempts" at transmission, since no transmission will occur on a link which is waiting for a RFNM return.)

The RFNM will also create a second aspect of concern, since each RFNM must be acknowledged. Therefore, a number of ACK's will be in contention for the service facility along with the messages themselves, and in heavy traffic conditions, will effectively increase each service time by the 3.0 msec. that is required to transmit a higher priority ACK. However, for lesser traffic levels, the ACK's can not be considered to be simply an additional portion of the average service time, but instead, appear to be separate high priority arrivals. These two effects can be combined by use of the priority model result that will be developed in Section 5.1.1. This result shows that the queueing delay for a priority message (the ACK in this case) will be:

$$W_1 = W_0/(1 - \alpha\lambda\bar{x}_1) \tag{4.3}$$

where W_0 is the expected time to complete a service, $\alpha\lambda$ is the average arrival rate of the ACK's, and $\bar{x}_1$ is the service time of the

ACK's. Similarly, the expected queueing delay of non-priority arrivals (the regular messages in this case) is shown to be:

$$W_2 = W_1/(1 - \rho) \tag{4.4}$$

where the value of ρ is a function of the composite arrival rate and service times (i.e., messages and ACK's). By substitution of Equation (4.3) into Equation (4.4) and recognizing that the arrival rate of messages and ACK's will be equal, we obtain:

$$W_2 = \frac{W_0}{(1 - \lambda_m X_a)(1 - \lambda_m \bar{x}_c)} \tag{4.5}$$

where X_a is the service time for an ACK, and $\bar{x}_c$ is the average composite service time for a message and an acknowledgement,

$$\bar{x}_c = X_a + \bar{x}_m \tag{4.6}$$

The value of W_0 can be shown to be:

$$W_0 = \lambda_m \overline{x_c^2}/2 \tag{4.7}$$

where the theoretical value of the average arrival rate is:

$$\lambda_m = N/T_a \tag{4.8}$$

with N being the number of generators utilized in the experiment. We also need an expression for $\overline{x_c^2}$, which must include both the ACK's and the messages. Since the two sets of arrivals are equally likely, one can show by Equation (A.6) from Appendix A, that:

$$\overline{x_c^2} = \frac{1}{2}(X_a)^2 + \frac{1}{2}\left[(X_a)^2 + 2\,X_a\,\bar{x}_m + 2(\bar{x}_m)^2\right] \tag{4.9}$$

which simplifies to:

$$\overline{x_c^2} = (X_a)^2 + X_a\,\bar{x}_m + (\bar{x}_m)^2 \tag{4.10}$$

Combining these equations produces an expression for the expected message queueing delay of:

$$W_2 = \frac{\frac{N}{2T_a}\left[(X_a)^2 + X_a\,\bar{x}_m + (\bar{x}_m)^2\right]}{\left[1 - \frac{N\,X_a}{T_a}\right]\cdot\left[1 - \frac{N(X_a + \bar{x}_m)}{T_a}\right]} \tag{4.11}$$

which is plotted as the theoretical queueing delay curve in Figure 4.1.2. Each of the test conditions used to obtain the plotted experimental data involved constant values of N/T_a and X_a, with the variable being the average message service requirement, $\bar{x}_m$. The figure shows that a rather poor match is obtained between the theory and experiment for the four generator case, with this discrepancy being due to the RFNM effect which increases the message interarrival times.* The other two test cases utilized a larger number of generators, which decreased this RFNM effect, and provided a much better agreement with the theoretical expectations. However, these two cases also exhibit a departure from the theoretical values for larger message lengths, and other test data confirmed that this behavior was also due to the RFNM effect.

Figure 4.1.3 shows a related effect that was observed in the experiments, i.e., that the effective value of ρ decreases due to the

*Each message generator utilizes a separate logical link which is "blocked" following a transmission until such time as the RFNM is returned.

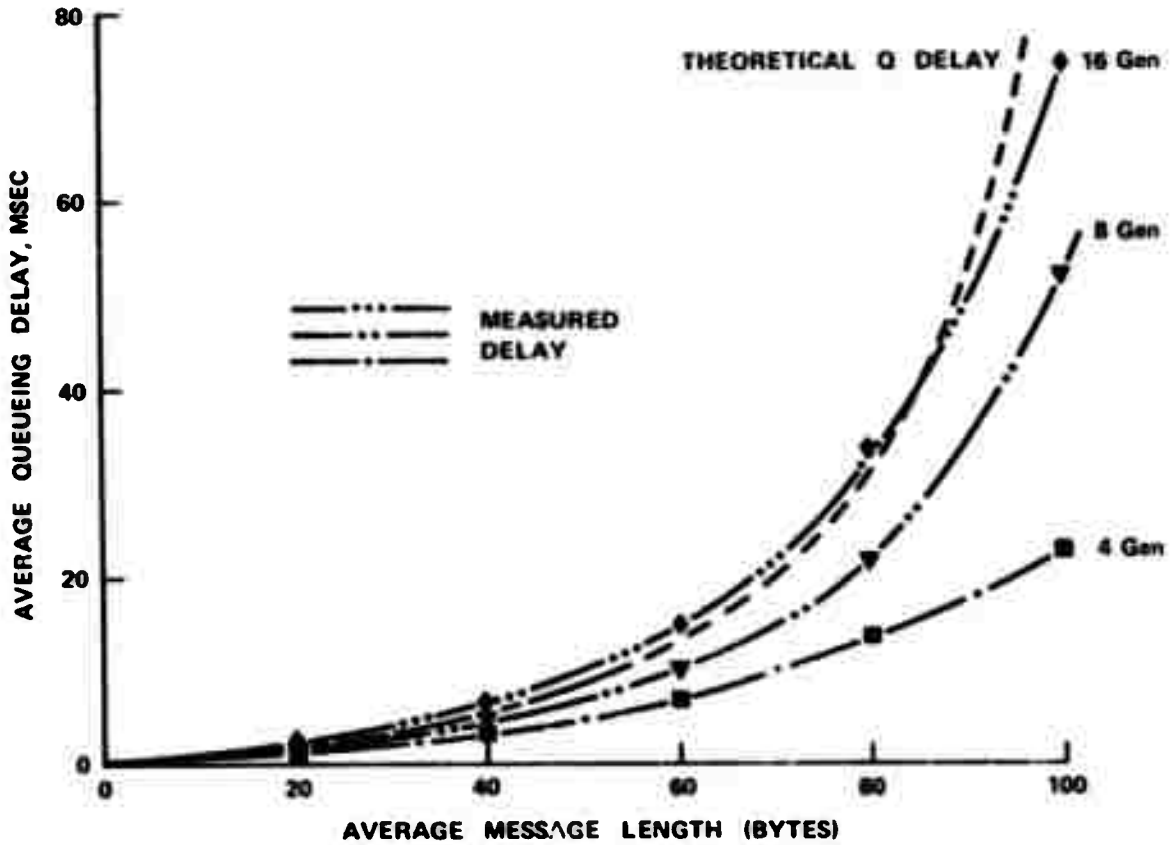


Figure 4.1.2 Queueing Delay as a Function of Message Length for a Fixed Average Interarrival Time

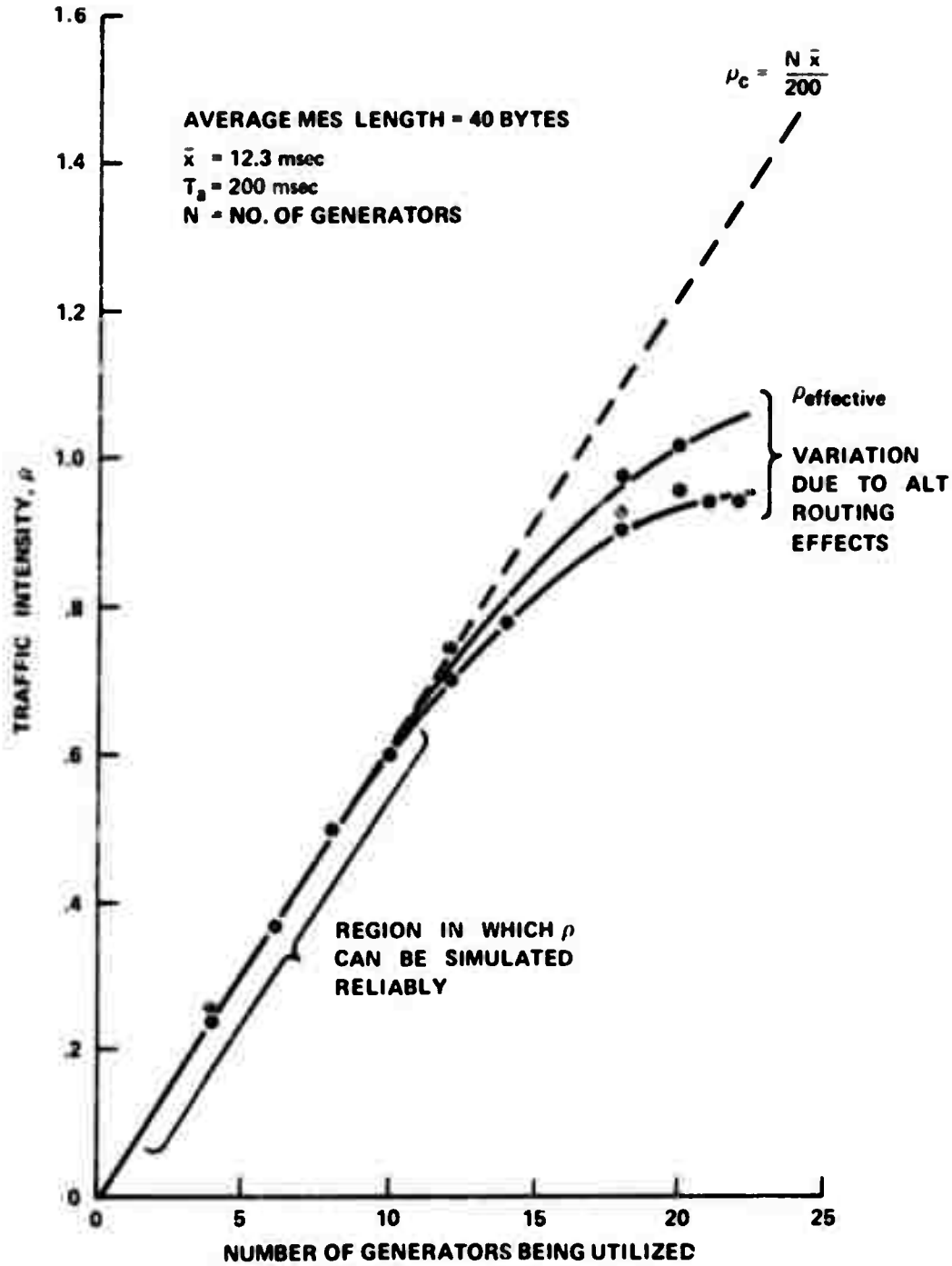


Figure 4.1.3 Theoretical and Measured Values of ρ for the Artificial Traffic Generator

increased interarrival times. In fact, the system approaches a "closed queueing system" in which the "customers" circulate through the server and then return for more service after some random idle period. For such a system the arrival rate must asymptotically approach the service rate, i.e., ρ approaches unity as shown in the figure. The effect of alternate routing perturbs this portion of the curve since it provides an instantaneous increase in the service rate, and makes the region rather unpredictable. However, the actual region of interest is the lower portion of the curve which approximates the more classical random arrival, infinite population, queueing model. This data will be utilized to select the experiment parameters, i.e., message length, number of generators, and the "transmission attempt interval", T_a, for subsequent experiments such as the priority model verification experiment of Section 5.1.2.5.

4.2 Measurements Related to Network Applications

Applications of the network were initially limited by the lack of an established protocol for HOST-to-HOST transmission, although several sites made use of the network by establishing special purpose protocol subsets for handling their particular functions of interest such as interactive work, file transmissions, or graphics display functions. Rather than summarizing the measurement facilities relative to a number of these application areas, we will present a more detailed accounting of the measurements that were taken on the activity of one set of users.

4.2.1 Measurements of Network Usage

The activity of interest involved the use of the DEC PDP-10 at the University of Utah and the XDS 940 at SRI, with the general nature of the interaction being as follows. A file would be sent from SRI to Utah, and then a number of interactive debug operations would be performed, with transmission from the XDS 940 to the PDP-10 being on a character-by-character basis, and responses being handled on a line at a time basis. The files were transmitted in blocks of 4608 bits, which was a convenient multiple of the 24-bit word length of the XDS 940 and the 36-bit word length of the PDP-10, and also provided the desired block size of 128 words for the PDP-10.

The established protocol involved several special characters to indicate the message type and length, such that a single character message required five IMP-words of transmitted data. Due to these character-by-character and file transmission protocol conventions, almost all of the SRI-to-Utah transmissions consisted of either 5-word messages or 5-packet messages respectively, as shown in the sample print-out of Table 4.2.1. Each line corresponds to one 12.8 second statistics message, with the time-of-day being determined by adding 12.8 second increments to the starting time since the data values were taken consecutively. The sequence of events which is shown includes a 10-word "file name" message, the file transmission of 67 5-packet messages plus a 4-packet file fragment, an "end of file" message, and a sequence of interactive traffic. This particular file transmission consisted of approximately 39,300 bytes and occurred over a 25.6 second period for an average bandwidth usage of about 12.5 Kbits/second. The

TABLE 4.2.1

A SUBSET OF THE ACCUMULATED STATISTICS MEASUREMENT

OF THE SRI-UTAH TRAFFIC

time-of-day	avg. RT time	avg. # words	single packet messages * 0-1	2-3	4-7	8-15	16-31	32-63	multi-packet messages ** 2	3	4	5	6	7	8
21:59.930	153.7	3[illegible].0	0	0	0	1	0	0	0	0	0	29	0	0	0
22:00.203	118.1	3[illegible].4	0	1	9	0	0	0	0	0	1	38	0	0	0
22:00.417	26.4	[illegible].0	[illegible]	[illegible]	32	0	0	0	0	0	0	0	0	0	0
22:00.630	0.0	[illegible].0	[illegible]	[illegible]	0	0	0	0	0	0	0	0	0	0	0
22:00.843	82.3	[illegible].0	0	0	11	[illegible]	[illegible]	[illegible]	[illegible]	0	0	0	0	0	0
22:01.057	23.9	[illegible].0	0	0	6	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
22:01.270	25.3	[illegible].0	0	0	12	0	0	0	0	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
22:01.483	27.6	[illegible].0	0	0	19	0	0	0	0	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
22:01.696	0.0	[illegible].0	0	0	0	0	0	0	0	0	0	0	0	0	0
22:01.910	0.0	[illegible].0	0	0	0	0	0	0	0	0	0	0	0	0	0
22:02.123	0.0	[illegible].0	0	0	0	0	0	0	0	0	0	0	0	0	0
22:02.336	133.3	37.8	0	1	0	1	0	0	0	0	0	34	0	0	0
22:02.550	0.0	[illegible].0	0	0	0	0	0	0	0	0	0	0	0	0	0
22:02.763	0.0	[illegible].0	0	0	0	0	0	0	0	0	0	0	0	0	0
22:02.976	29.4	[illegible].0	0	0	3	0	0	0	0	0	0	0	0	0	0
22:03.190	76.0	[illegible].0	0	0	11	0	0	0	0	0	0	0	0	0	0
22:03.403	55.5	[illegible].0	0	0	14	[illegible]	[illegible]	[illegible]	[illegible]	0	0	0	0	0	0
22:03.616	24.8	[illegible].0	0	0	8	[illegible]	[illegible]	[illegible]	[illegible]	0	0	0	0	0	0
22:03.830	60.0	[illegible].0	0	0	13	[illegible]	[illegible]	[illegible]	[illegible]	0	0	0	0	0	0
22:04.043	0.0	.0	0	0	0	0	0	0	0	0	0	0	0	0	0

end of file

interactive traffic

file name

file transmission

interactive traffic

* MEASURED IN TERMS OF IMP WORDS

** MEASURED IN TERMS OF THE NUMBER OF PACKETS PER MESSAGE

maximum file transmission bandwidth observed during any 12.8 second interval was one in which 73 5-packet messages were transmitted. Each message was the standard 576 byte record, for a total bandwidth usage of:

26.3 Kb/sec for data ("effective" data rate")
0.3 Kb/sec for marking, padding, and protocol format
3.9 Kb/sec for network header, checksum, etc.
30.5 Kb/sec total bandwidth used.

Transmission occurred during the entire interval since the file transmission was measured over three 12.8 sec. intervals, and this was the center interval. The average round trip time for a message was 159 msec. (measured from the time the source IMP receives the first packet until it receives the RFNM). The minimum round trip time would be about 138 msec. based on the time for serial transmissions and propagation delays. The extra delay of 159 minus 138, or 21 msec., was due to the delays in the PDP-10 acceptance of messages. The effective round trip cycle must include an additional 12 msec. for HOST processing of the RFNM and for the leader and first packet transmission from the HOST to the IMP. Therefore, the maximum single link data rate is approximately one 576 byte record every 150 msec. or 30.7 Kb/sec. of actual data, compared with the observed maximum effective data rate of 26.3 Kb/sec. A similar calculation using the measured average round trip time and the resulting effective data rate indicates that about 4 to 5 msec. are required for HOST processing of the RFNM and setting up the next transmission. These results are summarized below:

	Measured	Theoretical minimum (or maximum)
Round trip time	159 msec.	138 msec. min.
Effective data rate	26.3 Kb/sec.	30.7 Kb/sec. max.
Average HOST RFNM Proc. & mes. set up time	5 msec.	-
HOST delay in taking mes.	21 msec.	-

The round trip time required for messages between the two HOST's was plotted in Figure 4.2.1 as a function of message length L. Each point of this scatter plot represents one interactive message, i.e., a one-packet message. The scattering is an indication of the frequency of occurrence of both message length variations and round trip delay variations. A definite lower limit of delay was observed as indicated by the dashed line, and this limiting delay function is just what would be expected based on estimated values of the various components of delay. In this minimum delay case, there is no queueing time such that the expected delay consists of:

- Serial transmission at source IMP = 2.0 + 0.32L msec.
- Propagation and modem delay = 7.5 msec.
- Processing time at destination = 0.5 msec.
- Serial IMP-HOST transmission = 0.3 + 0.16L msec.
- Serial transmission of RFNM = 2.0 msec.
- Propagation and modem delay = 7.5 msec.
- Processing of RFNM = 0.2 msec.

Total 20.0 + 0.48L msec.

This estimated total delay agrees very well with the experimental

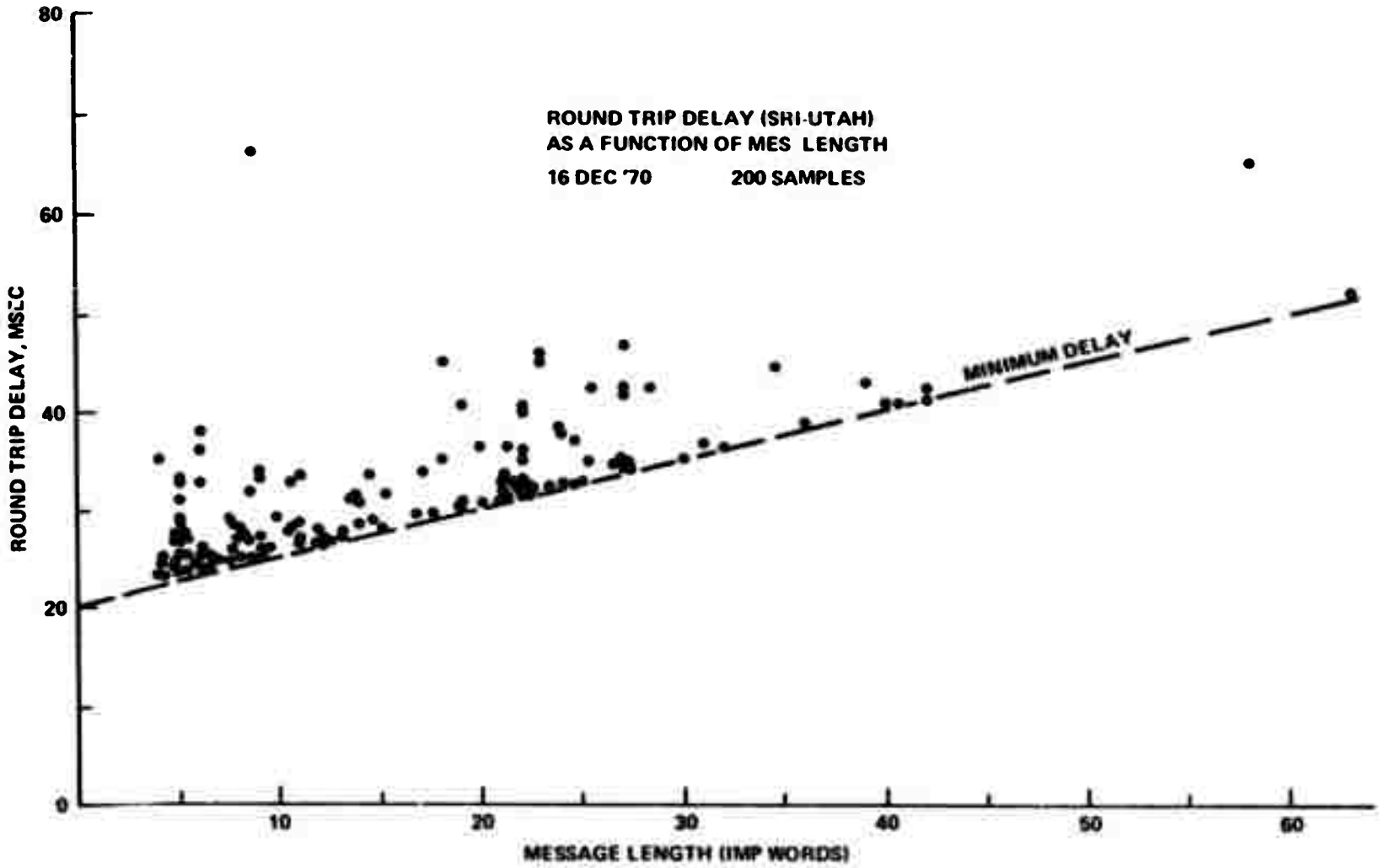


Figure 4.2.1 Round-Trip Delay Measurements of the SRI-Utah Traffic

minimum delay values as shown in Figure 4.2.1. The other points in the figure represent messages whose delay values exceeded the minimum due to queueing delays and/or HOST delays in taking the message from the IMP.

4.2.2 Estimating User Behavior

The example of the SRI-Utah interactive traffic measurements can also serve as an indication of the degree to which one can obtain information regarding the user behavior. Table 4.2.1 showed a pattern of data transfers which could readily be seen to be a sequence of file transfers followed by bursts of interactive traffic. Of course, no information could be obtained regarding the usage of the files, i.e., they could have been display images or small data bases, but the fact that they all consisted of five packet messages gives some information regarding the file transfer protocol. Similarly, the predominance of five word messages might lead one to speculate that the user-to-computer messages were sent on a character-by-character basis, while the computer response traffic was sent in longer and more variable length messages. (The latter were included in the measurements, but are not shown in the table.)

4.2.3 Estimating Traffic Statistics

The reduction of the data from the SRI-Utah traffic example would have been made more difficult if the traffic had been a composite representation of several simultaneous user transmissions, but some insights could probably have still been gained. The initial tests of user behavior were typically measurements of single user characteristics

due to the absence of heavy network usage, but as the traffic loads build up, such cases will occur with less frequency. However, in many instances the composite data transmission may be of particular interest, such as the case of a HOST site desiring to measure its total network activity over a given period of time. A hypothetical example of such data is shown in Figure 4.2.2 and is the result of artifically generated traffic. This particular example was selected at random from a sequence of test data involving messages with various average lengths, but in each case having an exponential distribution of lengths. The histogram data values were utilized (with the correction formulas as developed in Appendix D) to determine the mean message length, with an average length of 65.6 IMP-words being obtained compared to a theoretical expected value of 64.0 words. (The effect of the correction was to reduce the error from 3.2 to 1.6 words.) A second example is shown in Figure 4.2.3, with an estimated mean value of 25.9 compared to a theoretical value of 24.0. In this example the correction factor reduced the error from 5.0 to 1.9. This example also shows the peak of the log-histogram density function occurring at the mean of the exponential function; a result that was developed in Section 3.2.2.

Figure 4.2.3 also shows the density estimate from the log-histogram data compared with the theoretical density function, and a reasonably good match is seen. In contrast, the data from Figure 4.2.2 would not have a very recognizable density shape due to the small sample size, although such a plot has not been shown.

A sample histogram for a modem output channel is also shown in Figure 4.2.2, and serves as an example of a potential source of

1. MESSAGE SIZE STATISTICS

1.1 SINGLE PACKET MESSAGES

LENGTH (WORDS)	FROM HOSTS	
0 - 1	0	
2 - 3	1	
4 - 7	2	*
8 - 15	8	******
16 - 31	10	*******
32 - 63	26	**********

1.2 ALL MESSAGES

LENGTH (PKTS)	FROM HOSTS	
1	47	**********
2	19	********
3	5	**
4	2	
5	0	
6	0	
7	0	
8	0	

5. PACKET SIZE STATISTICS (LEAVING THE IMP)

LENGTH (WORDS)	CHANNEL 2 (TO UCSB)	
0 - 1	1	
2 - 3	5	*
4 - 7	8	**
8 - 15	13	****
16 - 31	15	****
32 - 63	66	**********
TOTAL # PKTS	108	

Figura 4.2.2 HOST-IMP and IMP-IMP Traffic Histograms

```
ACCUMULATED  STATISTICS    SOURCE = UCLA   TIME = 33869


1. MESSAGE  SIZE  STATISTICS

     1.1 SINGLE  PACKET  MESSAGES

          LENGTH (WORDS)          FROM  HOSTS

            0 -  1                 0
            2 -  3                39  *****
            4 -  7                61  ********
            8 - 15                87  ***********
           16 - 31               156  ********************
           32 - 63               104  *************

     1.2 ALL  MESSAGES

          LENGTH (PKTS)           FROM  HOSTS

            1                    447  ********************
            2                     36  *
            3                      2
            4                      0
```

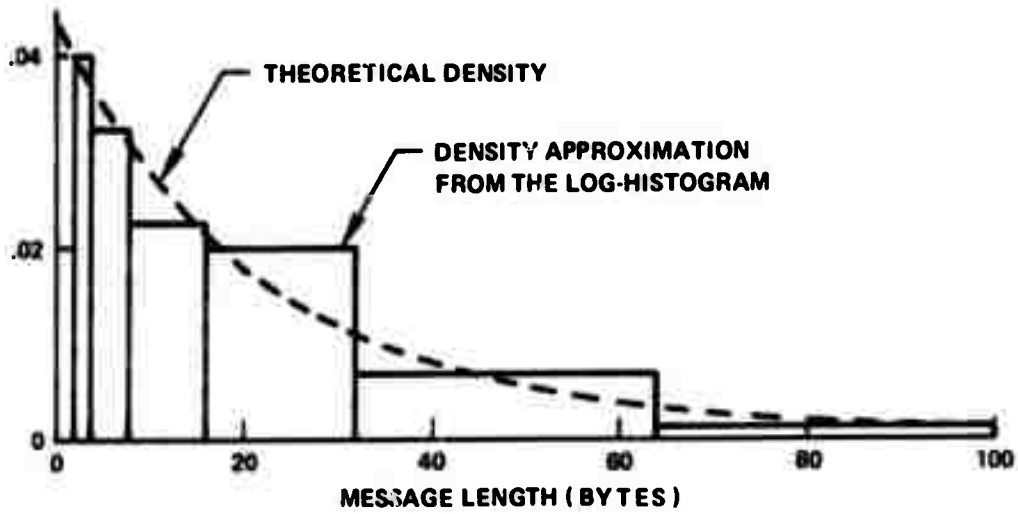


Figure 4.2.3 A Typical Log-Histogram from an Accumulated Statistics Message

error for the channel estimates. The histogram interval covering the range of 32 to 63 includes a large number of full 63-word packets, which should be considered separately in any estimates made from the histogram. In the example we can correct for this effect, since the HOST-to-IMP data shows that there were 35 full packets (i.e., 19 from the 2-packet messages, 10 from the 3-packet messages, and 6 from the 4-packet messages).* Using this information, the estimated length was 39.5 words compared to a theoretical value of 40.4, with the correction factor causing the error to increase from 0.8 to 0.9. The fact that this "correction" actually made the error somewhat larger is a reminder that the correction will improve the estimates on the average, but not necessarily in each individual case.

We were able to determine the number of full 63-word packets on the modem channel for the above example because those packets originated at the local HOST and additional length data was available. In general, we can only make such a calculation based on long term average values, and this limitation indicates a weakness in the channel data gathering facilities. An additional histogram interval for full 63-word packets would be a worthwhile extension of the measurement capabilities to avoid this uncertainty.

4.3 Interference Between Users

A major concern in the evaluation of a computer-communications network is the determination of how many simultaneous users the network can support. Several criteria for such an evaluation are possible,

*Each 3-packet message contributes two full packets, so that the five 3-packet messages result in ten full packets, etc.

ranging from being able to staisfy the peak requirement of all users simultaneously, to satisfying the average need for all users. In the latter case, if a user required an average of 500 bits/sec., then one might estimate that a 50 Kbit/sec. transmission facility could support 100 such users. Of course, such a system would not be workable since the service rate barely meets the average demand for service, and very large queues and delays would develop. The other case is overly conservative, since the probability of all users requiring their peak service simultaneously is very small, but these two conditions do tend to bound the number of users that such a network could support. In the following analysis we shall attempt to determine a better measure of the network capabilities in this regard.

4.3.1 Estimates of the Number of Users That a Communications Channel Can Support

The nature of the measured traffic in the SRI-Utah test of Section 4.2.1 involved both file transmissions and interactive messages, with the two types of traffic being seen to utilize the communications bandwidth in grossly differing ways. The file transmissions were quite infrequent, but utilized a large portion of the available bandwidth during bursts of activity. In contrast, the interactive usage consisted of longer intervals of low bandwidth transfers interspersed between idle or "think" periods. Due to these differences in bandwidth characteristics, and the differing consequences of delays for the two classes of traffic, one should not expect to utilize the same criteria for estimating the number of simultaneous users that could share the system. Therefore, we shall consider the two aspects of the

problem separately, and then attempt to combine the resulting estimates.

The interactive traffic utilization of the communication facility for several simultaneous users is analogous to the time-sharing of a computer system, and we shall take advantage of these earlier results to estimate the effects of sharing the communication channel. Scherr (SC67A) developed a time-sharing model which showed that the degradation of response due to mutliple users was low until the number of users exceeded a threshold or saturation point, above which the degradation rapidly increased. (Kleinrock (KL69C) generalized this model to evaluate multiple channels and investigated the notion of saturation.) The model considered exponential think time and service cycles for the various users, with an average think time of θ seconds and an average service requirement of $1/\mu C$ seconds per service cycle, where C is the rate at which work can be performed by the server, and $1/\mu$ is the average service requirement of an arrival. However, due to the sharing of the server, each user spends an average of T seconds in the server during a compute cycle, and then returns to the think state where he spends an average of θ seconds before making another service request. For a total of M simultaneous users, the number in the think state will, on the average, be equal to:

$$M\left[\frac{\theta}{T + \theta}\right] \tag{4.12}$$

which will produce an effective arrival rate for the server of approximately,

$$\lambda_{in} = M/(T + \theta) \tag{4.13}$$

The output rate of the server will be:

$$\lambda_{out} = \mu C(1 - p_0) \tag{4.14}$$

where μC is the service rate and p_0 is the probability that the server is idle. The input and output rates must be equal, such that:

$$\mu C(1 - p_0) = M/(T + \theta) \tag{4.15}$$

which when solved for a normalized value of T becomes:

$$\mu CT = \frac{M}{1 - p_0} - \mu C\theta \tag{4.16}$$

Since T is the average time spent in the processor to gain $1/\mu C$ seconds of service, the value of μCT is a measure of the system degradation due to the M users. For very small values of M, the degradation factor is near unity, but above a threshold or saturation value, the degradation increases rapidly and becomes approximately proportional to the number of users. This threshold value can be shown to be (KL69C):

$$M' = 1 + \mu C\theta \tag{4.17}$$

which can be written as:

$$M' = \frac{\tau + \theta}{\tau} \tag{4.18}$$

where τ is the average service time requirement,

$$\tau = 1/\mu C \tag{4.19}$$

Therefore, the relation becomes,

$$(M' - 1)\tau = \theta \tag{4.20}$$

which is intuitively satisfying since it says that the degradation which any user will see will begin to become excessive when the average service requirement of the other $M' - 1$ users equals his average think time.

To apply this theory to our shared communication channel model, we need only define the serial server by:

C = transmission rate in bits/sec.

$1/\mu$ = service requirement in bits/message

such that the average service requirement, τ, is:

$$\tau = 1/\mu C \text{ sec. per message} \tag{4.21}$$

The model assumes that this is the average service needed considering both user-to-computer and computer-to-user messages, which is a reasonable assumption for full duplex lines with users at both sites.

The measured data from the SRI-Utah test showed an average message length of about seven 16-bit words for interactive messages, which including the header, line control characters, the RFNM, and the ACK's results in an average total service requirement of:

$$\tau \cong 16.4 \text{ msec./message}$$

The think time was not measurable with the accumulated statistics due to their 12.8 second period, so the average value of 4.3 seconds observed by Jackson and Stubbs (JA69) was utilized, resulting in:

$$\theta = 4.3 \text{ seconds}$$

If we substitute these values into Equation (4.18), we obtain:

$$M' = \frac{\theta}{\tau} - 1 \cong 260 \text{ users.}$$

However, the message lengths in the observed activity were much shorter than would be expected for a more typical mix of users, and if we take Jackson and Stubbs average values of:

E[user-to-computer message] = 12 characters

E[computer-to-user message] = 154 characters

and assume an equal fraction of each type of message, we obtain an average service requirement, including the overhead, of:

$\tau \cong 24.7$ msec./message

For the same 4.3 second average think time, the critical number of users becomes:

$M' = 170$ users.

The effect of the file transfer bandwidth requirements is much more stringent on the number of users that could be supported simultaneously. A typical file length for the SRI-Utah test was about 25 K bytes, which for an effective channel bandwidth of about 40 Kbits/sec.,* would require 5 seconds of channel time for serial transmission. (The actual transmission would typically require considerably longer due to HOST delays in accepting and inserting messages.) The SRI-Utah file transmissions occurred at a frequency of about one every 12 minutes, during the period of active use, such that one user would require on the order of 1% of the effective channel bandwidth. This would set an upper limit of about 100 such users before exhausting the available bandwidth, but

---

*Based on a transmission overhead due to header, line control, etc. of about 6 msec. for each 20 msec. of effective data transmission on the 50 Kbits/sec. line.

for delay considerations, the maximum would be limited to about half that number.

We have therefore established some rather crude estimates of the number of users that such a network interconnection could support as being about 170 interactive users, or 50 file transmission users. Since users typically do some of each, we would like to combine these values in some proper manner. A reasonable combination would seem to be:

$$\frac{\text{\# interactive users}}{170} + \frac{\text{\# file transmit users}}{50} = 1 \qquad (4.22)$$

which satisfies the two end conditions, and allocates the bandwidth appropriately for a mix of the two classes of users.

4.3.2 Interference Tests Utilizing Artificial Traffic

The effect of multiple users can be simulated by utilizing many artificial traffic generators with random arrivals, but this situation differs from Scherr's model in the "think time" behavior. In his model, the think time period started at the completion of the previous service period, while in the artificial traffic generation scheme, the interval for the next "transmission attempt", T_a, is selected at the time the message is enqueued for transmission. As a result, the effective "think time" is the difference between T_a and the time spent obtaining service. We can include this effect in Kleinrock"s equation for the number of users at the "interference break point" from Equation (4.20);

$$(M' - 1)\tau = \theta$$

where τ is the average service requirement of a message (an average round trip time in our example), and θ is the think time. If we add τ to both sides of the equation we obtain:

$$M'\tau = \theta + \tau \tag{4.23}$$

in which $\theta + \tau$ is our T_a, such that:

$$M'\tau = T_a \tag{4.24}$$

For the examples shown in Figure 4.3.1, the value of T_a is 200 msec. and the average round trip time can be seen to be about 20 msec., resulting in a value of M' of 10, which agrees well with the experimental data for both exponential and fixed message lengths.

The theoretical model curves shown in the figure refer to the analysis of Equation (4.11) which includes the effect of the acknowledgement traffic. A particularly good match was obtained with the exponential message length experiment, although, as expected, the two results diverged at a higher number of generators since the model does not include the effect of Figure 4.1.3 which shows that ρ gradually approaches unity. A less satisfactory match between theory and experiment was obtained for the fixed length message experiment.

The linear slopes of the two experiment cases, i.e., with exponential (random variable) lengths and with fixed message lengths, were found to be approximately 10 msec./generator and 7.5 msec./generator respectively, although each has a mean service time of 12.3 msec. including the ACK for the RFNM. This difference is academically of interest and will be pursued in a later investigation, but the linear slope portion of the curves is not of particular concern for our present

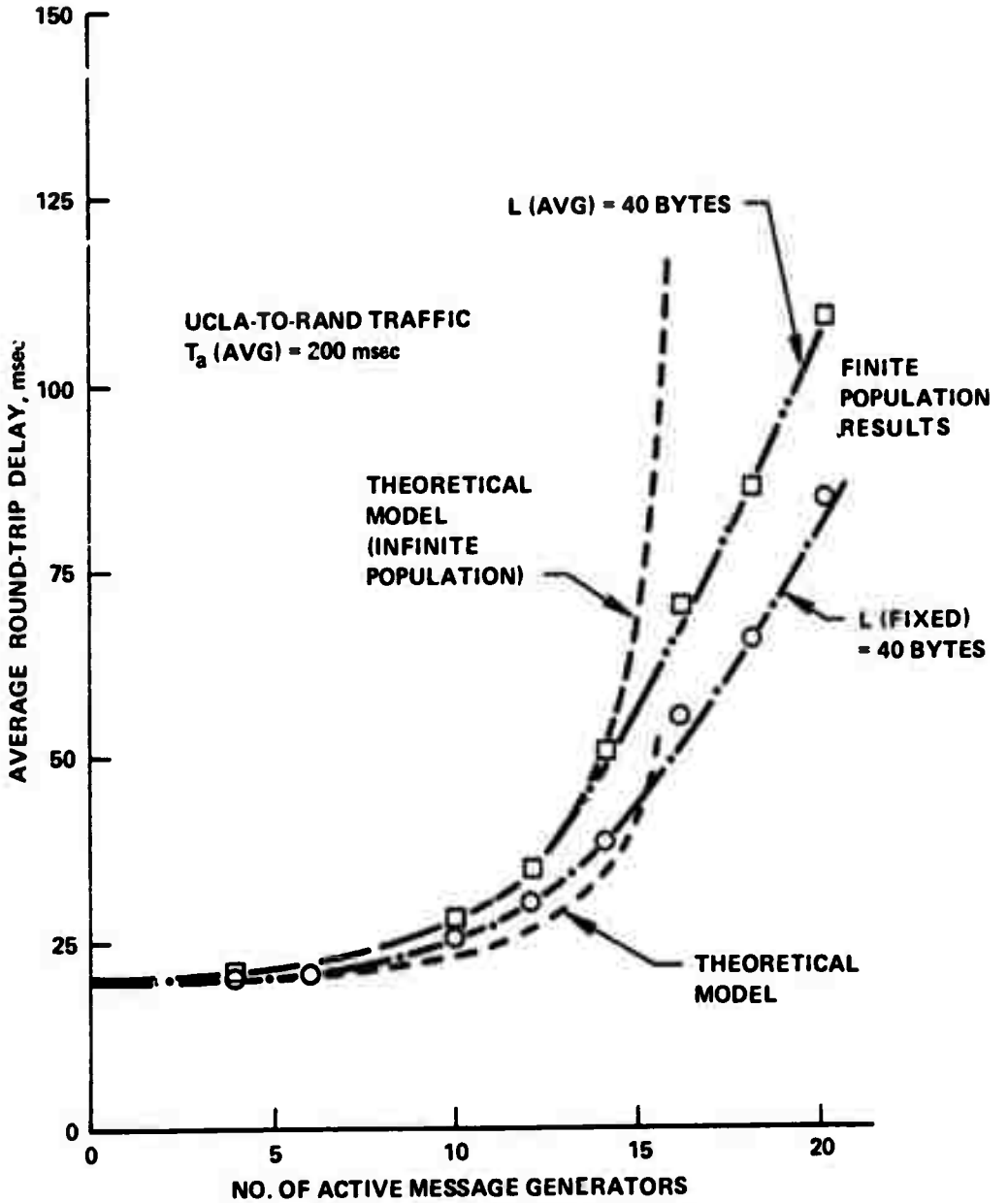


Figure 4.3.1 Average Round Trip Delay as a Function of the Number of Message Generators Being Utilized

needs of (1) finding the point at which interference occurs, and (2) establishing a predictable set of system stimuli for use in subsequent experiments.

4.3.3 Queueing Behavior in a Closed System

The ultimate user interference occurs in a closed or cyclic queueing system in which each user rejoins the queue after obtaining some amount of service. Such closed systems have been considered by Gordon and Newell (GO67), and more recently by a number of other queueing system analysts. Our model differs from theirs in several aspects, including the effect of non-zero arrival time, i.e., a message arrival occurs when the last bit of a serial bit-string has been received. Such arrival processes have been described by Cohen (CO70), who also considered the effect of finite storage capacities.

The non-zero arrival time functions can be utilized with unfinished work diagrams as shown in Figure 4.3.2, in which several different arrival rates are shown (relative to the service rate). In each example, three units share the server in a cyclic manner, but require a time interval of βT to rejoin the queue, where T is the fixed service time. In fact, an additional "arrival queue" is implicit in this scheme and corresponds to the HOST-IMP serial interface which may transmit only one message at a time, and which is in cascade with the IMP-IMP queue, with the latter being considered to be the actual service facility. In all three examples, the unfinished work builds up to a limiting value of approximately $2T$, and in Figure 4.3.3, this limiting value is shown to be $(N - 1)T$ for the general case of N items in the cyclic queue (with constant service times). As can be

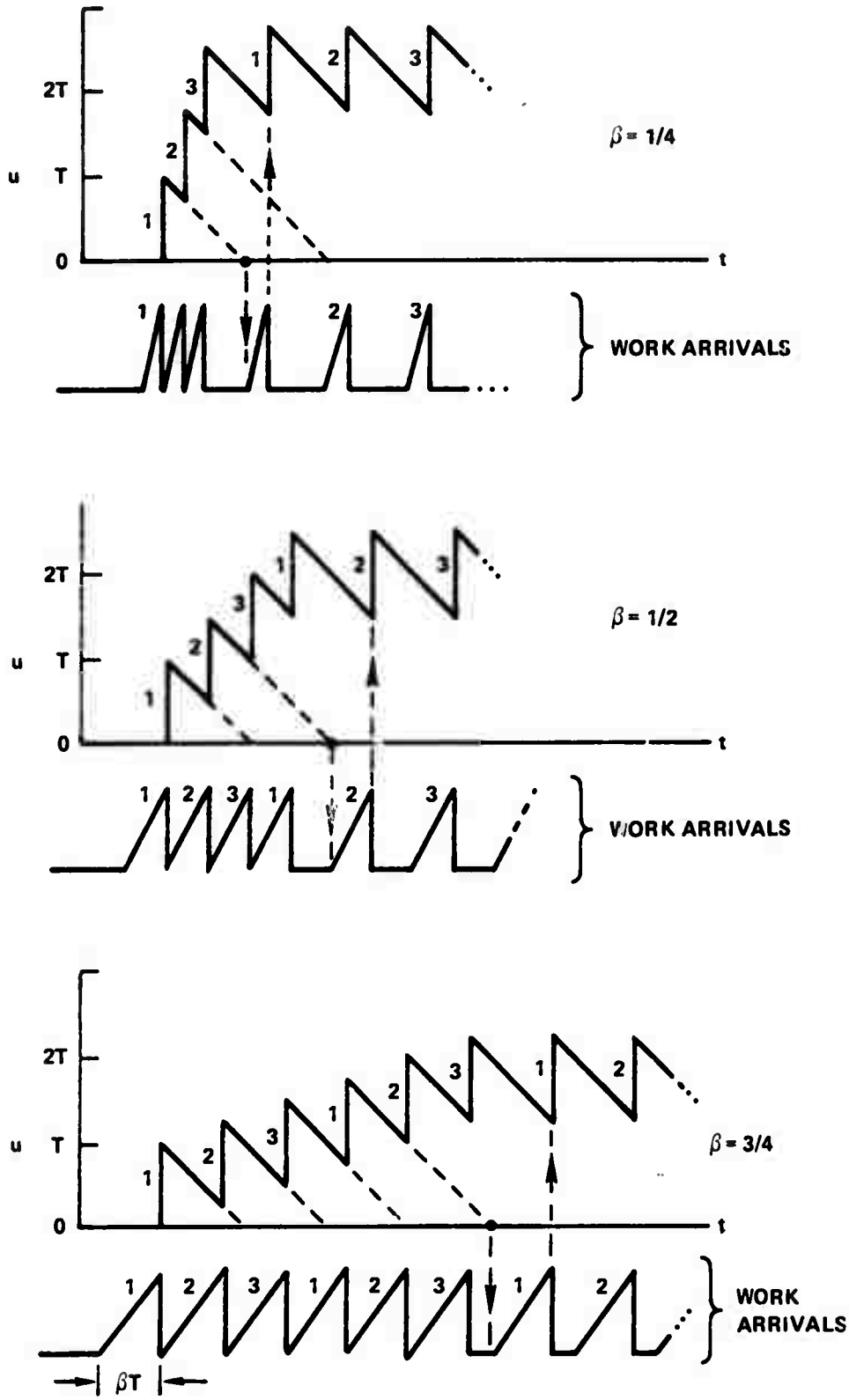


Figure 4.3.2 Arrival and Unfinished Work Diagrams for a Set of RFNM Driven Message Generators

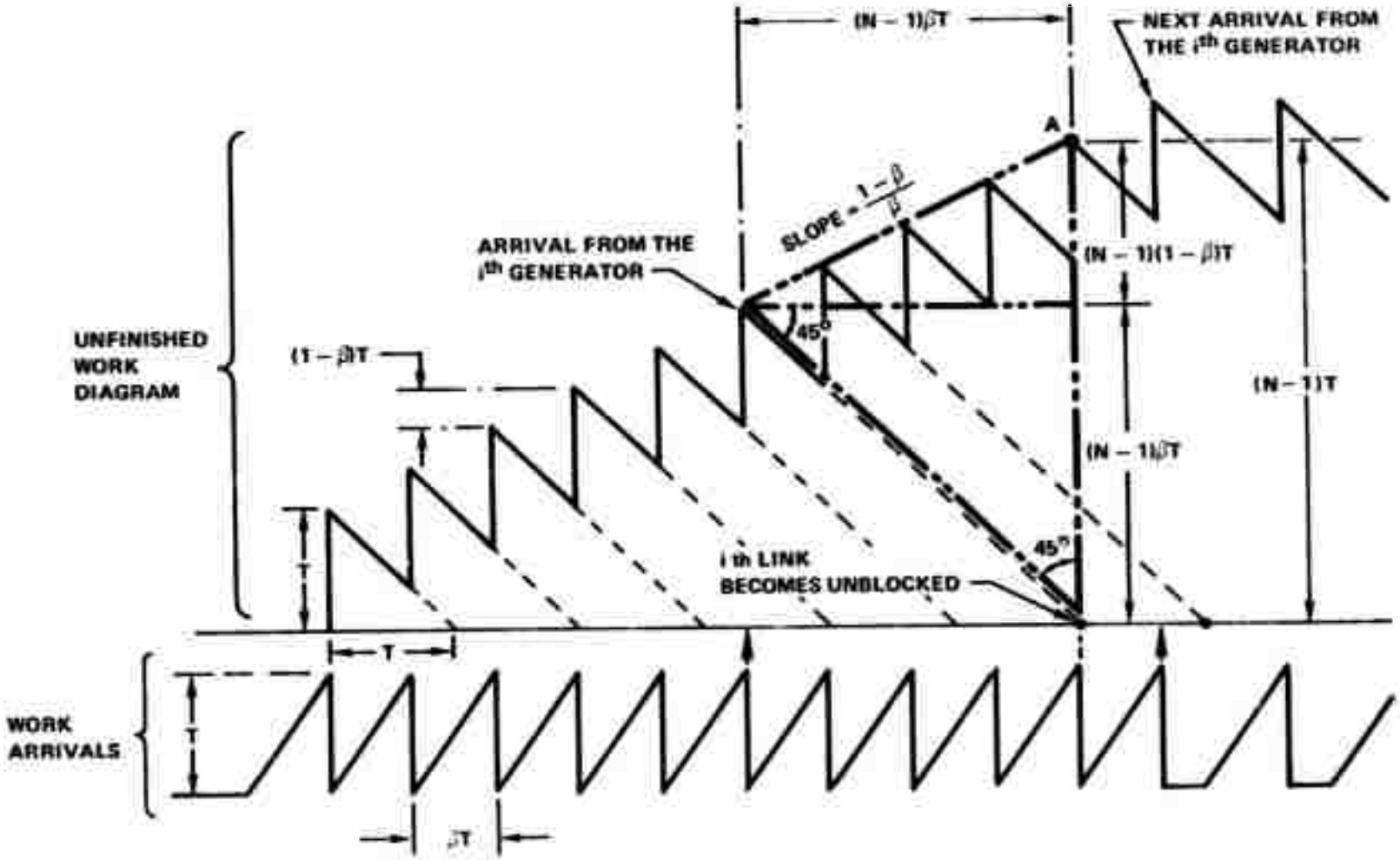


Figure 4.3.3 Determination of the Limiting Queue Level for a Set of N RFNM Driven Message Generators

seen in the previous case for N equal to three, this limit is only an approximation, but the peak values of the unfinished work function can be shown to range between the value of $N \cdot T$ for β equal to zero and $(N - 1) \cdot T$ for β equal to unity, which provides a reasonably tight bound for the larger values of N which are usually of interest.

Figure 4.3.3 indicates a geometric proof that this limiting value of unfinished work will be approximately $(N - 1) \cdot T$, by showing that the rate at which the work increases is $(1 - \beta)/\beta$ until such time as a message generator is expected to provide an arrival, but is blocked by not having completed its previous service. In the figure, the ith generator is available to provide an arrival, but the $i+1^{st}$ is not, and results in the steady-state condition of the unfinished work diagram. Knowing the slope at which the unfinished work accumulates, and the fact that for the critical timing condition, $(N - 1)\beta T$ seconds occur between the previous arrival from the ith generator and the time at which the next arrival should start across the HOST-IMP interface, then the height of the unfinished work can be seen to be:

$$\text{unfinished work} = (N - 1)(1 - \beta)T + (N - 1)\beta T$$

which simplifies to:

$$\text{unfinished work} = (N - 1)T. \tag{4.25}$$

This result was experimentally verified by the use of artificial traffic with various numbers of generators. For example, snap-shot measurements were utilized to measure the number of messages in the system for the case of ten traffic generators, and 43 of a total of 60 measurements indicated a value of nine, with eight other measurements

recording a length of ten messages and with the other nine samples being irrelevant due to alternate routing.

A similar application of the unfinished work model can be utilized to show that for the case of finite buffering, the limiting value of the unfinished work will be approximately $B - 1$ where B is the number of available buffers. Intuitively, this result says that one buffer must be allocated to the arriving message which is not included in the unfinished work until its arrival is complete.

Cyclic queueing behavior is quite artificial in the normal network environment, but does occur in certain test conditions such as the thru-put experiments which we describe in the following section.

4.4 Thru-Put Tests (RFNM Driven)

The communication lines between the IMP's have a bandwidth of 50 Kbits/second, which establishes an upper bound on the single path thru-put. However, the overhead associated with each message limits the total thru-put to some lesser value as shown in Figure 4.4.1. The same figure also indicates the measured thru-put for the case of alternate routing which exceeds the 50 Kbit/second value due to the bifurcation of the traffic flow. In all cases the messages were sent at the RFNM driven rate, i.e., as soon as a link would become unblocked.

Two different message sizes were included in the above test. In the first case, the messages were one full packet in length so that the effective packet overhead was minimized (relative to the packet size). The maximum thru-put for such messages was measured to be 38.6 Kbits/second, which agrees quite well with the theoretical value of 39.0 Kbits/second (based on 20.0 msec. of useful data transfer and

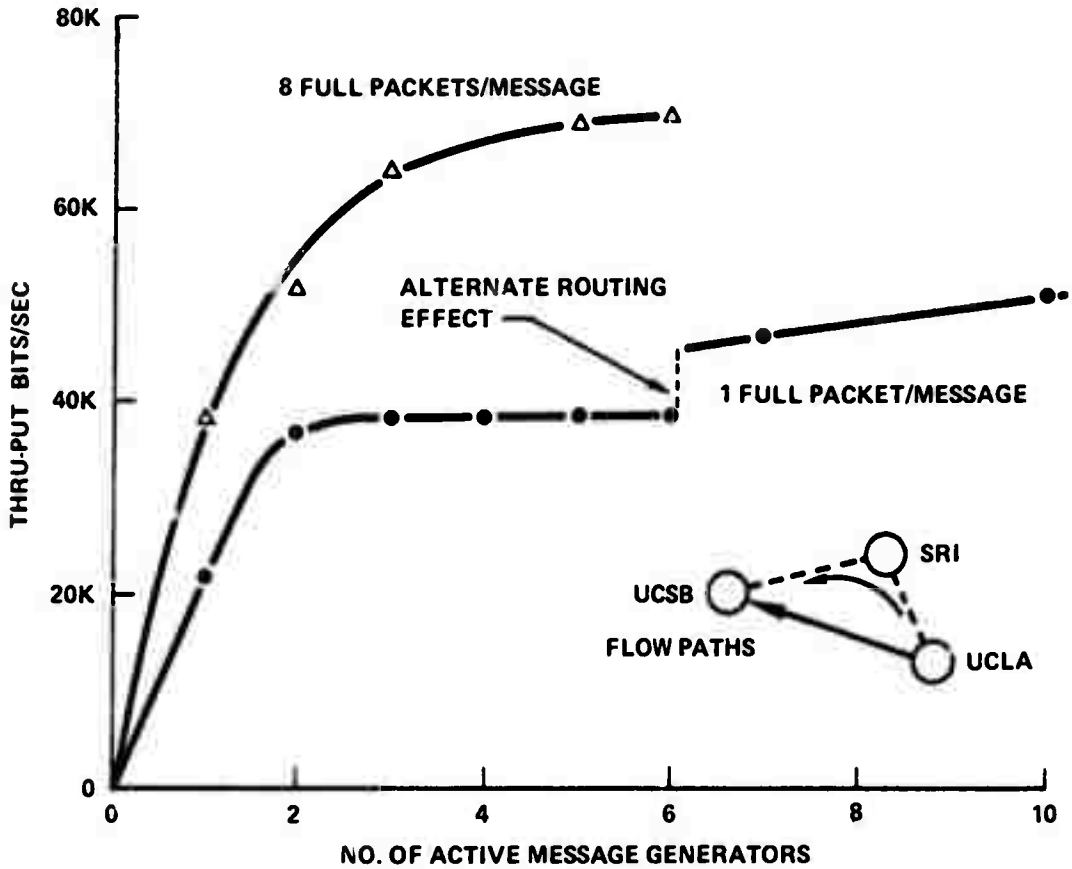


Figure 4.4.1 Thru-Put Measurements Between UCLA and the Neighboring UCSB IMP

5.7 msec. for the packet overhead and the acknowledgement for the RFNM). This maximum thru-put is not reached for the case of a single link due to the quiescent period while waiting for the RFNM to return. This delay is about 13.5 msec.,* which adds to the "non-productive"

---

*Based on a measured round trip time from UCLA to UCSB of 22.8 msec. for a 320 bit message (from Table 4.1.1) and the known service time of 9.3 msec. for a message of that length.

transmission period, and results in an expected single link thru-put of 25.5 Kbits/second. (The measured value was 22.2 Kbits/second.)

The thru-put for the one-packet message case can be seen to increase abruptly when the transition is made from six to seven traffic generators, due to alternate routing via the SRI path. Alternate routing occurs at this number of generators because the delay values in the routing table are numerically equal to the existing queue length (at the time of the routing update) plus four times the HOP number, i.e., the number of store-and-forward "hops" along the path. Equation (4.25) showed that the expected number of packets in the "system" is $N - 1$ where N is the number of generators (for single packet messages). Since the number in the queue is one less than the number in the "system" (one message is being serviced), the routing will change when $N - 2$ exceeds four, i.e., for N equal to seven as observed in the measurement data.

The increase in thru-put for the seventh generator was measured to be 8.1 Kbits/second, which is due to the fact that approximately five messages were enqueued for service when the routing was changed. Therefore, for a period of time, each channel will be transmitting with an effective instantaneous data rate of approximately 39 Kbits/second. This parallel transmission will last until the five or six messages are served (five in the queue plus one in the server), with an expected time to complete the service of about 125 msec. With routing updates of about every 640 msec., the "old routing" channel will contribute an added data rate of about:

$$(125/640) \cdot 39 \text{ Kbits/sec.} = 7.6 \text{ Kbits/sec.} \qquad (4.26)$$

which is approximately equal to the measured increase of 8.1 K/bits/ second.

The single link thru-put for the full 8-packet message case was seen to be approximately equal to the saturation value of the single packet case, i.e., 38.5 Kbits/second. This equality is due to the canceling effects of fewer RFNM's to acknowledge, and the short quiescent delay for the single link RFNM. No alternate routing was observed until the two generator test, above which the traffic split evenly between the direct UCLA-UCSB and the UCLA-SRI-UCSB routes.

The saturation effect for the 8-packet message case occurred at a lower bandwidth than was expected, since the direct and alternate route paths should each be able to deliver messages with an effective rate of about 43.5 Kbits/second (which is higher than the single-packet thru-put since there are fewer RFNM's to acknowledge). The measured limiting value of about 70 Kbits/second was therefore investigated further to try to determine the cause of this behavior. Several possible reasons for this discrepancy were considered and described below, since they represent the possible limiting factors in such transmissions and indicate the manner in which the statistics gathering tools can be utilized to determine the cause of anomalies.

a. Lack of store-and-forward buffers: In order to obtain the maximum thru-put, there must be a sufficient quantity of store-and-forward buffers to be able to create a queue that is long enough to keep the channel busy during the entire routing update interval which was thought to be about 520 msec., so with a packet service time of 23.0 msec., the queue would have to build up to a total of about 23 buffers.

Since we were told that there was a total of 25 store-and-forward buffers, this did not appear to be the limiting factor. Snap-shot measurements also showed that typically 18 to 20 store-and-forward buffers were in use, which seemed to substantiate this conclusion.

b. Lack of reassembly buffers: Snap-shot data at the UCSB destination IMP showed that typically either 8 or 16 buffers were in use, with a maximum of 24 being observed.* Since up to 32 buffers are available for reassembly, this was not the problem. The lack of any queue build up for the discard "fake HOST" was another indication that no problem existed in this area.

c. Loss of bandwidth due to routing loops: In a previous test, a considerable loss of effective bandwidth was noted due to loops in the routing (to be discussed in Section 4.5). However, the accumulated statistics at both UCLA and UCSB showed that the transmissions were all properly accounted for, i.e., no extra transmissions due to routing loops were seen.

c. Loss of bandwidth due to retransmissions: Many of the accumulated statistics data messages indicated transmission errors, with as many as six retransmissions occurring in a 12.8 second interval. However, these retransmissions required an insignificant bandwidth as evidenced by the thru-put measurements. (About 800-900 packets would be sent in one 12.8 second period, so that a few retransmissions had little effect.)

e. Loss of bandwidth for a multi-HOP path: The alternate routing path includes an extra store-and forward delay, but this delay

*Reassembly buffers are presently allocated in 8-packet blocks.

should not affect the thru-put since the channel is in effect a pipeline, and additive delays do not affect the thru-put of such a system.

f. Bandwidth limitations of the HOST-IMP interface: The HOST-IMP interface has a bandwidth of 100 Kbits/second, which can readily handle the two 50 Kbit/second channels since in the latter case there is an extra overhead due to the packet headers and line control characters. The interface is full duplex (i.e., data can pass in each direction simultaneously), so there is no conflict with the measurement data being collected.

g. Speed limitations of the Sigma 7 artificial traffic generator: The Sigma 7 computer must perform several functions including; (1) sending the artificial traffic, (2) collecting the statistics, and (3) driving the line printer (the output device for the statistics messages). Therefore, there was concern that the CPU was limited in its ability to generate the artificial traffic. This was shown not to be the case, by transmitting to the discard "fake HOST" at the UCLA IMP (instead of the similar discard facility at UCSB.) The measured thru-put for the local discard test was 91.5 Kbits/second, which verified the capability of the Sigma 7 to generate the necessary bandwidth.

h. Trace measurements to determine the detailed behavior: Since the accumulated statistics and the snap-shots both indicated that the limitation had to be in the UCLA IMP, a detailed investigation of the packet handling behavior was made by tracing all of the packets at the UCLA IMP for a brief instant of time. (At the traffic rates involved, about 100 packets/second were generated and each packet

resulted in a separate trace message.) This trace data indicated two errors in the values of the IMP parameters that had been used in the calculations of part (a) above. First the routing updates were found to occur at approximately 640 msec. intervals instead of the 520 msec. that this parameter was previously understood to be, and secondly, the upper limit on the store-and-forward buffers on an output queue must be approximately 20 (instead of 25),* since the interarrival times of packets were observed to increase from a value of 10.5 msec. to 11.5 msec. (the rate at which the two channels can just keep up with the arrivals). This condition will force the queue lengths to grow to their upper bound, and neither the snap-shot measurements nor the trace data showed an excess of 20 store-and-forward buffers in use. If we reconsider the effect of the finite buffering with these new parameter values, we find that the first channel will have approximately 18 packets to serve when the routing change occurs, and at 23.0 msec. of service time per packet, that channel will remain busy for about 420 msec.† Since the next routing update will occur after 640 msec., the first channel will be inactive for 220 of the 640 msec., for an effective thru-put contribution of:

$$\Delta \text{ thru-put} = \frac{420}{640} \cdot (43.5 \text{ Kbits/sec.}) = 28.5 \text{ Kbits/sec.} \quad (4.27)$$

for a total estimated thru-put of 72 Kbits/second, compared to the

*A subsequent check with BBN showed that the actual routing update interval is 655 msec. The upper limit on store-and-forward buffers is 25 (octal) which is 21 (decimal).

†The other two store-and-forward packets are assumed to be required for the SENT queue.

measured 70 Kbits/second, which is a reasonable match between the theory and measurement.

This test activity was described in some detail to indicate how the various measurement tools can be utilized to reconcile differences between the expected and experimental data values. However, it also points out an equally important aspect of the experimentation; namely that those people involved in the network measurement and modeling efforts must be aware of any changes in the network parameters if they are to be able to properly interpret the test results.

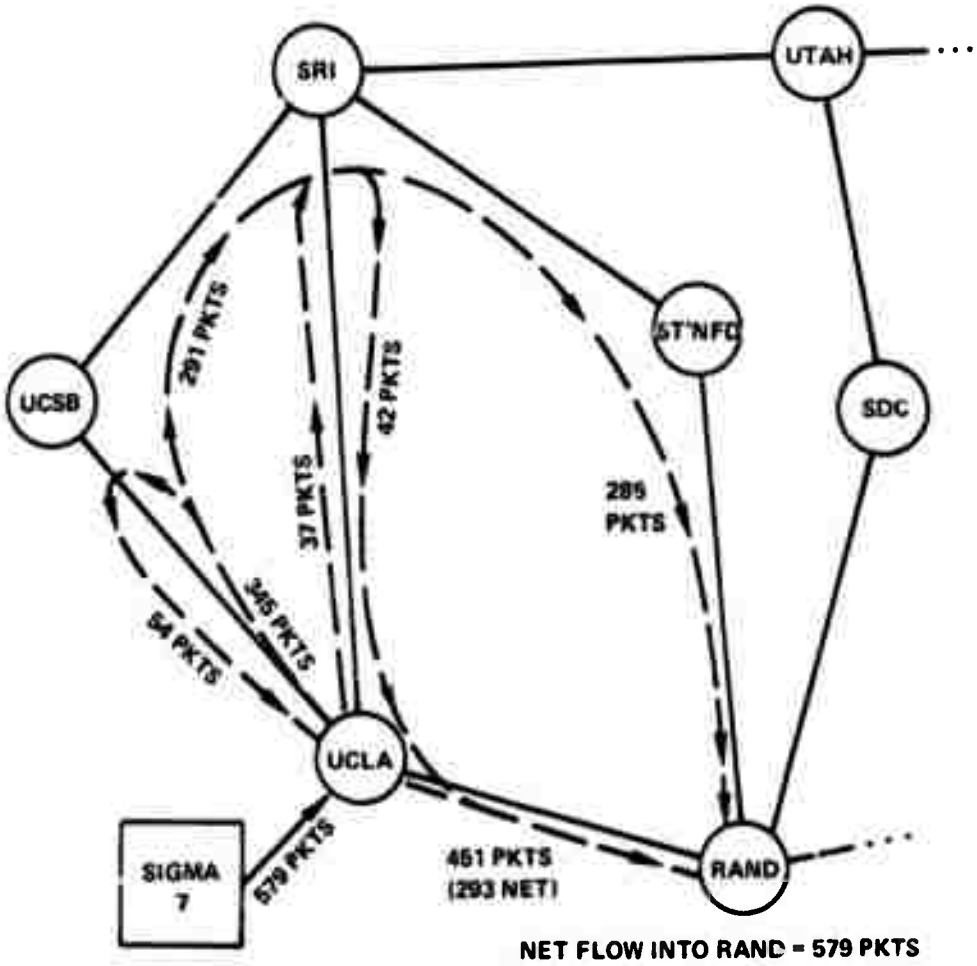
4.5 Alternate Routing Tests

The alternate routing mechanism of the IMP allows it to select a new path for traffic when congestion on a given modem channel exceeds a specified limit. However, any adaptive control system may become unstable, and in the case of the network, this instability is in the form of routing loops which can cause messages to circulate within the net rather than being routed towards their proper destination. Such loops do not have to occur endlessly to be of concern, since even transient loops add delay to the transmission due to the longer source-to-destination path, and by effectively increasing the traffic load within the net.

4.5.1 Detection of Loops in the Routing Procedure

Several interesting loops were observed in a test that was designed to check the alternate routing mechanism for UCLA to RAND message flow. This communication line was of marginal quality during the test, and numerous retransmissions were required on the direct UCLA

to RAND path. The combination of the retransmissions and the heavy traffic load lead to alternate routing both via the SRI node and via the UCSB node, and numerous routing loops were noted from the total packet counts on each channel as indicated in Figure 4.5.1. Most of



ARTIFICIAL TRAFFIC: 70 32-BIT WORDS PER MESSAGE
REGEN. PERIOD = 50 MSEC
5 LINKS ACTIVE

Figure 4.5.1 Loops in the Routing of Packets During a Special Test Condition (Not Typical of Network Flow)

the alternate routing was via UCSB, which is quite strange since it is a longer path than the route through SRI.

Accumulated data was only taken at the UCLA node, but the count of acknowledgements sent on each channel were good indications of the number of packets which had been received from SRI and UCSB, and these were counted as loops. In a test in which one expected loops, traces would also be taken to more closely measure the actual looping behavior, but unfortunately these loops were discovered well after the test had been run.

The combination of the repair of the UCLA to RAND communications line and a subsequent BBN rewrite of the IMP system program, resulted in completely differing results when the test was repeated at a later date. This latter test showed no looping, and all alternate routing went via the more direct SRI path. The test in each case consisted of sending messages with a length of 140 IMP-words on five links with a regeneration period of 50 msec. That is, every 50 msec. the artificial traffic generator would send a message on any or all of five links which were unblocked (had received the RFNM from the previous transmission). This network loading amounts to an effect data transmission rate of about 50 Kbits/second.

We should stress the fact that the loops shown in Figure 4.5.1 are quite unusual, and are certainly not typical of what one would expect to see in the network flow. However, they are indicative of the type of loops which can occur in a store-and-forward communication system, if the routing is not well behaved.

4.5.2 Delay Reduction Due to Alternate Routing

Figure 4.3.1 showed a linear increase in delay as additional message generators were introduced at high traffic levels. However, the alternate routing effect will eventually lessen this increase due to the bifurcation of the traffic flow into two or more channels, as shown in Figure 4.5.2. This same effect was not observed in the previous figure (and its related experiment) since the shortest alternate path between UCLA and RAND involves three HOP's, i.e., UCLA-SRI-Standord-RAND. Therefore reasonably large queues must build up (of the order of 10 or more packets enqueued), before alternate routing occurs, and when such alternate routing does occur, the delay is not reduced significantly due to the three store-and-forward transmissions. A third difference between the test conditions of Figures 4.3.1 and 4.5.2 involves the rate at which messages are transmitted. The UCLA-RAND test utilized random interarrival times with an average time between transmission attempts of 200 msec. for each generator. In contrast, the UCLA-UCSB experiment utilized RFNM driven traffic, i.e., messages were sent upon the receipt of the RFNM from the previous transmission.

Two other aspects of the tests are interesting to compare. The UCLA-UCSB example of Figure 4.5.2 has a delay slope of about 28 msec. per active generator, which is approximately equal to the 26 msec. service time of a full packet message and the ACK for the RFNM. The number of generators at which the interference begins to occur is seen to be about 2 generators, while with essentially zero "think time", the model would predict a value of 1. This difference is believed to be due to the "pipeline" effect of the system such that two message bit

streams can simultaneously be in various stages of the transmission, propagation, and RFNM return portions of the system.

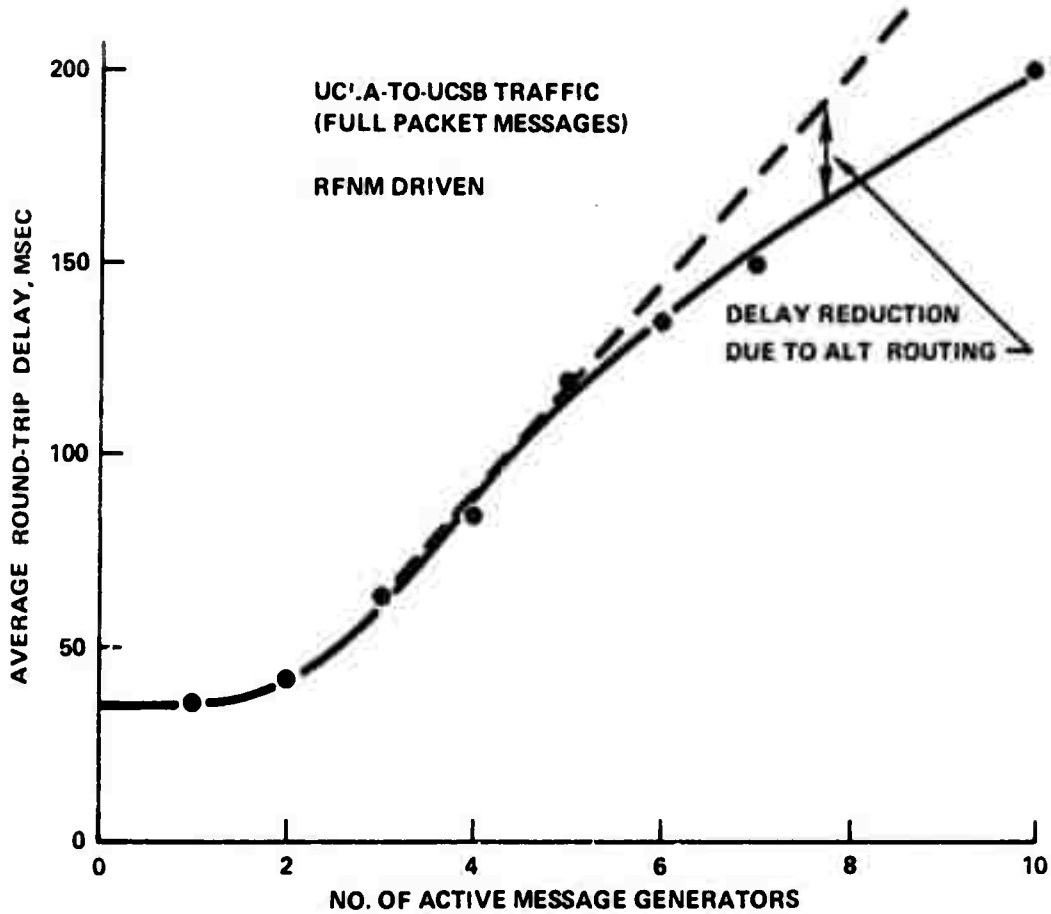


Figure 4.5.2 Average Round-Trip Delay as a Function of the Number of Active Message Generators for Full Packet Messages

CHAPTER 5

MEASUREMENTS AS RELATED TO ANALYTICAL MODELS

One of the primary functions of the network measurement effort is to provide a feedback mechanism for the development of analytic models. For example, a model governing some aspect of the IMP behavior might be hypothesized and subsequently either verified, corrected, or bounded by experimental test cases. On other instances, the tests may provide insights into the actual IMP behavior which would suggest extensions or refinements to the models, and hopefully this process could be used iteratively to develop models which satisfactorily resemble the actual operation of the subsystem being investigated.

We have attempted to develop a set of test cases to demonstrate the usage of measurements in providing insights and guidance into the development of models for a variety of different aspects of network behavior. Although these models were developed as a part of this study, they have, in some cases, been found to be quite similar to previously developed models of other application areas. However, our intent is primarily to utilize the models, both as a vehicle to demonstrate the measurement feedback process, and as approximations to the system behavior. These earlier references are cited in areas which were found to be "reinvented", with the other areas being felt to be original contributions.

Preceding page blank

5.1 A Priority Traffic Analysis Model

A common practice in message switching systems is to have a priority structure built into the queueing system. These priorities may be based on either service requirements, the relative urgency of messages, the need for interactive response, or any other appropriate rank ordering of message classes. However, we will assume that the priority class of a message is either defined by the message source or can be determined algorithmically upon injection of the message into the network, and further that the priority classification of a message remains fixed as it travels through the net.

Before defining our priority model, we will consider a wider scope of queue and service disciplines to indicate some of the alternative designs that might be considered, and to put the model in better context of the network application.

a. Queue disciplines and the conservation law

The selection of the appropriate priority mechanism is a subset of the more general problem of selecting a queue discipline from the multitude of available techniques including FCFS (first come, first served), LCFS (last come, first served), selection at random, etc. Kleinrock (KL64) has shown that the average queueing delay is independent of the queue discipline under the following "work" conserving conditions:

1. all messages remain in the system until serviced
2. there is a single server

3. non-preemptive (or preemptive-resume with exponential service)

4. Poisson arrivals.

This result is his well known conservation law which states that:

$$\sum_{p=1}^{P} \rho_p W_p = \text{constant with respect to the queue discipline} \quad (5.1)$$

where P is the number of priority classes, ρ_p is the traffic intensity, and W_p is the average queueing delay for units in the pth priority class.* The constant is evaluated by noting that for P equal to one, the system reverts to a FCFS discipline. An intuitive feeling for the meaning of the conservation law can be gained by rewriting the sum as:

$$\sum_{p=1}^{P} \rho_p W_p = \sum_{p=1}^{P} \lambda_p \bar{x}_p W_p \quad (5.2)$$

which by utilizing Little's result (LI61):

$$E[n] = \bar{n} = \lambda W$$

(where $\bar{n}$ is the expected number of messages in the system), becomes:

$$\sum_{p=1}^{P} \rho_p W_p = \sum_{p=1}^{P} \bar{n}_p \bar{x}_p \quad (5.3)$$

*We shall utilize the convention that the index 1 has the highest priority, with lower priority messages having larger subscripts.

This latter form of the equation is in terms of the average value of the unfinished work, and while not a formal proof of the conservation law, does give a feeling for what is being conserved. In essence, the conservation law reflects the fact that the amount of work remaining to be done will be independent of the order of service.

In the special case where the service density is the same for all classes, then:

$$\bar{x}_1 = \bar{x}_2 = \ldots = \bar{x}_p = \bar{x}$$

and Equation (5.3) reduces to:

$$\sum_{p=1}^{P} \lambda_p W_p = \lambda W(\text{FCFS}) = E[\text{no. in queue}] \tag{5.4}$$

which states that for this case, both the expected number of messages in the queue and the expected queueing delay are independent of the queue discipline.

Although these queue disciplines have the same mean queueing delay, they can be shown to have significantly different delay variances and therefore produce differing operational results.* However, for the special case in which one knows the service requirements in advance, a significant reduction can also be obtained in the mean queueing delay, and fortunately the message switching application is one such case. We shall exploit this situation to the fullest possible extent in our analysis, in considering both the message delay and buffer storage aspects of the network. Note that this reduction in the average

*Whitney tabulates these differences for a number of queue disciplines in Reference WH70.

queueing delay and the number in the system is not in conflict with the conservation law, since we are not changing the amount of unfinished work, but rather are taking advantage of the additional information about the service requirements.

b. Preemptive service disciplines

Just as there are several variations of the queue discipline in selecting the next message for service, there are also a number of possible service disciplines. We are concerned here with the preemptive aspects of the service discipline, i.e., the situation when the service of a message is prematurely terminated to handle a higher priority message. The possible preemptive situations can be categorized into four different types based on the decision algorithm to preempt and on the recovery mechanism for the preempted message. The former are:

1. preempt immediately if a lower priority class message is being serviced
2. decide to preempt or not based on some knowledge of the remaining or expended service on the message being handled.

and the recovery mechamisms are:

a. a preempted service must be restarted from its original starting point
b. a preempted message can be restarted at the point it was interrupted (or some roll-back point other than its origin.)

Although the network being considered does not utilize any form of preemptive priorities, both cases 1a and 2a would be feasible, while the preemptive-resume cases would not be unless major changes were made in the IMP-to-IMP packet handling protocol, buffer management methods, and header information. However, the segmentation of messages into packets, as will be discussed in Section 5.2, provides many of the preemptive priority advantages, so that the network application of preemptive priorities was not considered to be a fruitful area for further investigation.

c. The round trip delay considerations of conversational messages

A primary concern of the analysis will be with the added delay which the network introduces to the interactive conversational communications between a user and the computer. Therefore, we must consider the delays added to the total round trip of the user-to-computer command and the computer-to-user response. Since experience indicates that these two types of messages differ in length by an order of magnitude,* and because delays in either direction of transmission are of equal significance to the terminal user, we must consider both sources of delay in selecting a priority discipline to minimize the average interactive delay.

The total delay that the network adds to the interactive response will also include other factors such as propagation and modem delays, but these effects are outside of our domain of control, so we

---

*Test data taken by Bell Telephone Laboratories indicates a ratio of about 12 to 1 for such traffic. (See References FU70 and JA69.)

queueing delay and the number in the system is not in conflict with the conservation law, since we are not changing the amount of unfinished work, but rather are taking advantage of the additional information about the service requirements.

b. Preemptive service disciplines

Just as there are several variations of the queue discipline in selecting the next message for service, there are also a number of possible service disciplines. We are concerned here with the preemptive aspects of the service discipline, i.e., the situation when the service of a message is prematurely terminated to handle a higher priority message. The possible preemptive situations can be categorized into four different types based on the decision algorithm to preempt and on the recovery mechanism for the preempted message. The former are:

1. preempt immediately if a lower priority class message is being serviced
2. decide to preempt or not based on some knowledge of the remaining or expended service on the message being handled.

and the recovery mechanisms are:

a. a preempted service must be restarted from its original starting point
b. a preempted message can be restarted at the point it was interrupted (or some roll-back point other than its origin.)

shall concentrate on those delays which can be reduced by careful selection of parameters and message handling techniques. The primary source of delay which one can manipulate is that due to queueing, and hence this leads to our emphasis on that aspect of the problem. Our model will be designed to address this problem and will include the above mentioned variation between the command and response message lengths.

The service requirement of a message is a function of its length and the channel capacity, such that:

$$x = \ell/C$$

where:

x = service time required for serial transmission

ℓ = known message length in bits

C = channel capacity in bits per second

We consider here only the actual text portion of messages, i.e., the overhead due to communications control characters, header information, and the check sum is ignored since they are the same for all messages and are therefore merely an additive constant. (This assumes contiguous messages, and will change somewhat for the segmented message case considered in Section 5.2.) User-to-computer message lengths will be assumed to be random variables, with service requirements drawn from the density function; $b_u(x)$, and similarly the computer-to-user response service times are assumed to be from the density, $b_c(x)$.

We will assume that there are an equal number of user commands and computer responses, although we allow for an arbitrary difference

in their distributions and mean values. This assumption does not take into account computer output that occurs with several bursts in the total response; a situation which is analogous to the segmented message case of Section 5.2. However, there is a compensating user effect, namely the character-by-character transmission of some terminals. In this mode of operation each character is sent as a separate user-to-computer message, with a response being generated only after the entire multi-character command has been entered. Therefore, the assumption of an equal number of messages from the user and the computer seems to be reasonable, and it allows us to simplify the analysis by considering the composite density function:

$$b(x) = 0.5b_u(x) + 0.5b_c(x) \tag{5.6}$$

as the effective service time density of the interference traffic that a "tagged" message (i.e., one which we will follow through the system) will encounter upon arrival at the service facility. If we consider a two-priority system, i.e., the case of P equal to two, then the probability that a "tagged" user-to-computer command is of the priority class will be denoted as α_u,

$$\alpha_u = \Pr[\text{a user-to-comp. mes. is of the priority class}] \tag{5.7a}$$

and similarly, any computer-to-user response will be of the same class with probability, α_c, where:

$$\alpha_c = \Pr[\text{a comp.-to-user mes. is of the priority class}] \tag{5.7b}$$

The selection process for determining membership in the priority class can be based on message length, urgency, or any other criteria, and

will result in some average queueing delay for priority messages which will be denoted by:

$$W_1 = E[\text{queue delay} \mid \text{priority message}] \tag{5.8a}$$

with the average queueing delay for non-priority messages being denoted by:

$$W_2 = E[\text{queue delay} \mid \text{non-priority message}] \tag{5.8b}$$

At any given node, the expected queueing delay for a user-to-computer message will be:

$$E[W_u \mid \text{user-to-computer mes.}] = \alpha_u W_1 + (1 - \alpha_u)W_2 \tag{5.9a}$$

and for the computer response, the expected queueing delay will be:

$$E[W_c \mid \text{computer-to-user mes.}] = \alpha_c W_1 + (1 - \alpha_c)W_2 \tag{5.9b}$$

The queueing delay of interest is the total for the user command and the computer response, and at a given node along the store-and-forward path, is:

$$E[W_u + W_c] = (\alpha_u + \alpha_c)W_1 + [2 - (\alpha_u + \alpha_c)]W_2 \tag{5.10}$$

If we define the parameter, α, as the probability that an arbitrary message is of the priority class, then for equal rates for user and computer generated messages,

$$\alpha = 0.5\alpha_u + 0.5\alpha_c$$

or:

$$\alpha_u + \alpha_c = 2\alpha$$

Substituting into Equation (5.10), we obtain:

$$E[w_u + w_c] = 2[\alpha W_1 + (1 - \alpha)W_2] \tag{5.11}$$

which shows that the expected contribution to the total command-response delay is merely twice the expected unconditional queueing delay at the node, and therefore allows one to utilize conventional queueing theory results in the overall delay analysis.

The introduction of a composite service time density means that, at best, the queueing system model can be of the M/G/1 type (i.e., Markovian arrivals, general service times, and a single server). Therefore, such a model does not have the memoryless properties of the M/M/1 system which make queueing network analysis tractable as discussed earlier in Section 1.4, and the use of the model to develop procedures for delay minimization should be considered to be a local optimization which should lead to good, but not necessarily optimal, overall network performance.

5.1.1 The General Analysis of a Two Priority Class Queueing System

The effect of priorities can be to either give better service to certain selected messages, or to reduce the average delay for messages in general. Fortunately, if the higher priority class of messages is also of shorter length, then one obtains both improvements simultaneously. However, longer messages will have larger average delays in this priority situation as compared to a non-priority system. We shall consider the effect on each class of message in Section 5.1.2, but for now, we are concerned only with the overall average message queueing delay.

In the following theorem we shall investigate the change in this overall expected queueing delay for an arbitrary division of messages into two priority classes. The result will include the case of shorter messages having higher priority, but will also be valid for other cases such as giving priority to long messages or a random assignment of priorities.

Theorem 5.1.1

For a single server queueing system with Poisson arrivals, infinite queueing space, and a general service process, the expected queueing delay (averaged over all messages) for a two class priority system will be:

$$W = W(FCFS) \left[\frac{1 - \alpha \lambda \bar{x}}{1 - \alpha \lambda \bar{x}_1} \right]$$

using any arbitrary discrimination process to divide the messages into the two classes, where the notation is:

$$E[\text{fraction of priority messages}] = \alpha \tag{5.13a}$$

$$E[\text{service time of a priority message}] = \bar{x}_1 \tag{5.13b}$$

and in which:

λ = average arrival rate

$\bar{x}$ = E[service time of an arbitrary message]

$\overline{x^2}$ = E[square of the service time of an abritrary message]

W(FCFS) = E[queueing delay | FCFS discipline]

with:

$$W(FCFS) = \lambda \overline{x^2} / 2(1 - \rho)$$

and:

$$\rho = \lambda \bar{x} < 1$$

Proof:

From Cobham's result (CO54) we know that the expected delay for a priority message is:

$$W_1 = W_0 + \bar{n}_1 \bar{x}_1 \tag{5.14}$$

where:

W_1 = E[queueing delay of a priority message]

W_0 = E[time to complete an existing service]

$\bar{n}_1$ = E[number of priority messages in the queue]

and one can show that (e.g., see Reference SA61):

$$W_0 = \lambda \overline{x^2} / 2 \tag{5.15}$$

Next we utilize Little's result (LI61), which states that the expected number in the system (or queue) equals the average arrival rate times the expected time in the system (or queue). Applying these results, with an average arrival rate of priority messages of $\alpha\lambda$, we obtain:

$$W_1 = \frac{\lambda \overline{x^2}}{2} + \alpha\lambda W_1 \bar{x}_1$$

or in reduced form:

$$W_1 = \frac{(\lambda\ \overline{x^2}/2)}{1 - \alpha\lambda\bar{x}_1} \tag{5.16}$$

By a similar, but somewhat more involved procedure, we can calculate the expected delay for a non-priority message. One can show that:

$$W_2 = W_0 + \bar{n}_1\bar{x}_1 + \bar{n}_a\bar{x}_1 + \bar{n}_2\bar{x}_2 \tag{5.17}$$

which states than an arriving non-priority message will expect to wait for the completion of the service being performed, plus the service of the n_1 priority messages already in queue, plus that for the n_a subsequent priority arrivals, and finally, that for the n_2 non-priority messages that were already in the queue (which will each require an average of $\bar{x}_2$ seconds of service). Using Little's result the equation becomes:

$$W_2 = \frac{\lambda\ \overline{x^2}}{2} + \alpha\lambda\ W_1\bar{x}_1 + \alpha\lambda\ W_2\bar{x}_1 + (1 - \alpha)\lambda\ W_2\bar{x}_2$$

which upon combining terms becomes:

$$W_2 = \frac{(\lambda\ \overline{x^2}/2) + \alpha\lambda\ \bar{x}_1 W_1}{1 - \rho} \tag{5.18}$$

where:

$$\rho = \lambda\ \bar{x}$$

and with further reduction yields:

$$W_2 = \frac{W_1}{1 - \rho} \tag{5.19}$$

The expected delay for an arbitrary message is then found by combining these two components, since the unconditional delay is:

$$E[\text{delay}] = W = \sum_p E[\text{delay}|p^{th} \text{ priority class}] \cdot \Pr[\text{member of } p^{th} \text{ class}]$$

which becomes:

$$W = \alpha W_1 + (1 - \alpha) \frac{W_1}{1 - \rho}$$

or:

$$W = \frac{(\lambda \overline{x^2}/2)}{1 - \rho} \cdot \left[\frac{1 - \alpha\rho}{1 - \alpha\lambda\bar{x}_1}\right] \tag{5.20}$$

Since the first factor is the FCFS queue delay, and replacing ρ by $\lambda\bar{x}$ to give a similar form to both the numerator and denominator, we have:

$$W = W(\text{FCFS}) \cdot \left[\frac{1 - \alpha\lambda\bar{x}}{1 - \alpha\lambda\bar{x}_1}\right] \tag{5.21}$$

which completes the proof.

This theorem indicates that the expected queueing delay with priorities can be either less than, equal to, or greater than the FCFS delay depending on the relative values of $\bar{x}_1$ and $\bar{x}$. However, the delay will remain finite for all values of ρ less than unity since:

$$\frac{1 - \alpha\lambda\bar{x}}{1 - \alpha\lambda\bar{x}_1} \leq 1 \tag{5.22}$$

due to the fact that:

$$\bar{x} = \alpha\bar{x}_1 + (1 - \alpha)\,\bar{x}_2 \tag{5.23}$$

Similarly, the zero of the function does not occur in the range of $\rho < 1$ since,

$$\alpha\lambda\bar{x} = \alpha\rho \tag{5.24}$$

and α is at most equal to unity.

The result of Theorem 5.1.1 was developed independently, but was later found to be a generalization of an earlier result obtained by Morse (MO58) and another described by Conway, et al. (CO67). In each of these cases, the authors were considering a particular scheme of priority classification, and neither recognized that the result is applicable to any arbitrary separation of the items requiring service into two classes, with the only parameters of concern being the fraction of the items that are in the priority class, and their average service requirement.

The problem considered by Morse was the following. Priority messages have an exponential service time density with a mean service rate of μ. Non-priority service requirements are also exponential, but with a service rate $\beta\mu$. Note that a priority message may require a longer service time than some non-priority message; indeed, the priority messages may be longer on the average since β can either be less than or greater than unity. Examples of such a system would be those in which the sender of the message has paid a premium for faster service, has a degree of urgency, or outranks other users. Morse developed his solution by considering the state equations for the queues and solving for the expected value of the queueing delays from the generating functions. He then considered the ratio of delays with and

without the priorities and obtained as a result,

$$\text{ratio} = \frac{1 - \alpha\rho[\alpha + (1 - \alpha)(1/\beta)]}{1 - \alpha\rho} = \frac{1 - \alpha\,\rho_{eff}}{1 - \alpha\rho} \tag{5.25}$$

in which his ρ_{eff} corresponds to our ρ, and his value of ρ is λ/μ or our $\lambda\bar{x}_1$. His result, which was based on exponential service times for each of two priority classes, is an interesting special case of Theorem 5.1.1.

The problem described by Conway, et al. is the same as we will consider in Section 5.1.2, i.e., the situation when the two priority classes are distinguished by a threshold service requirement, such that messages requiring less service than this threshold value are considered to be of higher priority. We will discuss their result in more detail later, since the priority threshold case is the problem of particular concern to the ARPA network.

5.1.2 The Two Priority Class Case With Priority Based on Message Length

The two priority class case utilizes a priority threshold, X_T, to categorize messages into two groups; (1) priority messages for which $0 \leq x \leq X_T$, and (2) non-priority messages with $X_T < x$. For this special case, the value of α will be a function of X_T,

$$\alpha_1 = \Pr[\text{priority message}] = \int_0^{X_T} b(x)dx$$

$$\alpha_1 = \alpha$$

and:

$$\alpha_2 = \Pr[\text{non-priority message}] = \int_{X_T}^{\infty} b(x)dx$$

$$\alpha_2 = (1 - \alpha)$$

The general two class priority result of Theorem 5.1.1 applies here, and becomes a function of the priority threshold, X_T,

$$W(X_T) = W(FCFS) \cdot \frac{1 - \alpha(X_T)\lambda\bar{x}}{1 - \alpha(X_T)\lambda\bar{x}_1(X_T)} \tag{5.26}$$

where the functional relationships of α and $\bar{x}_1$ have been shown explicitly. However, we shall drop this functional notation in the sequel for notational convenience.*

We can determine a number of characteristics of the $W(X_T)$ function from the form of Equation (5.26). For example, for $X_T = 0$, there will be no priority messages, i.e., $\alpha = 0$, and we are left with:

$$W(X_T = 0) = W(FCFS) \tag{5.27}$$

---

*Our result was obtained independently from that of Conway, et al. (CO67), who also considered this special case of Theorem 5.1.2. Their result was of the form:

$$E[W] = \frac{\lambda E[p^2]}{2(1 - \rho)} \cdot \left(\frac{1 - \rho G(d)}{1 - \rho_1}\right)$$

where:

$$\rho_1 = \lambda G(d) E[p \mid p < d]$$

and $G(p)$ is the cumulative service distribution, such that $G(d)$ is equivalent to our α. Their p and d correspond to our x and X_T respectively. They noted that for any $G(p)$, d is an increasing function of λ, and is always greater than $E[p]$. They calculated the optimal priority threshold for the exponential and uniform densities using the approach of Cox (CO61).

The value of $\alpha\bar{x}_1$ can not be greater than $\bar{x}$ since:

$$\alpha\bar{x}_1 = \int_0^{X_T} xb(x)dx \leq \int_0^{\infty} xb(x)dx = \bar{x}$$

In the limit as X_T becomes infinitely large, $\alpha\bar{x}_1$ approaches the value of $\bar{x}$, so that:

$$W(X_T \to \infty) = W(FCFS) \tag{5.28}$$

The general shape of a plot of the expected delay, W, as a function of the priority threshold, X_T, is therefore known to be equal to the FCFS value at both extremes of X_T, and to have some lesser value at intermediate points. This general shape implies a minimal delay at some optimal value of X_T, and we shall investigate the location of this optimum and the degree of improvement that one can obtain. This investigation shall consist of four parts; (1) the application of Theorem 5.1.1 to some sample service density functions, (2) some additional theoretical results from these observations, (3) the introduction of a delay cost weighting factor, and (4) some experimental results pertaining to the two priority class model.

5.1.2.1 Examples of the Two Priority Class Case

We will consider several example service time density functions and the corresponding relationships between the queueing delay, W, and the priority threshold, X_T. These examples provide a more intuitive feel for the mechanisms involved and for the results that are obtained. Fortunately, they also provide some new insights and surprising generalizations to the analytical results.

a. The exponential service density case

If we consider the special case for which the service time density, $b(x)$, is exponentially distributed,

$$b(x) = \mu e^{-\mu x} \qquad 0 \leq x \tag{5.29}$$

we obtain the following function forms for α and $\alpha\bar{x}_1$

$$\alpha = \int_0^{X_T} \mu e^{-\mu x} dx = 1 - e^{-\mu X_T} \tag{5.30}$$

and:

$$\alpha\bar{x}_1 = \int_0^{X_T} x\mu e^{-\mu x} dx = \frac{1}{\mu}[1 - (1 + \mu X_T)e^{-\mu X_T}] \tag{5.31}$$

If we substitute these values into the expressions for W_1 and W_2 of Theorem 5.1.1, we can obtain the expected delays for the priority and non-priority messages respectively, as a function of X_T, namely:

$$W_1 = \frac{(\lambda\overline{x^2}/2)}{1 - \alpha\lambda\bar{x}_1}$$

or:

$$W_1 = \frac{\rho/\mu}{1 - \rho[1 - (1 + \mu X_T)e^{-\mu X_T}]} \tag{5.32}$$

and:

$$W_2 = \frac{W_1}{1 - \rho} \tag{5.33}$$

These equations are plotted as a function of X_T in Figure 5.1.1, with ρ fixed at 0.8. Both the delay and X_T scales are normalized

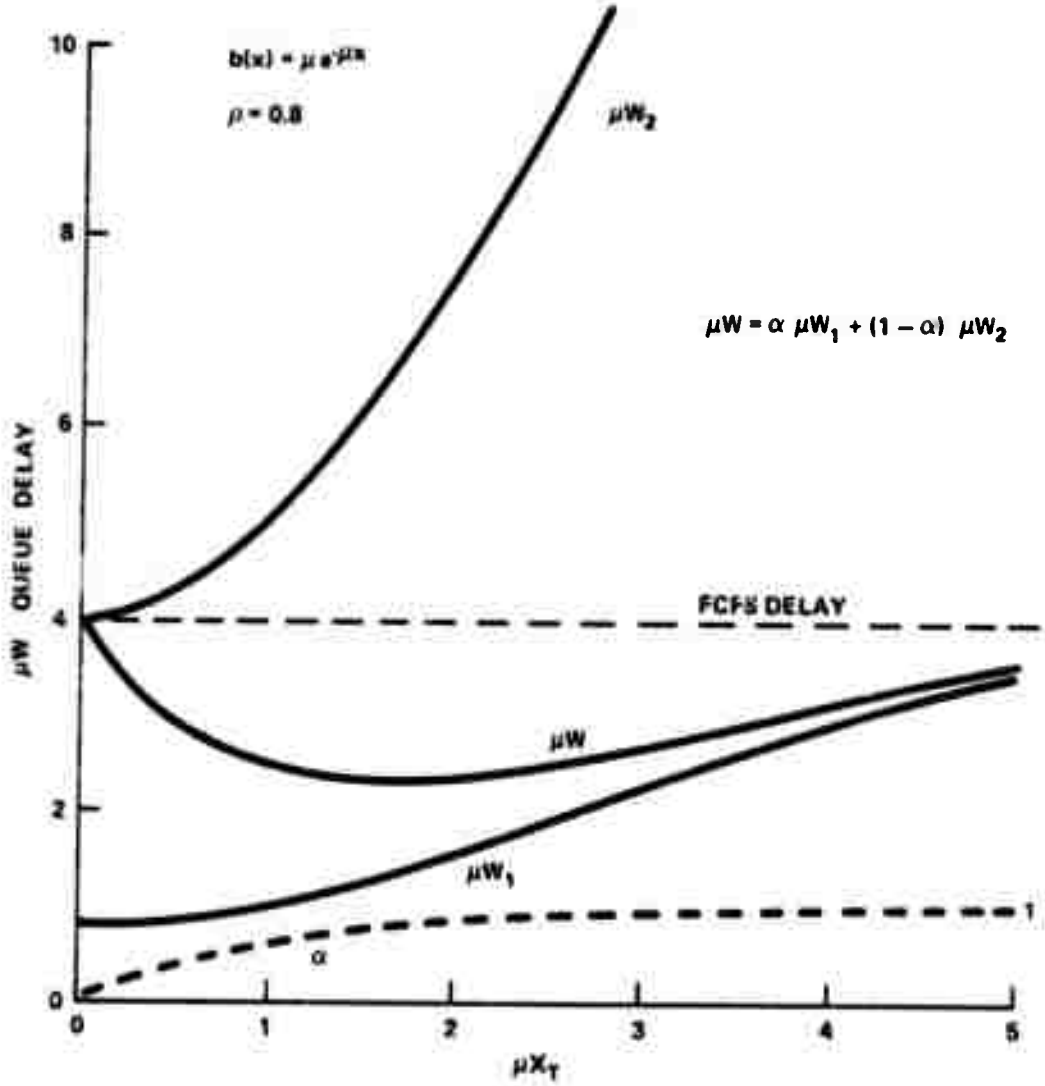


Figure 5.1.1 Plots of the Queueing Delays for Priority and Non-priority Messages as a Function of the Priority Threshold, X_T

with respect to the average service time.

The unconditional average delay, W, is also shown in the figure, where:

$$W = \alpha W_1 + (1 - \alpha)W_2 \tag{5.34}$$

since α is the probability that a message is of the priority class. The variable α is a function of X_T, being equal to the cumulative distribution function,

$$\alpha = B(X_T) \equiv Pr[x \leq X_T] \tag{5.35}$$

The optimal value of X_T can be seen to be at about $\mu X_T = 1.6$, which puts the priority threshold at 1.6 times the average message length. This value of the threshold is much higher than one would intuitively expect, and as can be seen from the corresponding value of α, means that about 80% of the messages in the system would be included in the priority class!

The optimal value of X_T is that which minimizes the function, $W(X_T)$, and Cox (C061) has shown that, although an explicit form can not be obtained, the optimal value of X_T is related to ρ by the equation:

$$\frac{1}{\rho} = 1 + \frac{e^{-\mu X_{T0}}}{\mu X_{T0} - 1}, \qquad \mu X_{T0} > 1 \tag{5.36}$$

b. <u>A comparison of results for the hyperexponential, exponential, and Erlang service densities</u>

A broad range of service time densities can be modeled by the use of hyperexponential, exponential, and Erlang functions, so these densities were selected as further examples of the optimal priority

threshold analysis. The hyperexponential density was assumed to have two equally liked exponential components with service rates of μ and $\gamma\mu$;

$$b(x;\gamma,\mu) = 0.5\mu e^{-\mu x} + 0.5\gamma\mu e^{-\gamma\mu x} \qquad 0 \leq x \tag{5.37}$$

For this service time density function, the result of Theorem 5.1.1 can be shown to be:

$$W(X_T) = W(FCFS) \cdot \left[\frac{1 - \alpha\rho}{1 - \rho\left(\frac{\gamma\psi}{1 + \gamma}\right)}\right] \tag{5.38}$$

where:

$$W(FCFS) = \frac{\rho}{1 - \rho} \cdot \frac{(1 + \gamma^2)}{\mu\gamma(1 + \gamma)} \tag{5.39}$$

$$\psi = [1 - (1 + \mu X_T)e^{-\mu X_T}] + \frac{1}{\gamma}[1 - (1 + \gamma\mu X_T)e^{-\gamma\mu X_T}] \tag{5.40}$$

and:

$$\alpha = B(X_T) = 0.5[1 - e^{-\mu X_T}] + 0.5[1 - e^{-\gamma\mu X_T}] \tag{5.41}$$

The corresponding relationship between ρ and the optimal priority threshold, X_{T0}, can also be shown to be:

$$\frac{1}{\rho} = 1 + \frac{0.5[e^{-\mu X_{T0}} + \frac{1}{\gamma} e^{-\gamma\mu X_{T0}}]}{\mu X_{T0} - \left(\frac{1 + \gamma}{2\gamma}\right)} \tag{5.42}$$

which for $\gamma = 1$, becomes the exponential result of Equation (5.36).

The form of the hyperexponential density of Equation (5.37) is useful when one is concerned with the ratio of the service rates, and expects the mean service time to vary as a function of this value,

since:

$$\bar{x} = E[x] = \frac{1 + \gamma}{2\gamma\mu} \tag{5.43}$$

However, if a constant mean service time is desired, Equations (5.38) through (5.42) can be modified by the substitution of:

$$\mu = \frac{1 + \gamma}{2\gamma\bar{x}} \tag{5.44}$$

which follows directly from Equation (5.43), with the expected value of x being a constant.

The hyperexponential density of Equation (5.37) is actually a family of functions, if we let the value of γ vary as a parameter. The mean value was given in Equation (5.43) and with the second moment being:

$$E[x^2] = \frac{1 + \gamma^2}{(\gamma\mu)^2} \tag{5.45}$$

the variance can be shown to be:

$$\sigma^2 = \frac{1}{4(\gamma\mu)^2}[3 - 2\gamma + 3\gamma^2] \tag{5.46}$$

such that the coefficient of variation is:

$$C \equiv \frac{\sigma}{\bar{x}} = \left[\frac{3 - 2\gamma + 3\gamma^2}{1 + 2\gamma + \gamma^2}\right]^{1/2} \tag{5.47}$$

The coefficient of variation will reach its maximum value of the square root of three at both extreme values of γ; namely at zero and as it becomes arbitrarily large, and will be minimum of one at γ equal to unity.

The Erlang density can also be considered to be a family of functions, with the parameter, K, being any positive integer greater than zero, such that:

$$b(x;K,\mu) = \frac{K\mu}{(K-1)!}(K\mu x)^{K-1}e^{-K\mu x} \qquad 0 \leq x \tag{5.48}$$

For K equal to unity, the function reverts to the simple exponential case, just as the hyperexponential did for γ equal to unity. Thus Erlang densities represent one half of the spectrum of functions which includes the exponential as a central function and the hyperexponential as the other half, with coefficients of variation of less than unity for the Erlang case, and greater than unity for the hyperexponential.

We will only consider the case of K equal to two for the Erlang family, since a general solution would require a closed form representation of the integral,

$$\alpha = \frac{K\mu}{(K-1)!}\int_0^{X_T}(K\mu x)^{K-1}e^{-K\mu x}dx \tag{5.49}$$

which is a variation of the incomplete gamma function and is known not to be expressible in closed form. However, the lack of a general solution is of little concern because; (1) our a priori expectation is that the service densities will be more hyperexponential-like due to the composite user and computer generated traffic, and (2) the effect of priority handling is most dramatic when there is a large coefficient of variation for the service times.

For the case of K equal to two, the queueing delay expression from Theorem 5.1.1 becomes:

$$W(X_T) = W(FCFS) \cdot \left[\frac{1 - \alpha\rho}{1 - \rho\phi}\right] \tag{5.50}$$

where:

$$W(FCFS) = \frac{(3/4)\rho}{\mu(1 - \rho)} \tag{5.51}$$

$$\phi = \left[1 + 2\mu X_T + \frac{(2\mu X_T)^2}{2}\right] e^{-2\mu X_T} \tag{5.52}$$

and

$$\alpha = 1 \cdot (1 - 2\mu X_T) e^{-2\mu X_T} \tag{5.53}$$

This result is plotted in Figure 5.1.2 for the case of ρ equal to 0.6, along with the corresponding delay functions for the exponential and hyperexponential service densities. The locus of the optimal threshold points is also shown and both sets of curves indicate the spectrum of results which are obtained from the Erlang and hyperexponential families.*

An interesting relationship can be noted between the degree of improvement relative to the FCFS delay and the location of the optimal threshold relative to the mean. For example, for ρ equal to 0.6, the exponential density has an optimal value of X_T of about 1.4 times the mean and the FCFS delay is about 1.4 times the delay at the optimal location. This observed relationship can be expressed as:

---

*The limiting case of γ equal to zero does not behave as might be expected near the origin due to the sharp cusp which forms, and eventually degenerates into an impulse function.

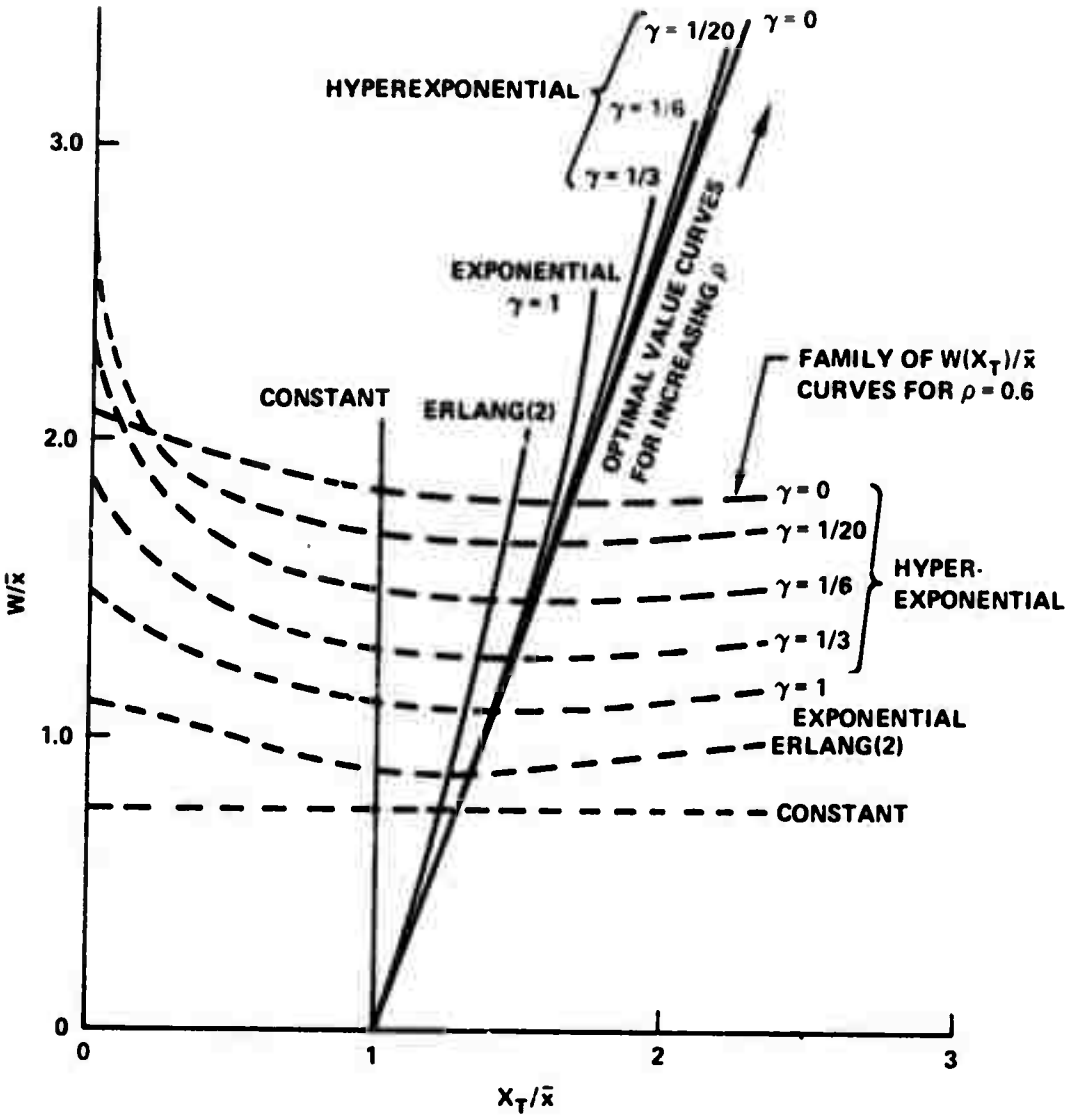


Figure 5.1.2 The Locus of Optimal Value Points for Various Service Time Density Functions

$$\frac{W(FCFS)}{W(X_{T0})} = \frac{X_{T0}}{\bar{x}}$$

and will later be shown to be true in general in Theorem 5.1.2.

The relationship between the value of ρ and the optimal threshold location can be shown to be:

$$\frac{1}{\rho} = 1 + \frac{(1 + \mu X_{T0})e^{-2\mu X_{T0}}}{\mu X_{T0} - 1} \tag{5.54}$$

for the Erlang (2) service density, and this function is plotted in Figure 5.1.3, which also shows a comparison with the hyperexponential family. The set of curves includes the exponential result of Cox (CO61), which again displays an intermediate behavior between that of the hyperexponential and Erlang functions.

c. <u>A comparison of three p.d.f.'s with same mean and variance</u>

The exponential, hyperexponential, and Erlang functions were of interest since they combine to cover a range of possible distribution shapes. However, they also have varying moments and it is difficult to separate the effects that are dependent on density shape versus those that are a function of the moments. To obtain some measure of this difference in effects, we shall consider three density functions that have the same first and second moments, but that differ significantly in their shape.

We know from the Pollaczek-Khinchine formula that the FCFS queue delay for all three densities will be the same, since this delay is a function of the first two moments of the service time for an M/G/1

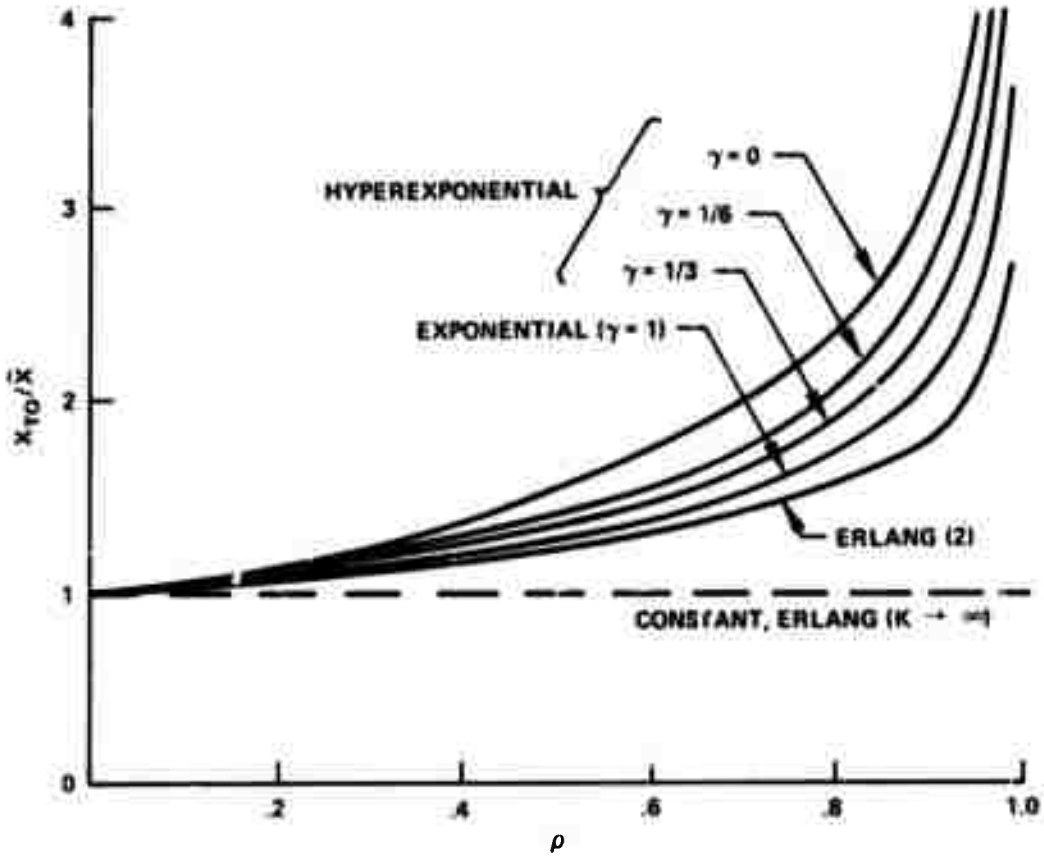


Figure 5.1.3 The Relationship Between the Values of ρ and the Normalized Priority Tnreshold for a Spectrum of Density Functions

system, i.e.:

$$E[\text{queue delay}] = \frac{\rho}{1-\rho} \quad \frac{\overline{x^2}}{2\bar{x}} \tag{5.55}$$

Therefore, the W versus X_T curves for all three of the densities will have the same starting and asymptotic values since this value is the FCFS delay. Our real concern will therefore be with the shape of the W versus X_T curve near the optimal value and with the resulting latitude in the selection of the priority threshold.

The three functions which we will consider are the hyper-exponential density, a truncated 1/x density, and a discrete distribution, with each distribution having a mean value of 30, a variance of 1350, and a resulting coefficient of variation of 1.22. The three functions are:

1. $b(x) = 0.5\mu e^{-\mu x} + 0.5\gamma\mu e^{-\gamma\mu x}, \qquad 0 \leq x$ (5.56a)

   for $\mu = 1/15$ and $\gamma = 1/3$

2. $b(x) = 1/(x \ln M), \qquad 1 \leq x \leq M$ (5.56b)

   for $M = 150$

3. $b(x) = 0.5\delta(x-5) + 0.25\delta(x-18) + 0.25\delta(x-98)$ (5.56c)

Where the delta function $\delta(x-a)$ is non-zero only for $x = a$, where it has unit probability area. The three functions are plotted in Figure 5.1.4 and can be seen to differ in shape by a considerable degree. Figure 5.1.5 shows the resulting plots of average queueing delay versus the location of the priority threshold, and these three curves are surprisingly similar, but do not have the exact same delay

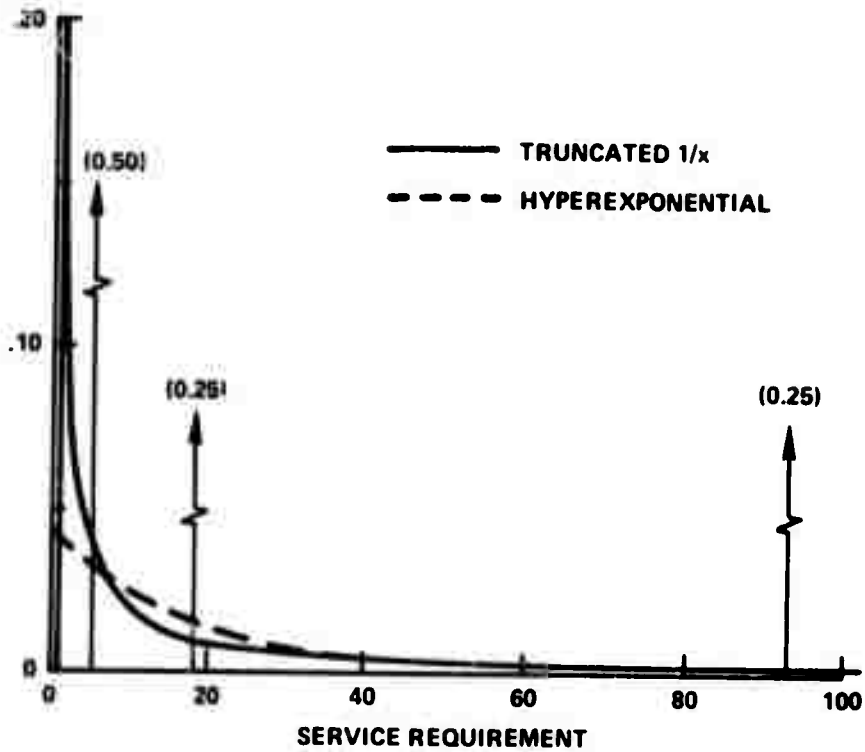


Figure 5.1.4 The Three Density Functions

reduction. We observed in the examples of part (b), that the delay reduction and the optimal priority threshold were related by:

$$\frac{W(FCFS)}{W(X_{T0})} = \frac{X_{T0}}{\bar{x}}$$

We shall later generalize this result (in Theorem 5.1.2), and it will therefore show that the optimal threshold differs only slightly for the three distributions.

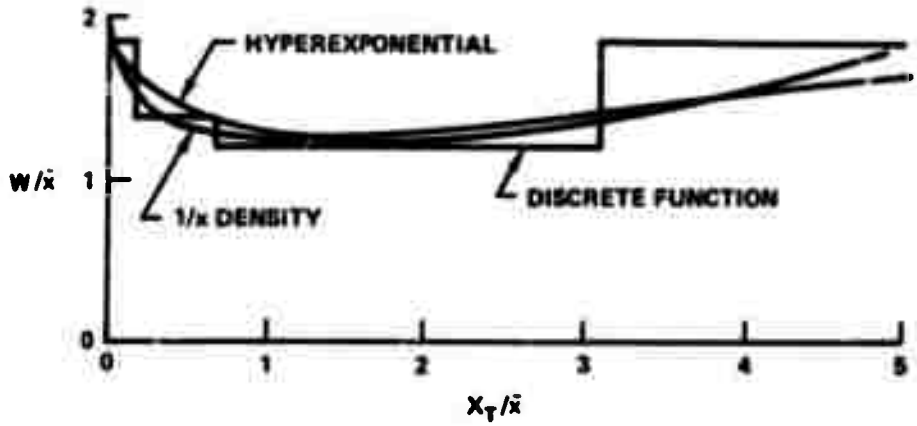


Figure 5.1.5 A Comparison of Queueing Delay Versus Priority Threshold for the Hyperexponential, Discrete, and 1/x Service Time Distributions

5.1.22 Further Analysis of the Two Priority Class Case

A close look at the priority analysis for the various different service time density functions suggests several areas of possible generalization. These observations include the equivalence of the two ratios,

$$\frac{W(FCFS)}{W(X_{T0})} = \frac{X_{T0}}{\bar{x}}$$

and the similarity of the equations relating ρ and X_{T0} for various service densities. We shall show that one can indeed generalize these relationships in the following analysis.

Theorem 5.1.2

The ratio of the minimal queueing delay, $W(X_{T0})$, to the FCFS

delay, is equal to the inverse of the ratio of X_{T0} to the mean service time, $\bar{x}$, for any arbitrary service density.

$$\frac{W(X_{T0})}{W(FCFS)} = 1\Big/\left(\frac{X_{T0}}{\bar{x}}\right) \tag{5.57}$$

Proof:

The optimal priority threshold location, X_{T0}, is found by differentiating the delay expression of Equation (5.12),

$$W(X_T) = \frac{W_0(1 - \alpha\lambda\bar{x})}{(1 - \rho)(1 - \alpha\lambda\bar{x}_1)}$$

However, one would expect that this differentiation could only be carried out for a particular service density, b(x). If we proceed with the differentiation in general, we must evaluate the roots of:

$$\frac{dW}{dX_T} = \frac{d}{dX_T}\left[\frac{W_0(1 - \alpha\rho)}{(1 - \rho)(1 - \alpha\lambda\bar{x}_1)}\right] = 0 \tag{5.58}$$

The W_0 and $1 - \rho$ factors are not functions of X_T, and do not need to be considered. We are left with:

$$\frac{d}{dX_T}\left[\frac{1 - \alpha\rho}{1 - \alpha\lambda\bar{x}_1}\right] = 0 \tag{5.59}$$

Where

$$\alpha = \int_0^{X_T} b(x)dx \tag{5.60}$$

and:

$$\alpha\bar{x}_1 = \int_0^{X_T} xb(x)\,dx \tag{5.61}$$

The differentiation results in:

$$\rho(1 - \alpha\lambda\bar{x}_1)\frac{d\alpha}{dX_T} - \lambda(1 - \alpha\rho)\frac{d}{dX_T}(\alpha\bar{x}_1) = 0$$

or when we substitute for α and $\alpha\bar{x}_1$, as above:

$$\rho(1 - \alpha\lambda\bar{x}_1)\frac{d}{dX_T}\int_0^{X_T} b(x)dx - \lambda(1 - \alpha\rho)\frac{d}{dX_T}\int_0^{X_T} xb(x)dx = 0$$

The differentiation of the integrals is easily handled by the relationship that:

$$\frac{d}{dy}\int_a^y f(z)dz = f(y)$$

and when common terms are factored out, we are left with:

$$\rho(1 - \alpha\lambda\bar{x}_1) - \lambda X_{T0}(1 - \alpha\rho) = 0 \tag{5.62}$$

The terms within the parentheses are those from the original expression, and are therefore the subject of our investigation. In particular, we are interested in the ratio:

$$\frac{1 - \alpha\rho}{1 - \alpha\lambda\bar{x}_1} = \frac{\rho}{\lambda X_{T0}}$$

Since we know from the definition of ρ

$$\rho = \lambda\bar{x}$$

Then:

$$\frac{1 - \alpha\rho}{1 - \alpha\lambda\bar{x}_1} = \frac{\bar{x}}{X_{T0}} \tag{5.63}$$

which when substituted into the delay equation produces:

$$W(X_{T0}) = \frac{W_0}{1 - \rho} \cdot \frac{1}{\left(X_{T0}/\bar{x}\right)} = W(FCFS) \Big/ \left(\frac{X_{T0}}{\bar{x}}\right)$$

This is then a proof of the theorem.

<u>Corollary 1</u>

The optimal priority threshold will always be greater than or equal to the mean service requirement, regardless of the service time distribution.

<u>Proof</u>:

The theorem proves that:

$$\frac{W(FCFS)}{W(X_{T0})} = \frac{X_{T0}}{\bar{x}}$$

As the arrival rate, λ, becomes very small, we know that $W(X_{T0})$ becomes approximately equal to $W(FCFS)$, since from Theorem 5.1.1,

$$\lim_{\lambda \to 0} \frac{W(FCFS)}{W(X_{T0})} = \lim_{\lambda \to 0} \frac{1 - \alpha\lambda\bar{x}_1}{1 - \alpha\lambda\bar{x}} = 1 \tag{5.64}$$

and that:

$$W(FCFS) \geq W(X_{T0}) \qquad \forall\ \lambda \tag{5.65}$$

because $\bar{x}_1 \leq \bar{x}$. Therefore, X_{T0} and $\bar{x}$ are related by the

inequality,

$$X_{T0} \geq \bar{x} \tag{5.66}$$

with equality for the special cases of $\lambda = 0$ or for the degenerate case of $\bar{x}_1 = \bar{x} =$ constant. This completes the proof of the corollary.

Another relationship which seemed to have some possible generality was the function relating ρ to X_{T0} with an observed form of:

$$\frac{1}{\rho} = 1 + \frac{f(X_{T0}, \bar{x})}{X_{T0} - \bar{x}} \tag{5.67}$$

This relationship is generalized in the following theorem.

<u>Theorem 5.1.3</u>

The general relationship between the optimal priority threshold, and the traffic intensity is:

$$\frac{1}{\rho} = 1 + \frac{(1 - \alpha)[\bar{x}_2 - X_{T0}]}{X_{T0} - \bar{x}} \tag{5.68}$$

for any arbitrary service density, $b(x)$ where α is the fraction of messages that are of the priority class, and $\bar{x}_2$ is the average service time of a non-priority message.

<u>Proof</u>:

We know from Equation (5.62) that,

$$\rho[1 - \alpha\lambda\bar{x}_1] = \lambda X_{T0}[1 - \alpha\rho]$$

where α and $\alpha\bar{x}_1$ are computed at X_{T0}, from Equations (5.60) and (5.61).

Expanding the above equation, we obtain:

$$\rho - \alpha\lambda\bar{x}_1\rho + \alpha\lambda X_{T0}\rho = \lambda X_{T0}$$

or, by recalling that $\rho = \lambda\bar{x}$,

$$\lambda[\bar{x} - \alpha\bar{x}_1\rho + \alpha X_{T0}\rho] = \lambda X_{T0}$$

which results in:

$$[\alpha X_{T0} - \alpha x_1]\rho = X_{T0} - \bar{x}$$

If we solve for $1/\rho$ we obtain:

$$\frac{1}{\rho} = \frac{\alpha(X_{T0} - \bar{x}_1)}{X_{T0} - \bar{x}} \tag{5.69}$$

which by adding and subtracting like quantities, one can write as:

$$\frac{1}{\rho} = \frac{X_{T0} - \bar{x}}{X_{T0} - \bar{x}} + \frac{\alpha(X_{T0} - \bar{x}_1) - (X_{T0} - \bar{x})}{X_{T0} - \bar{x}}$$

or:

$$\frac{1}{\rho} = 1 + \frac{(\bar{x} - \alpha\bar{x}_1) - (1 - \alpha)X_{T0}}{X_{T0} - \bar{x}} \tag{5.70}$$

We recognize from Equation (5.23), that:

$$\bar{x} - \alpha\bar{x}_1 = (1 - \alpha)\bar{x}_2 \tag{5.71}$$

such that

$$\frac{1}{\rho} = 1 + \frac{(1 - \alpha)[\bar{x}_2 - X_{T0}]}{X_{T0} - \bar{x}}$$

which is the desired result.

Theorem 5.1.3 can be utilized to determine the plot of X_{T0} versus ρ for any given service time density function. For the exponential case considered by Cox (C061), we can write the result by inspection since we know that:

$$\alpha(X_{T0}) = B(X_{T0}) = e^{-\mu X_{T0}} \tag{5.72}$$

and the mean of the exponential tail is,

$$\bar{x}_2 = X_{T0} + \bar{x}$$

such that:

$$\frac{1}{\rho} = 1 + \frac{\bar{x}\, e^{-\mu X_{T0}}}{X_{T0} - \bar{x}} \tag{5.73}$$

This equation is equivalent to the result given in Equation (5.36) and plotted in Figure 5.1.3.

5.1.2.3 Introduction of a Delay Cost Function

A surprising aspect of the priority threshold analysis of Section 5.1.2 was the fact that a large fraction of messages should be included in the priority class. A more intuitive approach might select only 5% to 20% of the messages for such preferential treatment, and while intuition should not rule over analysis, it does signal a potential oversight in the model.

If one considers the various classes of messages that might be serviced by a network, the list would include the short messages such as single character echoplex traffic, terse commands (such as /LOGIN), short lines of code, and brief computer responses, and would cover a wide range of message sizes up to the transmission of large files. These classes are ordered by message length, but this same ordering applies reasonably well to one's response time expectations, and therefore a weighting of delay as a function of the message length seems appropriate. One such weighting is introduced by the utilization of a delay cost function, $c(x)$, which is defined as the "cost" of delaying a message by one second given that the message requires x seconds of service. This cost can be expressed in arbitrary units, and is merely a relative weighting of the delay values for the various message lengths.

An alternative approach would be to implement N priority classes for the different message classifications, but the cost function approach may allow us to accomplish nearly the same effect without the additional overhead of a multiple priority scheme.* We consider only cost functions which are a monotonically non-increasing function of the service requirement, x, and therefore a family of exponential curves is felt to be adequate, but they should be normalized to have a common expected cost over all message lengths, with a convenient

---

*Cox and Smith (C061) consider the case of N priority classes, with the cost of delay for any message in the ith class being w_i units of cost per second. In our example, the cost function is continuous, rather than being discrete by class.

normalization being:

$$E[c(x)] = 1 \tag{5.74}$$

This expected value will be a function of the service time density, $b(x)$, but for the exponential cost function, the expected value takes on a particularly interesting form as shown in the following theorem.

Theorem 5.1.4

An exponential weighting for the cost of queueing delay as a function of the service requirement, x, and normalized such that the expected cost is unity, is:

$$c(x) = e^{-Kx}/B^*(K) \tag{5.75}$$

where $B^*(K)$ is the Laplace transform of the service time density, and K is an arbitrary positive constant that controls the degree of the weighting.

Proof:

An exponential cost function is desired of the form:

$$c(x) = a\, e^{-Kx}, \qquad 0 \leq x \tag{5.76}$$

with the constraint that:

$$E[c(x)] = 1.$$

The expected value of the cost function is, in general:

$$E[c(x)] = \int_0^\infty c(x)b(x)dx \tag{5.77}$$

With the exponential form of $c(x)$, we obtain:

$$E[c(x)] = \int_0^\infty a\ e^{-kx} b(x)dx \tag{5.78}$$

which can be recognized as being of the form of a Laplace transform,

$$L\{b(x)\} = B^*(s) = \int_0^\infty e^{-sx} b(x)dx \tag{5.79}$$

which for $s = K$ results in:

$$E[c(x)] = a\ B^*(K)$$

Since this expected value is equal to unity, we have the relationship:

$$a = 1/B^*(K) \tag{5.80}$$

and therefore the generalized exponential cost function is found to be:

$$c(x) = e^{-Kx}/B^*(K) \tag{5.81}$$

which proves the theorem.

For the exponential service density,

$$b(x) = \mu e^{-\mu x}, \qquad 0 \leq x$$

the Laplace transform is:

$$B^*(s) = \frac{\mu}{s + \mu}$$

and the cost function becomes:

$$c(x) = \frac{K + \mu}{\mu} e^{-Kx} \tag{5.82}$$

A more convenient notational form results if we make the substitution of variables,

$$k' = k/\mu \tag{5.83}$$

such that:

$$c(x) = (k' + 1)e^{-k'\mu x} \tag{5.84}$$

A family of exponential cost functions are thereby produced, ranging from the extreme case of a constant, $c(x)$ equal to unity, for k' equal to zero, to very sharply peaked functions near the origin. The latter would heavily weight the delay for messages requiring very short service times, as shown in Figure 5.1.6.

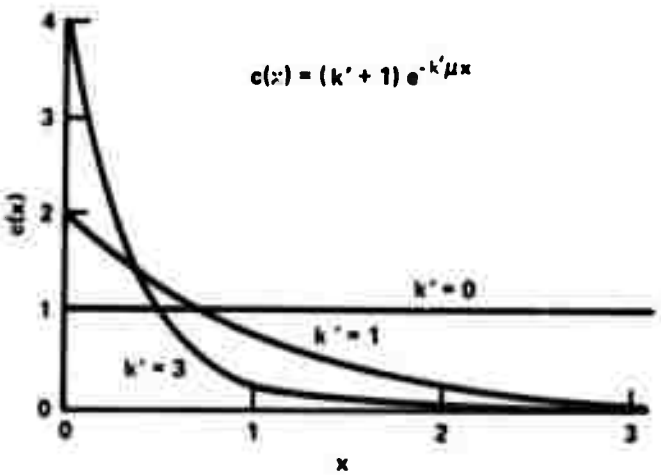


Figure 5.1.6 Examples of the Exponential Family of Cost Functions

The introduction of a delay cost function changes the priority analysis of Section 5.1.2, such that for a message requiring x seconds of service, the expected delay cost is:

$$E[\text{delay cost} \mid x] = c(x) \cdot E[\text{queue delay} \mid x] \tag{5.85}$$

For the two priority class system,

$$E[\text{queue delay} \mid x \leq X_T] = W_1$$

and:

$$E[\text{queue delay} \mid x > X_T] = W_2$$

such that the unconditional delay cost function becomes:

$$E[\text{delay cost}] = W_1 \int_0^{X_T} c(x)b(x)dx + W_2 \int_{X_T}^{\infty} c(x)b(x)dx$$

We shall introduce the notation:

$$\alpha_c \triangleq \int_0^{X_T} c(x)b(x)dx \tag{5.86}$$

resulting in:

$$E[\text{delay cost}] = \alpha_c W_1 + (1 - \alpha_c)W_2$$

The equations for W_1 and W_2 are not functions of $c(x)$, and therefore are as in Equations (5.16) and (5.19). Substitution and simplification produces:

$$E[\text{delay cost}] = \frac{W_0}{1 - \rho} \cdot \frac{1 - \alpha_c \rho}{1 - \alpha\lambda\bar{x}_1} \tag{5.87}$$

in which both α and α_c appear. For the exponential cost function and the exponential service density, the equation for α_c becomes:

$$\alpha_c = \int_0^{X_T} \mu(k' + 1)e^{-(k'+1)\mu x} dx$$

or:

$$\alpha_c = 1 - e^{-(k'+1)\mu X_T} \tag{5.88}$$

The resulting cost-weighted delay as a function of the priority threshold, is shown in Figure 5.1.7. The reader should recall that, with the

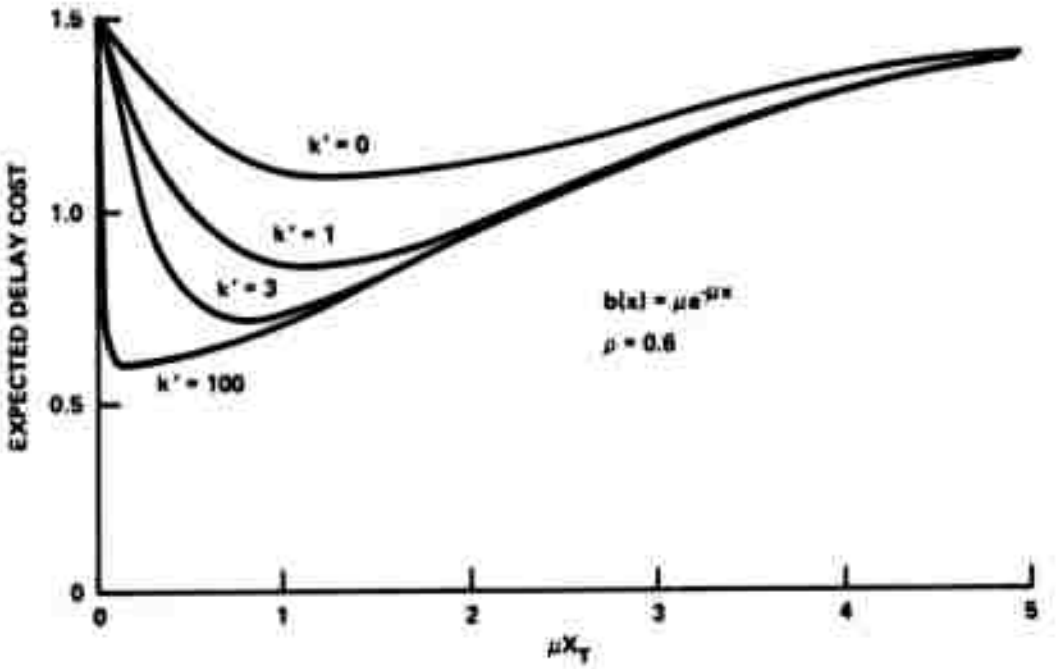


Figure 5.1.7 Optimal Priority Threshold Curves for Several Parameter Values of the Exponential Cost Function

exception of $k' = 0$, the delay is not scaled in normal time units, so the apparent delay reduction for $k' > 0$ is merely due to the lesser weighting of the delay for long messages. However, the net effect of the cost function is to decrease the optimal priority threshold, x_T, which also reduces the fraction of the messages in the priority class. This reduction is consistent with one's intuitive expectations, which implies that some similar cost-weighting may be inherent in the intuition.

5.1.2.4 Reduction in Buffering Needs Due to Priority Handling of Messages

Little's result, $\bar{n} = \lambda W$, would lead one to believe that we can reduce the buffering needs at the same time that we reduce the expected queueing delay, since the number of messages in the queue is proportional to the average queueing delay. However, the extent to which the delay reduction results in a similar reduction in the buffering requirements is dependent on the storage allocation scheme. For example, if storage is allocated with an arbitrarily small number of bits in a segment, then the priority system will have no storage advantage since bits will be freed at the service rate of the channel regardless of whether long or short messages are being served. The other extreme is when storage is allocated in fixed length blocks, in which case serving short messages frees these blocks of storage most rapidly. These two allocation schemes are bounds on the effect of priorities, and will be discussed in more detail in the following paragraphs.

a. The continuous storage allocation model

Although we know from the above argument that introduction of priorities should not effect the storage considerations for this case, let us look at the analysis in more detail since it brings out several interesting aspects of the problem. The expected storage requirement is:

$$E[\text{storage requirement}] = C[\bar{n}_1\bar{x}_1 + \bar{n}_2\bar{x}_2] \tag{5.89}$$

where C is the service rate in bits per second. Using Little's result again we can obtain:

$$E[\text{storage requirement}] = \lambda[\alpha W_1 C\bar{x}_1 + (1 - \alpha)W_2 C\bar{x}_2] \tag{5.90}$$

in which $C\bar{x}_1$ is the expected length (in bits) of a priority message, and W_1 is the expected time that such a message requires this storage due to its queue delay. (Storage is also required during service and until an acknowledgement has been received but we are not considering these effects here.)* These products of storage needed and the time period for which it is held form a measure of buffer utilization, namely the bit-seconds of storage requirement.

If we substitute for the values W_1, W_2, and $\bar{x}_2$ and simplify the resulting expression we obtain:

$$E[\text{storage requirement}] = \lambda C\bar{x}\left(\frac{W_0}{1 - \rho}\right) \tag{5.91}$$

*The time required for serial transmission, propagation delays, and acknowledgement return are independent of the factors being considered here.

which is merely the FCFS storage requirement. This result could have been obtained directly from the conservation law since this law is based on the fact that the expected amount of unfinished work in the queue is independent of the queue discipline.

b. The fixed length storage allocation model

If all message lengths are less than some finite maximal length, L, and if storage is allocated in segments of this maximal length, then the expected storage requirement is:

$$E[\text{storage requirement}] = L[\bar{n}_1 + \bar{n}_2] \tag{5.92}$$

or from Little's result,

$$E[\text{storage requirement}] = \alpha\lambda W_1 L + (1 - \alpha)\lambda W_2 L$$

which upon substitution for W_1 and W_2 becomes:

$$E[\text{storage requirement}] = \lambda L \frac{W_0}{1 - \rho} \frac{1 - \alpha\lambda\bar{x}}{1 - \alpha\lambda\bar{x}_1} \tag{5.93}$$

The expected reduction in the storage requirement due to the priorities is the same as the reduction in the expected queueing delay, as would be expected since in this case the storage requirement is directly proportional to the number of messages in the system. In fact, Equation (5.93) could have been obtained directly from Equation (5.12) and Little's result, since the proportionality factor is know to be L.

Since this is the same reduction factor as was previously seen in terms of the average queueing delay, all of those plots are equally applicable to the buffer considerations. However, one

additional plot is shown in Figure 5.1.8 to indicate the typical range of values of $\bar{n}$, and to show the relative insensitivity of the selection of the priority threshold. Even when the threshold is varied from one-half to twice the optimal value, the improvement factor is only slightly degraded for this example case.

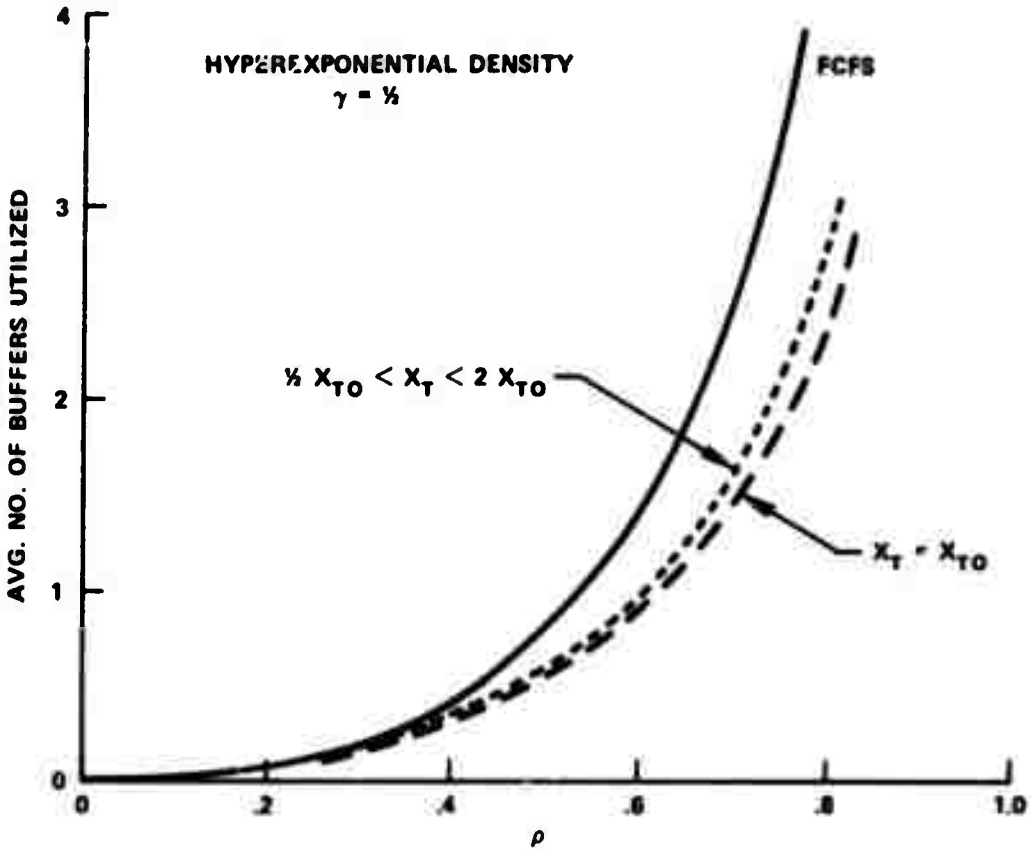


Figure 5.1.8. The Effect of Priorities on the Buffer Requirements for the Case of Fixed Length Buffer Allocation

5.1.2.5 Experimental Verification of the Priority Model

The priority model was evaluated via an experiment in which the priority threshold parameter of the IMP was varied and the resulting queueing delays were measured for several values of the traffic intensity, ρ. The values of ρ were selected using the techniques of Section 4.1.3, with the number of active generators being selected as the alterable parameter. (Changes of the message lengths would have complicated the comparison with the theoretical results since both α and $\bar{x}$ would vary.) The set of simulated parameter values were selected from Figure 4.1.3, with the simulated values of ρ of 0.3, 0.5, 0.6, and 0.8 being generated by the selection of 5, 8, 10, and 14 message generators respectively. The time parameter, T_a, was set accordingly at 200 msec. for all of the tests. An average message length of 320 bits was selected as a compromise between (1) desiring a reasonably large variable component for the message service time, and (2) being able to neglect the effect of message segmentation into packets. The service time for a message therefore consists of an average of $1/\mu = 6.4$ msec. for the exponentially distributed message length portion, plus an average overhead component of $X_0 = 2.9$ msec. for the header and line control characters, as shown in Figure 5.1.9. The figure also indicates the service time requirement for the acknowledgement traffic which occurs due to the RFNM (Request For Next Message) mechanism, (i.e., each RFNM must be acknowledged). Acknowledgements require a service time of $X_a = 3.0$ msec. for serial transmission, and occur with an average arrival rate, λ_a, that is equal to the average arrival rate of messages, λ_m, such that the composite average arrival rate

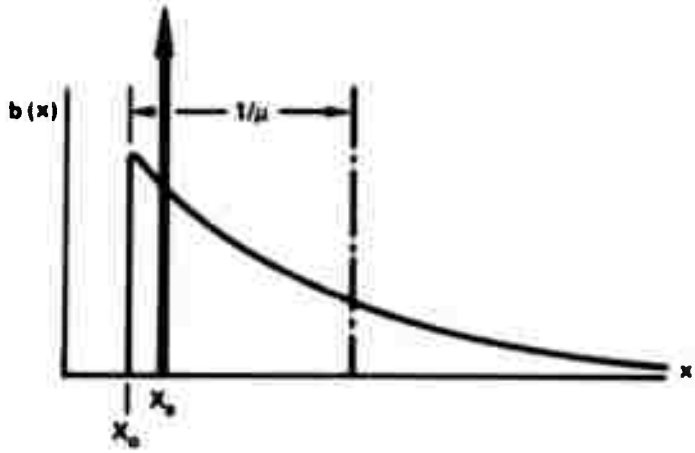


Figure 5.1.9 The Composite Service Time Density

becomes:

$$\lambda = \lambda_a + \lambda_m$$

and the composite traffic intensity is:

$$\rho_c = \lambda_a X_a + \lambda_m (X_0 + 1/\mu)$$

Since λ_a and λ_m are equal, the value of ρ_c can be written as:

$$\rho_c = \lambda \frac{X_a + X_0 + 1/\mu}{2}$$

which is of the usual form, i.e., the product of the average arrival rate and the average service time.

The effect of the acknowledgement traffic was shown in Equation (4.5) to result in an average message delay of:

$$W(\text{FCFS message traffic}) = \frac{W_U}{(1 - \rho_a)(1 - \rho_c)}$$

where ρ_a is the traffic intensity of acknowledgements,

$$\rho_a = \lambda X_a/2$$

and W_0 is the expected time to complete any existing service, which is from Equation (5.15):

$$W_0 = \lambda E[x^2]/2$$

Since the service is the composite of two components, W_0 becomes:

$$W_0 = \lambda_a E[x^2 \mid \text{an ACK}]/2 + \lambda_m E[x^2 \mid \text{a mes.}]/2$$

which can be shown to be:

$$W_0 = \frac{\lambda}{2}\left[(1/\mu)^2 + \frac{X_0}{\mu} + \frac{(X_0)^2 + (X_a)^2}{2}\right]$$

For the values of the service time variables given above, the expression reduces to:

$$W_0 = \frac{\lambda}{2}[68.3]$$

and results in an expected FCFS message delay of:

$$W(\text{FCFS}) = \frac{\lambda[34.2]}{(1 - \rho_a)(1 - \rho_c)}$$

For the $\rho = 0.8$ case, λ is equal to 0.13, and ρ_a becomes equal to 0.195, which results in a theoretical FCFS queueing delay of approximately 28 msec., compared to a measured value of 32 msec. This error is quite acceptable when one considers that a 1% error in the arrival rate at a value of ρ near 0.8 will result in a queueing delay error of about 1.5 msec. The fact that the shape of the

theoretical and measured delay curves match well as a function of the threshold value, and that the two sets of values agree quite well for $\rho = 0.6$, gives good support for the validity of the model. The relative error for the $\rho = 0.5$ case is surprisingly large, but is believed to be due to two sources of error; (1) the resolution of the round trip time measurement is 0.8 msec., and (2) the queueing delay value is determined by subtracting the relatively large fixed portion of the round trip time (about 20 msec.) from the average round trip measurement. Hence, the queueing delay values for the lower values of ρ are subject to much larger relative errors, i.e., the same fixed value error has a much larger apparent effect. More precise measurements could have been made by the use of trace data, but were not felt to be justified since differences between the queueing model and the actual system performance are typically more acute at the higher values of ρ.

The data of Figure 5.1.10 is shown with the queueing delay scaled in msec., but can be normalized by dividing by the average service time of the composite traffic,

$$E[x \mid \text{composite traffic}] = 0.5(3.0 \text{ msec.}) + 0.5(9.3 \text{ msec.})$$

for an average value of 6.2 msec. Therefore, the normalized queueing delay for the $\rho = 0.8$ case is 4.5 compared to a normalized value of 4.0 for the exponential example of Figure 5.1.1. The composite density has a larger delay value due to the added delays of the higher priority acknowledgement traffic, although the exponential service density has a larger variance and hence would otherwise have the greater queueing delay.

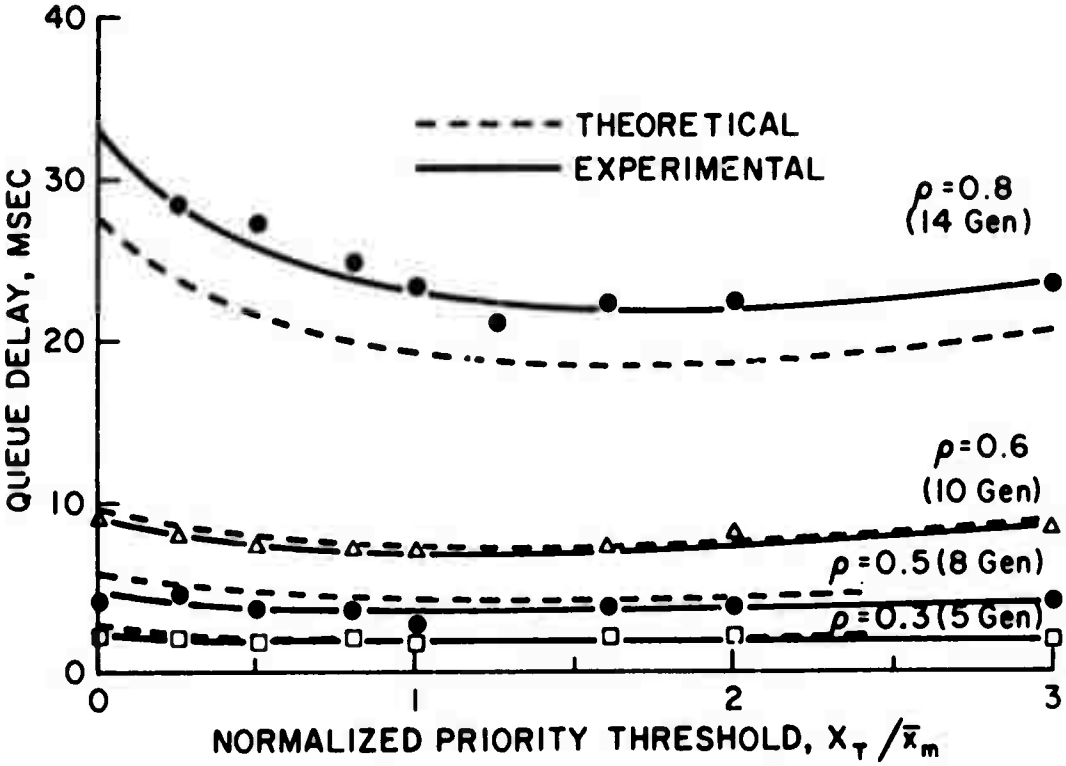


Figure 5.1.10 A Comparison of the Theoretical and Experimental Values of Queueing Delay Versus the Priority Threshold Location

The degree of improvement between the $\rho = 0.8$ cases of Figures 5.1.1 and 5.1.10 also makes an interesting comparison. For the exponential service density, the average delay was reduced by 42% compared to the FCFS delay, while the delay reduction for the composite service density was only 30%. This lesser reduction in delay is due to the fixed service time components, which cause the theoretical model of Equation (5.12) to take on the form:

$$W = W(FCFS)\ \frac{1 - \alpha\lambda[X + 1/\mu]}{1 - \alpha\lambda[X + \bar{x}_1]}$$

This relationship shows that as the fixed component, X, is increased relative to the variable component, $1/\mu$, the degree of improvement will be decreased accordingly. Future analysis and experimentation should probably consider the overhead due to the HOST-to-HOST protocol as part of this fixed component.

The general agreement between the predictions of the theoretical model and the experimental data was very encouraging, but an even more important aspect of the experiment was that it brought about improvements in the model. For example, the fixed time components that accompany each message had not been included in the original model, but became obvious when the experiment was being designed. However, a more subtle effect involved the handling of the acknowledgements for the RFNM's which had also been overlooked. This effect was uncovered while attempting to reconcile earlier experimental data, but was finally included in the model by utilizing part of the original priority analysis. This iterative and complementary usage of the measurement and modeling techniques is believed to be a significant advantage over the use of either approach by itself.

5.1.3 The Introduction of Additional Priority Classes

For the case of priorities being determined by an a priori knowledge of the service requirement, we can generalize the categorization process by defining the ith priority class to be that set of messages which have service requirements in the range, $x_{i-1} < x < x_i$. The fraction of the messages in the class is then:

$$\alpha_i = \int_{x_{i-1}}^{x_i} b(x)dx \tag{5.94}$$

and the expected service time, given that a message is in the ith class is:

$$\bar{x}_i = \frac{1}{\alpha_i} \int_{x_{i-1}}^{x_i} x\, b(x)dx \tag{5.95}$$

For a system with P priority classes, there are $P - 1$ priority threshold to select, subject only to the constraint that:

$$x_{T_1} < x_{T_2} < \ldots < x_{T_{P-1}} \tag{5.96}$$

This is a situation that leads to an increasingly complex optimization problem as the value of P becomes larger. To avoid this combinatorial explosion, we will only consider two special cases; (1) the three priority class case with $X_{T_2} = \beta X_{T_1}$ $\beta \geq 1$, and (2) the case of equal probability areas for each class.

5.1.3.1 An Example With Three Priority Classes

A priority threshold, X_T which divided messages into two priority classes was considered in Section 5.1.2, and we now wish to

extend this notion to additional priority classes. If we consider as an example, the case of two priority thresholds, X_{T_1} and X_{T_2} we will have three priority classes, and will define these classes and the associated variables as shown in Table 5.1.1.

TABLE 5.1.1

NOMENCLATURE FOR THE THREE PRIORITY CLASS CASE

	class 1	class 2	class 3
Priority	high	medium	low
Service times	$x \leq X_{T_1}$	$X_{T_1} < x \leq X_{T_2}$	$X_{T_2} < x$
Average queue delay	W_1	W_2	W_3
Fraction of messages	α_1	α_2	α_3
Arrival rate	$\alpha_1\lambda$	$\alpha_2\lambda$	$\alpha_3\lambda$

Following the approach of Section 5.1.1, we have:

$$W_1 = \frac{W_0}{1 - \lambda\alpha_1\bar{x}_1} \tag{5.97a}$$

$$W_2 = \frac{W_0 + \lambda\alpha_1\bar{x}_1W_1}{1 - \lambda(\alpha_1\bar{x}_1 + \alpha_2\bar{x}_2)} \tag{5.97b}$$

$$W_3 = \frac{W_0 + \lambda\alpha_1\bar{x}_1W_1 + \lambda\alpha_2\bar{x}_2W_2}{1 - \rho} \tag{5.97c}$$

and an expected queueing delay of:

$$W = \alpha_1W_1 + \alpha_2W_2 + \alpha_3W_3 \tag{5.98}$$

This result is shown in Figure 5.1.11 for the special case of an exponential service density, a traffic intensity of $\rho = 0.8$, and priority threshold of $X_{T_2} = 2X_{T_1}$. The two priority class case from Figure 5.1.12 has been superimposed to indicate the effect of the additional priority class, and as usual, the axes are normalized with respect to the mean service time. The comparison shows a relatively small improvement in the average queueing delay, but a somewhat extended latitude of threshold values for which good improvements were obtained. The delay of the third priority class shows a rapidly increasing value as X_{T_2} increases, but of course, there are relatively few such messages at high value of X_{T_2} for this distribution. The middle class messages have a delay varying from that of the highest priority messages for X_{T_1} near zero, to that of the two priority case for large values of X_{T_1}, since at both extremes, the system is essentially a two priority class system. The highest priority message delays are the same for both systems since the further subdivision of lower priority messages does not affect the class 1 messages.

In the above example, the two priority thresholds were related by $X_{T_2} = 2X_{T_1}$. We can expand this relationship by introducing the parameter, β such that:

$$X_{T_2} = \beta X_{T_1}$$

The effect of varying this parameter is illustrated in Figure 5.1.12 for a hyperexponential service density, and for a traffic intensity of $\rho = 0.6$. Note that for $\beta = 1$, the two thresholds are superimposed, and degenerate to the two priority class case. The improvement for

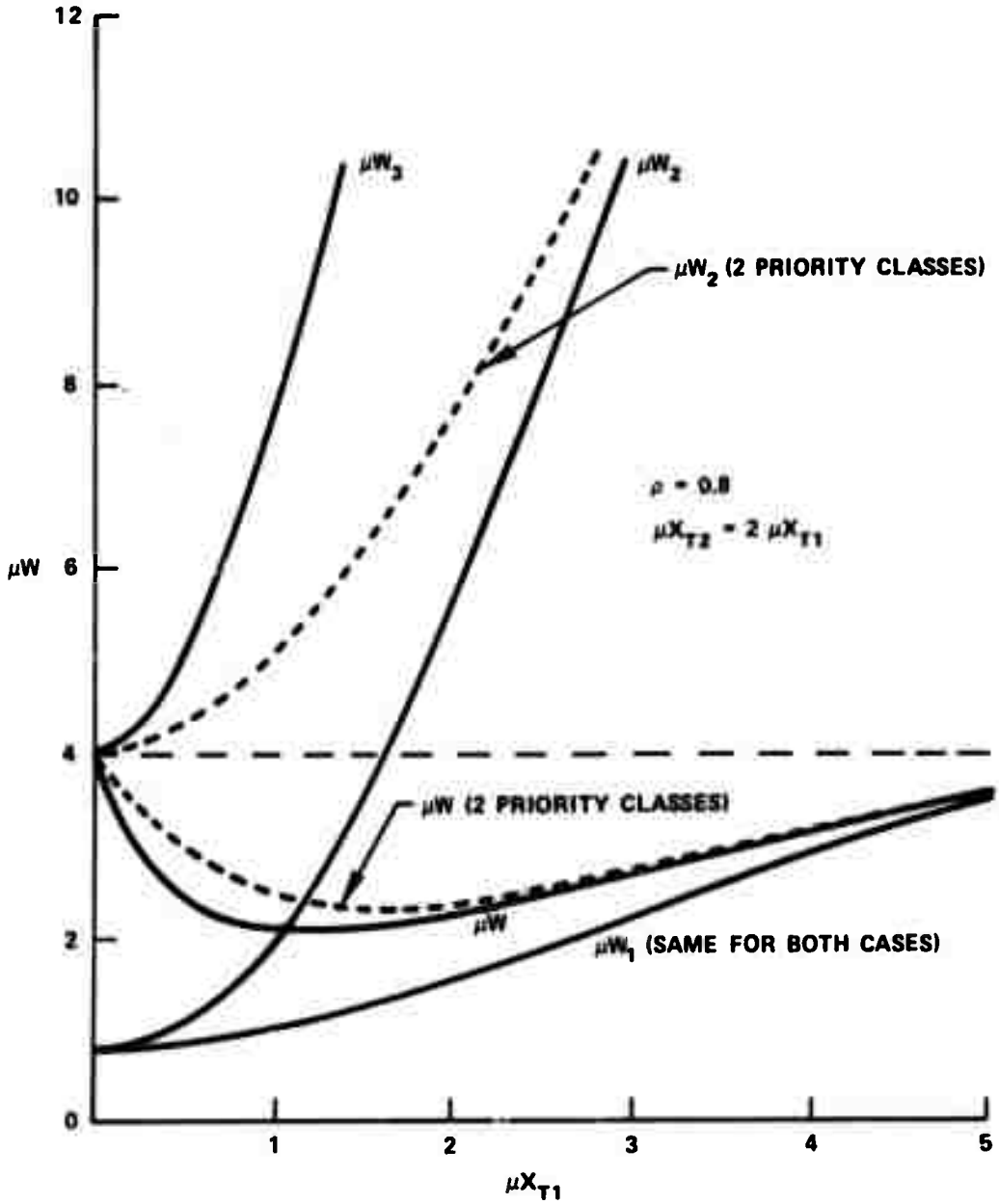


Figure 5.1.11, A Comparison of the Delay Curves for Two and Three Priority Classes. (Exponential service)

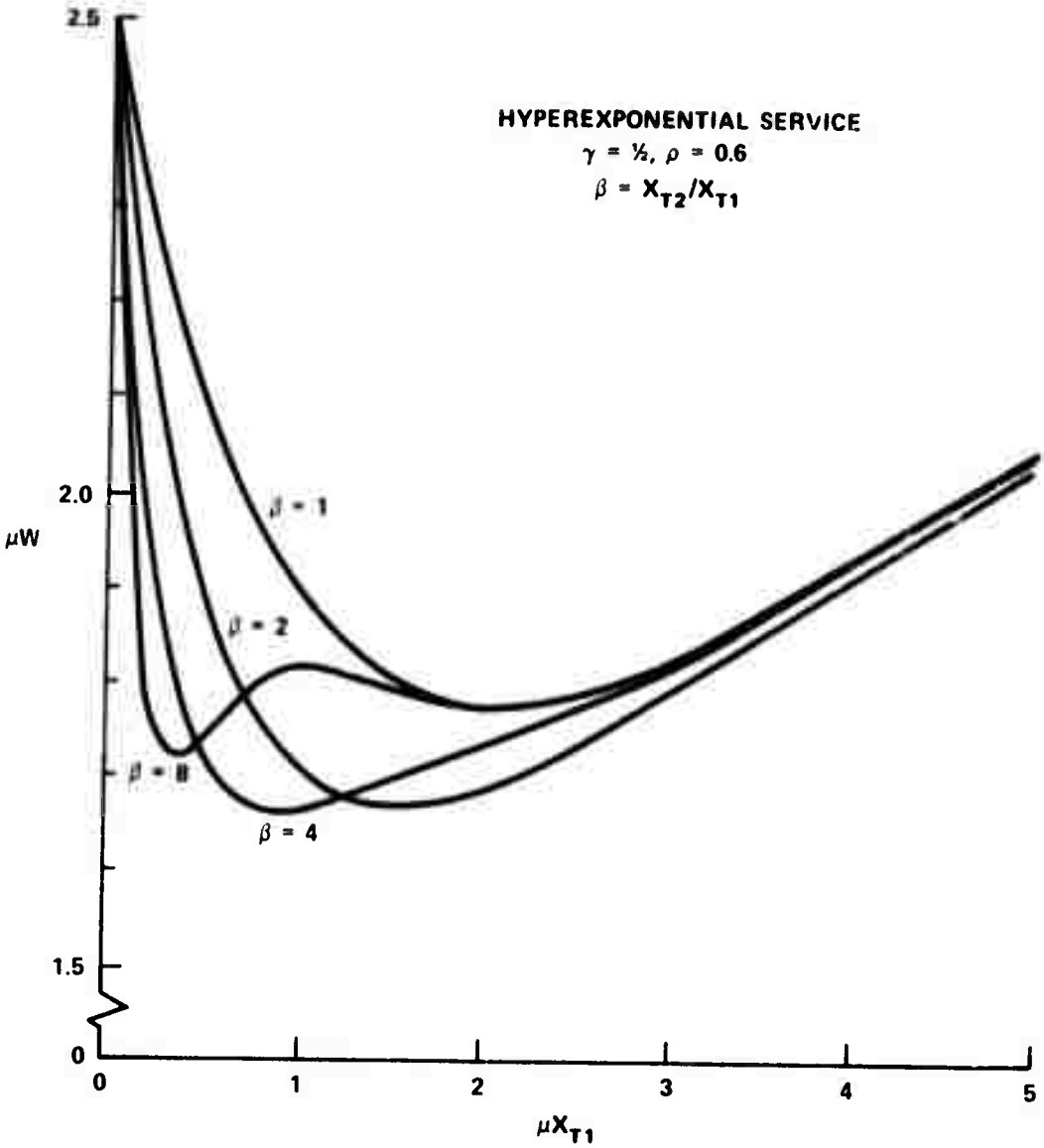


Figure 5.1.12. The Effect of a Second Priority Threshold, $X_{T2} = \beta X_{T1}$

$\beta = 2$ is similar to that observed for the exponential density in Figure 5.1.11 and a slight additional improvement is obtained for $\beta = 4$. However, for higher values of β, no further decreases were observed, and the curve begins to revert to the $\beta = 1$ case which it will approach asymptotically as β becomes infinitely large.

One should not be too dogmatic about conclusions drawn from these two examples, but they seem to indicate that additional priority classes provide little improvement in the queueing delay, but do increase the latitude of the improvement relative to the threshold location. That is, the precise threshold location is not as critical for a given service density, and conversely, a given set of thresholds may be reasonable for a wider class of service density functions.

5.1.3.2 The Analysis for P Priority Classes

The queueing delay expressions of Equation (5.97 a-c) can be readily extended to the case of P priority classes by first rewriting them in terms of W_0,

$$W_1 = \frac{W_0}{1 - \lambda\alpha_1\bar{x}_1} \tag{5.99a}$$

$$W_2 = \frac{W_0}{(1 - \lambda\alpha_1\bar{x}_1)[1 - \lambda(\alpha_1\bar{x}_1 + \alpha_2\bar{x}_2)]} \tag{5.99b}$$

$$W_3 = \frac{W_0}{[1 - \lambda(\alpha_1\bar{x}_1 + \alpha_2\bar{x}_2)]\cdot[1 - \lambda(\alpha_1\bar{x}_1 + \alpha_2\bar{x}_2 + \alpha_3\bar{x}_3)]} \tag{5.99c}$$

and noting that the results are of the form:

$$W_p = \frac{W_0}{\left[1 - \lambda \sum_{i=1}^{p-1} \alpha_i \bar{x}_i\right]\left[1 - \lambda \sum_{j=1}^{p} \alpha_j x_j\right]} \tag{5.99d}$$

which is Cobham's result (CO54). We are more concerned with the expected queueing delay averaged over all message classes,

$$W = \alpha_1 W_1 + \alpha_2 W_2 + \ldots + \alpha_P W_P \tag{5.100}$$

which after considerable algebra, can be represented in the form of the FCFS delay and an improvement factor,

$$W = \left[\frac{W_0}{1 - \rho}\right] \left\{ \frac{\sum_{p=1}^{P} \left[\alpha_p \prod_{j=1}^{p-2} \left[1 - \lambda \sum_{i=1}^{j} \alpha_i \bar{x}_i\right] \cdot \prod_{k=p+1}^{P} \left[1 - \lambda \sum_{i=1}^{k} \alpha_i \bar{x}_i\right]\right]}{\prod_{p=1}^{P-1} \left[1 - \lambda \sum_{i=1}^{p} \alpha_i \bar{x}_i\right]} \right\} \tag{5.101}$$

This is the generalization of Theorem 5.1.2 to the P priority class case, and has the same generality of priority classification as the original theorem.*

If one expands the products of Equation (5.101), the numerator and denominator cancel except for two factors leaving,

$$W = \left[\frac{W_0}{1 - \rho}\right] \sum_{p=1}^{P} \frac{\alpha_p (1 - \rho)}{\left[1 - \lambda \sum_{i=1}^{p-1} \alpha_i \bar{x}_i\right] \cdot \left[1 - \lambda \sum_{j=1}^{p} \alpha_j \bar{x}_j\right]} \tag{5.102}$$

in which the $1 - \rho$ factors cancel leaving a result given by Cox and Smith (CO61). However, the cancellation of the $1 - \rho$ factor in their

---

*Due to the generality of the priority classification of Equation (5.101) the factor within the braces will not necessarily be less than unity.

previous analysis lost the more explicit and intuitive form or representing the expected delay as the product of the expected FCFS delay and a reduction factor as shown in Equation (5.101).

5.1.3.3 The Limiting Case of a Continuum of Priority Classes

If the number of priority classes is allowed to increase indefinitely, the summation of Equation (5.102) approaches the integral result of Phipps (PH56),

$$W = \frac{\lambda \overline{x^2}}{2} \int_0^\infty \frac{dB(x)}{\left[1 - \lambda \int_0^x y \, d\, B(y)\right]^2} \tag{5.103}$$

where the expected remaining service time, W_0, is replaced by its equivalent,

$$W_0 = \lambda \overline{x^2}/2$$

and the α_n's become the elemental areas $dB(x)$, or in the case of continuous functions, $b(x)dx$.

If we consider the hyperexponential service time example,

$$b(x) = 0.5\, \mu\, e^{-\mu x} + 0.5\, \gamma\, \mu\, e^{-\gamma \mu x}$$

then we know from Equation (5.45) that:

$$\overline{x^2} = (1 + \gamma^2)/(\mu\, \gamma)^2$$

and one can show that:

$$\int_0^x y\, b(y) dy = \frac{\psi(\mu x)}{2\mu} \tag{5.104}$$

where:

$$\psi(\mu x) = [1 - (1 + \mu x)\exp(-\mu x)] + \frac{1}{\gamma}[1 - (1 + \gamma\mu x)\exp(-\gamma\mu x)]$$

The integral for the expected delay then becomes:

$$\mu\, E[\text{queue delay}] = \frac{\lambda}{2\mu} \cdot \frac{1 + \gamma^2}{\gamma^2} \int_0^\infty \frac{b(x)}{[1 - \frac{\lambda\psi}{2\mu}]^2}\, dx \qquad (5.105)$$

Due to the complexity of the integral when we include the expressions for $b(x)$ and $\psi(\mu x)$, the evaluation was performed numerically to obtain the results shown in Figures 5.1.13 and 5.1.14. The first figure

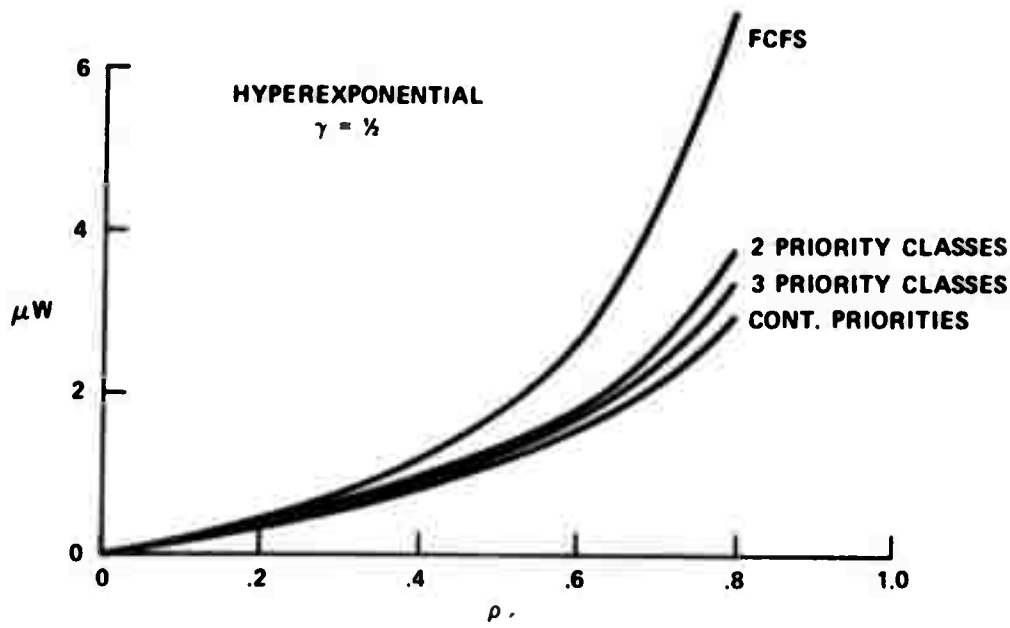


Figure 5.1.13. A Comparison of the Queueing Delays for the Cases of No Priorities, Two Priority Classes, Three Priority Classes, and a Continuum of Priority Classes

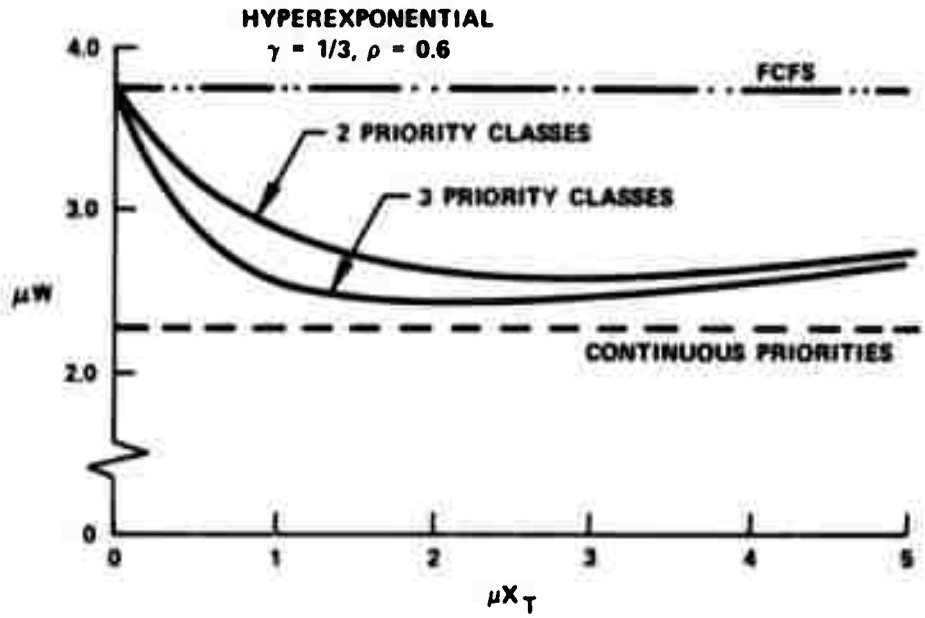


Figure 5.1.14. Typical Plots of Queueing Delay Versus Priority Threshold

is a comparison of the optimal delay values for the cases of two and three priority classes, with the FCFS delay and the continuous priority delay shown as bounds on the delays. The introduction of a single priority threshold, i.e., two priority classes, can be seen to accomplish the major portion of the available reduction in delay, but one should recall that the latitude of reasonably optimal values was also improved by further priority classifications. In cases where the density function is unknown or is variable, this increase in latitude may be particularly important. Figure 5.1.14 shows a typical plot of the delay

versus the threshold value and includes the continuous priority model as the lower bound. Here again, one can see the closeness to this limiting case for the case of merely two or three priority classes.

5.1.3.4 Considerations of the Cost to Implement an Additional Priority Class

The optimal priority system was shown to be one with a continuum of priority classes. However, if one associates a non-zero cost to implementing the priority classifications, the optimal value will be for some finite number of priority classes, and we will next consider the problem of how to find that optimal number of classes.

One of the problems in the optimization of multiple priority classes is the selection of the various priority thresholds, i.e., the borders between the classes. We shall arbitrarily resolve this by selecting equal probability areas for the priority classes. For example, if one has P different priority classes, the boundaries shall be selected such that:

$$\int_{X_{T_{i-1}}}^{X_{T_i}} b(x)dx = 1/P \qquad \text{for} \quad i = 1,2,\ldots,P \tag{5.106}$$

where:

$$X_{T_0} \equiv 0^- \quad \text{and} \quad X_{T_P} \equiv \infty$$

Cobham's result which was derived in Section 5.1.3.2 states that:

$$W_p = \frac{W_0}{\left[1 - \lambda \sum_{i=1}^{p-1} \alpha_i \bar{x}_i\right]\left[1 - \lambda \sum_{j=1}^{p} \alpha_j \bar{x}_j\right]}, \quad p = 1,2,\ldots,P \tag{5.107}$$

Where:

$$W_p = E[\text{queue delay for a priority class p message}]$$

and the priority precedence relations are such that class 1 messages have priority over class 2 messages, etc. The average queue delay is:

$$W = \alpha_1 W_1 + \alpha_2 W_2 + \ldots + \alpha_P W_P \tag{5.108}$$

but for the case of:

$$\alpha_1 = \alpha_2 = \ldots = \alpha_P = 1/P$$

then:

$$W = \frac{1}{P}\,[W_1 + W_2 + \ldots + W_P] \tag{5.109}$$

with each delay term, W_p, being as given in Equation (5.107). A plot of the delay, W, versus the number of priority classes is shown in Figure 5.1.15 for several values of ρ and an exponential service density. Once again the rapidity with which the delay reduces as the first few priority classes are introduced is apparent. The figure also shows the delay values for the optimal two priority-class case, which for ρ equal to 0.8, results in a further delay reduction of about 10% beyond that of the "equal probability area" threshold.

If we include a cost function to account for the overhead associated with the priority mechanism, we must establish two criteria. First the unit of cost must be selected, e.g., in actual dollar costs, in CPU utilization, in added message delay, or perhaps in terms of memory requirements; and secondly, the functional relationship between the cost and the number of priority classes must be approximated. An

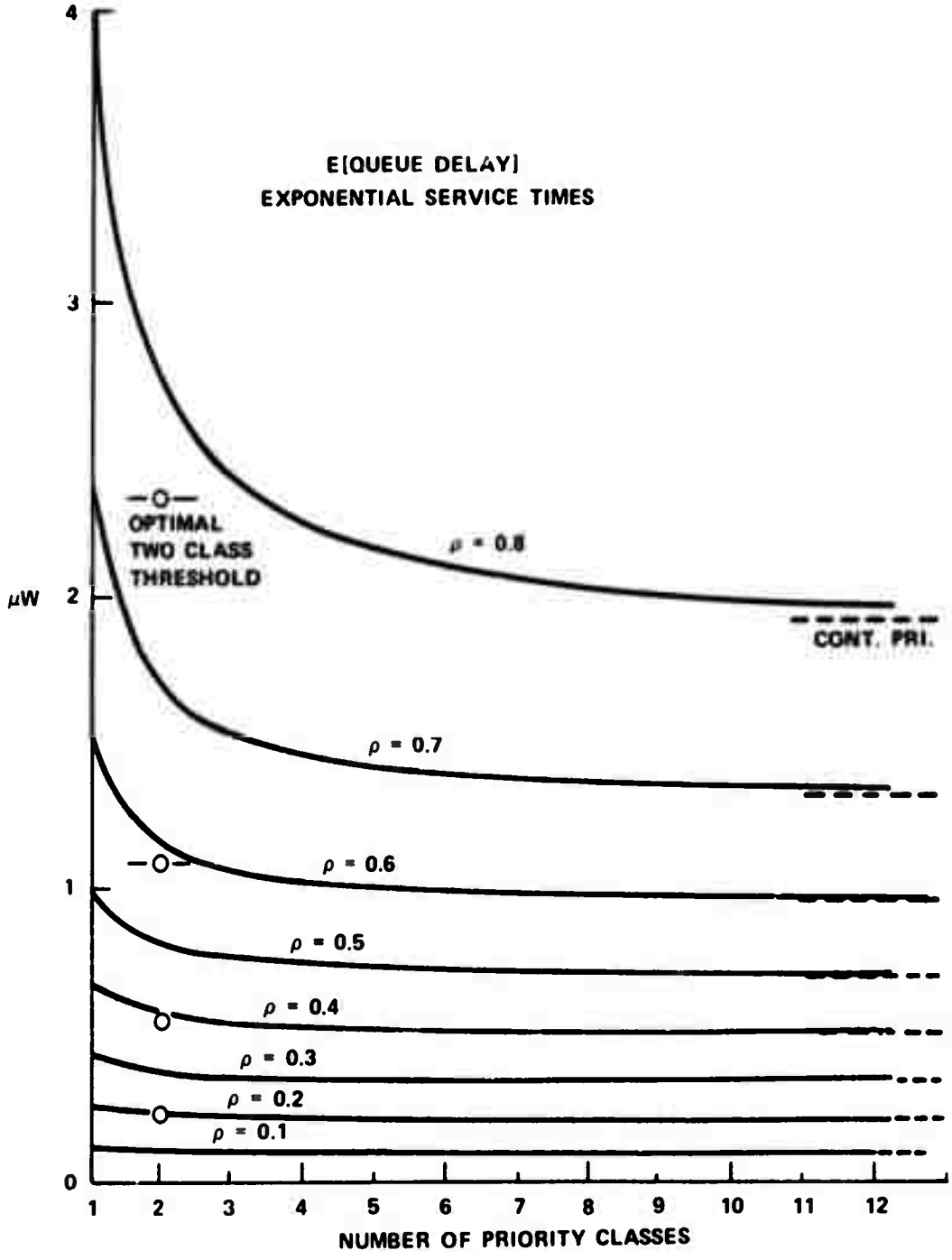


Figure 5.1.15. The Effect of the Number of Priority Classes on the Expected Queueing Delay

investigation of several possible priority implementation schemes indicated that the most critical cost effect was the core storage requirement for the priority determination and marking routines, and the additional linked lists to mechanize the priority queues. These costs grow in a manner which we will approximate by:

$$\text{cost} = K \ln(N) \tag{5.110}$$

where N is the number of priority classes and K is a free variable to be selected. The choice of the base for the logarithm is arbitrary, since it merely changes the value of K, but for convenience in handling the integer values of N, we will utilize base two logarithms. The cost function then becomes;

$$\text{cost (in memory requirements)} = K' \log_2(N) \tag{5.111}$$

where K' is equal to the memory requirement to implement the two priority class case, which we will refer to as M_2. The cost function to implement P priority classes can then be written as:

$$C_p = \text{Cost (in memory requirements)} = M_2 \log_2(P) \tag{5.112}$$

and is the cost that must be considered when one implements additional priority classes. The logarithmic cost function was chosen for a variety of reasons including; (1) it has a zero value for a single priority class, i.e., FCFS, (2) it has a decreasing slope as the number of priority classes is increased (which accounts for the use of more efficient priority queue handling mechanisms), and (3) because the time required for the priority classification via a binary search is known to be proportional to the logarithm of the number of categories.

If we assume, as in Section 5.1.2.4, that messages have some predefined maximal length, L, which defines the block size for storage allocation, then on the average, $\bar{n}$ such blocks will be linked together on an output queue, where by Little's result:

$$E[\text{number in queue}] = \bar{n} = \lambda W$$

If each such storage block is of length L, then the expected storage requirement for the enqueued messages is:

$$E[\text{storage for buffering}] = \lambda WL \tag{5.113}$$

The requirement that message lengths have some such finite upper bound is a practical necessity, but does not introduce any theoretical problem in the model since the message length distributions can include a discrete (delta function) component at the length, L. Under these conditions, the expected storage requirement for buffering can be reduced by the same priority techniques that reduce the average queueing delay, W, and when both the priority implementation and buffering considerations of Equations (5.112) and (5.113) are combined, the total storage requirement for a store-and-forward node with Q output queues becomes:

$$E[\text{storage}] = Q\lambda LW(P) + M_2\log_2(P) \tag{5.114}$$

If it were not for the $M_2\log_2(P)$ component, this expected value would be a continually decreasing function of the number of priority classes. However, the growing implementation cost component will result in a finite optimal number of priority classes, and it is this number of priority classes which we wish to determine.

The example of Figure 5.1.15 was extended to include this cost function, using the approximate implementation costs of the ARPA

network IMP, i.e., a requirement of about 25 IMP-words to implement the two-priority class mechanism, and $L = 63$ IMP-words, for a value of M_2 of:

$$M_2 = \frac{25}{63} L = 0.4L$$

The expression for the average storage requirement is then:

$$E[\text{storage}] = Q\lambda\bar{x} \frac{W(P)}{\bar{x}} L + 0.4L \log_2(P)$$

which can be written as:

$$E[\text{storage}] = Q\rho L \, \mu W(P) + 0.4L \log_2(P) \tag{5.115}$$

This function is plotted in Figure 5.1.16 for the exponential service example of Figure 5.1.15 and for $Q = 3$. Several values of ρ are shown, with definite optimal, i.e., minimal cost, values being indicated. As expected, the higher values of ρ show the greatest improvement due to the priority implementation, and show the benefit of introducing additional priroity classes, up to a maximum of about three or four classes. In contrast, the lower values of ρ indicate that the FCFS system has the minimal cost since the average number of messages in the queue is very small. However, the effects are of primary concern at the higher values of ρ, since effective buffer utilization is more critical under heavy loading.

5.1.3.5 Design of an Experiment to Verify the Multiple Priority Model

An experiment could be run to verify the multiple priority model by an approach quite similar to that utilized in Section 5.1.2.5 for the two priority model. Artificial traffic would be set up as

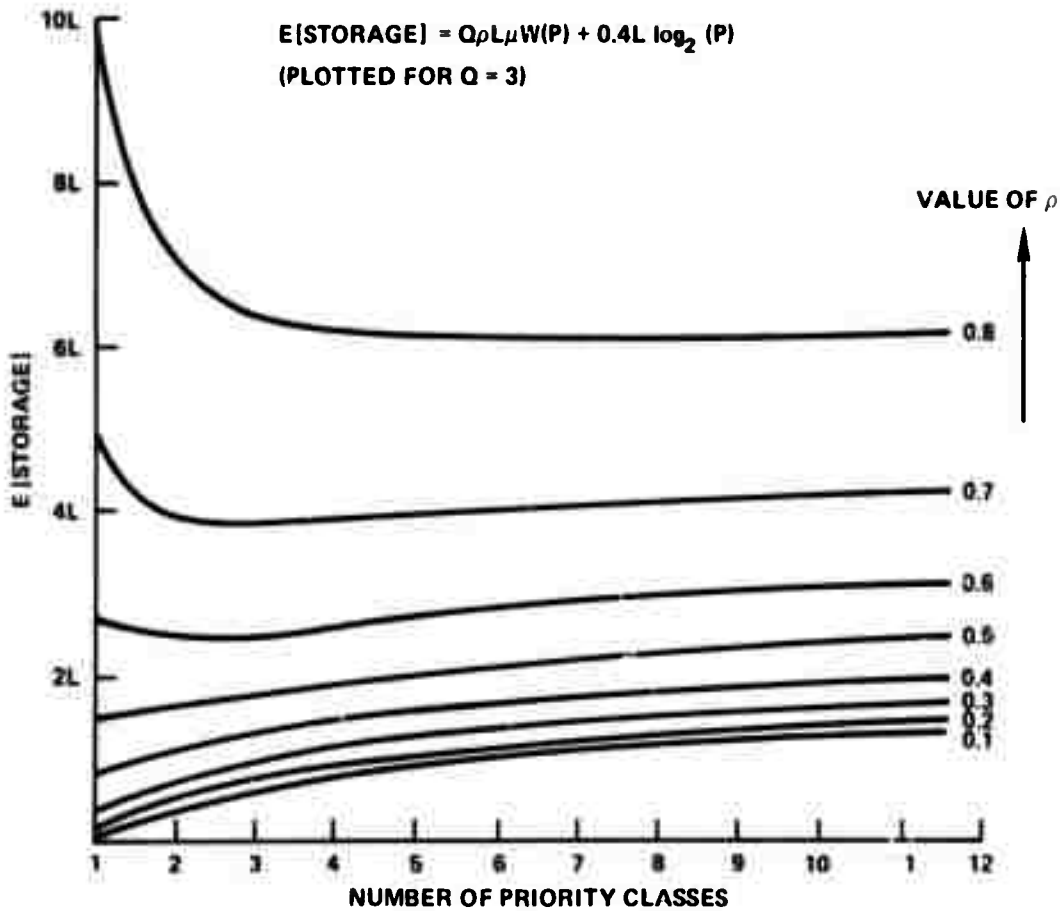


Figure 5.1.16 The Expected Storage Requirements Including the Priority Implementation Costs

before using fixed length messages and varying the number of active generators to establish the desired levels of the traffic intensity, ρ.

For purposes of the design of a suggested experiment, it will be assumed that a multi-priority system will be investigated using the equal probability area criterion for determining the priority classes. An exponential distribution of message lengths would be utilized with an average length of 40 bytes (to minimize the truncation effect of packet segmentation). Therefore, the priority threshold for the two priority class system would be set at a length, L_p, where:

$$e^{-L_p/\bar{\ell}} = 0.5$$

with $\bar{\ell}$ equal to 20, i.e., the average message length in terms of 16-bit IMP words. (The priority threshold should be expressed in terms of IMP words since that is the way in which the variations would be made during the experiment.) For the two priority class system the priority threshold is therefore,

$$L_p = 0.69\bar{\ell} = 14 \text{ words}$$

This value will be sub-optimal because the optimal threshold is known to be greater than the mean value. However, it should serve as an experimental test point to compare with the theoretical results of Figure 5.1.15. The thresholds for the three priority case can be determined in a similar manner by:

$$e^{-L_{p_1}/\bar{\ell}} = 0.67$$

and

$$e^{-L_{p_2}/\bar{\ell}} = 0.33$$

with resulting threshold values of:

$$L_{p_1} = 0.40\bar{\ell} = 8 \text{ words}$$

and

$$L_{p_2} = 1.11\bar{\ell} = 22 \text{ words.}$$

The thresholds for higher numbers of priority classes would be determined in a similar manner, with experimental data taken for each such set of values and compared with the theoretical results of Figure 5.1.15. Values of ρ equal to 0.4, 0.6, and 0.8 would be adequate to verify the model, and should give large enough queueing delays to give good results from the round-trip delay portion of the accumulated statistics. The UCLA-to-RAND path would again be utilized for the test to avoid alternate routing effects.

The largest error for such a test should occur for the highest value of ρ, which for ρ equal to 0.8 ± 0.05, would result in errors of about ±20% to 30% about the nominal delay value. The accuracy of creating a traffic level of ρ equal to 0.8 should be somewhat better than this ±0.05 value, so that the error would not be expected to be quite that large, but could easily be in the 10% to 15% region.

This experiment was not performed due to the difficulty of implementing the various priority mechanisms by IMP program changes, and the fact that such changes would have to be performed by the network contractor, BBN.

5.2 Models for the Segmented Message Case

The preceding analysis assumed that messages would be serviced in a contiguous manner, and that message buffers were of sufficient length to hold a maximal size message. A number of practical desires and constraints can not necessarily be met by such a scheme, and therefore most message switching systems are designed to handle fixed length message segments. Some of the consequences of such segmentation are considered in the following sections.

5.2.1 Rationale and Effects of Segmentation

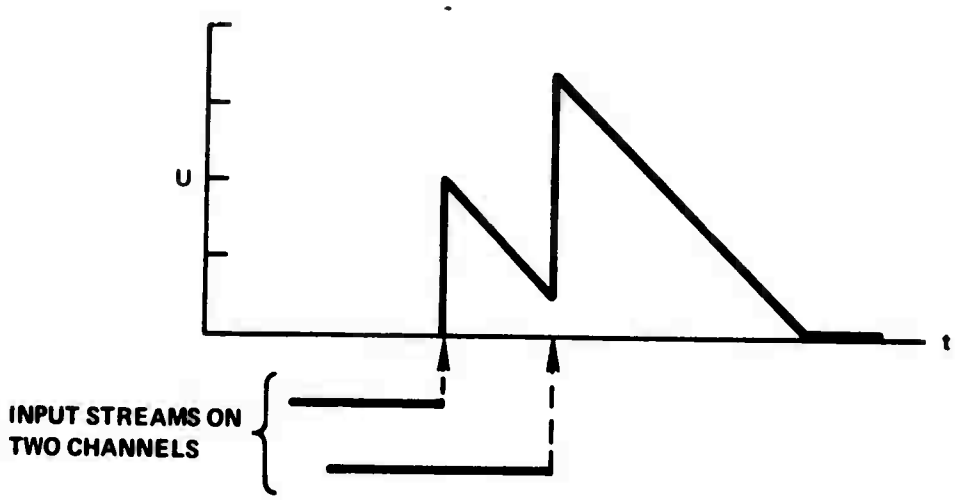
The wide variation in possible message lengths leads to the segmentation of messages into shorter fixed length blocks or packets, to smooth the service demands, and to give better response to short interactive messages. The buffer allocation problem may also be affected by the segmentation because of the need, in most equipment, to allocate space in fixed size blocks, but such segmentation does not necessarily affect the message flow since small buffers can be linked together to store a given length message. However, if the messages must be transmitted as a contiguous bit stream, interactive messages may be delayed excessively, so that some type of preemptive or preemptive-resume discipline might be considered. The segmentation of messages into packets is similar to the preemptive-resume situation, but preemption is allowed only at the packet boundaries. Some overhead is required in either case, with the individual packet overhead being the communication control characters and the header information. The advantages of segmenting messages into packets generally favor small

packet sizes, but the packet overhead cost must be balanced against these gains to determine the optimal packet size. We will consider some of these packet size selection criteria in Section 5.2.4, but for now will further explore the advantages of reducing the packet size.

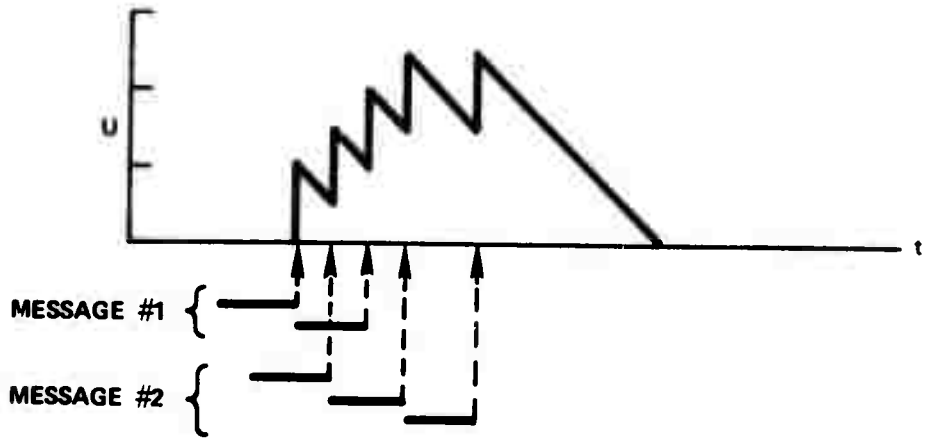
The analysis of Section 5.1.1 showed that the expected queueing delay is proportional to the average time to complete an existing service, W_0, which in turn is proportional to the second moment of the service time. Truncation of the service time density reduces this second moment, as shown in Appendix A, and thereby leads to shorter queueing delays for priority messages.

A second effect of interest involves the changes in the arrival times when messages are segmented. By definition, an arrival occurs when the last bit of a contiguous bit stream reaches the IMP, and the effect of the different arrival epochs in reducing the amount of unfinished work in the system is shown in Figure 5.2.1. The segmented message case also decreases the total delay by the fact that arrivals occur earlier, and that the input and output processes tend to be overlapped. The reduction in the amount of unfinished work implies that the storage requirement should also be less for the segmented case, a consideration that will be pursued in Section 5.2.3.1.

A third consequence of segmentation is its effect on the error control techniques used to detect and correct transmission errors. Longer segments are more likely to contain random errors and also make retransmission more costly, but allow the use of more efficient error control codes. An optimal balance of these factors depends on the error mechanism and rates involved, and has been considered in previous



(a) UNFINISHED WORK DIAGRAM FOR CONTIGUOUS MESSAGES



(b) UNFINISHED WORK DIAGRAM FOR SEGMENTED MESSAGES

Figure 5.2.1. Typical Unfinished Work Diagrams for Contiguous and Segmented Message Flows.

studies including that of Kirlin (KI69). The line quality and error control in the ARPA network have been quite effective and do not appear to be a determining factor for packet size, and therefore will not be considered further. (A 24-bit cyclic error checking code is appended to each packet, and is checked and regenerated at each store-and-forward IMP).

The segmentation of messages into separate packets which can flow through the network independently introduces the possibility of gaps between packet transmissions. These gaps are typically due to other traffic in competition for the service facility, and cause additional delay components at the destination because the message must be fully reassembled prior to transmission to the HOST. Aside from such gaps between packets, the individual packets may have arrived out of sequence, duplicate packets may have been recieved, or in rare cases one or more packets may have been lost, so that the reassembly process requires numbering the packets and indicating the final packet of a message, as well as time-outs to discard message fragments in the latter case. The selection of this time-out value and the mechanism for allocating reassembly buffers are important system parameters, and need careful attention. Several possible allocation procedures are possible including; (1) reserving the proper number in advance by a "request to send", (2) setting aside a full set of message reassembly buffers upon the receipt of the first packet of a message, and (3) allocating buffers on an "as arrived" basis. A concern in the latter case is that in periods of congestion, few complete messages might be reassembled due to the exhaustion of the buffer supply by numerous partially reassembled

messages.

The area of message segmentation is rich in possible advantages and pitfalls, which makes it a viable area of analysis and measurement, with the goal of improving the network performance by finding more optimal techniques and parameter values. Several such aspects are considered in the following sections, with both analytic and experimental methods being utilized.

5.2.2 The Separation Between Packets

If there were no other traffic in a store-and-forward network, the packets of a message would flow through the net in a continuous stream. However, when other traffic is introduced at the various nodes, the various message packets tend to become interspersed, and the packets of any given message may become separated. This intermixing does not necessarily change the average delay in flowing through the network, since the messages are essentially time-sharing the channel capacity, and analogous time-shared computer analysis results have indicated certain conservation relationships (KL64). However, the network problem differs in several ways including the lack of well ordered service scheduling, and also in the fact that the net involves a sequence of such facilities. There is an intuitive feeling that as packets become separated, interference may become more likely and therefore that the expected gap size may tend to grow as the message flows through a large network. We will show in Theorems 5.2.1 and 5.2.2 that the expected gap size is bounded for both the FCFS and priority disciplines respectively. Interestingly, this bound is the same in both cases, and is equal to:

$$\lim_{n\to\infty} E[\text{interpacket gap after traveling thourgh } n \text{ nodes}]$$

$$= \frac{\rho_i}{1 - \rho_i} \tag{5.116}$$

where ρ_i is the traffic intensity of the interfering traffic.

Theorem 5.2.1

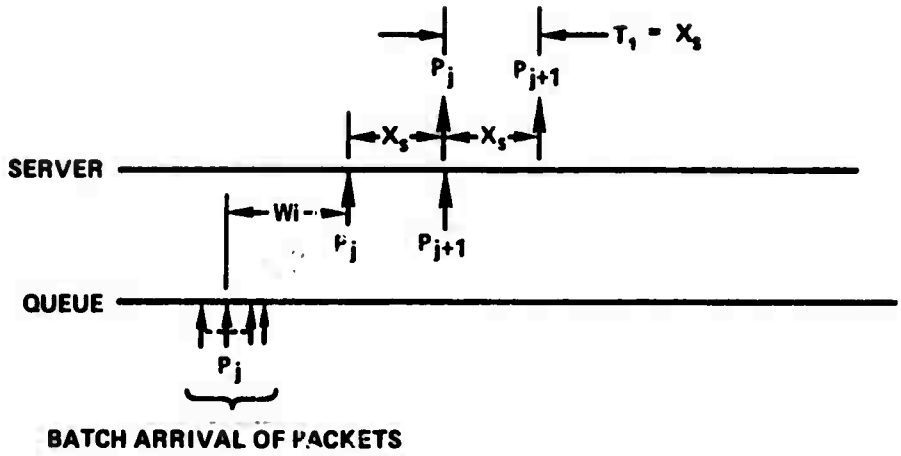
The expected interpacket gap for a segmented message, and a FCFS queue discipline, is:

$$E[\tau_n] = [1 - (\rho_i)^{n-1}] \frac{\rho_i}{1 - \rho_i} \cdot X_s \tag{5.117}$$

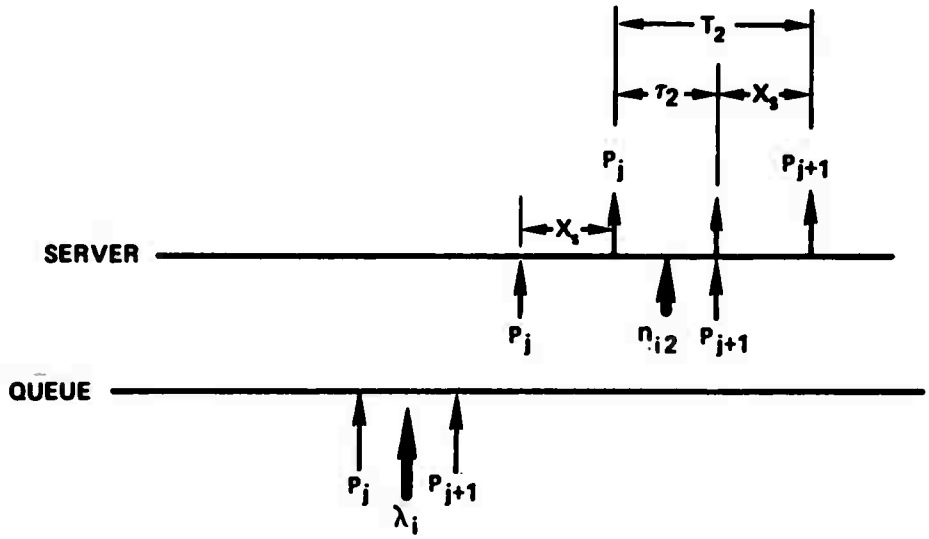
where τ_n is the extra interpacket spacing at the output of the nth node due to interfering traffic along a store-and-forward path, X_s, is the service time of a maximal length packet, and ρ_i is the traffic intensity in terms of the average arrival rate of interfering packets and their service requirements. All ρ_i's are assumed to be equal along the entire network path. Interfering packet arrivals at each node are also assumed to be Poisson, with parameter λ_i.

Proof:

If we consider a "tagged" multipacket message at the originating node, consisting of packets $P_1, P_2, \ldots, P_j, \ldots$, we find a situation as shown in Figure 5.2.2(a). The packets are presented to the queue as a group, and for the FCFS system, are transmitted continguously. The arrivals at the second node are therefore spaced at regular intervals of $T_1 = X_s$



(a) QUEUE AND SERVER AT THE SOURCE NODE



(b) THE INTERFERING ARRIVALS AT THE SECOND NODE

Figure 5.2.2. The Effect of Interfering Traffic to Create a Separation Between Arrivals of Packets of a Message.

seconds, where X_s is the time required to service one maximal length packet. However, at this second node there may be interfering traffic arriving on other input channels with an arrival rate of λ_i and as shown in Figure 5.2.2(b), these packets will become interspersed with the packets of the tagged message. The expected value of the time between the tagged packet departures will be:

$$\bar{T}_2 = X_s + \bar{\tau}_2 \tag{5.118}$$

where the gap between the now non-contiguous packets will have an expected value of:

$$\bar{\tau}_2 = \bar{n}_{i2}\, \bar{x}_i \tag{5.119}$$

with the subscripts i referring to the interfering traffic and the subscript 2 indicating that this delay is at the output of the second node along the path. The expected value $\bar{n}_{i2}$ is the average number of interfering arrivals during the interarrival time between the tagged packets, which for Poisson interference arrivals is:

$$\bar{n}_{i2} = \lambda_i\, X_s \tag{5.120}$$

and the value $\bar{x}_i$ is the average service requirement of the interfering traffic, such that:

$$\rho_i = \lambda_i\, \bar{x}_i \tag{5.121}$$

The expected gap between the packets at the output of the second node can then be written as:

$$E[\tau_2] = \rho_i\, X_s \tag{5.122}$$

and this gap becomes the interarrival gap at the third node. There is a non-zero probability that the interfering packets which separate the tagged packets will be routed to the same output channel at the third node, but we shall ignore this effect and assume that only new arrivals will be contending for the channel. Interfering traffic can occur anytime during the interarrival time which has now grown to $X_s + \tau_2$ such that the expected number of interfering arrivals becomes:

$$\bar{n}_{i3} = \lambda_i (X_s + \bar{\tau}_2) \tag{5.123}$$

and the resulting gap between the tagged packets at the output of the third node becomes:

$$E[\tau_3] = \lambda_i (X_s + E[\tau_2]) x_i$$

or substituting for $E[\tau_2]$, and simplifying:

$$E[\tau_3] = \rho_i (1 + \rho_i) X_s \tag{5.124}$$

A similar line of reasoning at the fourth node produces,

$$E[\tau_4] = \rho_i (X_s + E[\tau_3])$$

or:

$$E[\tau_4] = \rho_i (1 + \rho_i + \rho_i^2) X_s$$

and leads to the general iterative expression:

$$E[\tau_n] = \rho_i (X_s + E[\tau_{n-1}]) \tag{5.125a}$$

or:

$$E[\tau_n] = \rho_i \, X_s \sum_{k=0}^{n-2} (\rho_i)^k \tag{5.125b}$$

Equation (5.125) can be expressed in closed form as:

$$E[\tau_n] = \rho_i X_s \frac{1 - (\rho_i)^{n-1}}{1 - \rho_i}$$

and by grouping terms:

$$E[\tau_n] = [1 - (\rho_i)^{n-1}]\left(\frac{\rho_i}{1 - \rho_i}\right)X_s$$

with τ_n being the gap between a pair of tagged packets at the output of the nth store-and-forward node. This completes the proof of Theorem 5.2.1.

The expected interpacket gap is shown in Figure 5.2.3 for several values of ρ_i, and indicates several interesting aspects about the gap size. For relatively small values of ρ_i, the expected gap length reaches an asymptotic value after going through a few nodes, with larger values of ρ_i requiring long paths to reach a similar asymptote. The fact that in each case an asymptote exists is of interest, since it shows that the gap does not grow in an unbounded fashion, but of particular interest is the mathematical form of this asymptotic value,*

$$\lim_{n\to\infty} E[\tau_n] = \frac{\rho_i}{1 - \rho_i} X_s \tag{5.126}$$

It is interesting to note that for the expected interpacket gap to become within a factor, F, of the asymptotic value, $\bar{\tau}_\infty$, we need:

$$n \geq 1 + \frac{\log(1 - F)}{\log \rho_i}$$

*This form is believed to be analogous to the conservation results of Kleinrock (KL64) for various time-sharing queue disciplines.

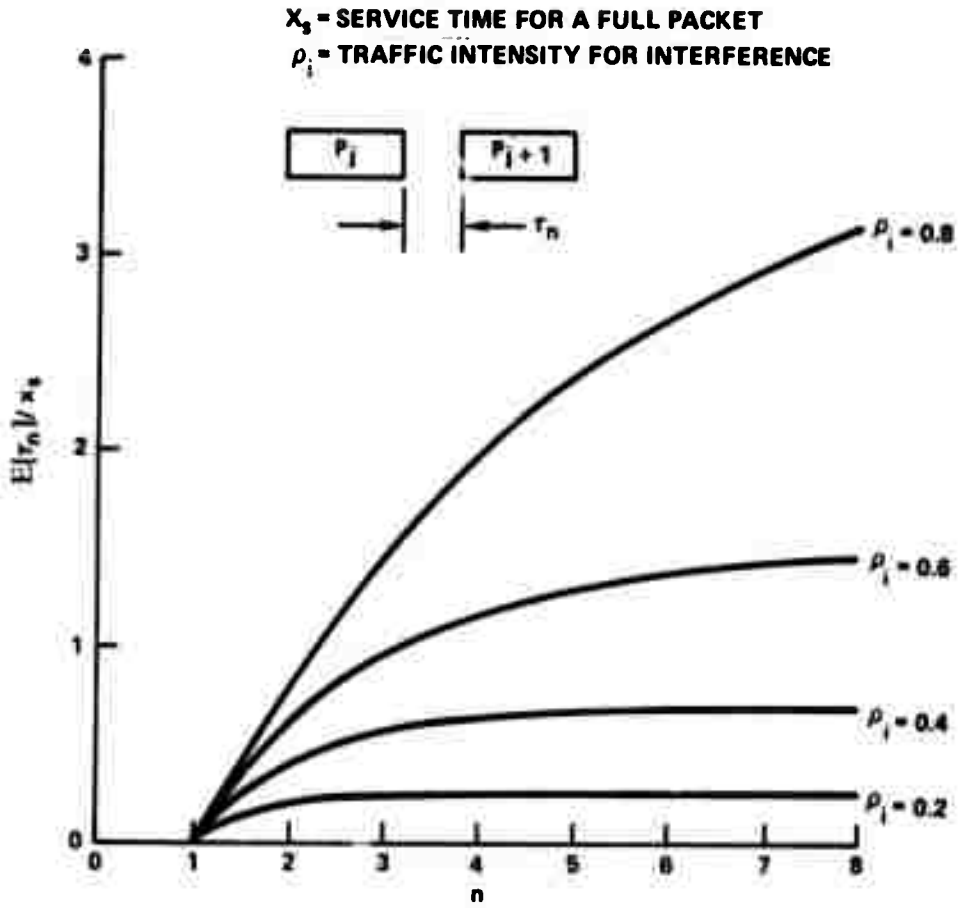


Figure 5.2.3. Expected Inter-Packet Separation After Going Through n Nodes.

This can be shown by noting that the expression for $E[\tau_n]$ can be written as:

$$E[\tau_n] = [1 - (\rho_i)^{n-1}]\ \bar{\tau}_\infty = F\ \bar{\tau}_\infty$$

such that:

$$(\rho_i)^{n-1} = 1 - F$$

and by taking logarithms of both sides,

$$n = 1 + \frac{\log(1 - F)}{\log \rho_i}$$

Therefore, any value of n equal to or greater than this value will cause the expected gap to be within the factor, F, of the asymptotic delay value.

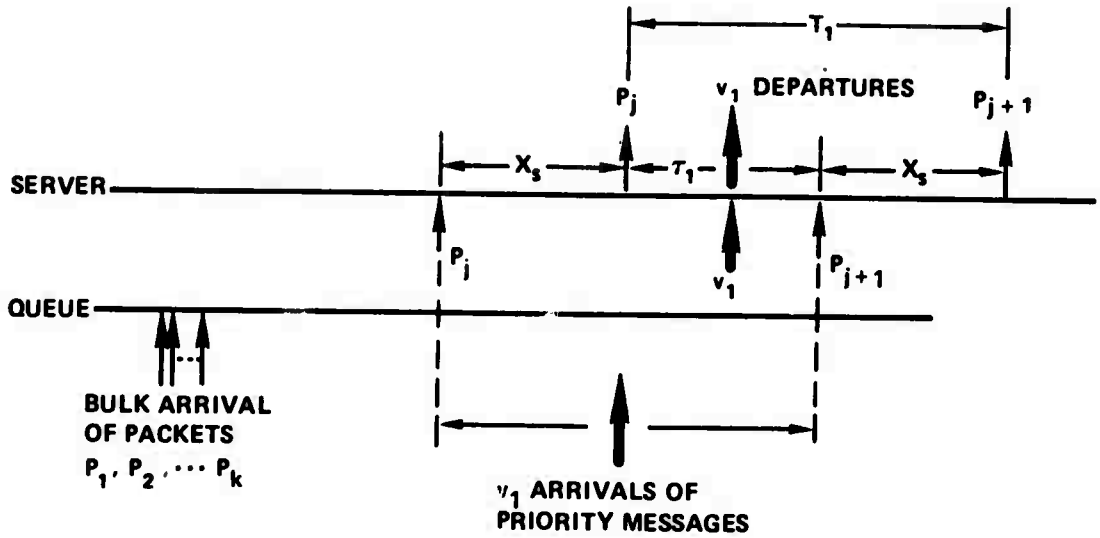
An extension of interpacket gap model was obtained by a colleague* who considered the case of non-uniform traffic intensities along the store-and-forward path. This change meant that the expected interpacket delay would not necessarily increase at each subsequent node, resulting in an extension of the iterative relationship of Equation (5.125a) to include this case,

$$E[\tau_n] = \max\{\rho_{in}(X_s + E[\tau_{n-1}] \ , \ E[\tau_{n-1}]\} \tag{5.127}$$

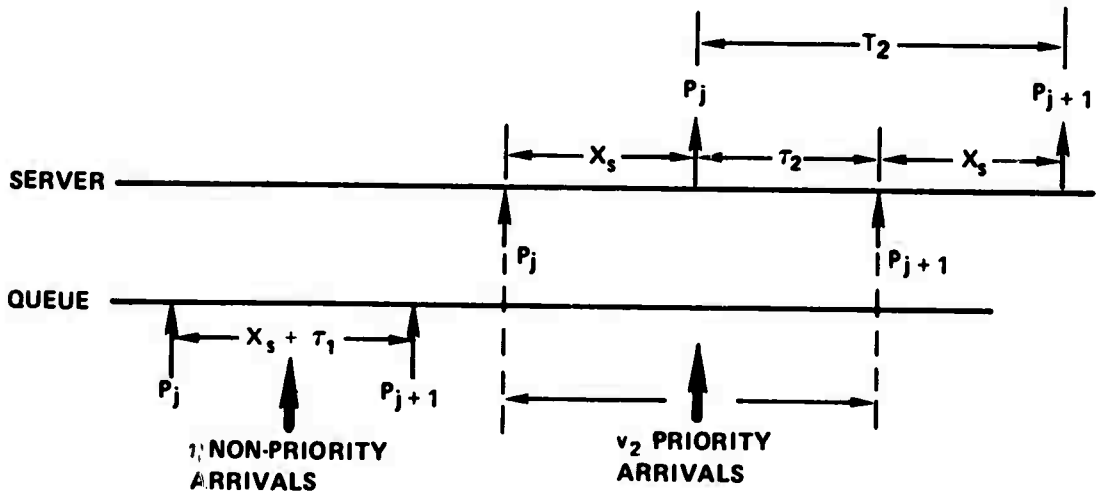
where ρ_{in} is the traffic intensity of the interfering packets at the nth node.

The interpacket gap analysis is a priority system differs in two aspects from that of the FCFS system. In the FCFS system, any arrivals occurring between the arrivals of packets P_j and P_{j+1} were also serviced between P_j and P_{j+1}. In the priority system, the interfering arrivals will be any non-priority packets between the arrivals of P_j and P_{j+1}, and any priority arrivals after P_j begins service but before P_{j+1} begins. The resulting interpacket gap given

*This extension was made by Gary Fultz, who was concerned with differing traffic intensities across the net in routing simulation studies. This interpacket gap time was of concern to him because of its effect on the reassembly time of messages at the eventual destination.



(a) THE QUEUE AND SERVER ACTIVITY AT THE SOURCE NODE



(b) THE QUEUE AND SERVER AT THE SECOND NODE

Figure 5.2.4. Arrivals and Departures for the Queue and Server in the Priority Queueing Case.

in Theorem 5.2.2 is shown to increase more rapidly than that of the non-priority case, but to have the same limiting value.

Theorem 5.2.2

The expected interpacket gap for a segmented message with higher priority traffic competing for the server is:

$$E[\tau_n] = \frac{X_s}{1 - \rho} \left[\rho - \left(\frac{\rho_{np}}{1 - \rho_p} \right)^n \right] \tag{5.128}$$

where the conditions and variables are as defined in Theorem 5.2.1 with ρ_p and ρ_{np} being the traffic intensities of priority and non-priority traffic respectively.

Proof:

Assume that a multipacket message is initially entered into the network as shown in Figure 5.2.4(a). The jth and $j+1^{st}$ packets of the message are not necessarily served contiquously (as in the FCFS case) since priority arrivals during the service of P_j will be serviced before P_{j+1} as will other priority arrivals during the service of the priority messages. If the arrival rate of interfering priority messages is λ_{ip} then the expected time between the departures of P_j and P_{j+1} is:

$$E[T_1] = X_s + E[\tau_1] \tag{5.129}$$

where τ_1 is the time to service the ν_1 priority arrivals, with:

$$E[\nu_1] = \lambda_{ip}(X_s + E[\tau_1]) \tag{5.130}$$

and since the arrivals and service requirements are independent,

$$E[\tau_1] = E[\nu_1] \cdot E[x \mid \text{mes. is priority}]$$

or:

$$E[\tau_1] = \lambda_{ip}(X_s + E[\tau_1])\overline{x_1} \tag{5.131}$$

Solving for $E[\tau_1]$ we obtain:

$$E[\tau_1] = \frac{\lambda ip\ \overline{x}_1\ X_s}{1 - \lambda_{ip}\ \overline{x}_1} = \frac{\rho_p\ X_s}{1 - \rho_p} \tag{5.132}$$

where ρ_p is the traffic intensity for interfering priority messages,

$$\rho_p = \lambda_{ip}\ \overline{x}_1$$

The expected interdeparture time of packets P_j and P_{j+1} at the first node is then:

$$E[T_1] = X_s + \frac{\rho_p\ X_s}{1 - \rho_p} = \frac{X_s}{1 - \rho_p} \tag{5.133}$$

so that at the second node we can expect to find η interfering arrivals of regular messages, where:

$$E[\eta] = \lambda_{inp}\ E[T_1] = \lambda_{inp}(X_s + E[\tau_1]) \tag{5.134}$$

Only the non-priority messages are considered to be interfering in the sense of adding to the interpacket gap. Any priority arrivals are either served prior to P_j or are considered in the ν_2 interfering priority arrivals, where ν_2 has an expected value of:

$$E[\nu_2] = \lambda_{ip}(X_s + E[\tau_2]) \tag{5.135}$$

If we again assume an independence of service and arrivals, the expected interdeparture time becomes:

$$E[T_2] = X_s + E[\eta_1] \cdot \bar{x}_2 + E[\nu_2] \cdot \bar{x}_1 \tag{5.136}$$

or substituting for the expected values,

$$E[T_2] = X_s + \lambda_{inp}\, \bar{x}_2 (X_s + E[\tau_1]) + \lambda_{ip}\, \bar{x}_1 (X_s + E[\tau_2]$$

However, since:

$$E[T_2] = X_s + E[\tau_2]$$

and substituting:

$$\rho_p = \lambda_{ip}\, \bar{x}_1 \quad \text{and} \quad \rho_{np} = \lambda_{inp}\, \bar{x}_2$$

we can solve for $E[\tau_2]$ as:

$$E[\tau_2] = \rho_{np}\, X_s + \rho_{np}\, E[\tau_1] + \rho_p\, X_s + \rho_p\, E[\tau_2]$$

or:

$$E[\tau_2] = \frac{\rho\, X_s - \rho_{np}\, E[\tau_1]}{1 - \rho_p} \tag{5.137}$$

where:

$$\rho_p + \rho_{np} = \alpha\lambda\bar{x}_1 + (1 - \alpha)\lambda\, \bar{x}_2 \tag{5.138}$$

and:

$$\alpha\bar{x}_1 + (1 - \alpha)\bar{x}_2 = \bar{x}$$

so that

$$\rho = \rho_p + \rho_{np}$$

By repeating the above arguments at subsequent nodes, the result of Equation (5.137) can be seen to be a general iterative relationship such that,

$$E[\tau_n] = \frac{\rho X_s + \rho_{np} E[\tau_{n-1}]}{1 - \rho_p} \tag{5.139}$$

with:

$$E[\tau_1] = \frac{\rho_p X_s}{1 - \rho_p} \tag{5.140}$$

This relationship can be expressed in a closed form* as:

$$E[\tau_n] = \frac{X_s}{1 - \rho} \left[\rho - \left(\frac{\rho_{np}}{1 - \rho_p} \right)^n \right]$$

which for ρ_p equal to zero, becomes the FCFS result of Equation (5.117).

If we compare the two results, we see that unlike the FCFS system, the priority system has a non-zero expected gap at the first node, but that both have the same limiting value as n becomes very large. The latter effect is not obvious, but can be shown by noting that:

$$\rho = \rho_p + \rho_{np}$$

such that:

$$\rho_{np} = \rho - \rho_p$$

---

*This form is a variation of the closed form representation developed by Fultz from the iterative result of Equation (5.139).

which is less than the denominator, $1 - \rho_p$, and therefore in the limit,

$$\lim_{n\to\infty} \left(\frac{\rho_{np}}{1 - \rho_p} \right)^n = 0$$

leaving:

$$\lim_{n\to\infty} E[\tau_n \mid \text{priority system}] = \frac{\rho}{1 - \rho} X_s$$

which is identical to the limit for the FCFS case

These effects are shown in Figure 5.2.5 for various values of ρ, and with ρ_p and ρ_{np} being arbitrarily made equal. This assumes that $\alpha\bar{x}_p$ is equal to $(1 - \alpha)\bar{x}_{np}$ in Equation (5.138), which from the priority threshold analysis, seems reasonable.

The preceding analysis was based on the assumption that all of the packets of a message arrive at the source IMP simultaneously, i.e., as a bulk arrival. If we assign a finite rate to these HOST-to-IMP transfers, such that a full packet requires X_1 seconds to cross this boundary, then the analysis must be modified to include this effect. The interpacket gap at the output of the originating node then becomes:

$$E[\tau_1] = \lambda_{inp}\, \bar{x}_2\, X_1 + \lambda_{ip}\, \bar{x}_1 (X_s + E[\tau_1]) \tag{5.141}$$

which is a modification to Equation (5.131) to include the non-priority interference. Solving for the average gap time, we obtain:

$$E[\tau_1] = \frac{\rho_{np}\, X_1 + \rho_p\, X_s}{1 - \rho_p} \tag{5.142}$$

The delays at subsequent nodes can be determined by the previous

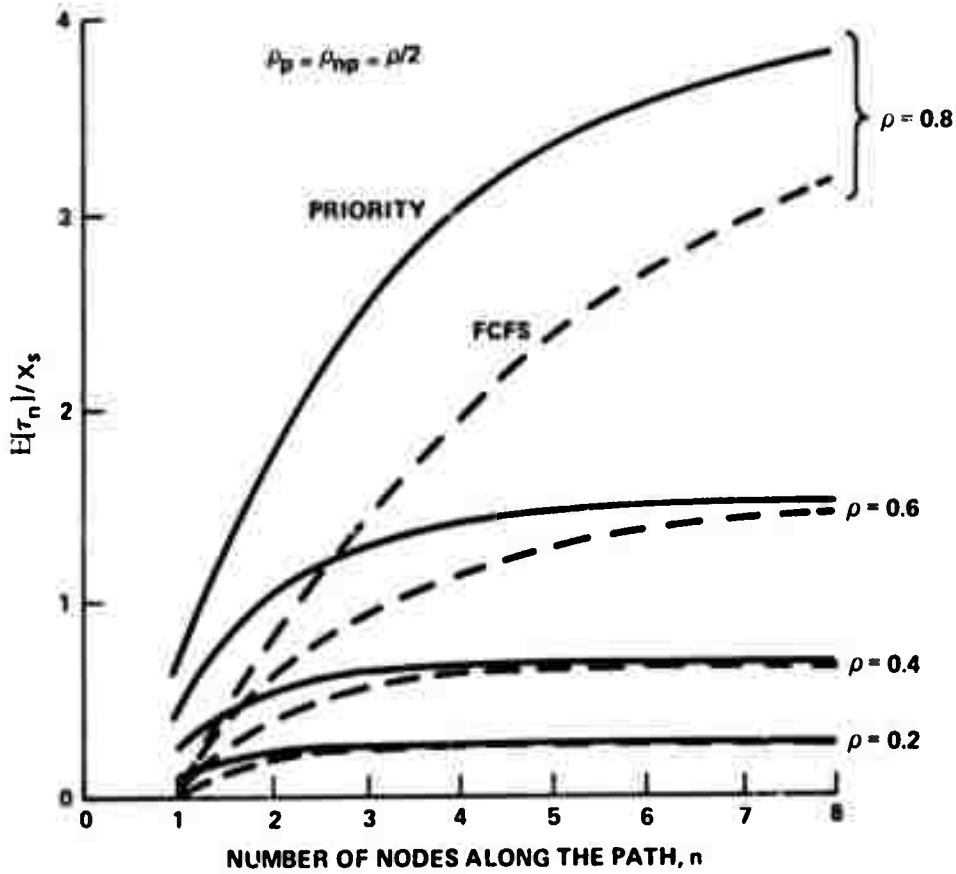


Figure 5.2.5. The Expected Interpacket Gap for the Priority System After Going Through n Nodes.

iterative relationship,

$$E[\tau_n] = \frac{\rho X_s + \rho_{np} E[\tau_{n-1}]}{1 - \rho_p}$$

which leads to the closed form result:

$$E[\tau_n] = \frac{X_s}{1-\rho}\left[\rho - \left(\frac{\rho_{np}}{1-\rho_p}\right)^n\right] + \left(\frac{\rho_{np}}{1-\rho_p}\right)^n X_1 \tag{5.143}$$

where X_s and X_1 are the full packet service times for the IMP-IMP and HOST-IMP transmissions respectively, then τ_n is the interpacket gap at the output of the nth store-and-forward node. (This result also shows that the non-zero HOST-IMP transfer time deos not effect the limiting value as n becomes arbitrarily large.)

5.2.3 Experimental Verification of the Packet Separation Model

A complete experimental verification of the packet separation model would require the ability to generate artificial traffic at each node along the store-and-forward path of a test message. Since such facilities are available at only one site, the UCLA Network Measurement Center, the verification test was much less extensive than would normally be desired.

The test was designed to test the validity of Equation (5.143) for the special case of $n = 1$,

$$E[\tau_1] = \frac{X_s}{1-\rho}\left[\rho - \frac{\rho_{np}}{1-\rho_p}\right] + \frac{\rho_{np}}{1-\rho_p} X_1 \tag{5.144}$$

The interference traffic was generated by the pseudo-random artificial traffic generator (as described in Section 2.5.8) with the traffic intensity, ρ, being determined by selection of the average message length. The number of active message generators was a compromise between, (1) the desire to avoid alternate routing, and (2) the desire to minimize the RFNM dependency of the arrivals. Eight generators were found to meet both desires satisfactorily.

The generation of the multi-packet test traffic was originally planned utilizing three different techniques to obtain a range of values of the parameter, X_1, (the time required for the serial "arrival" of a packet). The "fake HOST" pseudo-message generator facility within the IMP was selected to provide a minimal value of X_1 of 3.3 msec., which is the typical time required for the background program to generate a full packet of data. Similarly, the "fake HOST" generated traffic at a neighboring IMP was utilized for the maximal value of $X_1 = 23.0$ msec. which is the time required for the serial arrival of a full packet at the UCLA IMP, which in that case serves as a store-and-forward node. The third source of multi-packet traffic was to be from the network measurement center itself since the interference traffic and the test traffic could be created using separate message generators and message leaders (to trace only the test traffic). However, the latter condition does not provide a meaningful set of interference traffic since both the test traffic and the interference traffic must arrive on the same serial HOST-IMP interface, so only the theoretical curve is shown for the case of X_1 equal to 10.5 msec.

The theoretical values of the inter-packet gap time, τ, for various message lengths and values of X_1 are plotted in Figure 5.2.6 along with the test data from the experiment. The increase in the value of τ for very short messages is due to the fact that most of these messages are of the priority class, and therefore cause an increased amount of interference. Equation (5.144) indicates that for the special case in which all of the interference traffic is of the priority class, the value of τ will not be a function of X_1, and this effect was

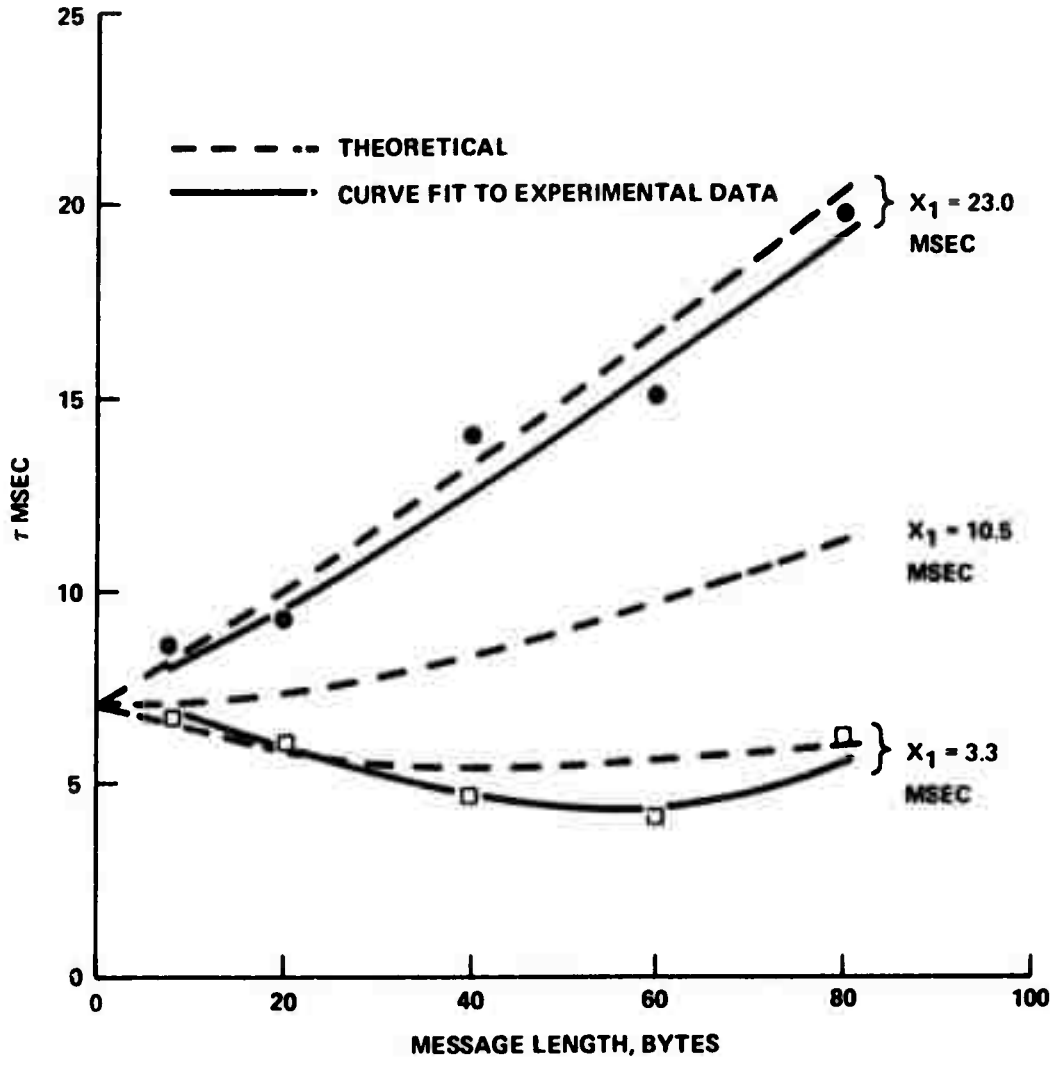


Figure 5.2.6. A Comparison of Measured and Theoretical Inter-Packet Gap Times Due to Interference Traffic.

also verified by the experimental data. This effect is due to the fact that the only arrivals that will contribute to the inter-packet gap are those that occur during the X_s seconds that a packet requires for service, and the subsequent arrivals during their service, etc., as expressed in the iterative result of Equation (5.131).

The experimental and theoretical results agree reasonably well for both the case of X_1 equal to 3.3 msec. and for X_1 equal to 23.0. Both the shape of the curves and the magnitude of the inter-packet gap (as measured by trace data at the destination) indicate that the model represents the actual system behavior satisfactorily.

The theoretical curves include the effects of the acknowledgements, the priority messsages, and the non-priority messages by considering the expected service time to be:

$$\begin{aligned} E[x] = {} & E[x \mid \text{an ACK}] \cdot Pr[\text{an ACK}] \\ & + E[x \mid \text{a pri. mes.}] \cdot Pr[\text{a pri. mes.}] \\ & + E[x \mid \text{non-pri. mes.}] \cdot Pr[\text{non-pri. mes.}] \end{aligned}$$

The acknowledgements are in response to the RFNM's (Request For Next Message), and therefore occur with the same frequency as messages, such that:

$$Pr[\text{an ACK}] = 0.5$$

Messages are considered to be in the priority class if they are less than or equal to 10 bytes in length, which for the exponential density of message length, results in:

$$\alpha_m = 1 - e^{-10/\bar{\ell}}$$

where $\bar{\ell}$ is the average message length in bytes, and α_m is the fraction of messages that are in the priority class. (The ACK's will also be in the priority class). The expected service time for a priority message can be found from the truncated exponential result of

Appendix A,

$$\bar{x}_p = E[x_m \mid x_m \leq X_T] = \frac{1}{\mu}\left[1 - \frac{\mu X_T e^{-\mu X_T}}{1 - e^{-\mu X_T}}\right]$$

and the expected service time for a non-priority message is also shown to be:

$$\bar{x}_{np} = E[x_m \mid x_m > X_T] = X_T + 1/\mu$$

Using these results, one can write the expected service time (of messages and ACK's) as:

$$E[x] = 0.5X_a + 0.5\alpha_m[X_0 + \bar{x}_p] + 0.5(1 - \alpha_m)[X_0 + X_T + 1/\mu]$$

where the first two terms represent the priority service times, and the last term considers the non-priority service times. The equation is of the form:

$$E[x] = \alpha\, E[x \mid \text{priority}] + (1 - \alpha)E[x \mid \text{non-priority}]$$

and if multiplied by the net arrival rate (of messages and ACK's), produces:

$$\lambda\, E[x] = \lambda\{0.5X_a + 0.5\alpha_m[X_0 + \bar{x}_p]\} + \lambda\{0.5(1 - \alpha_m)[X_0 + X_T + 1/\mu]\}$$

where:

$$\lambda = \lambda_a + \lambda_m = 2\lambda_m$$

with the subscripts referring to ACK's and messages. The values of the priority and non-priority interference traffic intensities can then be written as:

$$\rho_p = 2\lambda_m\{0.5X_a + 0.5\alpha_m[X_0 + \bar{x}_p]\}$$

and:

$$\rho_{np} = 2\lambda_m\{0.5(1 - \alpha_m)[X_0 + X_T + 1/\mu]\}$$

with the composite traffic intensity being:

$$\rho_c = \rho_p + \rho_{np}$$

If we consider an example in which the messages have an average length of 20 bytes, we find that 40% of the messages are in the priority class, and such messages have an average service time of 3.44 msec., compared to an average of 7.5 msec. for the non-priority messages (including the overhead characters for each). The service time of an acknowledgement is 3.0 msec., and when combined with the priority messages, produces a traffic intensity of:

$$\rho_p = \lambda_m[3.0 + 0.4(3.44)] = 0.175$$

where λ_m is 1/(25 msec.) for the artificial traffic being generated. The traffic intensity for the non-priority traffic can be found in a similar manner, since:

$$\rho_{np} = \lambda_m\{(1 - \alpha_m)[X_0 + X_T + 1/\mu]\}$$

which becomes:

$$\rho_{np} = \frac{0.6}{25}[2.9 + 1.6 + 3.2] = 0.185$$

such that:

$$\rho_c = \rho_p + \rho_{np} = 0.36$$

The expected value of τ can then be calculated for the known value of X_s equal to 23.0 msec., and for the desired value of X_1. These results are tabulated in Table 5.2.1 along with the associated values of ρ_p and ρ_{np}.

TABLE 5.2.1

THEORETICAL VALUES OF THE INTER-PACKET GAP, τ

Avg.Mes. Length (bytes)	ρ	α_m	ρ_p	ρ_{np}	Expected Value of τ		
					X_1=3.3	X_1=10.5	X_1=23.0
0	.23	1.00	.23	.00	6.9	6.9	6.9
8	.28	.72	.22	.06	6.7	7.3	8.3
20	.36	.39	.18	.18	5.8	7.3	10.1
40	.49	.22	.15	.34	5.3	8.3	13.3
60	.63	.15	.14	.49	5.6	9.7	16.8
80	.77	.12	.14	.63	6.0	11.3	20.4

5.2.4 The Special Case of Exponential Message Lengths

As in so much analysis, the assumption of an exponential service requirement simplifies the analysis by decoupling several effects which would otherwise be highly interdependent. This attribute of the exponential density is the well known memoryless property, i.e., the conditional density for the case of $x \geq X$ is also exponential with the same parameter value. This aspect of the exponential p.d.f. is useful in the segmented message case because of the fact that cutting off an arbitrary number of bits from the beginning of a message does not affect the statistics on the remaining portion of the message.

We will take advantage of this memoryless property in the following analysis, and subsequently will evaluate the results by an actual network experiment.

We will assume that message lengths are considered to be random variables, but that messages are partitioned into packet length segments for transmission through the network. That is, the bits of a message from the HOST are to be examined serially, until ℓ_p bits have been received, at which time a packet is formed, and the bit count is restarted. The service time for such a packet is assumed to be X_s seconds, such that for the exponential message length density with a mean message service requirement of $1/\mu$.

$$\Pr[\text{message service} \leq X_s] = 1 - e^{-\mu X_s} \tag{5.145}$$

and due to the memoryless property, this same probability applies to the remaining service requirement after an arbitrary number of packets have already been formed from the message. Therefore, since the probability that the message service will not be completed at any given packet, is:

$$\Pr[\text{service not completed}] = e^{-\mu X_s} \tag{5.146}$$

Such a process, in which the probability of continuing is the same at each epoch in time until a "success" is finally reached, leads to a geometric distribution of the number of trails, such that the number of packets in a message is governed by the probability law:

$$p(n) = \left(e^{-\mu X_s}\right)^{n-1}\left(1 - e^{-\mu X_s}\right) \qquad n = 1,2,3,\ldots \tag{5.147}$$

and the expected number of packets per message is simply:

$$E[n] = 1/\left(1 - e^{-\mu X_s}\right) \tag{5.148}$$

These results are well known, and have been utilized extensively in analysis, including as a model for the number of time-sharing quanta required to service a request (AD69). In many ways, the problem of multiplexing messages at a communications channel is similar to that of time-shared systems, and many of the following results may be applicable in both fields.

If messages are not segmented, then the queueing delay analysis is simply that of the well known M/M/1 case, i.e., messages arrive at an average rate of λ messages per second and require an average service time of $1/\mu$ seconds per job. The server utilization, ρ, for such a system is simply λ/μ, and we are interested in comparing this simple case with the segmented message situation. The following argument considers these factors for the two cases, and shows that for the M/M/1 system, the server utilization for the contiguous message case, ρ_m, is the same as for the segmented message case, ρ_p, in which the message is partitioned into fixed length packets with a service requirement for a full packet of X_s seconds.

We know that for the contiguous message case, the arrival rate and average service time are:

$$\lambda_m = \lambda$$

and

$$\bar{x}_m = 1/\mu$$

such that:

$$\rho_m = \lambda_m \bar{x}_m = \lambda/\mu$$

For the segmented message case, the arrival rate of packets is more than the corresponding rate for messages by a factor equal to the average number of packets per message,

$$\lambda_p = \lambda \cdot E[\text{no. of pkts. per mes.}]$$

which from Equation (5.148) is:

$$\lambda_p = \lambda/\left(1 - e^{-\mu X_s}\right) \tag{5.149}$$

The expected service time of a packet can be found from the service time distribution which is a truncated exponential with an impulse function at $x = X_s$. The first moment of such a function is shown in Appendix A to be:

$$\bar{x}_p = E[x \mid 0 \leq x \leq X_s] = \frac{1}{\mu}\left(1 - e^{-\mu X_s}\right)$$

such that:

$$\rho_p = \lambda_p \bar{x}_p = \frac{\lambda}{1 - e^{-\mu X_s}} \cdot \frac{1}{\mu}\left(1 - e^{-\mu X_s}\right)$$

or:

$$\rho_p = \lambda/\mu \tag{5.150}$$

which is precisely the same as the result for ρ_m and thereby proves that the two values of ρ are equal.

If we pursue this same line of reasoning, we can obtain some additional results relating the contiguous and segmented message

situations. One such case, which does not necessarily meet with one's intuitive expectations, relates the average number of packets in the queue for a segmented system to the number of messages in queue for a contiguous system. If there are an average of N packets per message, the intuitive expectation would be to find more than one, i.e., approximately N packets in the queue of the segmented system for each message in the queue of a comparable contiguous message system. However, we will show in Theorem 5.2.3 that just the opposite is true, if we can assume that the packet arrivals obey a Poisson process. This is not necessarily an acceptable assumption for the case where the network traffic is low, since in such cases the packets of a message would flow in a reasonably contiguous manner, and the interarrival times would be deterministic, i.e., X_s seconds apart (except for the arrival of the first packet). At higher traffic loads the packets tend to separate and would have more random interarrival times, which is encouraging since the analysis is of more concern at higher network loading. A second effect which introduces a randomization, is the multiplicity of input channels. Because the arrivals of interest are arrivals at the output modem queue, and a given packet may have reached the IMP from any one of several input channels, the composite arrivals to the queue have a more random pattern than the arrivals on any one input channel. However, the validity of the Poisson arrival assumption should be carefully considered in using the theorem result.

<u>Theorem 5.2.3</u>

The expected number of packets in queue for a segmented message case is less than or equal to the expected number of

messages in queue for a corresponding contiguous message system, for the case of Poisson arrivals and exponential message service requirements.

Proof:

The expected number of messages in the queue for a contiguous message M/M/1 system is known to be:

$$E[\text{no. of mes. in } Q \mid \text{contig. mes.}] = \frac{\rho^2}{1 - \rho} \tag{5.151}$$

The expected number of packets in the queue for the segmented case can be shown by use of the Pollaczek-Khinchine formula to be:

$$E[\text{no. of pkts in } Q \mid \text{seg. mes.}] = \frac{(\lambda_p)^2 \overline{x_p^2}}{2\,(1 - \rho_p)} \tag{5.152}$$

where λ_p is the arrival rate of packets, ρ_p is the server utilization factor for packets which was shown previously to be merely equal to ρ, and $\overline{x_p^2}$ is the second moment of the packet service time. We know from Equation (5.149) that the arrival rate of packets is-

$$\lambda_p = \lambda / \left(1 - e^{-\mu X_s}\right)$$

and from Appendix A, the second moment of the service time is:

$$\overline{x_p^2} = \frac{2}{\mu^2}\left(1 - (1 + \mu X_s)e^{-\mu X_s}\right)$$

such that the expected number of packets in the queue becomes:

$$\bar{n}_p = \left(\frac{\lambda}{1 - e^{-\mu X_s}}\right)^2 \frac{\frac{2}{\mu^2}\left(1 - (1 + \mu X_s)e^{-\mu X_s}\right)}{2(1 - \rho)}$$

or:

$$\bar{n}_p = \frac{\rho^2}{1 - \rho}\left[\frac{1 - (1 + \mu X_s)e^{-\mu X_s}}{\left(1 - e^{-\mu X_s}\right)^2}\right] \tag{5.153}$$

where we have replaced λ/μ by its equivalent, ρ. Since the factor involving ρ is equal to the expected number of messages in queue for a contiguous message system, we must then show that the second factor is always less than unity to prove the theorem. This can be done by showing that the numerator is never greater than the denominator for all $0 \leq X_s$, i.e.,

$$\left[1 - e^{-\mu X_s} - \mu X_s e^{-\mu X_s}\right] \overset{?}{\leq} \left[1 - 2e^{-\mu X_s} + \left(e^{-\mu X_s}\right)^2\right] \tag{5.154}$$

Subtracting the first two terms from both sides results in:

$$\left[-\mu X_s\, e^{-\mu X_s}\right] \overset{?}{\leq} \left[-e^{-\mu X_s}\left(1 - e^{-\mu X_s}\right)\right]$$

Dividing by the common term reverses the inequality due to the negative sign, such that:

$$\left[\mu X_s\right] \overset{?}{\geq} \left[1 - e^{-\mu X_s}\right]$$

For $\mu X_s \geq 1$, the inequality is obviously true. For $\mu X_s < 1$ we can expand the exponential in an infinite series, such that:

$$\left[\mu X_2\right] \overset{?}{\geq} \left[1 - \left(1 - \mu X_s + \frac{(\mu X_s)^2}{2!} - \ldots\right)\right]$$

or:

$$0 \overset{?}{\geq} \left| -\frac{(\mu X_s)^2}{2!} + \frac{(\mu X_s)^3}{3!} - \ldots \right|$$

and since $\mu X_s < 1$, the absolute value of each term decreases and so the first term is dominant, and the inequality must hold. Therefore, the average number of packets in the queue as expressed in Equation (5.153) must be less than, or at most equal to, that of the contiguous message case, which proves the theorem.

Corollary 1

The expected number of packets in the queue for the segmented message case of Theorem 5.2.3 will vary from that of the M/M/1 case for $X_s \to \infty$, to that of the deterministic service, M/D/1 case for $X_s \to 0$.

Proof:

The expected number of packets in the queue is expressed in Equation (5.153) for a full packet service requirement of X_s seconds. Both the numerator and the denominator of the term in brackets are functions of X_s and both go to zero as X_s becomes very small. The use of L'Hospital's rule produces:

$$\lim_{\mu X_s \to 0} \left[\frac{1 - (1 + \mu X_s)e^{-\mu X_s}}{\left(1 - e^{-\mu X_s}\right)^2} \right] = \lim_{\mu X_s \to 0} \left[\frac{(1 + \mu X_s)e^{-\mu X_s} - e^{-\mu X_s}}{2\left(1 - e^{-\mu X_s}\right)e^{-\mu X_s}} \right]$$

or:

$$\lim_{\mu X_s \to 0} \left[\frac{\mu X_s}{2\left(1 - e^{-\mu X_s}\right)} \right]$$

which is still indeterminate. A second application of the rule results in:

$$\lim_{\mu X_s \to 0} \left| \frac{1}{2\, e^{-\mu X_s}} \right| = \frac{1}{2}$$

Therefore, we have the limiting case of:

$$\lim_{\mu X_s \to 0} \bar{n}_p = \frac{\rho^2}{2(1 - \rho)} \tag{5.155}$$

which is the expected number in queue for an M/D/1 queueing system. This result can be seen intuitively by noting that as X_s becomes small, most of the packets will be full length packets and will therefore require a fixed (deterministic) service time.

At the other extreme, as X_s becomes very large, the system reverts to the contiguous message M/M/1 case, with:

$$\lim_{\mu X_s \to \infty} \bar{n}_p = \frac{\rho^2}{(1 - \rho)} \tag{5.156}$$

Thus the two extreme cases are shown to be the M/D/1 system for $\mu X_s \to 0$ and the M/M/1 system for $\mu X_s \to \infty$, which proves the corollary.

We noted in Section 5.2.1 that the arrival epochs changed when we segmented messages because, by definition, an arrival occurs when the last bit of a contiguous stream is received. Making these message segments shorter reduced the unfinished work in the system, and from the corollary result, reduces the expected unfinished work to an arbitrarily small value since we have a lesser number of message segments in the queue, and each segment is of arbitrarily small size. Of

course, in practice one must include some fixed length header and communications control information with each such segment, so it is not practical to segment messages to that extent. This leads one to assume that there must be some optimal packet length which will be a function of both the overhead and the average message length, and such an extension of this analysis is considered in Section 5.2.5.

5.2.5 The Optimal Segment Size

We observed in Section 5.2.3 that making a segment (or packet in ARPA network terminology) as small as possible improved the network performance, and therefore, indicated that segmentation should be done in very small increments. However, each packet must contain identification information (its header) and line control characters which constitute a fixed overhead per packet and preclude the usage of very small segment sizes. An optimal packet size must then exist, which will be a function of the message length distribution and the packet overhead.

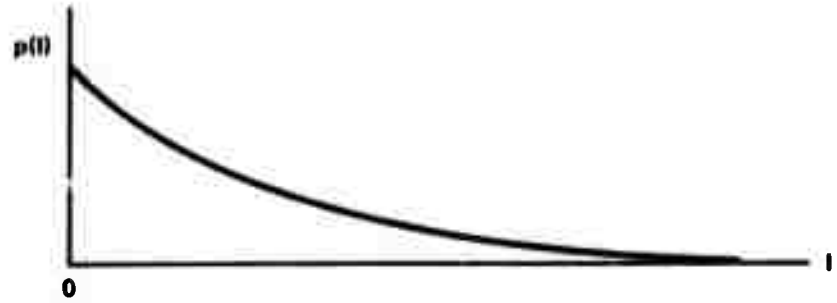
The criteria for the optimization will depend on the aspect of the network behavior which is considered to be most sensitive to this selection, and might include the preemptive-resume-like behavior described in Section 5.1, the need for retransmission due to errors as considered by Kirlin (KI69), or the efficient utilization of buffer storage. This latter effect is felt to be of particular significance in the ARPA network, and will be considered to be the criteria for the optimization.

5.2.5.1 The Optimal Segmentation for Exponential Message Lengths

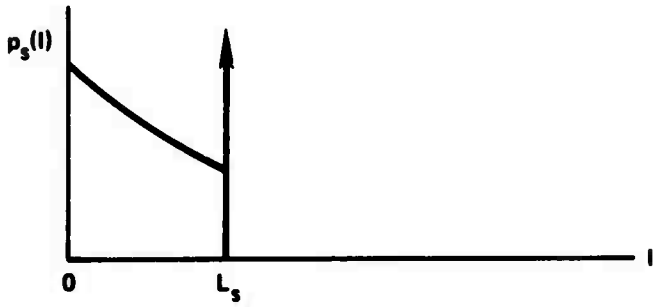
If we consider an exponential density of message lengths as shown in Figure 5.2.7(a), and the effect of segmenting messages into packets of length L_s, (e.g., by taking the first L_s units of the message and forming a packet) then we have the packet length density shown in Figure 5.2.7(b), which includes an impulse function whose area is equal to that of the portion of the density for $\ell > L_s$. The remaining portion of the density then becomes the new density of interest, conditioned on the fact that $\ell > L_s$. For the exponential density, the conditional form is identical to the original function, a characteristic which is unique to the exponential and leads to its "memoryless" property which makes it so tractable in analysis.

Another aspect of the optimization is the selection of the buffering attribute to be optimized, e.g., to minimize the total wasted space, or some other criteria of concern. The selected criterion was to optimize the efficiency of buffer storage utilization, i.e., to maximize the ratio of effective storage utilized for message text relative to the total storage required for the fixed length block, the header information, and any other packet overhead. (The line control characters do not need to be considered since they are hardware generated, and we are only concerned with memory utilization. However, any necessary pointers for linking buffers should be included.)

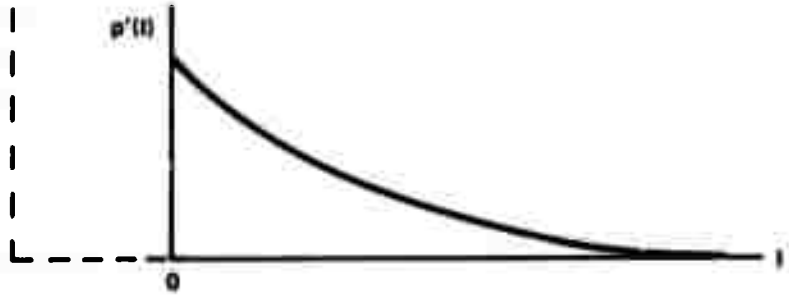
If we assume an exponential distribution of message lengths with a mean length of ℓ_m characters, a maximum text length of L_s characters per buffer, and a header of H characters, we will have an average buffer efficiency of:



(a) THE ORIGINAL EXPONENTIAL DENSITY



(b) THE TRUNCATED DENSITY



(c) THE CONDITIONAL DENSITY

Figure 5.2.7. The Effect of Segmentation on the Exponentential Density of Message Lengths.

$$\eta = \frac{\bar{\ell}}{L_s + H} = \frac{\ell_m[1 - e^{-L_s/\ell_m}]}{L_s + H} \tag{5.157}$$

where $\bar{\ell}$ is the mean of a truncated exponential as derived in Appendix A. This efficiency is shown in Figure 5.2.8 as a function of length L_s for several values of H. Both L_s and H are shown normalized with respect to the mean message length, ℓ_m.

The optimal value of L_s/ℓ_m is the point at which the efficiency is maximized. We can therefore calculate these optimal points by differentiating Equation (5.157) with respect to L_s, which yields:

$$\frac{d\eta}{dL_s} = \frac{e^{-L_s/\ell_m}}{L_s + H} - \frac{\ell_m[1 - e^{-L_s/\ell_m}]}{(L_s + H)^2} \equiv 0$$

such that:

$$e^{-L_s/\ell_m}[L_s + H] - \ell_m[1 - e^{-L_s/\ell_m}] \equiv 0 \tag{5.158}$$

A root of the equation will be the value of the length, L_s, at which the optimal efficiency occurs. If we find the proper root of this equation and then insert it in Equation (5.157), we can find the equation for the locus of the optimal points (shown as a dashed line in Figure 5.2.8). However, to avoid the necessity of solving Equation (5.158) for its roots, we will take advantage of the form of the numerator of Equation (5.157) and note that:

$$\frac{\ell_m[1 - e^{-L_{s0}/\ell_m}]}{L_{s0} + H} = e^{-L_{s0}/\ell_m}$$

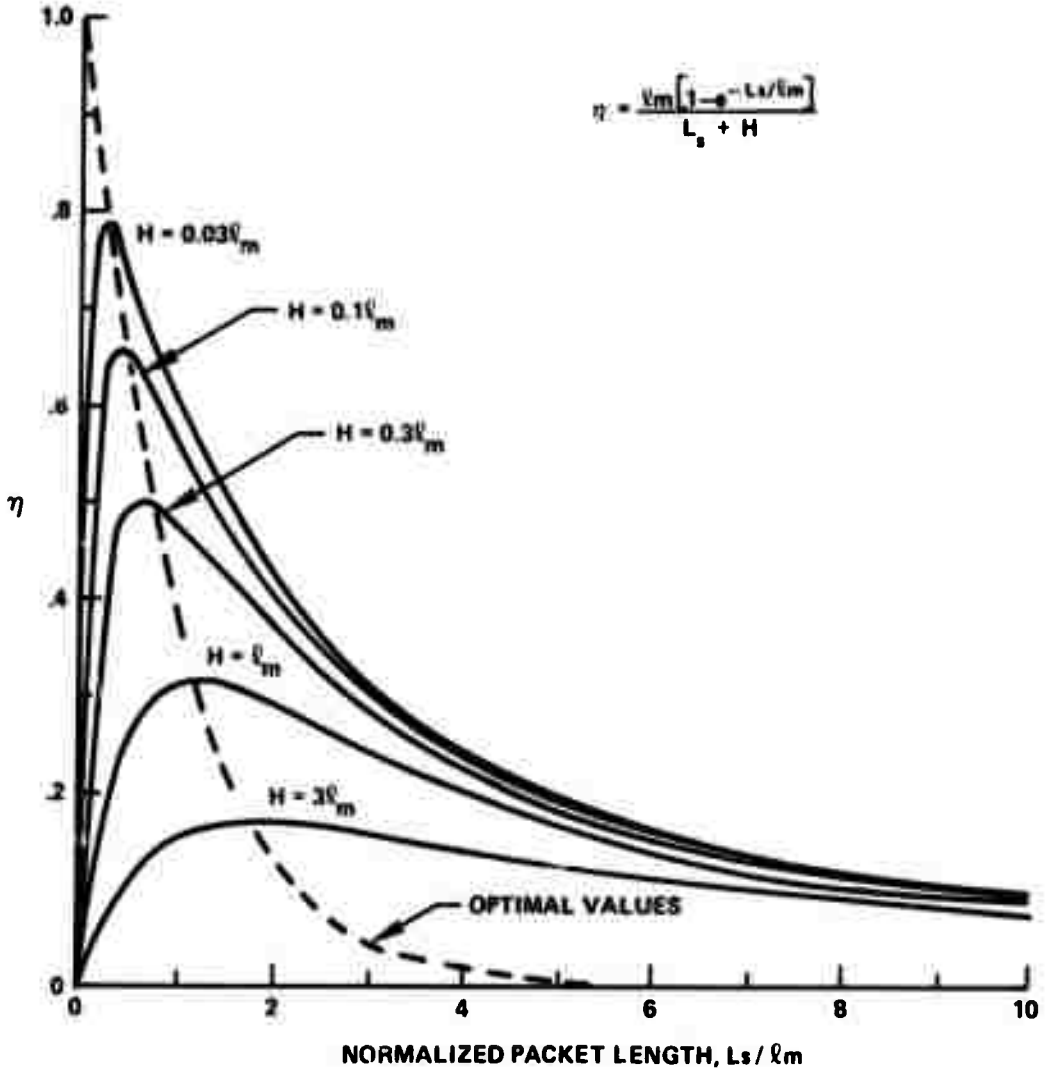


Figure 5.2.8. Buffer Utilization Efficiency as a Function of the Normalized Packet Length and Packet Storage Overhead.

with the subscripts indicating that L_{s0} is the length corresponding to an optimal value of the efficiency. If we substitute into Equation (5.157) we obtain:

$$\eta(\text{opt}) = e^{-L_{s0}/\ell_m} \tag{5.159}$$

which defines the locus of points shown in Figure 5.2.8.

The optimal buffer size as determined in the above analysis was later found to be consistent with an earlier result obtained by Wolman (WO65), although the two approaches were quite different. Wolman obtained a relationship between the parameters, which, using our symbols, is:

$$L_s \cong \sqrt{2\ H\ \ell_m}$$

and he found this result to be applicable to most "smooth" density functions, i.e., those which do not have large discrete or concentrated probability components. His approach was to minimize the wasted space, as compared to our maximization of the utilized space, and both analyses considered the same overhead components. His result has the advantages of some generality and of being an explicit equation for the segment size in terms of the average message length and the overhead per packet. However, it does not indicate the effect of non-optimal segment size selection, i.e., the sensitivity of the efficiency to the choice of the packet length as indicated in our Figure 5.2.8. The two results are, therefore, felt to be very complementary, and tend to confirm the resulting selection criteria for packet size.

5.2.5.2 Experimental Verification of the Model

The buffer utilization efficiency can be viewed as the ratio of the average packet size (of useful information) to the total storage requirement of a fixed length buffer. The latter figure includes any packet header information, marking, padding, and the necessary pointer(s) for linking the buffers on the queues. The model was developed to give some indication of the optimal buffer size for a given average message length (assuming an exponential density of lengths). A logical verification test would be to vary the buffer size and measure the resulting storage efficiency, η. However, it is difficult to change the buffer size since it is an IMP design parameter, so an alternative method of varying the average message size was pursued. Changes in the parameter, ℓ_m, are equally effective in changing the ratio L_s/ℓ_m as variations in L_s, and result in the efficiency curve shown in Figure 5.2.9. This plot was obtained by generating artificial traffic with an exponential distribution of lengths, and measuring the average information content of packets as the average message length was varied. Each buffer was assumed to consist of a total of 138 bytes of message content plus 12 bytes of overhead), with a resulting efficiency of:

$$\eta = \frac{(\text{\# bytes sent}/\text{\# pkts. sent})}{138} \tag{5.160}$$

The two measured values were the number of message content bytes transmitted on the modem channel, and the number of packets sent on that same channel. (Both of these values are available from the accumulated statistics as described in Section 2.5.1)

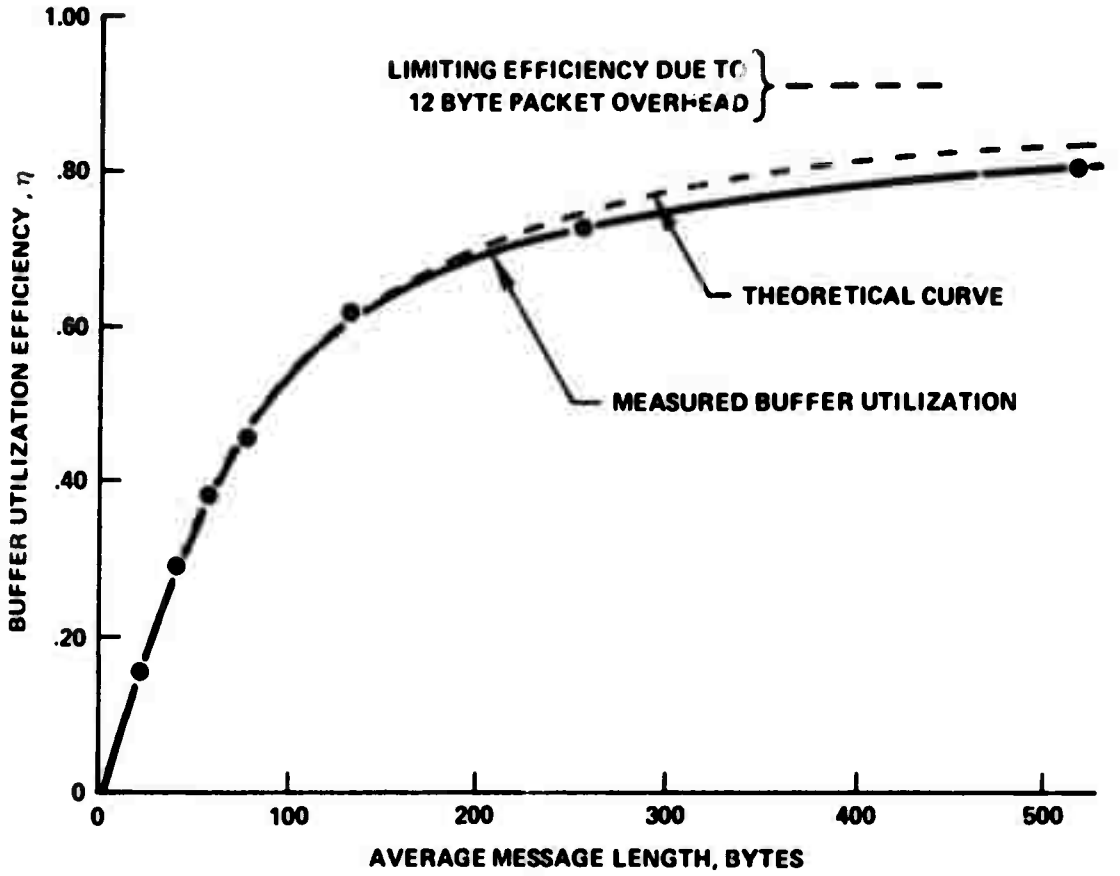


Figure 5.2.9. Buffer Utilization Efficiency as a Function of Average Message length (Exponential Density)

Figure 5.2.9 differs from the theoretical curve of Figure 5.2.8 since the latter was based on the independent variable being L_s. However, a second theoretical curve is shown in Figure 5.2.9 as a function of ℓ_m, and a reasonably close agreement can be seen between the theoretical and measured data. The difference at higher values of ℓ_m is due to the fact that message sizes are limited to a maximum of 1000 bytes, and this was not taken into account in the theoretical model.

This same data can be compared with the theoretical curve of Figure 5.2.8 by rescaling the effective overhead, H, to be:

$$\frac{H(\text{eff.})}{\ell_m} = \frac{6}{63} \tag{5.161}$$

That is, the value of H is six for L_s equal to 63 and should retain that same normalized value as ℓ_m is varied. Therefore,

$$H(\text{eff.}) = 6\ \ell_m/63 \tag{5.162}$$

will retain the effective value of H equal to approximately 0.1 as the ratio L_s/ℓ_m is varied. The resulting theoretical and measured values are compared in Figure 5.2.10, and again indicate a reasonably good match between theory and experiment. The same truncation effects that influenced the upper portion of the curve in Figure 5.2.9 are now seen in the lower values of L_s/ℓ_m, and cause the two curves to be somewhat different in that range.

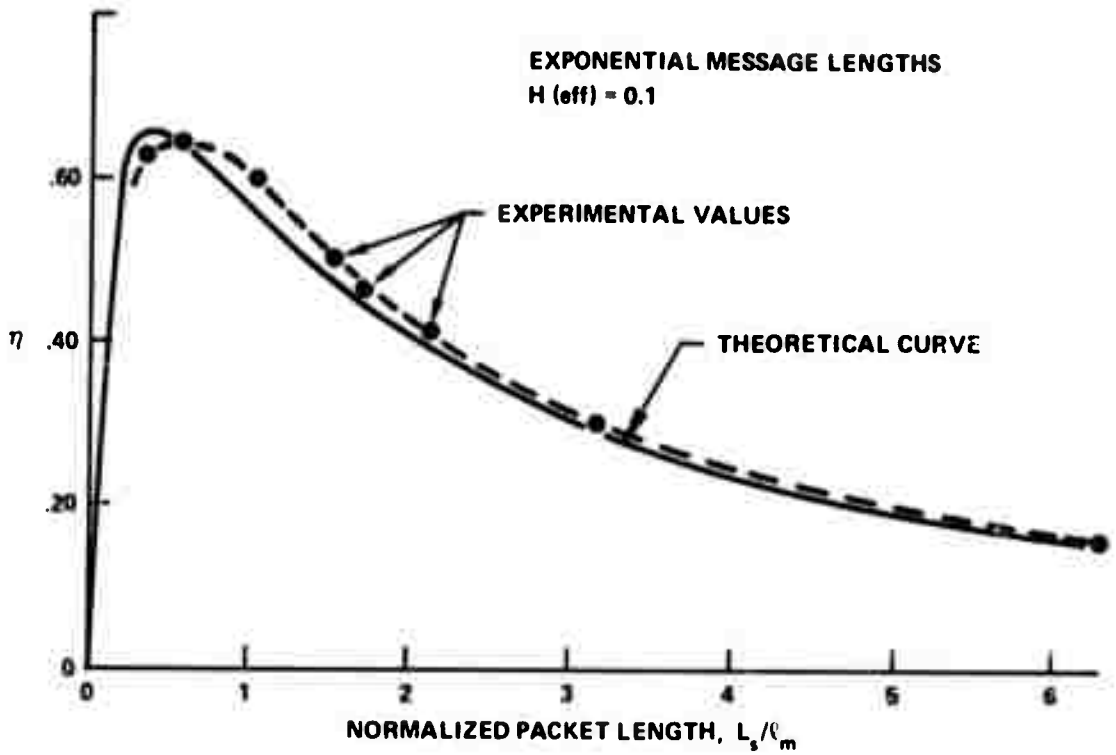


Figure 5.2.10. Buffer Utilization Efficiency as a Function of the Normalized Packet Length (Based on Adjusted Values of the Overhead, H)

CHAPTER 6

CONCLUSIONS AND SUGGESTED FUTURE RESEARCH

This research has shown that an extensive measurement capability can be developed within a store-and-forward communications network, and that with proper care, these measurement facilities can utilize the processing and transmission facilities of the data network. However, we found it necessary to work very closely with the network implementation group to ensure that the necessary data could be obtained with a minimum perturbation to the system behavior. These measurement considerations included the selection of the proper set of measurement facilities, and the introduction of the capability to selectively enable or disable the various measurement routines at each node in the network.

The measurement facilities were initially utilized to verify the design and operation of the network, and as part of this activity, an extensive artificial traffic generation capability was added to the set of measurement tools. This artificial traffic generation capability was later found to be extremely valuable for the simulation of traffic flow conditions, and was subsequently modified to include random message length and transmission time capabilities, in addition to the original deterministic traffic generation capabilities. The importance of artificial traffic generation was not fully appreciated initially, but evolved to be one of the more valuable network measurement tools.

Subsequent network measurement activities have demonstrated that the network performance can be evaluated, and that anomalous behavior can be discovered and information can be gained regarding its cause. The complementary nature of the measurement and modeling efforts was also demonstrated, with the experimental results being utilized in several instances to improve or correct the models. The models have, in turn, been utilized as the basis of further network experimentation, with the modeling and measurement functions being found to be mutually constructive in an iterative scheme of model development and verification testing.

Several application areas are envisioned for future network experimentation including the measurement of the network traffic for use in determining future topological and bandwidth changes, as well as to identify the different user transmission characteristics such as those due to interactive teletypewriter activity and graphic display data requirements. Measurements are also planned to determine areas in which dynamic control mechanisms could be effectively utilized to enhance the network performance, and to determine more extensive measures of such system performance.

A number of possible measurements of interest would require the cooperation of one or more HOST facilities to obtain the necessary data. These experiments include measurements of the performance of the overall network facilities including the HOST-implemented protocol functions, and the degree of system degradation imposed by such protocol implementations in terms of both the memory space and response time considerations. A final suggestion for future research which may be of

particular interest in determining the viability of network techniques, is to determine the degree to which users at the various sites are able to utilize each other's resources rather than regenerating a local version of the same facilities. Such measurements would also determine the features that should be available in future program and hardware developments that will make them more useful in a network environment. It is hoped that such future measurement activities can extend and build upon the measurement facilities which were developed as part of this research.

BIBLIOGRAPHY

AB33 Abernathy, J.R., "On the Elimination of Systematic Errors Due to Grouping," Annals of Math. Stat., Vol. 4, 1933, pp. 263-277.

AD69 Adiri, I., "Computer Time-Sharing Queues with Priorities," JACM, Vol. 16, No. 4, Oct. 1969, pp. 631-645.

BA64 Baran, P., "On Distributed Communications Networks," IEEE Trans. on Comm. Sys., Vol. CS-12, March 1964.

BA70 Batson, A., et al., "Measurements of Segment Size," Communications of the ACM, Vol. 13, No. 3, March 1970, pp. 155-159.

BE69 Benvenuto, A.A., et al., "System Load Sharing Study," The MITRE Corporation, Tech. Report, MTR-5062, 1969.

BE70A Bell, C.G., et al., "Computer Networks," Computer Group News, Sept.-Oct. 1970, pp. 13-23.

BE70B Bendick, M., "Final Report of the Electronic Industries Association Committee on Networking Computer Systems," System Development Corporation, SP-3514, Santa Monica, Calif., May 1970.

BH68 Bhusan, A.K. and Stotz, R.H., "Procedures and Standards for Intercomputer Communications," Proc. of the Spring Joint Comp. Conf., Vol. 32, Thompson Book Co., Washington, D.C., 1968, pp. 95-104.

BO69 Boehm, B.W. and Mobley, R.L., "Adaptive Routing Techniques for Distributed Communications Systems," IEEE Transactions on Communications Technology, Vol. Com-17, No. 3, June 1969, pp. 340-349.

BR67 Bryan, G.E., "Joss: 20,000 Hours at a Console - A Statistical Summary," Proc. of the Fall Joint Comp. Conf., Vol. 31, 1967, pp. 769-777.

BU56 Burke, P.J., "The Output of a Queueing System," Operations Research, Vol. 4, 1956, pp. 699-704.

CA68A Cantrell, H.N. and Ellison, A.L., "Multiprogramming System Performance Measurement and Analysis," Proc. of the Spring Joint Comp. Conf., Vol. 32, Thompson Book Co., Washington, D.C., 1968, pp. 213-221.

Preceding page blank

CA68B Campbell, D.J. and Heffner, W.J., "Measurement and Analysis of Large Operating Systems During System Development," Proc. of the Fall Joint Comp. Conf., Vol. 33, Thompson Book Co., Washington, D.C., 1968, pp. 903-914.

CA70 Carr, C.S., Crocker, S.D., and Cerf, V.G., "HOST-HOST Communication Protocol in the ARPA Network," Proc. of the Spring Joint Comp. Conf., Vol. 36, AFIPS Press, Montvale, N.J., 1970, pp. 589-597.

CO54 Cobham, A., "Priority Assignments in Waiting Line Problems," Operations Research, Vol. 2, 1954, pp. 70-76, "A Correction," Operations Research, Vol. 3, 1955, p. 547.

CO61 Cox, D.F. and Smith, W.L., Queues, Methuen & Co. Ltd., London, 1961.

CO67 Conway, R.W., Maxwell, W.L., and Miller, L.W., Theory of Scheduling, Addison-Wesley, Reading, Mass., 1967.

CO68 Coffman, E.G., and Kleinrock, L., "Feedback Queueing Models for Time-Shared Systems," JACM, Vol. 15, No. 4, Oct. 1968, pp. 549-576.

CO70 Cohen, J.W., "On Memory Units with Finite Capacity," Sixth International Teletraffic Congress, Technische Hochschule, Munich, Sept. 9-10, 1970, pp. 321/1-7.

CR61 Cramer, H., Mathematical Methods of Statistics, Princeton University Press, Princeton, N.J., 1961.

DA67 Davies, D.E., et al., "A Digital Communication Network for Computers Giving Rapid Response at Remote Terminals," ACM Symposium on Operating System Principles, Gatlinburg, Tenn., Oct. 1967.

DA68 Davies, D.W., "Communication Networks to Serve Rapid-Response Computers," IFIP Congress 1968, North Holland Publishing Co., Amsterdam, 1968, pp. 72-78.

DE69 Deniston, W.R., "SIPE: A TSS/360 Software Measurement Technique," Proc. of the 24th Natl. Conf., ACM, Assoc. for Comp. Mach., New York, N.Y., 1969, pp. 229-239.

EL33 Elderton, W.P., "Adjustments for the Moments of J-Shaped Curves," Biometrika, Vol. 25, 1933, pp. 179-180, "Note on Mr. Palin Elderton's Corrections to Moments of J-Shaped Curves," Biometrika, Vol. 25, 1933, pp. 180-184.

ES67A Estrin, G., et al., "Snuper Computer - A Computer Instrumentation Automaton," Proc. of the Spring Joint Comp. Conf., Vol. 30, Thompson Book Co., Washington, D.C., 1967, pp. 645-656.

ES67B Estrin, G. and Kleinrock, L., "Measures, Models and Measurements for Time-Shared Computer Utilities," Proc. of the 22nd National Conference, ACM, Thompson Book Co., Washington, D.C., 1967, pp. 85-96.

EV67 Evans, G.J. Jr., "Experience Gained From the American Airlines SABRE System Control Program," Proc. of the 22nd National Conference, ACM, Thompson Book Co., Washington, D.C., 1967, pp. 77-83.

FA65 Fano, R.M., "The MAC System: The Computer Utility Approach," IEEE Spectrum, Vol. 2, No. 1, Jan. 1965, pp.56-64.

FI59 Finch, P.D., "The Output Process of the Queueing System M/G/1," J. Roy. Stat. Soc., Ser. B, Vol. 21, No. 2, 1959, pp. 375-380.

FR65 Freidman, H.D., "Reduction Methods for Tandem Queueing Systems," Operations Research, Vol. 13, 1965, pp. 121-131.

FR70 Frank, H., et al., "Topological Considerations in the Design of the ARPA Net," Proc. of the Spring Joint Comp. Conf., Vol. 36, AFIPS Press, Montvale, N.J., 1970, pp. 581-587.

FU70 Fuchs, E. and Jackson, P.E., "Estimates of Distributions of Random Variables for Certain Computer Communications Traffic Models," Comm. of the ACM, Vol. 13, No. 12, 1970, pp. 752-757.

FU71A Fultz, G.L. and Kleinrock, L., "Adaptive Routing Techniques for Store-and-Forward Computer-Communications Networks," International Conference on Communications, Montreal, Canada, June 14, 1971.

FU71B Fultz, Gary, Adaptive Routing Techniques for Store-and-Forward Message Switching Computer-Communications Networks, Ph.D., Department of Computer Science, University of California, Los Angeles, 1971.

GA67 Gaver, D.P., "Probability Models for Multiprogramming Computer Systems," JACM, Vol. 14, No. 2, July 1967, pp. 423-438.

GE69 Gerhard, W.D. (Ed.), Proceedings of the Invitational Workshop on Networks of Computers (NOC-68) 1968, National Security Agency, Fort George G. Meade, Maryland, Sept. 1969.

GO67 Gordon, W.J. and Newell, G.F., "Closed Queueing Systems with Exponential Servers," Operations Research, Vol. 15, 1967, pp. 254-265.

GR68 Gruenberger, F. (Ed.), Computers and Communications - Toward a Computer Utility, Prentice-Hall, Englewood, N.J., 1968.

HA68 Hamsher, M. (Ed), Communication System Engineering Handbook, McGraw-Hill, New York, N.Y., 1968.

HA69 Hamming, R.W., "One Man's View of Computer Science," JACM, Vol. 16, No. 1, Jan. 1969, pp. 3-12.

HA70 Hamaker, R.F., "Distributed Computer Systems," Telecommunications, March 1970, pp. 25-30.

HE70 Heart, F.E., et al., "The Interface Message Processor for the ARPA Computer Network," Proc. of the Spring Joint Comp. Conf., Vol. 36, AFIPS Press, Montvale, N.J., 1970, pp. 551-567.

HE71 Hersch, P., "Data Communications," IEEE Spectrum, Vol. 8, No. 2, Feb. 1971, pp. 47-60.

HI68 Hildebrand, D., "On the Capacity of Tandem Server, Finite Queue Service Systems," Operations Research, Vol. 16, No. 1, 1968, pp. 72-82.

HU56 Hunt, C.G., "Sequential Arrays of Waiting Lines," Operations Research, Vol. 4, 1956, pp. 674-683.

JA57 Jackson, J.R., "Networks of Waiting Lines," Operations Research, Vol. 5, 1957, pp. 518-521.

JA69 Jackson, P.E. and Stubbs, C.D., "A Study of Multi-Access Computer Communications," Proc. of the Spring Joint Comp. Conf., Vol. 34, AFIPS Press, Montvale, N.J., 1968, pp. 491-504.

JO64 Johnson, N.L. and Leone, F.C., Statistics and Experimental Design, Wiley, New York, N.Y., 1964.

KA69 Karush, A.D., "Two Approaches for Measuring the Performance of Time-Sharing Systems," System Development Corp. Report SP-3364 Santa Monica, Calif., 1969.

KE57 Kesten, H. and Runnenburg, J.,"Priority in Waiting Line Problems," Koninkl. Ned. Akad. Wetenschap., Proc., Vol. 60, 1957, pp. 312-336.

KI69 Kirlin, R.L., "Variable Block Length and Transmission Efficiency," IEEE Trans. on Comm. Tech., Vol. COM-17, No. 3, June 1969, pp. 340-349.

KL64 Kleinrock, L., Communication Nets: Stochastic Message Flow and Delay, McGraw-Hill, New York, N.Y., 1964.

KL67 Klir, J., "Processing of General System Activity," General Systems, Vol. 12, 1967, pp. 193-198.

KL69A Kleinrock, L., "Models for Computer Networks," Proc. of the IEEE International Conf. on Communications, Boulder, Colo., June 1969.

KL69B Kleinrock, L., "Comparison of Solution Methods for Computer Network Models," Proc. of the Computers and Communications Conference, Rome, N.Y., Sept. 1969.

KL69C Kleinrock, L., "Certain Analytic Results for Time-Shared Processors," IFIP Congress 1969, North Holland Publishing Co., Amsterdam, 1969, pp. 838-845.

KL70 Kleinrock, L., "Analytic and Simulation Methods in Computer Network Design," Proc. of the Spring Joint Comp. Conf., Vol. 36, AFIPS Press, Montvale, N.J., 1970, pp. 551-567.

LI61 Little, J.D.C., "A Proof for the Queueing Formula $L = \lambda W$," Operations Research, Vol. 9, 1961, pp. 383-387.

LI68 Licklider, J.C.R., Taylor, R.W. and Herbert, E., "The Computer as a Communication Device," Science and Technology, No. 76, April 1968, pp. 21-31.

MA66 Marill, T. and Roberts, L.G., "Toward a Cooperative Network of Time-Shared Computers," Proc. of the Fall Joint Comp. Conf., Vol. 29, Thompson Book Co., Washington, D.C., 1966, pp. 425-431.

MA69 Martin, J., Telecommunications and the Computer, Prentice-Hall, Englewood Cliffs, N.J., 1969.

MA70 Martin, J., Teleprocessing Network Organization, Prentice-Hall, Englewood Cliffs, N.J., 1970.

MO58 Morse, P.M., Queues, Inventories and Maintenance, Wiley, New York, N.Y. 1958.

MU69 Murphey, R.W., "The System Logic and Usage Recorder," Proc. of the Fall Joint Comp. Conf., AFIPS Press, Montvale, N.J., 1969, pp. 219-229

PA60 Parzen, E., Modern Probability Theory and Its Application, Wiley, New York, N.Y., 1960.

PA65A Payne, A. H., "On Measuring the Value of Information - With Implications for Communications Systems," Center for Naval Analysis, Report No. AD624-785, Jan. 1965.

PA65B Papoulis, A., Probability, Random Variables, and Stochastic Processes, McGraw-Hill, New York, N.Y., 1965.

PH56 Phipps, T.E. Jr., "Machine Repair as a Priority Waiting Line Problem," Operations Research, Vol. 4, 1956, pp. 76-85.

PI71 Pierce, J.R., et al., "An Experiment in Addressed Block Data Transmission Around a Loop," Digest of the IEEE 71 Inter-National Convention, Inst. of Electrical and Electronic Engr., Inc., New York, 1971, pp. 222-223.

PR62 Prosser, R.T., "Routing Procedures in Communications Networks," Parts 1 and 2, IRE Trans. on Comm. Systems, CS 10, 1962, pp. 322-329 and pp. 329-335.

PR69 Presser, L. and Melkanoff, M.A., "Software Measurements and Their Influence Upon Machine Language Design," Proc. of the Spring Joint Comp. Conf., AFIPS Press, Montvale, N.J., 1969, pp. 733-737.

RE57 Reich, E., "Waiting Times When Queues Are in Tandem," Annals of Math. Stat., Vol. 28, No. 3, Sept. 1957, p. 768.

RO67 Roberts, L.G., "Multiple Computer Networks and Intercomputer Communications," ACM Symposium on Operating System Principles, Gatlinburg, Tenn., Oct. 1967.

RO70 Roberts, L.G. and Wessler, B.D., "Computer Network Development to Achieve Resource Sharing," Proc. of Spring Joint Comp. Conf., Vol. 36, AFIPS Press, Montvale, N.J., 1970, pp. 543-549.

RU69A Russell, E.C. and Estrin, G., "Measurement Based Automatic Analysis of FORTRAN Programs," Proc. of the Spring Joint Comp. Conf., AFIPS Press, Montvale, N.J., 1969, pp. 723-732.

RU69B Rutledge, R.M., et al., "An Interactive Network of Time-Sharing Computers," Proc. of the 24th National Conference, ACM, Assoc. for Comp. Mach., New York, N.Y., 1969, pp. 431-441

SA61 Saaty, T.L., Elements of Queueing Theory, McGraw-Hill, New York, N.Y., 1961.

SC67A Scherr, A.L., An Analysis of Time-Shared Computer Systems, The MIT Press, Cambridge, Mass., 1967.

SC67B Schulman, F.D., "Hardware Measurement Device for the IBM System 360 Time-Sharing Evaluation," Proc. of 22nd National Conference, ACM, Thompson Book Co., Washington, D.C., 1967, pp. 103-109.

SH98 Sheppard, W.F., "On the Calculation of the Most Probable Values of Frequency Constants for Data Arranged According to Equidistant Divisions of a Scale," Proc. of London Math. Soc., Vol. 29, 1898, pp. 353-380

TA67 Tarter, M.E., et al., "After the Histogram, What? A Description of New Computer Methods for Estimating the Population Density," Proc. of the 22nd National Conference, ACM, Thompson Book Co., Washington, D.C., 1967, pp.511-519.

TO65 Totschek, R.A., "An Empirical Investigation into the Behavior of the SDC Time-Sharing System," System Development Corp. Report SP-2191, AD622003, Santa Monica, Calif., 1965.

UZ70 Uzgalis, R., "A Short History of Computer System Modeling and Measurement at the University of California, Los Angeles," Symposium on Structure of Computers and Operating Systems, Erlangen, Germany, Oct. 1970.

WA66 Wallace, V.G. and Rosenberg, R.S., "Markovian Models and Numerical Analysis of Computer Systems," Proc. of the Spring Joint Comp. Conf., Vol. 28, Thompson Book Co., Washington, D.C., 1966, pp. 141-148

WE64 Weingarten, A., "Storage Requirements for a Message Switching Computer," IEEE Trans. on Comm. Sys., Vol. CS-12, No. 6, June 1964, pp. 191-195.

WH58 White, H. and Christie, L.S., "Queueing With Pre-emptive Priorities or With Breakdown," Operations Research, Vol. 6, 1958, pp. 79-95.

WH70 Whitney, V. Kelvin Moore, A Study of Optimal File Assignment and Communication Network Configuration in Remote-Access Computer Message Processing and Communication Systems, Ph.D. in Electrical Engineering, University of Michigan, Ann Arbor, Mich., 1970.

WO65 Wolman, E., "A Fixed Optimum Cell-Size for Records of Various Lenghts," JACM, Vol. 12, No. 1, Jan. 1965, pp. 53-70.

ZE71 Zeigler, J.F. and Kleinrock, L., "Nodal Blocking in Large Networks," International Conference on Communications, Montreal, Canada, June 14, 1971.

APPENDIX A

THE MOMENTS OF THE TRUNCATED EXPONENTIAL DENSITY

The analysis of both the two class priority queueing system and the segmented message service case required that one calculate the moments of truncated density functions. Such moments have been derived by Adiri (AD69) and also by Coffman and Kleinrock (CO68), and are summarized in the following equations, although the notation and form of the results have been modified to suit our purposes more directly.

a. The Truncated Exponential With a Deterministic Component

When messages are segmented into fixed length segments or packets, some fraction of the segments are of the maximal length, and result in a deterministic component of the service time density function as shown in Figure A.1(b). The density also has a fixed offset at the origin to account for the overhead service requirement such as the line control characters and the header.

The moments of this function can be shown to be:

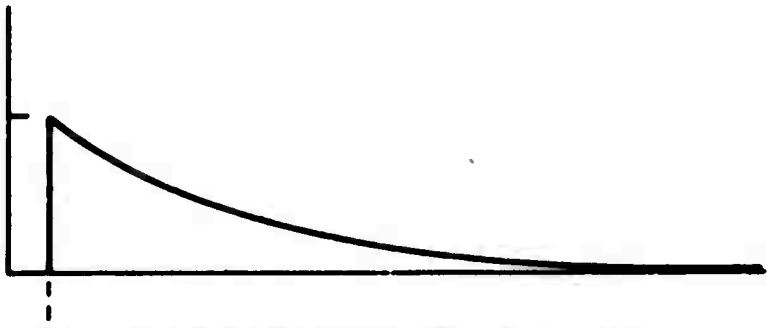
$$E[x] = X_0 + \frac{1}{\mu}\,[1 - e^{-\mu X_s}] \tag{A.1}$$

and:

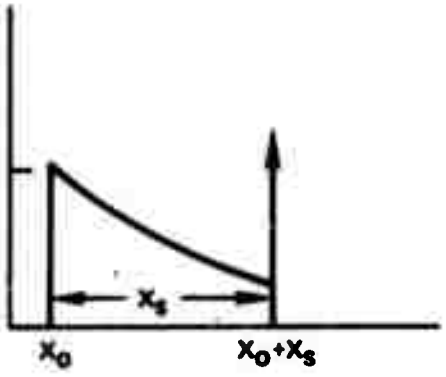
$$E[x^2] = X_0^2 + \frac{2}{\mu^2}\,[1 - (1 + \mu X_s)e^{-\mu X_s}]$$

$$+ \frac{2X_0}{\mu}\,[1 - e^{-\mu X_s}] \tag{A.2}$$

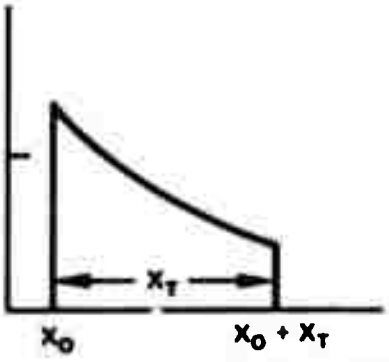
Preceding page blank



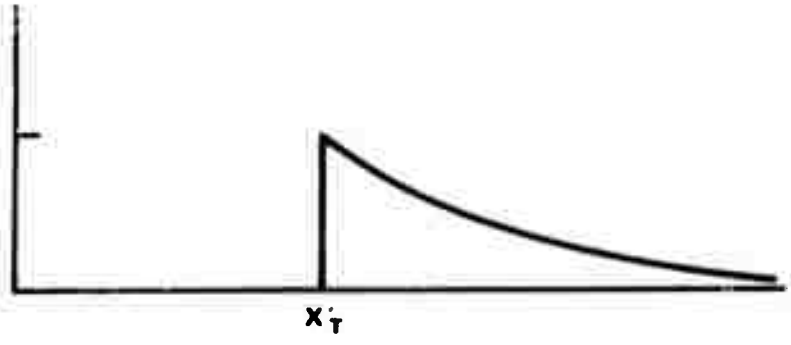
(a) THE EXPONENTIAL DENSITY FUNCTION



(b) TRUNCATED EXPONENTIAL WITH A DETERMINISTIC COMPONENT



(c) TRUNCATED AND RENORMALIZED DENSITY



(d) A RENORMALIZED EXPONENTIAL TAIL FROM THE TRUNCATION

Figure A.1. The Exponential Density and Several Variations of its Truncated Form.

b. The Truncated and Renormalized Density

The truncation effect in the priority analysis did not involve an impulse function at the truncation value, and therefore resulted in the two conditional densities shown in Figure A.1 (c) and (d). To differentiate between the two types of truncation, and to be consistent with the notation used in the priority analysis, we shall consider this truncation value to be $X_0 + X_T$. The moments of the first of these two truncated functions are:

$$E[x \mid x \leq X_0 + X_T] = X_0 + \frac{1}{\mu}\left[1 - \frac{\mu X_T e^{-\mu X_T}}{1 - e^{-\mu X_T}}\right] \tag{A.3}$$

and:

$$E[x^2 \mid x \leq X_0 + X_T] = X_0^2 + \frac{2}{\mu^2}\left[1 - \frac{\left(\mu X_T + \frac{(\mu X_T)^2}{2}\right)e^{-\mu X_T}}{1 - e^{-\mu X_T}}\right]$$

$$+ \frac{2X_0}{\mu}\left[1 - \frac{\mu X_T e^{-\mu X_T}}{1 - e^{-\mu X_T}}\right] \tag{A.4}$$

Although Adiri did not consider the second function directly, it can be considered to be a special case of part (a) with X_0 becoming equal to X_T' and X_s becoming arbitrarily large. The moments are then:

$$E[x \mid x \geq X_T'] = X_T' + \frac{1}{\mu} \tag{A.5}$$

and:

$$E[x^2 \mid x \geq X_T'] = (X_T')^2 + \frac{2X_T'}{\mu} + \frac{2}{\mu^2}$$

or:

$$E[x^2 \mid x \geq X_T'] = \frac{2}{\mu^2}\left[1 + \mu X_T' + \frac{(\mu X_T')^2}{2}\right] \tag{A.6}$$

APPENDIX B

THE TRUNCATED 1/x DENSITY FUNCTION

An interesting density function for much of our analysis is the truncated density,

$$f(x) = 1/x \ln M \qquad 1 \leq x \leq M \tag{B.1}$$

The function has a very sharp peak at its initial value, $x = 1$, and a relatively flat slope near its maximum of $x = M$.

The moments of the function can be found by:

$$\overline{x^n} = \int_1^M \frac{x^n}{x \ln M} dx$$

which can be integrated for the general case, resulting in:

$$\overline{x^n} = \left. \frac{x^n}{n \ln M} \right|_1^M = \frac{M^n - 1}{n \ln M} \tag{B.2}$$

Using the first and second moments, we can find an expression for the variance,

$$\sigma^2 = \frac{(M^2 - 1)\ln M - 2(M - 1)^2}{2(\ln M)^2} \tag{B.3}$$

and for the coefficient of variation,

$$c = \left[\frac{(M + 1)\ln M}{2(M - 1)} - 1 \right]^{1/2} \tag{B.4}$$

The truncation of the function at some finite upper bound is necessary to meet the unit area requirement. Also, since the cumulative distribution function is:

$$F(x) = \ln x / \ln M \tag{B.5}$$

the lower bound can not be less than unity (to avoid negative values). For our purposes, this lower bound shall always be considered to be unity.

The $1/x$ density has several valuable features which make it of interest, including the fact that it transforms into a uniform density for the transformation of variable,

$$\xi = \log_b x$$

which is the continuous equivalent of the log histogram reshaping effect. The availability of readily calculated higher moments is also a convenience for use as a test case in the moment transformation work of Section 3.3.2.1.

APPENDIX C

TEST CASES FOR THE MOMENT TRANSFORMATIONS

Pairs of $f(x)$ and $\phi(\xi)$ functions are needed for evaluation of the moment transformations derived in Section 3.3.2.1. Since these equations require the calculation of many higher moments of $\phi(\xi)$, we will investigate functions with readily obtainable moments, and in particular, will consider uniform and exponential forms of the function, $\phi(\xi)$.

a. Uniform $\phi(\xi)$ Density

The logarithmic transform function pair for which $\phi(\xi)$ is uniform, has an $f(x)$ function of the form:

$$f(x) = 1/x \ln M \qquad 1 < x < M \tag{C.1}$$

and from the transformation of variables,

$$\phi(\xi) = \ln 2/\ln M \qquad 0 < \xi < \log_2 M \tag{C.2}$$

The moments of the function, $\phi(\xi)$ can be shown to be:

$$\overline{\xi^n} = \left[\frac{\ln M}{\ln 2}\right]^n \Big/ (n + 1) \tag{C.3}$$

with the central moments being:

$$\mu_n = \begin{cases} 0 & \text{for } n = \text{odd} \\ \left[\dfrac{\ln M}{2 \ln 2}\right]^n \Big/ (n + 1) & \text{for } n = \text{even} \end{cases} \tag{C.4}$$

From Equation (3.44), we have the relationship:

$$\bar{x} = 1 + \bar{\xi} \ln 2 + \frac{\overline{\xi^2}}{2!} (\ln 2)^2 + \ldots + \frac{\overline{\xi^n}}{n!} (\ln 2)^n + \ldots \qquad (C.5)$$

which upon substitution for the moments of $\phi(\xi)$ becomes:

$$\bar{x} = 1 + \frac{\ln M}{2} + \frac{(\ln M)^2}{3(2!)} + \ldots + \frac{(\ln M)^n}{(n+1)(n!)} + \ldots \qquad (C.6)$$

A more convenient form can be obtained by adding and subtracting unity, resulting in:

$$\bar{x} = \frac{1}{\ln M} \left[(1 + \ln M + \frac{(\ln M)^2}{2!} + \frac{(\ln M)^3}{3!} + \ldots) - 1 \right] \qquad (C.7)$$

The expression within the parentheses can be recognized as the infinite series equivalent of the exponential, which further simplifies to:

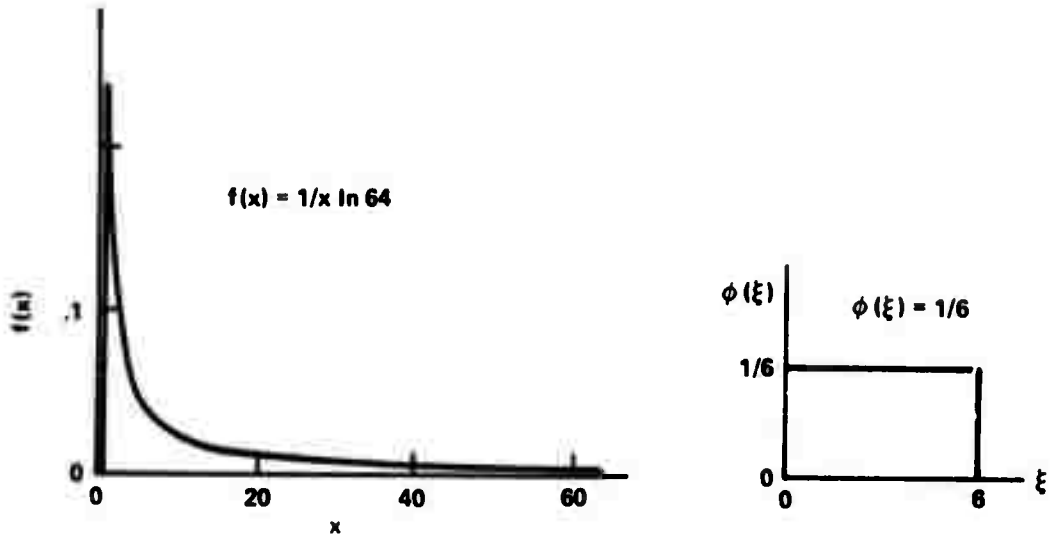
$$e^{\ln M} = M$$

The resulting expression for $\bar{x}$ is:

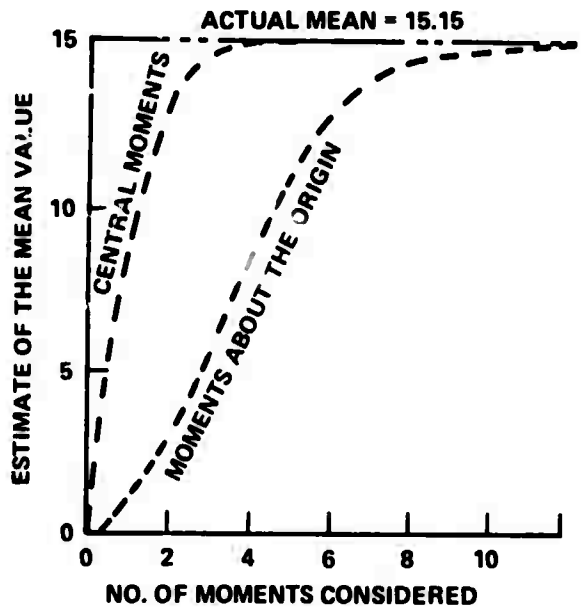
$$\bar{x} = (M - 1)/\ln M \qquad (C.8)$$

which from Equation (B.2) is known to be the mean value of the $1/x$ density. This result serves as a check on the correctness of the expansion, but does not answer the real question of concern, namely, how rapidly does the series converge?

We could pursue this problem for the general value of M, but since such a result would still be a special case as far as general p.d.f.'s are concerned, we shall not bother with such a pseudo-generality. Instead, we shall consider the special case shown in Figure C.1 for which the mean value of x is known to be 15.15. When we



(a) THE f(x) DENSITY AND ITS LOGARITHMIC TRANSFORMED EQUIVALENT DENSITY.



(b) THE RELATIVE CONVERGENCE OF THE ESTIMATES OF $\bar{x}$ FROM THE MOMENTS OF $\phi(\xi)$.

Figure C.1. The Density Functions and Convergence of the Estimate of the Mean for the Example Considered.

calculate the moments of $\phi(\xi)$ and substitute these moments into Equations (3.44) and (3.50), we find that the former converges very slowly, requiring the first 6 to 10 moments to obtain a reasonable estimate of the mean, while the latter converges much more rapidly. In fact, the expansion in terms of the central moments converges to within 0.3% of the actual value when only three moments are considered, compared to a 1.0% error in the estimate when the first ten moments about the origin are considered. However, these figures should not be considered to hold with any generality, or as necessarily being typical of the convergence of each series.

b. Exponential $\phi(\xi)$ Density

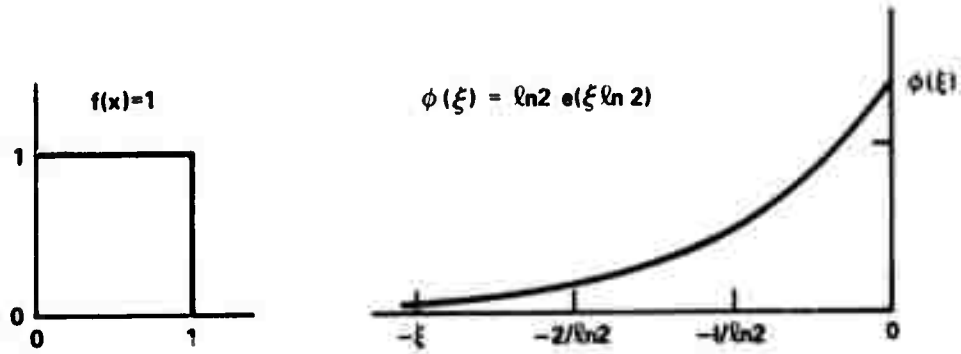
A second transform pair involves a uniform $\phi(\xi)$ density with its logarithmic transform function being an exponential in the left half plane. The $f(x)$ function is:

$$f(x) = \begin{cases} 1 & 0 < x < 1 \\ 0 & \text{otherwise} \end{cases} \tag{C.9}$$

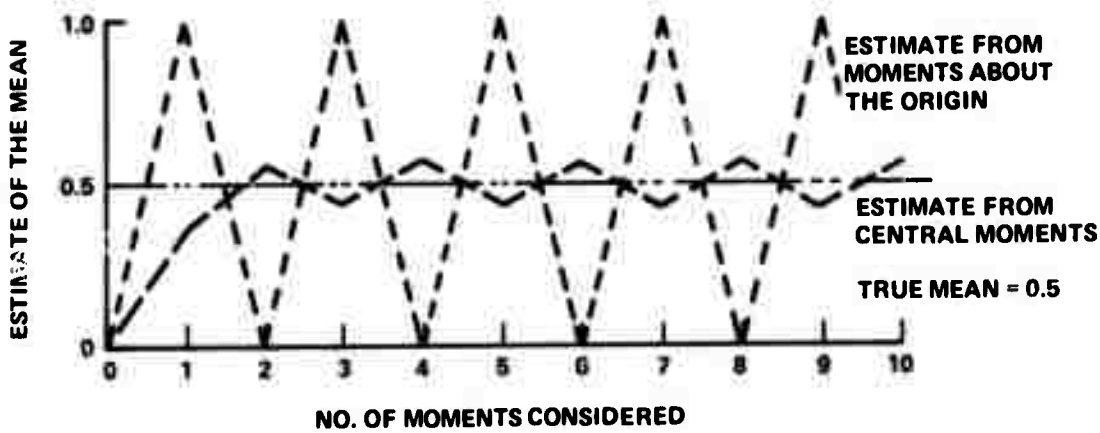
and the corresponding $\phi(\xi)$ function can be shown to be:

$$\phi(\xi) = \ln 2 \, e^{\xi \ln 2} \qquad -\infty < \xi < 0 \tag{C.10}$$

This example is of particular interest because it does not result in convergence of either Equation (3.44) in terms of moments about the origin, or Equation (3.50) in terms of the central moments. However, the latter expansion does oscillate in a reasonably restricted range about the mean as shown in Figure C.2. The expansion in terms of moments about the origin oscillates between zero and unity because of the



(a) The f (x) DENSITY AND ITS LOGARITHMIC TRANSFORMED EQUIVALENT DENSITY



(b) THE RELATIVE CONVERGENCE OF THE ESTIMATES OF $\bar{x}$ FROM THE MOMENTS OF $\phi(\xi)$

Figure C.2. An Example in Which the Moment Transforms Do Not Converge.

alternating series which results for these values of the moments of $\phi(\xi)$. The functions used in the example are not considered to be representative of actual densities which would be encountered in practice, but are a convenient function-pair to demonstrate the convergence problem.

APPENDIX D

MOMENT CORRECTIONS FOR GROUPED DATA

The calculation of moments from grouped data such as histograms can result in systematic bias errors depending on the shape of the density function and the order of the moment being estimated. Sheppard (SH98) developed correction formulas for gaussian-like densities using the Euler-MacLaurin sum formula as a discrete approximation to the definite integral representation of the moments. The necessary and sufficient conditions for the applicability of the correction formulas are met by functions which are continuous and finite over the range of interest, and which taper to zero at each extreme.* For such functions, the first moment is unbiased, and the variance has a correction of:

$$(\sigma^2)_c = (\sigma^2)_{uc} - h^2/12 \qquad \text{(D.1)}$$

where the subscripts c and uc denote the corrected and uncorrected estimates respectively, and h is the histogram interval size.

Most of the density functions that we have considered do not meet the above conditions, since the exponential and other "J-shaped" densities do not taper to zero smoothly at both extremes. Elderton (EL33) has made adjustments for these shape differences using an empirically derived rule based on observations of the exponential density,

---

*A more formal description of Sheppard's corrections can be found in Cramér (CR61).

but in a companion paper, Pearson showed that these corrections degrade considerably for densities with shapes that radically depart from that of the exponential. (One such deviation is the so-called "twisted J-shaped" curve which has a large, but finite, slope at its origin.) Pearson also compared Elderton's corrections with other adjustment formulas by Pearse and by Martin which were more accurate, but required considerably more computational effort. The Pearse method also included an "abruptness" coefficient for truncated density functions. Abernathy (AB33) took a different approach to densities which do not meet the restriction of Sheppard's formulas, in that he removed the need for the slowly tapering tails of the density at each extreme, and only required the existence of the moments of the density function.

These moment corrections as developed by Sheppard and as modified by Elderton, Abernathy, and others, have all been based on uniform interval histograms. We will take a different approach which is not limited to uniform intervals, and which takes advantage of the information available in the neighboring interval heights. For example, if we considered the ith interval by itself, we could only assume a uniform distribution across the interval. However, if we know that the interval to the left is greater, and the interval to the right is smaller than the ith interval, we have additional information regarding the most likely distribution of the probability mass across the ith interval. This usage of neighboring interval heights to determine a linear slope function will be referred to as a first-order approximation to the density shape, as opposed to the zero-order approximation of the uniform density assumption. This terminology and approach is

analogous to that used in sampled data reconstruction, and in keeping with the analogy, higher order approximations are of rapidly increasing complexity, and are felt to have marginal value considering this added complexity.

In order to develop techniques which are applicable to both uniform and log-histograms, we will first consider a more general approach to histograms as shown in Figure D.1. The intervals are

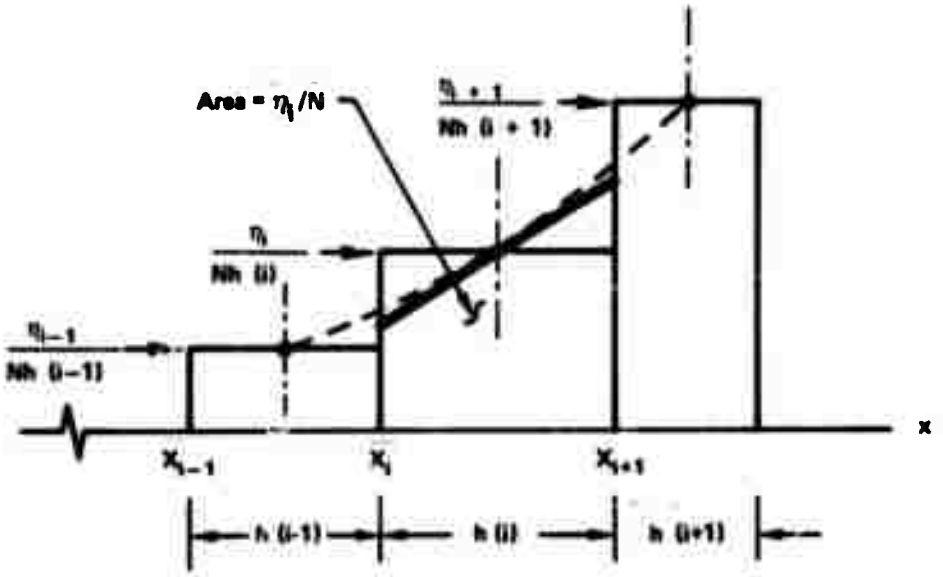


Figure D.1. A First-Order Approximation to a Generalized Histogram Format.

determined by a functional relationship such that $h(i)$ is the width of the ith interval. If η_i out of a total of N observations occur in the ith interval, then the probability associated with the interval is η_i/N, and for the zero-order approximation, the elemental density function is:

$$f_i(x) = \eta_i/Nh(i) \qquad x_i < x < X_{i+1} \tag{D.2}$$

The component of the nth moment due to this interval for any approximation, $f_i(x)$, will be denoted by:

$$\Delta\hat{E}[x^n]_i = \int_{X_i}^{X_{i+1}} x^n f_i(x)\, dx \tag{D.3}$$

and since the moment contributions are additive, the moment estimates will be:

$$\hat{E}[x^n] = \sum_{i=0}^{N-1} \Delta\hat{E}[x^n]_i \tag{D.4}$$

The approximate slope for the ith interval, m_i, will be the weighted average of the slopes to the midpoints of the two neighboring intervals,

$$m_i = w_i \left\{ \frac{\dfrac{\eta_i}{Nh(i)} - \dfrac{\eta_{i-1}}{Nh(i-1)}}{\frac{1}{2}[h(i-1) + h(i)]} \right\} + (1 - w_i) \left\{ \frac{\dfrac{\eta_{i+1}}{Nh(i+1)} - \dfrac{\eta_i}{Nh(i)}}{\frac{1}{2}[h(i) - h(i+1)]} \right\} \tag{D.5}$$

If we select the weighting according to the length of each slope segment, i.e.,

$$w_i = \frac{h(i-1) + h(i)}{h(i-1) + 2h(i) + h(i+1)} \tag{D.6}$$

and:

$$1 - w_i = \frac{h(i) + h(i+1)}{h(i-1) + 2h(i) + h(i+1)}$$

then the slope becomes:

$$m_i = \frac{2\left[\frac{\eta_{i+1}}{Nh(i+1)} - \frac{\eta_{i-1}}{Nh(i-1)}\right]}{h(i-1) + 2h(i) + h(i+1)} \tag{D.7}$$

The first order approximation function for the ith interval is:

$$f_i(x) = m_i x + B_i \qquad X_i < x < x_{i+1} \tag{D.8}$$

for which:

$$f_i\left(X_i + \frac{h(i)}{2}\right) = \eta_i / Nh(i) \tag{D.9}$$

such that:

$$B_i = \frac{\eta_i}{Nh(i)} - m_i\left(X_i + \frac{h(i)}{2}\right) \tag{D.10}$$

The function can then be written as:

$$f_i(x) = m_i x + \frac{\eta_i}{Nh(i)} - m_i\left(X_i + \frac{h(i)}{2}\right) \tag{D.11}$$

The component of the nth moment is from Equation (D.3),

$$\Delta\hat{E}[x^n]_i = \int_{X_i}^{X_{i+1}} x^n [m_i x + B_i]\, dx \tag{D.12}$$

or:

$$\Delta\hat{E}[x^n]_i = m_i \frac{x^{n+2}}{n+2} + B_i \frac{x^{n+1}}{n+1} \Bigg|_{X_i}^{X_{i+1}}$$

which becomes:

$$\Delta\hat{E}[x^n]_i = \frac{m_i}{n+2}\left[(X_{i+1})^{n+2} - (X_i)^{n+2}\right] + \frac{B_i}{n+1}\left[(X_{i+1})^{n+1} - (X_i)^{n+1}\right] \quad (D.13)$$

where m_i, B_i, and X_i are as defined in Equations (D.7), (D.10), and Figure D.1 respectively. This is as far as we can carry the development without introducing a functional relationship between the $h(i)$'s and the X_i's so we will next consider the two special cases of interest, namely the uniform and log-histograms.

a. <u>The Uniform Interval Histogram</u>

For an M-interval uniform histogram, all of the intervals are of length h such that:

$$h(i) = h \qquad i = 0,1,2,\ldots,M-1 \quad (D.14)$$

and:

$$X_i = ih \quad (D.15)$$

with the range of x equal to:

$$0 \leq x < Mh$$

If we evaluate Equation (D.12) for the expected value of x, in the ith interval, we obtain:

$$\Delta\hat{E}[x]_i = \frac{m_i}{3}\left\{[(i+1)h]^3 - [ih]^3\right\} + \frac{B_i}{2}\left\{[(i+1)h]^2 - [ih]^2\right\}$$

or:

$$\Delta\hat{E}[x]_i = \frac{m_i h^3}{3}\left\{3i^2 + 3i + 1\right\} + \frac{B_i h^2}{2}\left\{2i + 1\right\} \quad (D.16)$$

Since we know that B_i is equal to:

$$B_i = \frac{\eta_i}{Nh} - m_i[ih + \frac{h}{2}]$$

then:

$$\Delta\hat{E}[x]_i = \frac{m_i h^3}{3}(3i^2 + 3i + 1) + \left|\frac{\eta_i}{Nh} - m_i(ih + \frac{h}{2})\right|\left(\frac{h^2}{2}\right)\left(2i + 1\right)$$

and by collecting like terms:

$$E[x]_i = \frac{m_i h^3}{12} + \frac{h^2}{4}(2i + 1)\left[\eta_i/Nh\right] \tag{D.17}$$

which upon substituting for m_i,

$$\Delta\hat{E}[x]_i = (ih + \frac{h}{2})\frac{\eta_i}{N} + \frac{h}{24N}[\eta_{i+1} - \eta_{i-1}] \tag{D.18}$$

becomes the desired result. The first term is the conventional component of the mean, and the second term is the first-order correction component. Note that if η_{i+1} is greater than η_{i-1}, i.e., for a function with a positive slope, the correction is positive since more of the mass is beyond the midpoint, and conversely, if the slope is negative, the mass is nearer the origin so the correction component is negative. The total correction will be the sum of these components, but we must also account for the end conditions where the slope functions are undefined.* A reasonable approach was found to be to assume that, for slope considerations, we should let $\eta_{-1} = \eta_0$ and $\eta_M = \eta_{M-1}$, such that the correction formula,

---

*The end conditions correspond to the "abruptness" coefficient mentioned earlier.

$$\text{correction} = \frac{h}{24N} \sum_{i=0}^{M-1} (\eta_{i+1} - \eta_{i-1}) \tag{D.19}$$

becomes:

$$\text{correction} = \frac{h}{24N} [\eta_{M-1} - \eta_0] \tag{D.20}$$

This result is consistent with Sheppard's correction for functions which taper to zero at the two extremes since for such functions the correction would be zero.

The second moment correction is more complicated because of the terms being raised to the higher powers, but can otherwise be evaluated in a similar manner. However, there are two choices for the conventional representation; (1) assuming that all of the probability mass is located at the center of the interval, at $ih + h/2$, and (2) assuming that the mass is spread uniformly over the interval, in which case the effective mass location would be $ih + h/\sqrt{3}$. After considerable algebra, one can show that the component contributions for the first case are:

$$\Delta\hat{E}[x^2]_i = \left[\left(ih + \frac{h}{2}\right)^2 \frac{\eta_i}{N}\right] + \frac{h}{12N} [\eta_i h + (ih + \frac{h}{2})(\eta_{i+1} - \eta_{i-1})]$$

for which the total correction becomes,

$$\text{correction} = \frac{-h^2}{12N} (\eta_0 + \eta_1 + \eta_2 + \ldots + \eta_{M-1})$$

or simply,

$$\text{correction} = \frac{-h^2}{12} \tag{D.21}$$

which is equivalent to Sheppard's correction, since the estimate of variance is:

$$\hat{\sigma}^2 = \hat{E}[x^2] - \{\hat{E}[x]\}^2 \tag{D.22}$$

and the estimate of the mean was unbiased for the set of applicable density functions. The combination of the two correction formulas of Equations (D.20) and (D.21) are applicable to any density functions for which the moment integral of Equation (D.3) is defined.

For the second case, i.e., where the effective probability mass location is $ih + h/\sqrt{3}$, the component contributions are:

$$\Delta\hat{E}[x^2]_i = \left[(ih + h/\sqrt{3})^2 \frac{\eta_i}{N}\right] + (ih + h/\sqrt{3}) \frac{h\sqrt{3}}{48N}\left[\eta_{i+1} - \eta_{i-1}\right]$$

for which the total correction is:

$$\text{correction} = \frac{h^2\sqrt{3}}{48N} \sum_{i=0}^{M-1} \left(i + \frac{1}{\sqrt{3}}\right)\left(\eta_{i+1} - \eta_{i-1}\right)$$

and results in a first-order correction of:

$$\text{correction} = \frac{-h^2\sqrt{3}}{24} \tag{D.23}$$

This correction is of the same sign as in case (1), but is of smaller magnitude since the initial estimate is based on a better approximation of the density function. Intuitively, one prefers an estimate requiring a minimal correction, so the latter method was used in the comparative Monte Carlo tests of Section 3.3.2.4.

b. The Logarithmic Interval Histogram

Data taken in log-histogram form can be considered in either of two domains, as discussed in Section 3.2.2. In the ξ domain, the histogram has uniform intervals of unit length such that:

$$\pi_i = Pr[i \leq \xi < i+1] \tag{D.24}$$

The equivalent probability density in the x domain is:

$$P_i = \pi_i/h(i) \qquad (b)^i \leq x < (b)^{i+1} \tag{D.25}$$

where the interval boundaries are:

$$X_i = (b)^i \qquad i = 0,1,2,\ldots,M-1 \tag{D.26}$$

and the interval length functions are:

$$h(i) = (b-1)(b)^i \tag{D.27}$$

for all $b > 1$, where b is the base of the logarithmic transformation,

$$\xi = \log_b x$$

For these values of X_i, Equation (D.12) becomes:

$$\Delta\hat{E}[x]_i = \frac{m_i}{3}\left[(b^{i+1})^3 - (b^i)^3\right] + \frac{B_i}{2}\left[(b^{i+1})^2 - (b^i)^2\right]$$

or:

$$\Delta\hat{E}[x]_i = \frac{m_i}{3}(b^3 - 1)(b)^{3i} + \frac{B_i}{2}(b^2 - 1)(b)^{2i}$$

The values of m_i and B_i can be shown to be:

$$m_i = \frac{2b[P_{i+1} - P_{i-1}]}{(b-1)(b+1)^2 (b)^i} \tag{D.28a}$$

and:

$$B_i = P_i - m_i \frac{(b+1)(b)^i}{2} \tag{D.28b}$$

from Equations (D.7) and (D.10) respectively, such that:

$$\Delta\hat{E}[x]_i = P_i \frac{(b^2-1)(b)^{2i}}{2} + m_i (b)^{3i}\left[\frac{b^3-1}{3} - \frac{(b+1)(b^2-1)}{4}\right]$$

or:

$$\Delta\hat{E}[x]_i = \frac{P_i (b^2-1)(b)^{2i}}{2} + \frac{m_i (b)^{3i}}{12}(b-1)^3$$

and substituting for m_i, we obtain:

$$\Delta\hat{E}[x]_i = \frac{P_i (b^2-1)(b)^{2i}}{2} - \frac{b(b-1)^2 (b)^{2i}}{6(b+1)^2}[P_{i+1} - P_{i-1}] \tag{D.29}$$

Equation (D.29) is of the desired form in that it consists of a conventional zero-order estimate term,

$$\frac{P_i (b^2-1)(b)^{2i}}{2} = \int_{(b)^i}^{(b)^{i+1}} P_i \, x \, dx \tag{D.30a}$$

and a first-order correction term,

$$i^{th} \text{ correction} = \frac{b(b-1)^2 (b)^{2i}}{6(b+1)^2}[P_{i+1} - P_{i-1}] \tag{D.30b}$$

A more convenient form of the correction equation involves the π_i values since those represent the actual measured data. The correction

equation then becomes:

$$i^{th} \text{ correction} = \frac{b(b-1)^2(b)^{2i}}{6(b+1)^2} \left| \frac{\pi_{i+1}}{(b-1)(b)^{i+1}} - \frac{\pi_{i-1}}{(b-1)(b)^{i-1}} \right|$$

or:

$$i^{th} \text{ correction} = \frac{(b-1)(b)^i}{6(b+1)^2} [\pi_{i+1}' - b^2\pi_{i-1}] \tag{D.31}$$

The end conditions for the log-histogram do not simplify as readily as they did for the uniform interval case, but are particularly important due to the potential moment contribution of such large segments at a large distance from the origin. (Note that the end condition at the origin is not significant due to its relatively small effect on the moments.)

At the upper end of the histogram, we will assume that the nonexistent M^{th} interval mirrors the value of the last actual interval, i.e., that for slope considerations;

$$P_M = P_{M-1} \tag{D.32}$$

Since the Mth interval is b times as long as the $M\text{-}1^{st}$ interval, the π_M value must then be:

$$\pi_M = b\pi_{M-1} \tag{D.33}$$

such that the slope correction to the $M\text{-}1^{st}$ interval becomes:

$$\frac{(b-1)(b)^{M-1}}{6(b+1)^2} [b\pi_{M-1} - b^2\pi_{M-2}] \tag{D.34}$$

The resulting total correction is therefore,

$$\frac{b-1}{6(b+1)^2}\left\{\sum_{i=1}^{M-2}[\pi_{i+1}-b^2\pi_{i-1}](b)^i+[b\pi_{M-1}-b^2\pi_{M-2}](b)^{M-1}\right\} \tag{D.35}$$

with the uncorrected estimate being:

$$\hat{E}[x]_{uc}=\frac{b^2-1}{2}\sum_{i=0}^{M-1}P_i(b)^{2i}$$

or:

$$\hat{E}[x]_{uc}=\frac{b+1}{2}\sum_{i=0}^{M-1}\pi_i(b)^i \tag{D.36}$$

For the practical case of concern to us, the base of the logarithmic transform is equal to two, with the uncorrected estimate being:

$$\hat{E}[x]_{uc}=\frac{3}{2}\sum_{i=0}^{M-1}\pi_i(2)^i \tag{D.37}$$

and the corresponding correction formula being:

$$\frac{1}{54}\left\{\sum_{i=1}^{M-2}[\pi_{i+1}-4\pi_{i-1}](2)^i+[2\pi_{M-1}-4\pi_{M-2}](2)^{M-1}\right\} \tag{D.38}$$

The second moment estimate and correction can be found by evaluating Equation (D.12) for $n = 2$, with the ith component being:

$$\Delta\hat{E}[x^2]_i=\frac{m_i}{4}\left[(b^{i+1})^4-(b^i)^4\right]+\frac{B_i}{3}\left[(b^{i+1})^3-(b^i)^3\right]$$

which simplifies to:

$$\Delta\hat{E}[x^2]_i = \frac{m_i(b^4 - 1)}{4}(b)^{4i} + \frac{B_i(b^3 - 1)}{3}(b)^{3i} \tag{D.39}$$

Substituting the value of B_i from Equation (D.28b), we obtain:

$$\Delta\hat{E}[x^2]_i = P_i\left[\frac{(b^3 - 1)(b)^{3i}}{3}\right] + \frac{m_i(b)^{4i}}{12}\left[(b - 1)^2(b^2 - 1)\right]$$

which upon substitution for m_i from Equation (D.28a) becomes:

$$E[x^2]_i = P_i\left[\frac{(b^3 - 1)(b)^{3i}}{3}\right] + \frac{b(b - 1)(b^2 - 1)}{6(b + 1)^2}(b)^{3i}[P_{i+1} - P_{i-1}] \tag{D.40}$$

Equation (D.40) consists of a zero-order estimate component,

$$P_i\left[\frac{(b^3 - 1)(b)^{3i}}{3}\right] = \int_{(b)^i}^{(b)^{i+1}} x^2 P_i \, dx \tag{D.41}$$

and a first-order correction component:

$$i^{th} \text{ correction} = \frac{b(b - 1)^2(b)^{3i}}{6(b + 1)}[P_{i+1} - P_{i-1}]$$

If we substitute for the π_i values as in Equation (D.31) we obtain:

$$i^{th} \text{ correction} = \frac{(b - 1)(b)^{2i}}{6(b + 1)}[\pi_{i+1} - b^2\pi_{i-1}] \tag{D.42a}$$

except for the $N\text{-}1^{st}$ term which is:

$$\frac{(b - 1)(b)^{2N-2}}{6(b + 1)}[b\pi_{N-1} - b^2\pi_{N-2}] \tag{D.42b}$$

The zero-order estimate can be also obtained in terms of π_i by substitution in Equation (D.41) which leads to an uncorrected estimate of:

$$\hat{E}[x^2]_{uc} = \frac{b^2 + b + 1}{3} \sum_{i=0}^{M-1} \pi_i (b)^{2i} \tag{D.43}$$

For the limiting case of b approaching unity, the histogram becomes arbitrarily close to the continuous case, and the zero-order second moment is seen to be unbiased. The correction term goes to zero in this limiting case, as can be seen from Equations (D.42a, b). For the practical case of concern, $b = 2$, and the equations become:

$$\hat{E}[x^2]_{uc} = \frac{7}{3} \sum_{i=0}^{M-1} \pi_i (2)^{2i} \tag{D.44}$$

and:

$$\text{correction} = \frac{1}{18} \left\{ \sum_{i=1}^{M-2} [\pi_{i+1} - 4\pi_{i-1}] (4)^{2i} \right.$$

$$\left. + [2\pi_{M-1} - 4\pi_{M-2}] (4)^{M-1} \right\} \tag{3.45}$$

The correction formulas for the first and second moments of both uniform and log-histograms were utilized in the Monte Carlo tests as described in Section 3.3.2.4, and on the average, produced significant improvements in the moment estimates. In some cases the resulting estimates were overcorrected, i.e., produced equally large errors but of the opposite sign, and in isolated cases, produced even larger errors than the "uncorrected" estimates. However, more typical correction results were to reduce estimate errors as shown in Table 3.3.2.

The effect of the first-order correction for the exponential density is shown in Figure D.2, and indicates a reasonable slope curve fit over each segment of the log-histogram. However, the last interval

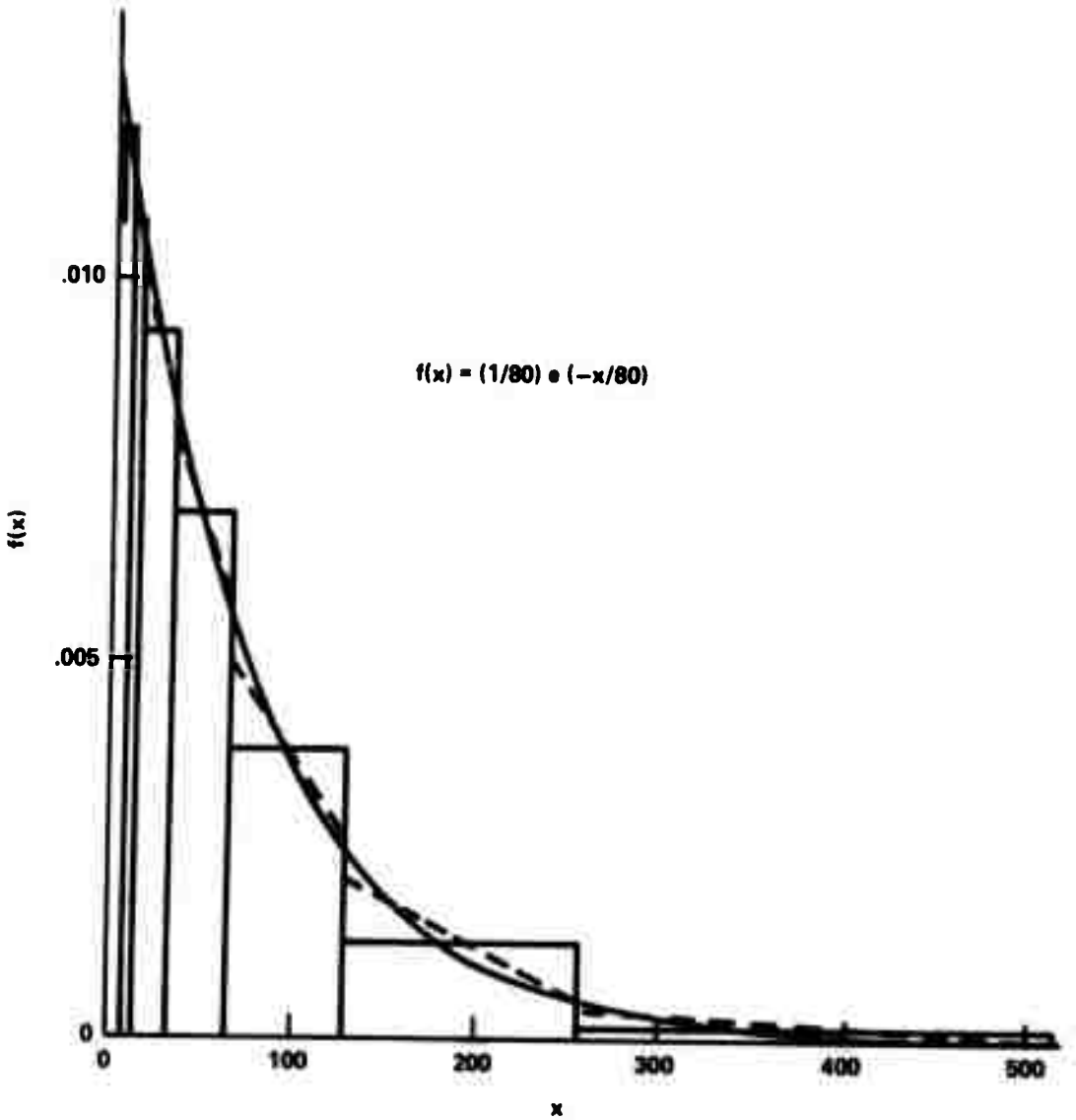


Figure D.2. An Exponential Density Shown with the Theoretical Log-Histogram Values and the First-Order Slope Approximation.

shows an effect which can result in significant errors, and is shown on a magnified scale in Figure D.3. The first-order slope, in this case, causes the approximating function to become negative, and introduces extra positive mass at the lower portion of the interval. Both effects tend to bias the moment estimates to be low, with an increasingly important effect on higher moments. Such a condition can be determined from the probability heights, i.e., it will occur if:

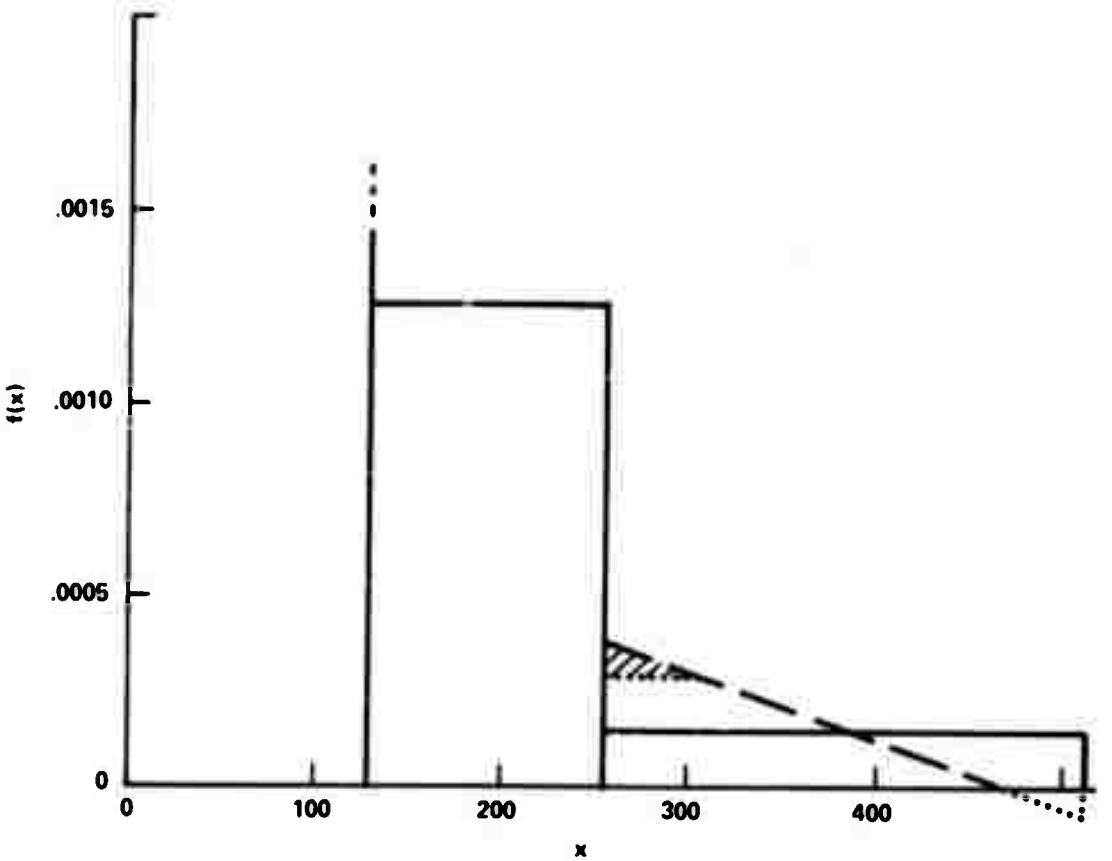


Figure D.3. An Enlarged Portion of the First-Order Approximation of Figure D.2.

$$m_i \frac{(b)^i}{2} > P_i \tag{D.46}$$

or substituting for m_i and reducing, if:

$$b[P_{i+1} - P_{i-1}] > P_i[(b - 1)(b + 1)^2] \tag{D.47}$$

In terms of the π_i probabilities, the test becomes:

$$\frac{[\pi_{i+1} - b^2\pi_{i-1}]}{(b - 1)} > (b + 1)^2 \pi_i \tag{D.48}$$

The appropriate action to take when the test fails is not clear, and probably should be handled on an individual basis.

APPENDIX E

MOMENT ESTIMATION CAPABILITIES OF UNIFORM AND LOG-HISTOGRAMS

Monte Carlo tests were run to evaluate the moment estimating capabilities of relatively coarse histograms with both uniform and logarithmic intervals. These tests were devised to check for any particular sensitivities to sample size, location of the mean, or density shape.

a. Overview of the Monte Carlo Tests

Uniform random variables over the range of 0 to 1 were generated by a conventional psuedo-random number generator, and were converted into the density of interest by a transformation of variables of the form,

$$f(x)\ dx = g(y)\ dy \tag{E.1}$$

where $f(x)$ is the desired density function, and

$$g(y) = \begin{cases} 1 & 0 < y < 1 \\ 0 & \text{otherwise} \end{cases} \tag{E.2}$$

The random variables were sorted into both uniform and logarithmic histograms, and also were accumulated as a sum and a sum of squares to provide a known record of the true sample mean, mean-square, and variance. Estimates of these moments were then made from the histograms and were compared with the known sample statistics. The test included the first-order moment corrections for both the uniform and

log-histograms, as well as evaluating the superposition method of Section 3.3.2.2.

b. Selection of the Parameters for the Experiment

A variety of density shapes was desired, so the tests included the uniform, Erlang(2), exponential, hyperexponential, and truncated $1/x$ density functions. The uniform and truncated $1/x$ densities were optimal for density estimates because of the equal area segments for the uniform and log-histograms respectively, and therefore were included in the moment estimate tests as well. The Erlang(2), exponential, and hyperexponential were considered to be more typical of the densities that might be encountered and represent an interesting spectrum of possible density shapes.

The range of the random variables was set fairly arbitrarily at 0 to 500, and for base two logarithms, resulted in 10 increments for the log-histogram. Ten intervals of size 50 were then selected for the uniform histogram. The values of the moments for the uniform and $1/x$ densities were defined by the range, and so were not free variables. However, the Erlang(2), exponential, and hyperexponential densities had no such constraints, but their parameters were arbitrarily set to have approximately the same mean value as the $1/x$ density. The means are only approximately equal due to the truncation effects, i.e., no values over 500 are allowed. This selection of intervals and mean values should be a highly unfavorable test for the log-histogram since seven of its ten intervals fall well below the mean value.

The density functions being considered are shown in Table E.1 which lists their parameter values and the resulting mean, variance,

and coefficient of variation, where these latter values include the truncation effect.

TABLE E.1

THEORETICAL VALUES OF THE MOMENTS OF THE TRUNCATED DENSITY FUNCTIONS BEING CONSIDERED

Density	Range	Parameters	Mean	Variance	Coefficient of Variation
Uniform	0-500	-	250.0	20,830	0.58
Erlang (2)	0-500	80	80.0	3,170	0.70
Exponential	0-500	80	79.0	5,960	0.98
Hyperexponential	0-500	40 and 120	76.0	8,820	1.23
1/x density	1-500	-	80.3	13,670	1.46

c. Generation of the Desired Densities

The random numbers that were obtained from the random number generator were uniformly distributed over the range of zero to one. If we call this uniform density $g(y)$ where:

$$g(y) = \begin{cases} 1 & 0 < y < 1 \\ 0 & \text{otherwise} \end{cases}$$

and desire a transformed density, $f(x)$, then we can relate the two densities by:

$$f(x)\,dx = g(y)\,dy$$

which for $g(y) = 1$ becomes:

$$y = \int_0^x f(x)\,dx = F(x) \qquad \text{(E.3)}$$

We want the inverse function, i.e., the relationship of x to some function of y, and can most easily obtain this by a slight change of variable,

$$y' = 1 - y = 1 - F(x) \tag{E.4}$$

where y' will also be uniformly distributed over the unit interval. For example, if $f(x)$ is the exponential density,

$$f(x) = \mu\, e^{-\mu x} \qquad 0 \leq x \tag{E.5}$$

then,

$$F(x) = 1 - e^{-\mu x}$$

and

$$y' = 1 - F(x) = e^{-\mu x}$$

so that:

$$x = \frac{-1}{\mu} \ln(y') \tag{E.6}$$

A similar transformation can be found for the truncated $1/x$ density,

$$f(x) = 1/x \ln M \tag{E.7}$$

with:

$$y' = 1 - F(x) = \ln x/\ln M$$

or:

$$x = \exp(y' \ln M) \tag{E.8}$$

The transformations for the Erlang and hyperexponential densities can readily be found from exponential components; the Erlang (2) being the sum of two exponential random variables, and the hyperexponential being the random selection between two equally likely exponential random variables with different means.

d. Moment Estimates Using the Superposition Method

The technique described in Section 3.3.2.2, in which the log-histogram components were transformed into $1/x$ density segments, was evaluated by a series of Monte Carlo tests with uniform, Erlang (2), exponential, hyperexponential, and truncated $1/x$ density functions. The method produced excellent results for the $1/x$ density, as would be expected, and somewhat surprisingly, also gave good results for the uniform density. However, for the Erlang (2), exponential, and hyperexponential cases, the estimates of the mean were in error by about 5% to 6%, and the variance estimates were about 30% too large.

The reason for both the good estimates of the uniform density case, and the poor estimates of the "long tail" densities, is the fact that the $1/x$ density is reasonably flat at the larger values of x. This upper region provides a predominant contribution to the moment calculation, and must be approximated quite closely if good estimates are to be obtained. Since the $1/x$ density approximation has a significant weakness in this area, it was not pursued any further.

e. Moment Estimates Using the First-Order Correction Equations

The five density functions shown in Table E.1 were utilized as test cases to evaluate and compare the moment estimation capabilities of the first-order correction equations for both uniform and log-histograms. The resulting estimates of the mean values are shown in Table E.2 with the variance estimates being shown in Table E.3. The surprising result is that the estimates from the log-histograms were often actually better than those from the uniform histogram, even though the

TABLE E.2

A COMPARISON OF ESTIMATES OF THE MEAN FROM UNIFORM AND LOG-HISTOGRAMS

Density Function	Lot No.	Estimates of the Mean				
		Sample Mean	Uniform Histogram		Log-Histogram	
			Estimate	% Error	Estimate	% Error
Uniform	1	242	242	+0.1%	244	+0.7%
	2	250	250	+0.1%	247	-0.9%
	3	243	244	+0.5%	250	+2.8%
	4	253	254	+0.6%	256	+1.2%
	Total	247	248	+0.3%	249	+1.0%
Erlang (2)	1	84.5	84.5	+0.0%	84.7	+0.2%
	2	90.1	89.6	-0.6	87.0	-3.5%
	3	75.0	72.3	-3.7%	74.6	-0.6%
	4	78.4	77.4	-1.2%	79.3	+1.1%
	Total	82.0	80.9	-1.4%	81.4	-0.7%
Exponential	1	81.8	83.7	+2.3%	81.8	+0.0%
	2	82.4	85.1	+3.3%	81.9	-0.7%
	3	80.5	81.1	+0.7	81.2	+0.9%
	4	78.1	80.2	+2.7%	76.0	-2.8%
	Total	80.7	82.5	+2.3%	80.2	-0.6%
Hyper-exponential	1	84.9	86.9	+2.4%	84.1	-0.9%
	2	73.3	75.4	+2.8%	74.3	+1.3%
	3	81.7	82.6	+1.0%	82.8	+1.3%
	4	83.4	86.7	+3.9%	83.0	-0.6%
	Total	80.8	82.9	+2.5%	81.0	+0.3%
Truncated 1/x density	1	80.2	87.9	+9.6%	81.7	+1.9%
	2	91.2	98.8	+8.3%	93.8	+2.8%
	3	73.5	80.0	+8.9%	71.5	-2.8%
	4	78.6	85.8	+9.0%	82.0	+4.3%
	Total	80.9	88.1	+9.0%	82.2	+1.6%

TABLE E.3

A COMPARISON OF ESTIMATES OF THE VARIANCE FROM UNIFORM AND LOG-HISTOGRAMS

Density Function	Lot No.	Estimates of the Variance				
		Sample Variance	Uniform Histogram		Log-Histogram	
			Estimate	% Error	Estimate	% Error
Uniform	1	21,800	23.300	+ 6.6%	22,100	+ 1.2%
	2	23,700	25,100	+ 5.8%	22,800	- 3.8%
	3	19,600	20,900	+ 6.6%	21,000	+ 7.1%
	4	20,200	22,100	+ 9.1%	21,600	+ 6.8%
	Total	21,300	22,800	+ 7.0%	21,900	+ 2.8%
Erlang (2)	1	3,470	4,110	+18.6%	2,200	-36.5%
	2	4,050	4,810	+18.9%	2,390	-41.0%
	3	2,470	3,020	+22.2%	1,080	-56.3%
	4	3,130	3,810	+21.5%	2,940	- 6.1%
	Total	3,280	3,940	+20.3%	2,150	-35.0%
Exponential	1	5,650	6,110	+ 8.3%	5,300	- 6.1%
	2	6,950	7,450	+ 7.1%	6,820	- 1.9%
	3	5,610	6,200	+10.4%	6,010	+ 7.1%
	4	6,160	6,530	+ 6.1%	4,890	-20.6%
	Total	6,090	6,570	+ 8.0%	5,760	- 5.4%
Hyper-exponential	1	7,100	7,510	+ 5.7%	6,250	-12.0%
	2	7,090	7,270	+ 2.6%	7,260	+ 2.3%
	3	7,540	7,880	+ 4.5%	7,410	- 1.8%
	4	7,510	7,860	+ 4.6%	6,940	- 7.6%
	Total	7,310	7,630	+ 4.3%	6,960	- 4.8%
Truncated 1/x density	1	14,500	13,900	- 4.7%	15,000	+ 2.8%
	2	17,200	16,500	- 4.2%	17,600	+ 2.3%
	3	12,500	12,000	- 4.1%	12,000	- 3.8%
	4	12,300	12,000	- 3.2%	14,200	+15.2%
	Total	14,100	13,600	- 4.0%	14,700	+ 4.1%

latter has much better resolution at the higher data values, and consequently should yield better estimates.

Some insight into this apparent discrepency can be gained from Table E.4 which compares the corrected and uncorrected estimates

TABLE E.4

THE EFFECT OF THE FIRST-ORDER MOMENT CORRECTIONS FOR AN EXPONENTIAL DENSITY

	Mean	Mean Square	Variance
Uniform intervals			
• uncorrected	+3.5%	+ 7.9%	+ 9.2%
• corrected	+2.3%	+ 6.3%	+ 8.0%
Log intervals			
• uncorrected	+6.9%	+21.4%	+29.3%
• corrected	-0.6%	- 3.2%	- 5.4%

for the two types of histograms with an exponential density function. The uncorrected estimates of the first and second moments have relatively large errors for the log-histogram procedure, but the correction can be seen to be quite effective in decreasing the error; and in fact tends to somewhat overcorrect the values,producing an error of the opposite sign. In contrast, the uniform histogram corrections are not nearly large enough and Tables E.2 and E.3 show that this was typically the case. The log-histogram corrections were far more effective and produced overall estimates which appeared to be reasonably unbiased. The same basic correction technique was used for both types of histograms, as described in Appendix D, but the data indicates that some

other weighting of the adjacent slopes would probably give better results for the uniform histogram, since they should inherently give better moment estimates than the log-histograms. This subject was not pursued, since for our purposes the log-histograms were shown to be preferable for density estimates, and when corrected, produce good estimates of at least the first and second moments.

The five density functions were selected to have a variety of density shapes, and indeed some shape-dependent errors were seen in the test data. For example, the Erlang (2) density test was found to have very large errors in the variance estimates. This effect was due to two factors; (a) the second moment about the origin is somewhat over-corrected, and (b) the variance is the difference between two estimates,

$$\hat{\sigma}^2 = \hat{E}[x^2] - \{\hat{E}[x]\}^2 \tag{E.7}$$

and since the variance of the Erlang density is relatively small, this becomes a large percentage error. For example, in the first sample, the errors in the mean and meansquare values were 0.2% and -11.6% respectively, but the resulting error in the variance was -36.5%. Fortunately, the Erlang density shape is not expected in any of the message size data, so the log-histogram data reduction for the network measurements should not involve this kind of error.

f. A Comparison of the Estimates for Different Mean Values

In many measurement situations, one knows the range of allowable values but does not have much insight into the expected value of the variable. We will now compare the uniform and log-histogram for such a case, with the value constrained to be within the range of 0 to

500, but with various values of the mean. The resulting errors are plotted in Figure E.1 as a function of the sample mean (for an exponential density). The log-histogram estimates have errors in the estimate of the mean value which are reasonably independent of the actual mean value, while the uniform histogram estimates are a strong function of the location of the mean. While this example does not necessarily inciated a general relationship, it does seem to be reasonable to expect many distributions to have such a characteristic because of the nature of the errors involved. That is, the uniform histogram will be expected to have a reasonably constant absolute error in its estimates, which will be a decreasing percentage error as the nominal mean value is increased. In contrast, the log-histogram has an absolute error which is approximately proportional to the nominal value, and hence produces a fairly constant percentage error.

The second moments are seen to have a larger percentage error in both cases, although the log-histogram has somewhat less error than the uniform histogram. In contrast, the variance estimates of the uniform histogram are reasonably constant over the entire range, and are often less than those of the log-histogram. This is due to the fact that, for the exponential density, the error in the variance is proportional to the difference between the percentage errors of the first and second moments,

$$\frac{er(\sigma^2)}{\sigma^2} \cong 2\left[\frac{er(\overline{x^2})}{\overline{x^2}} - \frac{er(\bar{x})}{\bar{x}}\right] \tag{E.8}$$

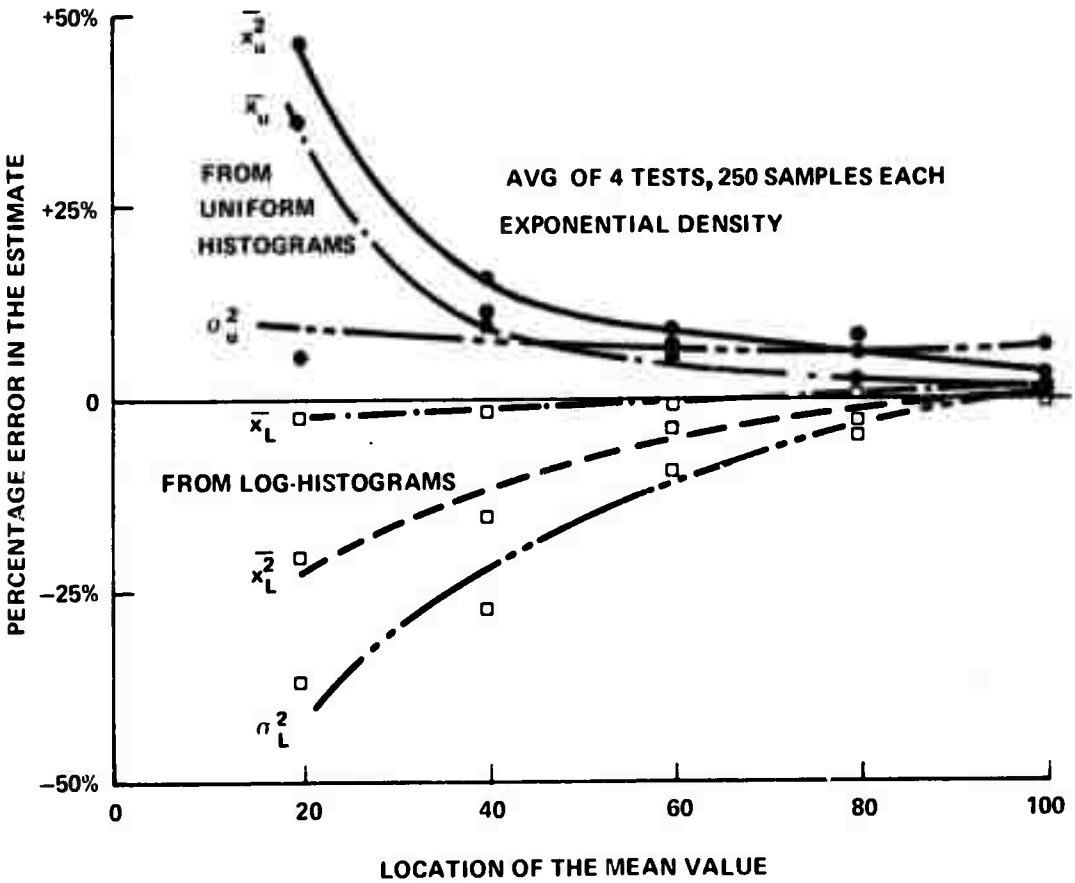


Figure E.1. A Comparison of the Errors in the Moment Estimates from Uniform and Log-Histograms.

and this difference is fairly constant for the uniform histogram case. This relationship can be shown by considering the error of the estimates:

$$er(\bar{x}) = \hat{E}[x] - E[x] \tag{E.9a}$$

$$er(\overline{x^2}) = \hat{E}[x^2] = E[x^2] \tag{E.9b}$$

$$er(\sigma^2) = \hat{\sigma}^2 - \sigma^2 \tag{E.9c}$$

The estimated variance is:

$$\hat{\sigma}^2 = \hat{E}[x^2] - (\hat{E}[x])^2 \tag{E.10}$$

and the actual variance is:

$$\sigma^2 = E[x^2] - (E[x])^2 \tag{E.11}$$

If we subtract Equation (E.11) from Equation (E.10), we obtain the error in the variance,

$$\hat{\sigma}^2 - \sigma^2 = \{\hat{E}[x^2] - E[x^2]\} - \{(\hat{E}[x])^2 - (E[x])^2\}$$

which can be written as:

$$er(\sigma^2) = er(\overline{x^2}) - \{\hat{E}[x] + E[x]\} \cdot er(\bar{x}) \tag{E.12}$$

For small errors, the sum term will be approximately equal to twice the mean value, such that:

$$er(\sigma^2) \cong er(\overline{x^2}) - 2\,\bar{x}\,er(\bar{x}) \tag{E.13}$$

This expression simplifies for the case of the exponential density with,

$$\overline{x^2} = 2\sigma^2 = 2(\bar{x})^2 \tag{E.14}$$

resulting in:

$$\frac{er(\sigma^2)}{2\sigma^2} \cong \frac{er(\overline{x^2})}{\overline{x^2}} - \frac{er(\bar{x})}{\bar{x}} \tag{E.15}$$

which yields Equation (E.8).

GLOSSARY

Accumulated statistics: The measurement routine which gathers counts, sums, and histograms of activity in an IMP over a time period (presently 12.8 seconds).

ACK, acknowledgement: An IMP-to-IMP transmission from IMP_i to IMP_j that verifies that a packet was received and accepted by IMP_i from IMP_j along the store-and-forward path.

ARPA: The Advanced Research Projects Agency, the agency of the Department of Defense which sponsored the network development.

Artifact: Any measurement effect which results in a difference between the observed system variables and the actual system variables (as they would exist if the measurements had not been made).

Artificial traffic: Traffic that is generated solely for purposes of network loading and tests.

BBN: Bolt Beranek and Newman, Inc., the network contractor and a node in the network.

Blocking: The condition that occurs when an IMP cannot accept transmissions from its neighbors due to the lack of buffers.

Buffer: Storage space for a full packet.

Fake-HOST: An IMP routine which performs one or more of the HOST-like functions, e.g., the source or destination of messages such as the statistics data, the IMP console Teletype, and the

Preceding page blank

discard facility.

FCFS: First-Come-First-Served queue discipline.

First-order moment correction: A correction technique for moment estimates from histogram data using a linear slope approximation based on neighboring interval heights.

GHOST: Same as a Fake-HOST.

Header: A 64-bit record preceding each packet, which contains the source, the destination, the message number, etc.

HOP: One IMP-to-IMP path or leg. The HOP # is the number of such legs between two nodes.

HOST: One of the independent computing centers of the network.

Hello: A periodic transmission between adjacent IMP's to pass routing information and to test the operational status of the lines.

HOST-to-HOST protocol: The prescribed manner in which HOST computers communicate.

IHY: I Heard You; the response to a Hello message.

IMP: Interface Message Processor; the store-and-forward message switches for the network.

Inter-packet gap: The time interval that occurs between otherwise contiguous packet flow due to other traffic on the communication channels.

Leader: The 32-bits of information which precede each message and define its destination, message number, etc.

Link: A logical (software) connection between the IMP's at the source and destination nodes of a message transmission. The link becomes "blocked", (i.e., cannot be used again) until a RFNM is returned.

L L: Lincoln Laboratories

Log-histogram: A histogram in which the intervals are scaled in a logarithmic (or exponential) manner. In the case of binary log-histograms, each interval is twice the size of the preceding interval.

Marking: A bit-string 00...01 preceding the first bit of a message (for transmissions between HOST computers having differing word lengths).

MIT: Massachusetts Institute of Technology

Message: A logical record of up to 8095 bits which is used for HOST-to-HOST transmissions.

Modem: A modulator-demodulator used to convert between digital data and the frequency-modulated telephone carrier signal.

Moment corrections: The addition of term(s) to compensate for the systematic bias in moment estimates from grouped data such as histograms.

msec: millisecond, i.e., 10^{-3} seconds.

μsec: microsecond, i.e., 10^{-6} seconds.

Net, network: The communications facilities (including IMP's) which interconnect the HOST computers. Sometimes used in a larger

scope including the HOST computers as well.

NMC: The Network Measurement Center (at UCLA).

Node: One of the sites involved in the network, i.e., typically referring to both the IMP and HOST(s).

Packet: A message segment of up to 1008 bits of information, (the standard IMP-to-IMP transmission block size).

Padding: A bit-string 10...00 following the last bit of a message to fill out the remaining portion of the IMP (or destination HOST) word length.

Poisson arrivals: Independent random arrivals to a queueing system (with an exponential density of interarrival times).

Propagation delay: The transmission delay due to the finite propagation time of electrical energy along a transmission line (about 10 μsec. per mile).

Protocol: The established procedures and standards for network transmissions.

Pseudo-messages: Fixed length artificial traffic generated by the IMP's.

RAND: The RAND Corporation.

Reassembly: The process of collecting packets and placing them in the proper sequence to recreate the segmented message.

RFNM: Request For Next Message; the response which is returned to the message source to allow further transmission along a given

link. (A congestion and flow control mechanism.)

RFNM driven traffic: Traffic flow that occurs as rapdily as the RFNM mechanism will allow.

Routing: The selection of each store-and-forward "hop" along the source-to-destination path.

Routing table: A table listing the estimated best known communication channels for transmission to the various destinations.

Routing update: The process of exchanging routing information between the nodes in a network.

SDC: System Development Corporation.

Sigma 7: The computer system at the UCLA Network Measurement Center.

Segment: A portion of a message; often synonymous with packet.

Segmented messages: The division of messages into fixed size blocks (called packets).

SENT queue: A linked list of packets which have been transmitted, but not yet acknowledged.

Snap-shots: The measurement routine which records the state of the IMP queues and routing tables at periodic intervals (presently every 0.8 sec.).

SRI: Stanford Research Institute.

Store-and-forward: The receipt and subsequent transmission of a packet by an intermediate node in a network source-destination path.

Thru-put: The effective rate at which data can be transmitted through the network, i.e., between a given source and destination.

Traces: Measurement data taken at an IMP recording the time at which a given packet was received and "processed".

UCLA: University of California at Los Angeles.

UCSB: University of California at Santa Barbara.

Unfinished work diagram: A diagram showing the amount of the service "backlog" which is present in the system as a function of time.

Uniform histogram: A conventional histogram in which all of the intervals are of equal size.

Utah: The University of Utah.