TRAFFIC METRICS FOR
ADAPTIVE ROUTING

By
Mital Parikh
B.S., Bombay University, India

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF MASTER OF SCIENCE IN ELECTRICAL ENGINEERING
UNIVERSITY OF MASSACHUSETTS-LOWELL

Signature of Author:_____________________________Date______________

Signature of Thesis Supervisor:____________________________

Kavitha Chandra, D.Eng., Associate Professor

Signature of Other Thesis Committee Members:

______________________________________

Charles Thompson, Ph.D, Professor

______________________________________

Vineet Mehta, D.Eng., Adjunct Professor
TRAFFIC METRICS
FOR ADAPTIVE ROUTING

By

MITAL PARIKH

ABSTRACT OF THESIS
SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR
THE DEGREE OF MASTER OF SCIENCE IN ELECTRICAL ENGINEERING
UNIVERSITY OF MASSACHUSETTS-LOWELL
ABSTRACT

A measurement based analysis and modeling of HTTP traffic statistics is presented. The objectives are to characterize and model the non-stationarity in the mean value of the packet arrival process and to apply the model in adaptive routing algorithms. To this end, recursive estimation algorithms are implemented and used to determine characterizing the mean number of packets in a prescribed interval time. Discrete time Markov chains and first order autoregressive process are shown to adequately capture the temporal variation. The AR model is implemented in a state-space formation for estimating the mean arrival rate and buffer occupancy at a network node. Kalman estimators are used to predict the optimal size of the state vector. The application of this technique in adaptive routing is demonstrated.
ACKNOWLEDGEMENTS

First and foremost I would like to express my sincere appreciation and thanks to my advisor, Prof. Kavitha Chandra. I am grateful for her guidance and for introducing me to the world of traffic-engineering. I am also thankful to Prof. Charles Thompson for providing useful insights to my work and for being a good friend. I would like to thank Gbenga and Max for being good friends and helping me with the data collection process. I would like to thank Parita for her moral support. I would also like to thank Jimmie Davis for being a good friend and helping me with my courses. I would like to thank Prof Martin Patt for providing me with financial support. I am grateful to Dr. Vineet Mehta and Prof Krishnan for their support. I would also like to thank my mom and grandparents for their support.
TABLE OF CONTENTS

Abstract iii
Acknowledgements iv
Table of Contents v
List of Figures vii
List of Tables xi
List of Symbols xii

Chapter 1. Introduction
  1.1 Background 1
  1.2 Internet routing 3
  1.3 Thesis objectives 9

Chapter 2. Network Traffic Visualization and flow characterization
  2.1 Introduction 11
  2.2 Traffic Visualization 12
  2.3 Traffic metrics for flow characterization 30
  2.4 Short time scale packet arrival statistics 41
  2.5 Summary 52

Chapter 3. Recursive Estimation of Packet Count Statistics
  3.1 Introduction 54
  3.2 Traffic Statistics - Packet count mean and Variance 55
  3.3 Recursive Estimation of the Mean 58

Chapter 4. Application of Kalman Estimators for Adaptive routing
  4.1 Introduction 72
  4.2 State space model 73
  4.3 Kalman Filter Estimation 75
  4.4 Adaptive routing 80
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.5 Summary</td>
<td>86</td>
</tr>
<tr>
<td>Chapter 5. Conclusion and future work</td>
<td>87</td>
</tr>
<tr>
<td>Appendix I</td>
<td>90</td>
</tr>
<tr>
<td>Appendix II</td>
<td>96</td>
</tr>
<tr>
<td>References</td>
<td>100</td>
</tr>
<tr>
<td>Biography</td>
<td>102</td>
</tr>
</tbody>
</table>
## List of Figures

| Figure 2.1 | UML Campus Measurements | 13 |
| Figure 2.2a | Outbound traffic due to TCP, UDP and OTHER protocols | 16 |
| Figure 2.2b | Inbound traffic due to TCP, UDP and OTHER protocols | 16 |
| Figure 2.3a | Traffic contributed by different applications (Outbound) | 18 |
| Figure 2.3b | Traffic contributed by different applications (Inbound) | 19 |
| Figure 2.4a | Packet size distribution for aggregate outbound traffic | 20 |
| Figure 2.4b | Packet size distribution for aggregate inbound traffic | 21 |
| Figure 2.5a | Packet size vs application port during peak hour (outbound) | 22 |
| Figure 2.5b | Packet size vs application port during peak hour (inbound) | 22 |
| Figure 2.6 | Dominant destinations for Outbound traffic | 23 |
| Figure 2.7 | Dominant source subnets for outbound traffic | 24 |
| Figure 2.8a | Inbound traffic packets per hour | 26 |
| Figure 2.8b | Outbound traffic packets per hour | 26 |
| Figure 2.9a | Inbound traffic bytes per hour | 27 |
| Figure 2.9b | Outbound traffic bytes per hour | 27 |
| Figure 2.10 | Percentage of traffic generated vs number of flows | 33 |
| Figure 2.11 | Model fit to (70%) of total traffic | 35 |
| Figure 2.12 | Approximate model fit to entire distribution | 36 |
| Figure 2.13 | Minimum mean square error between the data and model | 37 |
| Figure 2.14 | Readjusted model fit | 37 |
| Figure 2.15 | Regression model of $a$ vs $\delta t_f$ | 38 |
| Figure 2.16a | Comparison of Model and Observation (mar01) | 39 |
| Figure 2.16b | Comparison of model and Observation (mar15) | 39 |
| Figure 2.16c | Comparison of model and Observation (apr19) | 39 |
| Figure 2.17 | Hierarchy of UML network | 43 |
| Figure 2.18a | Cov for hosts (feb16) | 48 |
| Figure 2.18b | Cov for hosts (mar01) | 48 |
| Figure 2.18c | Cov for hosts (mar15) | 48 |
| Figure 2.18d | Cov for hosts (apr19) | 48 |
| Figure 2.19a | Cov for Subnets (feb16) | 49 |
| Figure 2.19b | Cov for Subnets (mar01) | 49 |
| Figure 2.19c | Cov for Subnets (mar15) | 49 |
| Figure 2.19d | Cov for Subnets (apr19) | 50 |
| Figure 2.20 | Cov at aggregate level | 51 |
| Figure 3.1a | Time series (feb16) | 57 |
Figure 3.1b  Time series (mar01)  57
Figure 3.1c  Time series (mar15)  57
Figure 3.1d  Time series (apr19)  57
Figure 3.2  Recursive estimate of the mean under stationarity  60
Figure 3.3a  Recursive estimate with exponential windowing (apr19)  62
Figure 3.3b  Recursive estimate with exponential windowing (feb16)  63
Figure 3.3c  Recursive estimate with exponential windowing (mar01)  63
Figure 3.3d  Recursive estimate with exponential windowing (mar15)  63
Figure 3.4a  Residual error of recursive mean process  64
Figure 3.4b  CDF of the residual and the Gaussian process  64
Figure 3.5  NACF of the residual error  65
Figure 3.6  Mean estimate for 30 seconds  67
Figure 3.7a  ICDF of delay using fluid buffer and simulation(apr19)  72
Figure 3.7b  ICDF of delay using fluid buffer and simulation(feb16)  72
Figure 3.7c  ICDF of delay using fluid buffer and simulation(mar01)  72
Figure 3.7d  ICDF of delay using fluid buffer and simulation(mar15)  72
Figure 4.1  Estimated packet count process using Kalman Process  78
Figure 4.2  Estimated buffer occupancy using Kalman Process  78
| Figure 4.3 | CDF from Kalman estimator and data | 79 |
| Figure 4.4 | Error between model and estimator for buffer occupancy | 79 |
| Figure 4.5 | 3-Node Network | 80 |
| Figure 4.6a | Performance of buffer 1 | 84 |
| Figure 4.6b | Performance of buffer 2 | 84 |
| Figure 4.7 | Rerouted traffic | 86 |
| Figure A1.1 | OSI layer 7 model | 90 |
| Figure A1.2 | Ethernet IEEE 802.3 frame | 91 |
| Figure A1.3 | IP header format | 92 |
| Figure A1.4 | Example | 93 |
## List of Tables

| Table 2.1 | Raw header traces | 14 |
| Table 2.2 | Mapping of source port index to bin value for visualization | 17 |
| Table 2.3 | Mapping of packet size to bin index for visualization | 19 |
| Table 2.4 | UML aggregate Inbound traffic statistics | 28 |
| Table 2.5 | UML aggregate Outbound traffic statistics | 28 |
| Table 2.6 | IP address classification | 31 |
| Table 2.7 | Modeling parameters for flows (aggregate traffic) | 40 |
| Table 2.8 | HTTP flow characterization parameters | 41 |
| Table 2.9 | Traffic statistics | 42 |
| Table 2.10a | Statistics for 129.63.<sip3>.xx(feb16) | 43 |
| Table 2.10b | Statistics for 129.63.<sip3>.xx(mar01) | 44 |
| Table 2.10c | Statistics for 129.63.<sip3>.xx(mar15) | 44 |
| Table 2.10d | Statistics for 129.63.<sip3>.xx(apr19) | 45 |
| Table 2.11a | Mean and Variance 129.63.153.<sip4> (feb16) | 46 |
| Table 2.11b | Mean and Variance 129.63.153.<sip4> (mar01) | 46 |
| Table 2.11c | Mean and Variance 129.63.153.<sip4> (mar15) | 47 |
| Table 2.11d | Mean and Variance 129.63.153.<sip4> (apr19) | 47 |
| Table 3.1 | Mean and Variance of residual error | 61 |

| Table 2.11a | Mean and Variance 129.63.153.<sip4> (feb16) | 46 |
| Table 2.11b | Mean and Variance 129.63.153.<sip4> (mar01) | 46 |
| Table 2.11c | Mean and Variance 129.63.153.<sip4> (mar15) | 47 |
| Table 2.11d | Mean and Variance 129.63.153.<sip4> (apr19) | 47 |
| Table 3.1 | Mean and Variance of residual error | 61 |
List of Symbols

$\delta t_f$ Selected time duration

$N_{\delta t_f}$ Number of flows in a selected time duration

$F_v(n_f)$ CDF of byte traffic generated by number of flows $n_f$

$a$ parameter of the CDF of byte traffic

$\gamma$ parameter of the CDF of byte traffic

$\beta$ function of $a$ and $\gamma$

$N_s$ number of observations needed for linear approximation

$\varepsilon$ least square error estimator

$r^2$ square of error

$\theta$ parameter vector

$A$ matrix consisting of number of flows

MMSE Minimum mean square error

$\tau_i$ packet interarrival times

$k$ consecutive intervals for COV

$C_k^2$ coefficient of variation

$S_k$ sum of $k$ consecutive interarrivals
\( C \) capacity
\( \lambda \) average arrival rate
\( \rho \) link utilization
\( y_k \) packet count time series
\( a_k \) mean of \( k \) observations
\( \hat{a}_k \) estimate of mean \( a_k \)
\( P_k \) weighting coefficients
\( T_s \) Sampling interval
\( T_e \) exponential time constant
\( \alpha \) exponential windowing constant
\( \varepsilon_k \) residual error in the estimation
\( R_1(\varepsilon(\alpha)) \) NACF of the residual error
\( S_\varepsilon(\alpha) \) Sum of NACF of residual error
\( D \) Maximum difference between gaussian distribution and error distribution
\( D_c \) Critical value for Kolmogrov-Smirnov test
\( B \) buffer occupancy
\( \tau_b \) packet delays
\( A_i(k) \) average arrival rate at node \( i \)
\( \mu_i \) mean of AR process

\( \alpha_i \) AR coefficient

\( g_i(k) \) iid noise process

\( \sigma_i^2 \) variance of noise process

\( B_i(k) \) buffer occupancy or number of packets queued at time \( k \)

\( C_{ij} \) Capacity between link \( i \) and \( j \)

\( \chi(k) \) state vector

\( \hat{B}_k \) estimated buffer occupancy

\( S_{ij} \) cost function between links \( i \) and \( j \)

\( C_f \) cost factor
CHAPTER 1
INTRODUCTION

1.1 BACKGROUND

Computer communications over the Internet requires the coordination of protocols, switching and routing functions. The delivery of packets from source to destination end-points typically takes place using best-effort service. The packets are queued at a router for access to an outgoing link and are serviced on a first-in-first-out (FIFO) basis. There is work underway to add quality-of-service measures \cite{1} to the best-effort service paradigm. These activities are being coordinated by the Internet Engineering Task Force (IETF) and include work related to (i) Traffic Engineering (TE) and Multiprotocol Label Switching (MPLS) \cite{2} (ii) Integrated Services (Intserv) \cite{3} (iii) Differentiated Services (Diffserv) \cite{3} and (iv) IP Performance Metrics (IPPM) \cite{4}.

Traffic engineering is the application of technology and scientific rules to optimize and control network performance. A well engineered application must jointly address the performance requirements of end users (traffic management) and efficient usage of network resources (capacity management). Performance measures are typically the end-to-end delay, delay variation, packet loss and throughput. Network efficiency is measured with respect to the usage of buffer space, link bandwidth and computational resources. The Internet is subjected to traffic surges, link failures etc. A traffic engineered system
will have to function as a closed-loop control system operating on network state variables, input functions, decision and policy information. In addition to enforcing traffic and network management functions, a well engineered system must also evaluate network performance to assess the effectiveness of traffic engineering rules.

The Intserv model was first proposed for Internet QOS. This model proposes the reservation of network resources based on traffic flow QOS requirements and characteristic traffic features prior to transmission. The functions of packet classification, packet scheduling and admission control are salient features of the proposal. The approach offers guaranteed service with deterministic bounds on end-to-end delays and controlled service for applications that can tolerate delay variations. Intserv requires resource reservation on each node in the source to destination path. Scalability is a problem with this approach. Significant overhead is generated by the signaling protocol during bandwidth reservation for the thousands of micro-flows that may exist at any given time. Modifications to the original proposal are currently underway to address the scalability issue.

The Diffserv model, proposed subsequent to Intserv, attempts to remove some of the limitations of Intserv by considering the aggregation of flows based on their traffic characteristics and/or performance requirements. The differentiated services approach to providing quality of service in networks employs a small, well defined set of building blocks from which a variety of traffic types may be aggregated. A six bit pattern in each IP packet is used to mark a packet for forwarding treatment, or per-hop behaviour (PHB)
at each network node. A number of PHB groups have been standardized along with differentiated services for each of these groups. Unlike the Intserv model, packet classification and traffic shaping takes place only at the edge of the network or at the ingress to a Diffserv network. Additional components necessary to support differentiated services include traffic shapers and packet markers that could be used at the boundaries of networks. A behaviour aggregate (BA) is formed at the edge of a network by selecting certain packets through the use of classifiers and by imposing rules on those packets via traffic conditioners. The aforementioned activities are supported by the IPPM working group that is focussed on defining a set of standard metrics for monitoring the performance of Internet services.

In this thesis the focus is on the problems of Internet traffic characterization and identification of suitable traffic metrics. These metrics are to be used in state based network control functions. Techniques for improving current routing approaches using dynamic traffic features are considered. An overview of the fundamentals of packet switching and IP packets is provided in Appendix I. The basic requirements of Internet routing are briefly discussed next.

1.2 Internet Routing

The Internet is an interconnection of autonomous systems. Interior and exterior gateway protocols define the routing within and between the autonomous systems (AS).
Currently, the Routing Information Protocol (RIP) and Open-Shortest-Path-First (OSPF) serve as the interior routing protocols, while the Border Gateway Protocol (BGP) and Inter-Domain Routing Protocol (IDRP) are used for routing between AS’s\textsuperscript{[5]}. In a packet switching network, routing refers to the process of choosing a path over which to send packets, and router refers to a computer that makes such a choice. The function of a router is to transfer packets from one network to another. Routing may be characterized as direct delivery and indirect delivery. Direct delivery transmission of an IP datagram refers to the transmission between two machines on a single physical network and does not involve routers. The sender encapsulates the datagram in a physical frame, binds the destination address to a physical hardware address, and sends the resulting frame directly to the destination. IP addresses are divided into a network-specific prefix and a host-specific suffix. The chief advantage of dividing an IP address into two parts arises from a reduction in the size of the routing tables required in routers. Instead of keeping one routing entry per destination host, a router can keep one routing entry per network, and examine only the network portion of the destination address when making routing decisions. To see if a destination lies on one of the directly connected networks, the sender extracts the network portion of its own IP address. A match in the network portion means that the datagram can be sent directly without routing.

Indirect routing is required when many networks connected by routers separate the source and destination hosts. When one host wants to send data to any other host, an en-
capsulated datagram is sent to the nearest router. All physical networks are interconnected via routers attached to each network. Thus, the originating host can reach a router using a single physical network. When a data frame reaches a router the datagram is encapsulated and the IP selects the next router along the path towards the destination. The datagram is again placed in a frame and sent over the next physical network to a second router, and so on, until it is delivered.

Routing techniques may be broadly classified as fixed and dynamic or adaptive routing. Route calculation may be distributed or centralized, which determines the location of the computational engine in the routing system. Adaptive routing selects routes based on the current state information for the network. The state information permits the calculation of a cost metric can be minimized to determine the optimal route. Routing on the Internet has evolved to dynamic routing algorithms based on shortest path algorithms using costs assigned to each router interface. A description of the interface would include for example, the IP address of the interface, the mask, the type of network it is connected to, and the routers connected to that network. The collection of all these link-states form a link-state database that is maintained by each router.

The selection of a route can be based on the least number of hops required to reach the destination. Typically performance measures are also considered \cite{6}. Costs are assigned to links to support one or more design objectives. For example the cost could be inversely related to the link capacity (i.e. the higher the capacity, lower the assigned cost.
of the link) or the current queuing delay on the link. These constraints would serve to maximize throughput or minimize delay respectively. Vendors favour maximized throughput\(^7\) while ignoring variations in traffic demand.

Adaptive routing has been used in packet switched networks since the work on ARPANET proposed in 1980 by McQuillan et al\(^8\). In the approach proposed in \(^8\) each node maintains a database of complete network topology and network delays on all links. This information was used to generate the minimum delay path from a given node to a destination node. The delays in the database were measurement values at each node that were propagated through the network. Every 10 seconds, the node computed the average delay on each outgoing link. When new information arrived, it recomputed its routing table using Dijkstra’s algorithm\(^9\) which calculates the shortest path between two nodes. This was a effective routing mechanism only if there was some correlation between the reported delay values and those actually experienced after rerouting. The correlation was found to be high for network under light and moderate traffic loads. However, at heavy load levels, there was little correlation.

Load balancing techniques have also been used in telephone systems, since their inception. When no direct trunks are available for a telephone call, the originating switch checks the availability and load conditions of all two-link paths on a per-call basis. If any of the two-link paths are available, the call is set-up over the least loaded two-link path. The routing is generated for each call in real-time and discarded\(^10\).
The OSPF protocol is the standard Internal Gateway Protocol (IGP) for the TCP/IP networks, within a single autonomous system \cite{11}. It is based on the link-state technology and allows for a better load balancing based on the actual cost of the link. The OSPF algorithm uses link-state routing, where each router on initialization sends the link cost on each of its network interfaces to all other routers in the topology. This information is sent only when significant changes occur. Since each router receives the link-costs of all other routers in the topology, it can construct the topology of the entire configuration and calculate the shortest path to each destination network. Any routing algorithm, such as for example Dijkstra’s algorithm can be used to determine the next-hop. OSPF provides five possible cost metrics corresponding to the type of service (TOS) field set in the IP packet header. These include: (a) \textit{TOS 0}, the normal or default metric set by network administrators based on their policies and typically some performance objective; (b) \textit{TOS 2} which aims to minimize monetary cost; (c) \textit{TOS 4} used to maximize reliability based on known information such as history of outages or packet losses; (d) \textit{TOS 8} to maximize throughput using knowledge of data rate of the interface; and (e) \textit{TOS 16} which minimizes delay using transit time or delay through a particular hop. The delays in (e) are typically measured by routers at each networks interface. Therefore a router has a flexibility to set upto five routing paths based on each of the aforementioned cost metrics. Most service providers presently use the \textit{TOS 0} default metric set in a centralized off-line computation. Research studies are underway to design a more dynamic distributed algorithm for meet-
The utility of using traffic demand at each router to assign link costs has been discussed in recent studies. Traffic state information may comprise both measured and predicted values obtained from the network and from external sources. A new routing technology referred as Multi Protocol Label Switching (MPLS) has been recently proposed to incorporate more dynamic link costs and provide support to differentiated services and traffic engineering. The packets are classified upon entrance into an MPLS domain and prefixed with an explicit label which is a short fixed length number that does not include any network layer address. Within the MPLS domain the packet is processed and forwarded through the MPLS domain based only on the label using a "label swapping" technique. The label is removed by the egress label switching router (LSR) when the packet departs from the MPLS domain. The use of the label eliminates the need for packet classification at each router. Classification is done only once at the MPLS domain ingress. Within an MPLS domain, each LSR examines the label of an incoming packet and uses it as an index into an LSR’s forwarding table. Classification of packets at the ingress into an MPLS domain can be coarse, i.e. all packets with the same destination address or all packets belonging to a particular application between two hosts. Fortz and Thorup use a demand matrix to characterize traffic demand between a pair of source and destination nodes to specify OSPF weights on the links. Using this metric for minimizing a cost function, load-balancing that was substantially better than the MPLS ap-
approach was shown. The demand matrix modeled predictable periodic changes, but did not accommodate unpredicted bursts in traffic. The utility of using traffic demand computed from measurements in assigning the OSPF weights is also discussed by Feldmann et. al \cite{14}. Adaptive next-hop routing for type-of-service (TOS) classes was proposed by Matta et. al \cite{15} who considered separate queues for each TOS class at the outgoing link. The next-hops were updated based on link traffic changes. The IPV4 packet header allowed specification of a TOS field.

1.3 Thesis objectives

The Internet service migration approach under consideration by the IETF and major vendors of networking equipment and software is one that will require accurate characterization of traffic flows, methods for flow aggregation, simple parametric models for relevant traffic statistics and techniques for implementing these models in policing, admission control and routing functions. This work proposes methodologies for developing the four important components of TE. These are the measurement of traffic statistics, traffic characterization and modeling, identification of traffic states and their implementation in an adaptive routing scheme. These techniques may also be useful for specific functions within the Diffserv and Intserv implementations.

Based on prior work by You \cite{16} who identified a stationarity criteria for aggregating flows based on application ports and Olowoyeye \cite{17} who proposed dynamic models
for HTTP traffic, this work considers techniques for identifying the states of HTTP flows and their application for adaptive routing. A traffic characterization and visualization study is first presented in Chapter 2. The results allow the identification of long-term traffic trends and the applications that generate the dominant traffic. New traffic metrics for characterizing flows based on source-destination addresses are also proposed. Chapter 3 considers the estimation of traffic metrics when the statistics such as mean and variance of arrival processes are time-varying. Recursive estimation methods are applied to identify time-varying traffic parameters. The time-variation is captured in the form of a autoregressive time-series model that allows definition of traffic states. Chapter 4 implements a Kalman predictor for router state estimation and discusses the application of this technique in a simple three-node network. The simulation results are discussed. Finally Chapter 5 summarizes and concludes the thesis.
Chapter 2

Network Traffic Visualization and Flow Characterization

2.1 Introduction

Computer communications traffic exhibit features that are typically a function of the time-scale at which the traffic analysis is being carried out. Aggregating traffic at daily and hourly time intervals yields characteristic trends that allow identification of long term features. These trends exhibit quasi-deterministic seasonality that results from daily access patterns of users. Analysis of traffic at the packet arrival time scale yields information on the short-time scale dynamics. The short time scale is related to the client application, network and server response times. Intermediate to these time-scales, the traffic may be classified as flows, the average duration of which represent the time-scale over which two end-systems communicate. Quality of service based capacity allocation and routing will generally operate at the time-scales that characterize these flows. This chapter presents first in Section 2.2, a visualization study of network data traffic that allows for interpretation of the hourly traffic trends and dependencies between traffic parameters. The traffic statistics are a function of several parameters such as the time of the day, the application port, protocol and packet size. The data used in this work is obtained from packet header traces measured at the University of Massachusetts Lowell (UML) router that con-
nects the campus network to the Internet. Following this in Section 2.3, a flow characterization study is undertaken to determine the load generated by unique source-destination pairs. Finally in Section 2.4, the short time-scale packet features are examined by analysis of packet inter-arrival times.

2.2 TRAFFIC VISUALIZATION

Visualization studies are developed to extract meaningful insights from the large amount of network data available. A basic problem is determining the degree of predictability in the traffic data. Visualization of important long-term features of traffic generated from the campus to the Internet (outbound) as well as that of traffic in the reverse direction (inbound) is important in this regard. Visualization is implemented using the Visual Advizor software developed at Lucent Technologies. Traffic between the Internet and the University of Massachusetts Lowell (UML) network edge router flows through four T1 lines. The UML network is connected to the edge router through 100 Mbps fiber optic links. One of the ports on the router is dedicated to monitoring the uplink and downlink traffic. The packet header traces are obtained using the tcpdump utility. This utility captures to a file a summary of all the packets that traverse the network being monitored. The measurement location and data archive in relation to the UML network architecture is depicted in Fig 2.1. Data coming into and out of UML is forwarded to one of the ports on the UML Internet router (R). This monitoring port is connected to a hub. An optical
Transceiver is connected to the auxiliary interface (AUI) port of the hub and from here, a 10 Mbps optical cable carries the inbound and outbound traffic to the data collection and storage area.

![Diagram](image_url)

**Fig 2.1: UML Campus Measurements**

Traffic measurements are collected 24 hrs a day, seven days a week. The measurements of packet headers include timestamps at microsecond resolution, source and destination IP addresses, source and destination port numbers, packet size and protocol index. An example of the data trace is shown in Table 2.1.
Table 2.1 Raw Header Traces

Daily and hourly trends in the traffic to and from the Internet are useful to determine the peak usage hours and the off-peak hours. The results of the study are presented for one weekday (Wednesday, 10/6/99) in the following section. The traffic characteristics are influenced by features such as the time-of-day, protocol, application ports, and packet sizes. To determine the influence of protocol, the number of packets and bytes generated were classified according to the protocol field. Figs. 2.2(a) and (b) show the three-dimensional representation of the number of packets generated per hour on the vertical axis as a function of the protocol index and time of day. The two graphs represent the outbound and inbound directions respectively. In these graphs index 6 represents transmission control protocol (TCP), 17 represents user datagram protocol (UDP), and the other protocols that occurred in the measurements are marked index 1. On the time axis, hour 0 represents midnight 00:00. Over 77% of the traffic is generated by the TCP protocol, followed by UDP which accounts for about 15 – 20%. The daily usage pattern is distinctly different from that observed on telephone networks where peak hours can be clearly distinguished. In the case of data network traffic, a comparable volume exists during most hours of the day. In the case of outbound traffic volume (Fig. 2.2(a)) a dip occurs during the early hours around 06:00 as well as around the 12:00 – 15:00, 18:00 and 21:00 time
frame due to the reduction in number of users on the network for reasons such as lunch break, fewer afternoon classes, end of classes, and dinner times. The peak usage occurs during the evening hours from 18:00 onwards, which can be an expected trend for campus networks. This is the time that students spend browsing the Internet. The pattern will be expected to be different for usage from a corporate network, where the peak hours would typically occur between 09:00 to 17:00. The visualization of traffic generated from the Internet shows similar patterns to the outbound traffic. Except that for certain applications such as HTTP, the volume is much higher since this traffic is primarily server generated information for which both the average packet size and packet rate is larger than that of client packets generated in the outbound direction. Since TCP is the dominant protocol, the influence of different TCP applications on the traffic mix is considered next.
Fig. 2.2(a): Outbound traffic due to TCP, UDP and other protocols.

Fig. 2.2(b): Inbound traffic due to TCP, UDP and other protocols.
The TCP applications may be identified by considering the source port index in the traffic header. The source port index ranges from 1 to over 50000. The well known ports correspond to applications such as HTTP (80), FTP (20), SMTP (25), NNTP (19), Telnet (20) etc.. Indices in the range 1024 - 49151 are registered port numbers used by applications executed by users. For visualization purposes the source port indices are grouped and mapped to a set of eight integer values as shown in Table 2.2. Under this mapping, for example, HTTP sources will appear in bin 10, FTP sources in bin 1, SMTP in bin 2 etc.. Figs. 2.3(a) and 2.3(b) depict the outbound and inbound traffic packets contributed by the applications during each hour of the day. The application port axis has been reordered in Fig. 2.3(a) to depict the aggregate traffic count of each application group in descending order. The outbound traffic pattern shows that registered ports (1000 - 10000) in bins 25 and 30 contribute a significant traffic volume. This represents mainly client based communication generated from applications such as HTTP, FTP etc.. In the inbound direction, the HTTP port in bin 10 dominates the traffic flow from 09:00 onwards, contributing over 70% of traffic bytes. This indicates that a significant portion of outbound traffic arises due to UML clients accessing servers and the inbound traffic is comprised mainly of the response of these servers residing on the Internet. One can also observe in the hourly patterns, a high level of correlation between the client generated traffic in bin 25 of Fig. 2.3(a) and the corresponding server response in bin 10 in Fig. 2.3(b). A significant portion of the inbound traffic also arises from bin 25 representing port indices
1001 – 5000 which may be applications generated by Internet clients.

<table>
<thead>
<tr>
<th>Src Port Index</th>
<th>Bin</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-24</td>
<td>1</td>
</tr>
<tr>
<td>25-50</td>
<td>5</td>
</tr>
<tr>
<td>51-100</td>
<td>10</td>
</tr>
<tr>
<td>101-500</td>
<td>15</td>
</tr>
<tr>
<td>501-1000</td>
<td>20</td>
</tr>
<tr>
<td>1001-5000</td>
<td>25</td>
</tr>
<tr>
<td>5001-10000</td>
<td>30</td>
</tr>
<tr>
<td>10001+</td>
<td>35</td>
</tr>
</tbody>
</table>

Table 2.2. Mapping of source port index to bin value for visualization

Fig. 2.3(a): Traffic contributed by different applications (outbound)
Next the characteristic features of packet sizes as a function of both time and application ports are examined to determine any correlation that may exist between particular applications and packet sizes. To facilitate visualization, the packet sizes which typically range from 40-1500 bytes are mapped to a set of seven indices as tabulated in Table 2.3.

<table>
<thead>
<tr>
<th>packet size</th>
<th>Bin</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-40</td>
<td>1</td>
</tr>
<tr>
<td>40-100</td>
<td>5</td>
</tr>
<tr>
<td>101-300</td>
<td>10</td>
</tr>
<tr>
<td>301-600</td>
<td>15</td>
</tr>
<tr>
<td>601-900</td>
<td>20</td>
</tr>
<tr>
<td>901-1200</td>
<td>25</td>
</tr>
<tr>
<td>1201+</td>
<td>30</td>
</tr>
</tbody>
</table>

Table 2.3: Mapping of packet size to bin index for visualization.
The packet size distribution is one of the features that exhibits the most consistent behavior, even among different networks. The packets are typically concentrated around the 40, 576, and 1500 byte regions and these values represent the IP packet header size, the maximum transfer unit supported by all IP routers and maximum packet size for IP networks respectively. Figs. 2.4(a) and 2.4(b) depict the volume of packets of a particular size along the vertical axis with the packet size range and hour of the day given on the other two axes. For the outbound direction the packet sizes in the range 0-100 contribute to over 60% of the traffic, indicating that this direction is characterized by client traffic. These packets are mainly comprised of TCP acknowledgement and HTTP document access related packets. On the other hand, in Fig. 2.4(b) the server traffic that dominates the inbound direction includes in addition to the 40 byte packets, a large percentage of data packets of size 576 and 1500 bytes.

![Fig. 2.4(a): Packet size distribution for aggregate outbound traffic.](image-url)
To determine the relation between packet sizes and application ports if any, Figs. 2.5(a) and 2.5(b) depict the packet count generated during one of the peak usage hours (11:00) as a function of the source port index and packet size. The outbound traffic is dominated by the cylinder at the intersection of bins one and twenty five. These attributes represent packet sizes in the 0−40 byte range and source ports ranging from 1001−5000. The corresponding graph for the inbound traffic shows that the 1500 byte packets predominantly belong to HTTP server traffic in bin 10. The source port index 25 exhibits dependence on the 40 byte packet similar to the client pattern observed in Fig. 2.5(a). A small fraction of 1500 byte packets are seen to also belong to the applications belonging to index 1, which may be FTP, SMTP etc.
Next, leading to the characterization of flows, the analysis of outbound traffic is
carried out to determine the destinations to which most of the UML traffic is directed to. Fig 2.6 shows the traffic byte count between the UML network and a destination distinguished by its network address $<\text{dip} > .xx.xx.xx$. The notation $< xx >$ denotes that this field is not considered. The data analyzed here is for one peak hour of traffic. The vertical axis shows the number of bytes generated between each source destination pair during the hour. A few destinations have been found to be the dominant traffic routes in all of the data analyzed. The destinations correspond to network prefixes 24, 130, 207 and 209 which consist typically of sites such as Yahoo.com, Hotmail.com and music sites hosting MP3 servers. These sites are typically used by students to check their Emails and download free music.

![Dominant Destinations for Outbound traffic](image)

**Fig. 2.6:** Dominant Destinations for Outbound traffic

Finally the typical traffic matrix between UML subnets and destination networks is
depicted in Fig. 2.7. The vertical axis represents the byte count between the corresponding source-destination pair. The UML subnet is represented by the third suffix \( sip3 \) of source IP address 129.63.\( sip3 \cdot xx \) and the destination address is same as in Fig. 2.6. The largest traffic volume is found to be generated from Concordia dormitory with subnet id \( sip3 = 208 \) towards \( dip1 = 130 \) which hosts the Napster music site.

![Graph showing traffic volume]

**Fig 2.7:** Dominant source subnets for Outbound traffic

The basic features of long-term Internet traffic generated from UML networks have been identified using a single days worth of traffic measurements. The long-term feature analysis is concluded by summarizing the traffic behavior over a week. The inbound and out-bound traffic was recorded at the access point of the UML network from 00:00 on 10/5/99 to 23:59 on 10/11/99. The data is processed to create a time-series. Each sample
in the time-series is comprised of the byte-count/packet-count summed over a one hour period. This results in one sample per hour. Fig 2.8 and 2.9 show an example of the week long UML traffic pattern during the 10/5/99 - 10/11/99 period. The seven consecutive days correspond to Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday and Monday. The low traffic on October 11 (Monday) was due to a holiday due to Columbus day. Fig 2.8(a) denotes the packet count time-series for the inbound traffic and 2.8(b) represents the same for outbound traffic. The inbound traffic volume is summarized in Table 2.4 and the outbound traffic in Table 2.5. Each table gives the total number of packets and bytes for each day. In addition, hourly averages are given. One can estimate that a typical work-day generates an aggregate of approximately 40.5 mega-packets per day transferring about 23 Gigabytes in each direction.
Fig. 2.8(a): Inbound traffic packets per hour.

Fig. 2.8(b): Outbound traffic packets per hour.
Fig. 2.9(a): Inbound traffic bytes per hour.

Fig. 2.9(b): Outbound traffic bytes per hour.
<table>
<thead>
<tr>
<th>Day</th>
<th>Tot Pkts /10^7</th>
<th>Tot Bytes /10^10</th>
<th>Avg Pkts/hr /10^6</th>
<th>Avg Bytes/hr /10^9</th>
</tr>
</thead>
<tbody>
<tr>
<td>10/5/99 (T)</td>
<td>4.58</td>
<td>2.45</td>
<td>1.91</td>
<td>1.02</td>
</tr>
<tr>
<td>10/6/99 (W)</td>
<td>4.47</td>
<td>2.28</td>
<td>1.86</td>
<td>0.95</td>
</tr>
<tr>
<td>10/7/99 (R)</td>
<td>4.61</td>
<td>1.69</td>
<td>1.92</td>
<td>0.71</td>
</tr>
<tr>
<td>10/8/99 (F)</td>
<td>4.50</td>
<td>2.43</td>
<td>1.88</td>
<td>1.01</td>
</tr>
<tr>
<td>10/9/99 (S)</td>
<td>2.96</td>
<td>1.42</td>
<td>1.23</td>
<td>0.59</td>
</tr>
<tr>
<td>10/10/99 (S)</td>
<td>2.66</td>
<td>1.40</td>
<td>1.11</td>
<td>0.58</td>
</tr>
<tr>
<td>10/11/99 (M)</td>
<td>2.96</td>
<td>1.56</td>
<td>1.23</td>
<td>0.65</td>
</tr>
</tbody>
</table>

| Avg Daily Volume (pkts) | 3.82 x 10^7 |
| Avg Daily Volume (bytes) | 1.89 x 10^9 |
| Avg Daily Rate (pkts/sec) | 442 |
| Avg Daily Rate (bits/sec) | 1.75 x 10^5 |

Table 2.4: UML Aggregate Inbound Traffic Statistics

<table>
<thead>
<tr>
<th>Day</th>
<th>Tot Pkts /10^7</th>
<th>Tot Bytes /10^10</th>
<th>Avg Pkts/hr /10^6</th>
<th>Avg Bytes/hr /10^9</th>
</tr>
</thead>
<tbody>
<tr>
<td>10/5/99 (T)</td>
<td>4.67</td>
<td>2.53</td>
<td>1.95</td>
<td>1.05</td>
</tr>
<tr>
<td>10/6/99 (W)</td>
<td>4.66</td>
<td>2.71</td>
<td>1.94</td>
<td>1.13</td>
</tr>
<tr>
<td>10/7/99 (R)</td>
<td>4.34</td>
<td>2.68</td>
<td>1.81</td>
<td>1.11</td>
</tr>
<tr>
<td>10/8/99 (F)</td>
<td>4.49</td>
<td>2.63</td>
<td>1.87</td>
<td>1.09</td>
</tr>
<tr>
<td>10/9/99 (S)</td>
<td>2.99</td>
<td>1.87</td>
<td>1.25</td>
<td>0.78</td>
</tr>
<tr>
<td>10/10/99 (S)</td>
<td>2.69</td>
<td>1.32</td>
<td>1.12</td>
<td>0.55</td>
</tr>
<tr>
<td>10/11/99 (M)</td>
<td>2.84</td>
<td>1.44</td>
<td>1.18</td>
<td>0.60</td>
</tr>
</tbody>
</table>

| Avg Daily Volume (pkts) | 3.81 x 10^7 |
| Avg Daily Volume (bytes) | 2.17 x 10^10 |
| Avg Daily Rate (pkts/sec) | 441 |
| Avg Daily Rate (bits/sec) | 2.0 x 10^6 |

Table 2.5: UML Aggregate Outbound Traffic Statistics

In both tables, the first column represents the total packets transmitted for each day while the second column shows the total bytes transmitted for each day. The average number of
packets and bytes per hour for each day are listed in the third and fourth columns respectively. The daily average rates were obtained by summing the total number of bytes/packets for each hour and dividing by 24. The average daily volumes were calculated by summing the total number of bytes or packets per day and dividing by the total number of days in the observation period. The average daily rates were similarly determined by summing the average hourly rates for each day and dividing by the total number of days in the observation period. The average in bytes per hour was converted to the more commonly used bits per second by first multiplying the byte-rate value by 8 bits/byte to convert the amount to bits per hour. The product is then divided by 3600 seconds to convert from bits per hour to bits per second. The volume of in- and outbound traffic is comparable in magnitude in terms of the packet count. However the outbound traffic transmits $2.17 \times 10^{10}$ bytes compared to the $1.8 \times 10^9$ bytes coming into the campus.

The cyclical patterns observed are typical of most local area networks, however the peak hours of activity may vary for corporate and university networks. The cyclical structures are particularly evident in the packets generated per hour. The peak activity is seen to typically begin during 10:00-11:00 and is sustained during the work day until about 20:00-21:00. The graphs show that the packet count trace exhibits less variability than the byte counts and may provide a more predictable component for traffic modeling. The next section considers the analysis of flows between UML sources and Internet destinations.
2.3 Traffic Metrics for Flow Characterization

Following the basic understanding of long-term hourly and daily trends in the network traffic, attention is next shifted to the characterization of medium and short-time scale behavior of the traffic. The short-time scale features are determined by the inter-arrival times and sizes of individual packets. The basic features of these random variables are examined in the next section. The medium time-scale is considered in this section. These features may be characterized by defining an entity referred to as a flow. At the basic level, a flow may represent the stream of packets with a common source and destination address. The typical duration of flows and the fraction of traffic contributed are important metrics for the design of resource allocation techniques and QOS based routing. The number and intensity of flows are particularly important for an understanding of the number of traffic states that will be required to be maintained for optimized routing. Claffy et al.\textsuperscript{[18]} proposed profiling of flows using timeout durations and determined that host pair flow timeout values between 16 and 128 seconds seemed to be appropriate tradeoff between router processing and memory. These durations were determined based on minimizing the setting up and tearing down of flow states at a router.

Each host on a TCP/IP Internet is assigned a unique 32-bit Internet address that is used in all communication with the host. The IP addresses of all hosts on a given network share a common prefix. The classification of IP addresses is as shown in Table 2.6.
<table>
<thead>
<tr>
<th>Class</th>
<th>Lowest Address</th>
<th>Highest Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.1.0.0</td>
<td>126.0.0.0</td>
</tr>
<tr>
<td>B</td>
<td>128.0.0.0</td>
<td>191.255.0.0</td>
</tr>
<tr>
<td>C</td>
<td>192.0.1.0</td>
<td>223.255.255.0</td>
</tr>
<tr>
<td>D</td>
<td>224.0.0.0</td>
<td>239.255.255.255</td>
</tr>
<tr>
<td>E</td>
<td>240.0.0.0</td>
<td>247.255.255.255</td>
</tr>
</tbody>
</table>

Table 2.6: IP address classification

The address 127.0.0.0, is reserved for loopback and is intended for use in TCP/IP and for inter-process communication on the local machine. The 32 bit IP address has an Internet portion and a local portion, where the Internet portion is the first 16 bits and identifies a site possibly with multiple physical networks. The local portion given by the last 16 bits identifies a physical network and a host on that site. This form of hierarchical addressing leads to hierarchical routing. The top level of routing uses the first two octets when routing, and the next level (i.e. the local site) uses an additional octet. Finally the lowest level (i.e. delivery across one physical network) uses the entire address \[19\].

The characterization of traffic flows generated between the UML network and Internet destinations is based on extraction of source and destination addresses from the packet header. The traffic is restricted to TCP flows and one hour of data during peak usage hours of 12:00 − 13:00 pm is analyzed. Four traces obtained on 02/16/00, 03/01/00, 03/15/00 and 04/19/00, each corresponding to a Wednesday were considered for this analysis. The date 03/15/00 however corresponds to a University holiday due to spring break. The flows are characterized using the pair \(< src – addr, dest – addr >\) where
src − addr = 129.63.<sip3>.<x>

dest − addr = <dip1>..<dip2>..XX.XX , where <dip1> and <dip2> refers to all network prefixes except <129> and <63>. All packets satisfying unique <sip3>,<dip1> values are classified as belonging to a unique flow. This allows the determination of number of flows that are active in a selected time duration $\delta t_f$. A given $\delta t_f$ leads to some maximum of $N_{\delta t_f}$ number of flows, where $N_{\delta t_f}$ is a random variable. These flows are sorted according to the number of bytes of traffic generated between the respective source and destination and a cumulative count of the percentage of byte traffic as a function of number of flows is obtained. Since this metric can change randomly with successive intervals of $\delta t_f$, the average cumulative distribution is calculated over all of the intervals in a one-hour time period. The average cumulative distribution of the traffic bytes as a function of the number of flows is shown in Fig. 2.10 for $\delta t_f = 10, 30$ and 300 seconds. The horizontal axis shows the number of flows and the vertical axis represents the fraction contributed to the total traffic. Both axes are scaled logarithmically. This result shows for example that 90% of the load is generated by 47 flows on a 10 second time scale while the number increases to 57 flows on a 30 second time scale and 94 flows on a 300 second time scale. Also, over 25 % of the total traffic is generated by a single dominant flow. A quantitative characterization of the curves in Fig. 2.10 will enable a determination of the number of flow states to be maintained on average for a given time-scale. The modeling approach is discussed next.
Fig 2.10: Percentage of traffic generated vs number of flows on X-axis.

The cumulative distribution of byte traffic volume \( F_v(n_f) \) generated by number of flows \( n_f \) is modeled as,

\[
\hat{F}_v(n_f, \alpha, \gamma) = \frac{n_f^\gamma}{(a + n_f)^\gamma}
\]

(2.1)

where \( \alpha \) and \( \gamma \) are the parameters to be determined. Equation (2.1) satisfies the boundary constraint that in the limit as \( n_f \to \infty \), \( \hat{F}_v \) tends to unity. Taking the logarithm on both sides of Eq. (2.1) yields,

\[
\ln \hat{F}_v(n_f, \alpha, \gamma) = \gamma \ln(n_f) - \gamma \ln(a + n_f)
\]

(2.2)

For values \( n_f \ll a \), approximating \( \ln(a + n_f) \approx \ln(a) \),

\[
\ln \hat{F}_v \approx \gamma \ln(n_f) - \gamma \ln(a)
\]

(2.3)

The observations from monitoring traffic are uniform increments of \( F_v \) ranging from \{0:1\} in increments of 0.01. Let these discrete values be denoted by
For each successive increment the number of flows contributing to the traffic is denoted by $n_f(i)$. Denoting, $\tilde{n}_f(i) = \ln(n_f(i))$ and $\tilde{F}_v(i) = \ln(F_v(i))$ and using the notation $\beta = -\gamma \ln(a)$ which gives $a = \exp(-\beta/\gamma)$.

$$\tilde{F}_v(i) = \gamma \tilde{n}_f(i) + \beta \quad i = 1, 2, \ldots N_s \quad (2.4)$$

where $N_s$ is the number of observations of $\{ \tilde{F}_v(i), \tilde{n}_f(i) \}$ that are appropriate for linear approximation of the model. The value of $N_s$ was determined using $F_v(i) = 0.7$ as the cut-off. Thus approximately 70% of the traffic contributes to the transient regime of $F_v(n_f)$. The set of $N_s$ equations can be represented in matrix-vector form as,

$$\begin{bmatrix}
\tilde{F}_v(1) \\
\tilde{F}_v(2) \\
\vdots \\
\tilde{F}_v(N)
\end{bmatrix} =
\begin{bmatrix}
1 & \tilde{n}_f(1) \\
1 & \tilde{n}_f(2) \\
\vdots & \vdots \\
1 & \tilde{n}_f(N)
\end{bmatrix}
\begin{bmatrix}
\beta \\
\gamma
\end{bmatrix} \quad (2.5)$$

In compact notation Eq. (2.5) is represented as,

$$b = A\theta \quad (2.6)$$

where $\theta = [\beta \gamma]^T$ the parameter vector to be estimated. The least square error estimator is considered, where the error is

$$\epsilon = b - A\theta \quad (2.7)$$

and the error square,

$$\epsilon^T \epsilon = (b - A\theta)^T (b - A\theta) = r^2$$

$$r^2 = [b^T b - b^T A\theta - \theta^T A^T b + \theta^T A^T A \theta]$$

Setting the derivative of $r^2$ with respect to $\theta$ equal to zero, yields the solution $^{[20]}$
\[ \vartheta = (A^T A)^{-1} A^T b \]  
(2.8)

Solving Eq. (2.8), the parameters \( a \) and \( \gamma \) may be estimated. The estimated values for \( \delta t_f = 30 \) seconds are \( a = 30.79 \) and the \( \gamma = 0.411750 \).

The solution in Eq. (2.8) is valid in the region where \( n_f < a \). Figure 2.11 shows the region of validity which includes approximately 70% of the total traffic. It is observed that the model is a reasonably good fit to the observations.

![Fig 2.11: Model fit to (70%) of total traffic (\( \delta t_f=30 \) secs)](image)

If the same parameters were to be used to model the total traffic distribution, the resulting fit to the observations is as shown in Fig 2.12.
It is clear that the tail of the distribution is poorly modeled. To extrapolate the model for remainder of the traffic distribution, the parameters will need to be adjusted to minimize the error over the entire distribution. The parameter $\gamma$ which gives an estimate of the slope of the data is a good fit for the model. Thus keeping the parameter $\gamma$ the same, the scaling parameter $a$ is adjusted to minimize the error over the entire distribution. The minimum mean square error is calculated as the error of the function.

$$MMSE(a) = \frac{1}{N} \sum_{i=1}^{N} \left[ F_v(i, a, \gamma) - \hat{F}_v(i, a, \gamma) \right]^2$$

(2.9)

Using the observations $F_v$ and the approximate model estimates $\hat{F}_v$, Eq. (2.9) was evaluated for $17.97 < a < 30.22$ in increments of 0.25. A plot of the mean square error a function of $a$ is shown in Figure 2.13.
Fig 2.13: Minimum Mean Square error between the actual and the model

From this process, the mean square error was found to be minimum at \( a = 22.793 \). Using this readjusted model, the plot of the cumulative traffic bytes against the number flows is shown in Fig. 2.14 for the entire distribution. The model is seen to be considerably improved from that shown in Fig. 2.12.

Fig 2.14: Readjusted model fit.

The final process in deriving the model relating the number of flows that contribute
to a given fraction of the traffic is to model the influence of the time interval. To this end, the relationship between the parameter $a$ and $\delta t_f$ is sought. Again since the time interval influences the region of saturation more than the transient regime, $\gamma$ is assumed to be independent of $\delta t_f$. The time dependence on parameter $a$ is modeled as $\hat{a}(\delta t_f) = \alpha (\delta t_f)^\beta$ was assumed and the parameters $\alpha$ and $\beta$ were estimated using 10 values of $\delta t_f$. The minimum mean square estimation yields values of $\alpha = 13.947$ and $\beta = 0.1377$ respectively. Fig. 2.15 compares the model $\hat{a}(\delta t_f)$ and the values derived using Eqns. (2.1) and (2.9). The X-axis represents $\delta t_f$ in seconds and y-axis is estimate of $a$.

![Graph](image)

**Fig 2.15:** Regression model of $a$ vs $\delta t_f$.

Thus Eq. 2.1 can be rewritten as

$$F_v(n_f, \alpha, \beta, \gamma) = \frac{n_f'\gamma}{[n_f + \alpha (\delta t_f)^\beta]^{\gamma}}$$

(2.11)

The flow characterization was applied on the three other data sets as well. Figs. 2.16(a-c)
depict the performance of the model in comparison with the observations.

Fig 2.16a: Comparison of model and observations (Mar 01)

Fig 2.16b: Comparison of model and observation (Mar 15)

Fig 2.16c: Comparison of model and observation (Apr 19)
The parameters for the four data sets are tabulated in Table 2.7. The last column $mse$ denotes the mean square error between the cdf formed by the observations and the hypothesized model.

<table>
<thead>
<tr>
<th>day</th>
<th>$\gamma$</th>
<th>$\alpha$</th>
<th>$\beta$</th>
<th>$mse$</th>
</tr>
</thead>
<tbody>
<tr>
<td>feb16</td>
<td>0.41175</td>
<td>13.947</td>
<td>0.1377</td>
<td>0.0259</td>
</tr>
<tr>
<td>mar01</td>
<td>0.28014</td>
<td>19.332</td>
<td>0.1612</td>
<td>0.0157</td>
</tr>
<tr>
<td>mar15</td>
<td>0.37660</td>
<td>6.536</td>
<td>0.4307</td>
<td>0.0324</td>
</tr>
<tr>
<td>apr19</td>
<td>0.47356</td>
<td>25.596</td>
<td>0.1832</td>
<td>0.0427</td>
</tr>
</tbody>
</table>

**Table 2.7:** Modeling Parameters for flows (Aggregate traffic)

The modeling methodology described above can also be applied to the characterization of flows generated from specific applications. For example, the analysis of flows due to HTTP client traffic yields parameter and error values given in Table 2.8. It is observed that there is a greater degree of consistency in the parameter estimates when individual applications are considered rather than the aggregate traffic. This is particularly true for dominant applications such as HTTP. A notable feature in Table 2.8 is the deviation from the workday magnitudes of $\alpha$ and $\beta$ for mar15 which was a holiday. The flow characterization approach proposed in this section is suitable for determining the approximate size of link-state database that is to be maintained at edge routers. A quantitative model for $F_v(n_f)$ will allow its inversion for a specified fraction of traffic bytes and yield $n_f$ the number of flows contributing to this traffic for a given time scale. In the next section, the packet arrival statistics of HTTP flows generated by UML clients are examined.
Table 2.8: HTTP flow characterization parameters

<table>
<thead>
<tr>
<th>day</th>
<th>$\gamma$</th>
<th>$\alpha$</th>
<th>$\beta$</th>
<th>mse</th>
</tr>
</thead>
<tbody>
<tr>
<td>feb16</td>
<td>0.547</td>
<td>21.847</td>
<td>0.2465</td>
<td>0.0136</td>
</tr>
<tr>
<td>mar01</td>
<td>0.555</td>
<td>16.019</td>
<td>0.2816</td>
<td>0.0807</td>
</tr>
<tr>
<td>mar15</td>
<td>0.517</td>
<td>8.3803</td>
<td>0.3497</td>
<td>0.0254</td>
</tr>
<tr>
<td>apr19</td>
<td>0.579</td>
<td>26.147</td>
<td>0.2262</td>
<td>0.0335</td>
</tr>
</tbody>
</table>

2.4 Short Time-Scale Packet Arrival Statistics

In this section the focus is on characterizing the short time-scale packet arrival statistics and understanding the influence of aggregating flows. The attention is on the HTTP client traffic generated by UML subnetworks. To determine at what time scale and where traffic control is to be enforced one must examine the flow aggregation results at successively higher levels in the network hierarchy. For optimal multiplexing efficiency, the aggregation of flows must lead to a new flow that has a reduced degree of temporal dependence in its arrival rates. This feature will be examined through the magnitudes of the squared coefficient of variation of the packet inter-arrival times, also referred to as the index of dispersion for intervals. Specifying the sequence of packet interarrival times for the monitored flow as $\tau_i, i = 1, 2, \ldots$, the dependence statistics of this point process may be examined using the k-interval squared coefficient of variation defined as \[^{21}\]

$$c^2_k = \frac{k \text{Var}(S_k)}{\left[ E(S_k) \right]^2} \quad k = 1, 2, \ldots$$

where $S_k$ is the sum of $k$ consecutive interarrival times. The sequence $c^2_k = c^2_1$ if the $\{ \tau_i \}$ are generated by a renewal process. For processes with dependence $c^2_k, k > 1$ models the...
the cumulative effect of covariances of the interarrival times $^{[22]}$. 

The four representative data sets considered in the previous section are again considered in this analysis. Table 2.9 depicts for each of these days, the number of unique subnets found communicating to Internet destinations and the number of bytes and packets of data generated for each case. Except for 03/15/00 the statistics for each day are similar. Around 60 unique subnets communicate to roughly 80 destinations generating about 1.4 million packets. Each subnet generates about 140 MB of traffic within the hour. The order of magnitude decrease found for 03/15/00 is due to the fact that this day corresponded to spring break which eliminated much of the daily traffic generated from classes and dormitories.

<table>
<thead>
<tr>
<th>day</th>
<th>Packets</th>
<th>Bytes</th>
<th>#sources</th>
<th># dest</th>
</tr>
</thead>
<tbody>
<tr>
<td>feb16</td>
<td>1353286</td>
<td>139M</td>
<td>62</td>
<td>80</td>
</tr>
<tr>
<td>mar01</td>
<td>1388834</td>
<td>157M</td>
<td>61</td>
<td>78</td>
</tr>
<tr>
<td>mar15</td>
<td>540144</td>
<td>56M</td>
<td>57</td>
<td>61</td>
</tr>
<tr>
<td>apr19</td>
<td>1500607</td>
<td>150M</td>
<td>66</td>
<td>80</td>
</tr>
</tbody>
</table>

Table 2.9: Traffic Statistics

Flow aggregation is examined at three stages corresponding to the UML network, UML subnet and UML host level. The hierarchy of present flow aggregation is shown in the block diagram of Fig. 2.17.
Fig 2.17: Hierarchy of UML network.

The subnets that are dominant in generating traffic are indicated in the second tier. Each subnet corresponds to a geographically distinct location, corresponding to different academic departments, dormitories etc.. For example, subnet 153 which exists in Olsen building housing the Computer Science, Mathematics and other departments generates traffic flow aggregated from a number of individual hosts which may be workstations, servers or personal computers. Tables 2.10(a-d) depict for each day, the 10 subnets that are dominant in descending order of number of packets generated.

<table>
<thead>
<tr>
<th>subnet</th>
<th>cum_p</th>
<th>pkts</th>
<th>cum_b</th>
<th>bytes</th>
<th>iatmean</th>
<th>iatvar</th>
</tr>
</thead>
<tbody>
<tr>
<td>122</td>
<td>0.0858</td>
<td>116209</td>
<td>0.0869</td>
<td>12.1M</td>
<td>0.0309</td>
<td>0.0052</td>
</tr>
<tr>
<td>184</td>
<td>0.1677</td>
<td>110853</td>
<td>0.1681</td>
<td>11.1M</td>
<td>0.0327</td>
<td>0.0037</td>
</tr>
<tr>
<td>80</td>
<td>0.2298</td>
<td>83994</td>
<td>0.2337</td>
<td>9.0M</td>
<td>0.0428</td>
<td>0.0053</td>
</tr>
<tr>
<td>206</td>
<td>0.2855</td>
<td>75322</td>
<td>0.2916</td>
<td>8.0M</td>
<td>0.0477</td>
<td>0.0196</td>
</tr>
<tr>
<td>200</td>
<td>0.3364</td>
<td>68870</td>
<td>0.3419</td>
<td>7.2M</td>
<td>0.0522</td>
<td>0.0225</td>
</tr>
<tr>
<td>196</td>
<td>0.3866</td>
<td>68044</td>
<td>0.3899</td>
<td>6.5M</td>
<td>0.0529</td>
<td>0.0164</td>
</tr>
<tr>
<td>96</td>
<td>0.4356</td>
<td>66332</td>
<td>0.4490</td>
<td>8.3M</td>
<td>0.0547</td>
<td>0.0268</td>
</tr>
<tr>
<td>153</td>
<td>0.4829</td>
<td>63944</td>
<td>0.5078</td>
<td>8.1M</td>
<td>0.0562</td>
<td>0.0275</td>
</tr>
<tr>
<td>202</td>
<td>0.5197</td>
<td>49867</td>
<td>0.5404</td>
<td>4.5M</td>
<td>0.0721</td>
<td>0.0340</td>
</tr>
<tr>
<td>178</td>
<td>0.5519</td>
<td>43538</td>
<td>0.5635</td>
<td>3.2M</td>
<td>0.0826</td>
<td>0.0178</td>
</tr>
</tbody>
</table>
Table 2.10a: Statistics for 129.63. <sip3> .xx (Feb 16)

<table>
<thead>
<tr>
<th>subnet</th>
<th>cum_p</th>
<th>pkts</th>
<th>cum_b</th>
<th>bytes</th>
<th>iatmean</th>
<th>iatvar</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td>0.0876</td>
<td>121687</td>
<td>0.0677</td>
<td>10.6M</td>
<td>0.0291</td>
<td>0.0016</td>
</tr>
<tr>
<td>122</td>
<td>0.1727</td>
<td>118284</td>
<td>0.1617</td>
<td>14.8M</td>
<td>0.0299</td>
<td>0.0048</td>
</tr>
<tr>
<td>184</td>
<td>0.2403</td>
<td>93814</td>
<td>0.2221</td>
<td>9.5M</td>
<td>0.0377</td>
<td>0.0082</td>
</tr>
<tr>
<td>198</td>
<td>0.3008</td>
<td>84074</td>
<td>0.2632</td>
<td>6.4M</td>
<td>0.0421</td>
<td>0.0382</td>
</tr>
<tr>
<td>200</td>
<td>0.3544</td>
<td>74399</td>
<td>0.3081</td>
<td>7.0M</td>
<td>0.0476</td>
<td>0.0133</td>
</tr>
<tr>
<td>96</td>
<td>0.4046</td>
<td>69685</td>
<td>0.3555</td>
<td>7.4M</td>
<td>0.0508</td>
<td>0.0413</td>
</tr>
<tr>
<td>206</td>
<td>0.4533</td>
<td>67616</td>
<td>0.4065</td>
<td>8.0M</td>
<td>0.0524</td>
<td>0.0182</td>
</tr>
<tr>
<td>202</td>
<td>0.4951</td>
<td>58144</td>
<td>0.4435</td>
<td>5.8M</td>
<td>0.0610</td>
<td>0.0171</td>
</tr>
<tr>
<td>144</td>
<td>0.5369</td>
<td>58080</td>
<td>0.4776</td>
<td>5.3M</td>
<td>0.0651</td>
<td>0.1148</td>
</tr>
</tbody>
</table>

Aggregate pkts: 1353286
Aggregate bytes: 139 M
iat mean: 0.00266
iat var: 0.00009

Table 2.10b: Statistics for 129.63. <sip3> .xx (Mar 01)

<table>
<thead>
<tr>
<th>subnet</th>
<th>cum_p</th>
<th>pkts</th>
<th>cum_b</th>
<th>bytes</th>
<th>iatmean</th>
<th>iatvar</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td>0.2173</td>
<td>117400</td>
<td>0.1834</td>
<td>10M</td>
<td>0.0281</td>
<td>0.0070</td>
</tr>
<tr>
<td>144</td>
<td>0.2896</td>
<td>39030</td>
<td>0.2491</td>
<td>3.7M</td>
<td>0.0848</td>
<td>0.1926</td>
</tr>
<tr>
<td>153</td>
<td>0.3544</td>
<td>35031</td>
<td>0.3254</td>
<td>4.3M</td>
<td>0.0944</td>
<td>0.1309</td>
</tr>
<tr>
<td>184</td>
<td>0.4161</td>
<td>33331</td>
<td>0.3951</td>
<td>3.9M</td>
<td>0.0992</td>
<td>0.1261</td>
</tr>
<tr>
<td>114</td>
<td>0.4652</td>
<td>26522</td>
<td>0.4482</td>
<td>2.9M</td>
<td>0.1248</td>
<td>1.0310</td>
</tr>
<tr>
<td>176</td>
<td>0.5107</td>
<td>24560</td>
<td>0.5039</td>
<td>3.1M</td>
<td>0.1347</td>
<td>0.5625</td>
</tr>
<tr>
<td>152</td>
<td>0.5501</td>
<td>21305</td>
<td>0.5343</td>
<td>1.7M</td>
<td>0.1552</td>
<td>0.3657</td>
</tr>
<tr>
<td>150</td>
<td>0.5876</td>
<td>20234</td>
<td>0.5727</td>
<td>2.1M</td>
<td>0.1635</td>
<td>0.4695</td>
</tr>
<tr>
<td>123</td>
<td>0.6218</td>
<td>18461</td>
<td>0.6122</td>
<td>2.2M</td>
<td>0.1790</td>
<td>1.6679</td>
</tr>
<tr>
<td>200</td>
<td>0.6515</td>
<td>16055</td>
<td>0.6402</td>
<td>1.5M</td>
<td>0.2061</td>
<td>0.3702</td>
</tr>
</tbody>
</table>

Aggregate-Pkts: 1388834
Aggregate bytes: 157 M
iat mean: 0.0025
iat var: 0.00004

Table 2.10c: Statistics for 129.63. <sip3> .xx (Mar 15)

<table>
<thead>
<tr>
<th>subnet</th>
<th>cum_p</th>
<th>pkts</th>
<th>cum_b</th>
<th>bytes</th>
<th>iatmean</th>
<th>iatvar</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td>0.2173</td>
<td>117400</td>
<td>0.1834</td>
<td>10M</td>
<td>0.0281</td>
<td>0.0070</td>
</tr>
<tr>
<td>144</td>
<td>0.2896</td>
<td>39030</td>
<td>0.2491</td>
<td>3.7M</td>
<td>0.0848</td>
<td>0.1926</td>
</tr>
<tr>
<td>153</td>
<td>0.3544</td>
<td>35031</td>
<td>0.3254</td>
<td>4.3M</td>
<td>0.0944</td>
<td>0.1309</td>
</tr>
<tr>
<td>184</td>
<td>0.4161</td>
<td>33331</td>
<td>0.3951</td>
<td>3.9M</td>
<td>0.0992</td>
<td>0.1261</td>
</tr>
<tr>
<td>114</td>
<td>0.4652</td>
<td>26522</td>
<td>0.4482</td>
<td>2.9M</td>
<td>0.1248</td>
<td>1.0310</td>
</tr>
<tr>
<td>176</td>
<td>0.5107</td>
<td>24560</td>
<td>0.5039</td>
<td>3.1M</td>
<td>0.1347</td>
<td>0.5625</td>
</tr>
<tr>
<td>152</td>
<td>0.5501</td>
<td>21305</td>
<td>0.5343</td>
<td>1.7M</td>
<td>0.1552</td>
<td>0.3657</td>
</tr>
<tr>
<td>150</td>
<td>0.5876</td>
<td>20234</td>
<td>0.5727</td>
<td>2.1M</td>
<td>0.1635</td>
<td>0.4695</td>
</tr>
<tr>
<td>123</td>
<td>0.6218</td>
<td>18461</td>
<td>0.6122</td>
<td>2.2M</td>
<td>0.1790</td>
<td>1.6679</td>
</tr>
<tr>
<td>200</td>
<td>0.6515</td>
<td>16055</td>
<td>0.6402</td>
<td>1.5M</td>
<td>0.2061</td>
<td>0.3702</td>
</tr>
</tbody>
</table>

Aggregate-Pkts: 540144
Aggregate-bytes: 56 M
iat mean: 0.00612
iat var: 0.00012

Table 2.10c: Statistics for 129.63. <sip3> .xx (Mar 15)
<table>
<thead>
<tr>
<th>subnet</th>
<th>cum_p</th>
<th>pkts</th>
<th>cum_b</th>
<th>bytes</th>
<th>iatmean</th>
<th>iatvar</th>
</tr>
</thead>
<tbody>
<tr>
<td>202</td>
<td>0.0774</td>
<td>116213</td>
<td>0.0690</td>
<td>10.4M</td>
<td>0.0295</td>
<td>0.0035</td>
</tr>
<tr>
<td>80</td>
<td>0.1536</td>
<td>114311</td>
<td>0.1371</td>
<td>10.2M</td>
<td>0.0300</td>
<td>0.0013</td>
</tr>
<tr>
<td>122</td>
<td>0.2283</td>
<td>112160</td>
<td>0.2202</td>
<td>12.5M</td>
<td>0.0306</td>
<td>0.0036</td>
</tr>
<tr>
<td>184</td>
<td>0.2997</td>
<td>107147</td>
<td>0.2956</td>
<td>11.3M</td>
<td>0.0320</td>
<td>0.0035</td>
</tr>
<tr>
<td>200</td>
<td>0.3517</td>
<td>78053</td>
<td>0.3491</td>
<td>8.0M</td>
<td>0.0440</td>
<td>0.0054</td>
</tr>
<tr>
<td>198</td>
<td>0.4037</td>
<td>77934</td>
<td>0.3852</td>
<td>5.4M</td>
<td>0.0525</td>
<td>0.0173</td>
</tr>
<tr>
<td>178</td>
<td>0.4472</td>
<td>65403</td>
<td>0.4279</td>
<td>6.4M</td>
<td>0.0560</td>
<td>0.0175</td>
</tr>
<tr>
<td>210</td>
<td>0.4881</td>
<td>61324</td>
<td>0.4687</td>
<td>6.1M</td>
<td>0.0592</td>
<td>0.0286</td>
</tr>
<tr>
<td>192</td>
<td>0.5268</td>
<td>58017</td>
<td>0.5124</td>
<td>6.5M</td>
<td>0.0604</td>
<td>0.0117</td>
</tr>
</tbody>
</table>

| Aggregate pkts | 1500607 |
| Aggregate bytes | 150 M  |
| iat_mean        | 0.00289 |
| iat_var         | 0.00001 |

**Table 2.10d:** Statistics for 129.63. < sip3 > . xx (Apr 19)

Column two indicates the cumulative count of packets generated from these local area networks. Column five specifies the number of megabytes carried by the traffic. The mean and variance of the inter-arrival times are given in columns six and seven respectively. Also indicated for each day are the aggregate count of the packets and bytes generated as well as the mean and variance of the aggregate traffic inter-arrival times. Here aggregate refers to HTTP flows generated by the UML network. It can be observed that a specific set of subnets characteristically generate dominant traffic flows for all days considered. These are for example subnets 122, 80, 184, 200 and 153. It is also observed that as the traffic count representing particular subnets decrease, there is a corresponding increase in the mean packet interarrival time and its variance. This suggests that the intrinsic burstiness of the flows increases with the progressive sparseness of the traffic arrival.
rates and not due to instantaneous transfer of big bursts of data.

The k-interval coefficient of variation $c_k^2$ was calculated for the one-hour of interarrival times for each of the four days considered. First the features at the host level were examined. The results are discussed with respect to subnet 153. For each trace, the hosts in subnet 153 were sorted in descending order using the number of packets generated as the criteria. Approximately 3 – 6 number of hosts contributed to about 50% of the total packet count. Table 2.11(a-d) depicts the mean and variance of the interarrival times for each of the hosts that make up the dominant component of subnet 153. The average interarrival time is on the order of 0.5 seconds with relatively large magnitudes of variance. The time-scales of analysis at the host level therefore corresponds to an average range 0.25 – 25 seconds.

<table>
<thead>
<tr>
<th>host</th>
<th>iatmean</th>
<th>iatvar</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>0.2333</td>
<td>8.957</td>
</tr>
<tr>
<td>25</td>
<td>0.5190</td>
<td>128.16</td>
</tr>
<tr>
<td>26</td>
<td>0.7201</td>
<td>411.52</td>
</tr>
<tr>
<td>27</td>
<td>0.3324</td>
<td>17.233</td>
</tr>
</tbody>
</table>

**Table 2.11a:** Mean and Variance 129.63.153. <sip4> (Feb 16)

<table>
<thead>
<tr>
<th>host</th>
<th>iatmean</th>
<th>iatvar</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>0.4020</td>
<td>18.36</td>
</tr>
<tr>
<td>14</td>
<td>0.7325</td>
<td>391.71</td>
</tr>
<tr>
<td>1</td>
<td>0.3175</td>
<td>2.77</td>
</tr>
<tr>
<td>40</td>
<td>1.2228</td>
<td>26.59</td>
</tr>
<tr>
<td>41</td>
<td>0.7403</td>
<td>13.76</td>
</tr>
<tr>
<td>24</td>
<td>1.3407</td>
<td>1069.35</td>
</tr>
</tbody>
</table>

**Table 2.11b:** Mean and Variance 129.63.153. <sip4> (Mar 01)
Table 2.11c: Mean and Variance <sip4> (Mar15)

<table>
<thead>
<tr>
<th>host</th>
<th>iatmean</th>
<th>iatvar</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.4266</td>
<td>10.30</td>
</tr>
<tr>
<td>3</td>
<td>0.6182</td>
<td>12.26</td>
</tr>
<tr>
<td>9</td>
<td>0.6286</td>
<td>33.06</td>
</tr>
</tbody>
</table>

Table 2.11d: Mean and Variance <sip4> (Apr19)

<table>
<thead>
<tr>
<th>host</th>
<th>iatmean</th>
<th>iatvar</th>
</tr>
</thead>
<tbody>
<tr>
<td>24</td>
<td>0.4322</td>
<td>6.47</td>
</tr>
<tr>
<td>9</td>
<td>0.5758</td>
<td>90.79</td>
</tr>
<tr>
<td>15</td>
<td>0.2002</td>
<td>0.912</td>
</tr>
<tr>
<td>90</td>
<td>0.4786</td>
<td>38.33</td>
</tr>
<tr>
<td>1</td>
<td>0.7823</td>
<td>216.93</td>
</tr>
<tr>
<td>21</td>
<td>0.3940</td>
<td>7.152</td>
</tr>
</tbody>
</table>

For each of these hosts $c_k^2$ was evaluated for $k = 1, 2, \ldots, 50$. Figs 2.18(a-d) show the results for the four traces. The vertical axis denotes the magnitude of $c_k^2$ and the horizontal axis denotes $k$ the number of intervals aggregated. The results for normal working days in Figs. 2.17(a,b,d) show that for the most part the interarrival times may be characterized as renewal processes. A renewal process is a point process where the interarrivals are independent identically distributed (iid). Although the magnitude of the coefficient of variation is significantly high suggesting a large variance and bursty nature of the arrival times. In contrast the results for trace 03/15/00 depicted in Fig. 2.18(c) show non-renewal features wherein the $c_k^2$ exhibits increasing magnitudes with $k$. 
Fig 2.18a: COV for hosts (feb16)

Fig 2.18b: COV for hosts (mar01)

Fig 2.18c: COV for hosts (mar15)

Fig 2.18d: COV for hosts (apr19)
Next the analysis is repeated at the subnet level, where the coefficient of variation was calculated for the host-aggregate flow generated by each subnet. The results for five of the subnets 80, 122, 184, 153, 200 that are common in all of the traces are depicted in Figs. 2.19(a-d).

Fig 2.19a: COV for subnets (feb16)

Fig 2.19b: COV for subnets (mar01)

Fig 2.19c: COV for subnets (mar15)
It is seen that the process of aggregating flows from several hosts has considerably reduced the magnitude of coefficient of variation, although the process has also induced non-renewal properties in the subnet traffic. The $c_k^2$ range from 1 to about 40 for most of the subnets, except for subnet 122 on 03/15/00. These magnitudes may be compared to values of 100 to 800 seen for flows from individual hosts. Referring to Tables 2.10 ($a - d$) the average inter-arrival time at the subnet level is on the order of $10^{-2}$ seconds. The calculation of the $c_k^2$ therefore corresponds to average time durations ranging from $10^{-2}$ to 0.5 seconds. The dependence features of the interarrival times are very much evident even at these smaller time-scales. According to the central limit theorem if there were sufficient samples for the host level, the coefficient of variation would converge to a steady state value. However due to insufficient number of hosts the convergence in the coefficient of variation is not observed.

Finally, the behavior of the HTTP flows aggregated from all subnets is considered. Fig 2.20 shows the trend in the k-interval coefficient of variation for the four traces. The
results for three normal workdays show approximately renewal process behavior. The result for 03/15/00 continues to exhibit dependence features even at this smaller time scale. Referring to the average interarrival times noted in Tables 2.10 (a-d) for the four days, the range of time-scales for Fig. 2.19 correspond to $10^{-3}$ to 0.1 second.

![Figure 2.20: COV at Aggregate level](image)

It is observed that the coefficient of variation at the host level is high, in the order of 100-800 compared to coefficient of variation at the subnet level. The coefficient of variation at the network level is the least. The aggregation of large number of subnet flows has resulted in a decrease in the variability at the network level. This feature indicates that the predictability of flows and their control may be carried out most efficiently at the network level, rather than at the subnet level.
2.5 SUMMARY

A visualization study of some of long-term traffic features that characterize Internet related traffic and statistical analysis of flow and packet time-scale features have been presented. The hourly variation in the traffic between UML and the Internet shows several invariant features that may be useful for development of traffic and performance models. The visualization of upstream and downstream traffic shows that the TCP protocol is the dominant traffic generator at all times of the day. Among the TCP applications, HTTP, FTP, and SMTP generate over seventy percent of the hourly byte and packet volume. The distribution of packet size over time is also found to be a predictable component for both inbound and outbound directions. This study also examined the relationship between applications and the packet sizes they generate. A strong correlation is shown to exist between the dominant applications in each direction and the packet size. On an hourly time scale, over 90% of traffic in the outbound direction is in the 0-100 byte range and is generated by client traffic from web and email related applications. In the inbound direction, over 75% of the traffic is in the 576 and 1500 byte range and arise from server responses to the outbound requests. These dependent relationships between packet size and applications are found to be consistent features in the day to day patterns of bi-directional traffic between the campus network and the Internet.

A flow characterization study was undertaken and a quantitative model for the cumulative byte count as a function of number of flows was derived. Parametric invariance
was shown in four days of traces when HTTP flows alone were considered, rather than
the mix of all applications. The use of this model for determining the number of flow
states as a function of time duration was discussed. Finally the dependence features of
HTTP packet interarrival times were analyzed using the k-interval coefficient of variation
metric. It was observed that the aggregation of flows at the network level leads to a re-
newal process behavior at short time scales ranging from 1 to 100 milliseconds. However
the aggregation of flows at the subnet level did not yield such a feature. This may be at-
tributed to the sparseness of the number of host level flows, typically in the 10 – 20 range.
The host level flow characteristics were shown to exhibit approximate renewal features
but had a high interarrival time variance.
Chapter 3

Recursive Estimation of Packet Count statistics

3.1 Introduction

One of the primary goals of the differentiated services proposal is to develop aggregation and resource allocation strategies for flows at the edge of the network. The analysis of traffic from Chapter 2 shows that HTTP traffic is dominant throughout the day. Typically over 77% bytes and packets account for HTTP traffic. HTTP flow characterization parameters obtained in chapter 2 showed that the relationship between number of flows and the amount of contribution to the total bytes, remained nearly invariant for the workday traces. It was also shown that at the packet arrival time scale the aggregate traffic exhibited close to renewal behaviour. To this end modelling of aggregated HTTP flows is considered. This traffic is extracted by identifying all packets having a source IP address equal to 129.63.xx.xx with destination port 80. In recent work, Olowoyeye [17] showed that the HTTP packet-count time-series exhibited a time-varying mean and variance. These statistics change on a time-scale on the order of several seconds. These variations result in the long-range correlation features observed in the packet-count time-series. The dynamics at these time-scales are important to network management and routing functions. These processes typically operate at these longer time-scales allowing stable operating points. To facilitate the modeling of these time-varying features, recursive estima-
tors for the mean and variance of the arrival process are designed to permit online estimation of the mean packet-rate of HTTP traffic. Estimation results show that both finite state Markov chains and first order autoregressive processes can adequately model the observed time variations. The parametric models of HTTP arrival rate are used in the design of a Kalman state estimator for prediction of traffic and buffer states in a queue.

3.2 Traffic Statistics: Packet count Mean and Variance

The first two moments of the arrival process strongly impact the queuing delays and losses. They are also important in application of admission control, for assessing multiplexing efficiency and allocating capacity on an outgoing link. The operating load or link utilization $\rho$ is determined from the ratio of the average arrival rate $\lambda$ and average service rate $C$. For finite capacity links $C$ represents a constant service rate and can be calculated for a required utilization, $\rho$, as $C = \lambda / \rho$. The provision of capacity is made under the assumption that the mean arrival rate or load on the system is a constant during the hourly time scale. If however the mean changes on a faster time scale congestion levels higher than predicted would be observed. For arrival and service processes that are not Poisson, the variance of the distribution also plays an important role in determining queue statistics. In particular for $M/G/1$ queues with Markovian input and general service times the average queuing delay is proportional to the variance of the service times $^{[23]}$.

To accurately estimate changes in the first and second moments a recursive estima-
tion procedure is developed. The packet arrivals are first aggregated into a time series. A time series model specifies the number of packets that arrive within successive time intervals of a selected duration $\Delta t$. While the conversion to a time series is not necessary it is useful for later discussion on prediction where time discretization is required. Thus the packet count time series $y_k$ represents the number of packets arriving in a time interval $(k-1)\Delta t < t \leq k\Delta t$. Olowoyeye [17] has shown that an interval of $\Delta t = 1$ second captures the important temporal dynamics of HTTP traffic. The time series for the four days 02/16/00, 03/01/00, 03/15/00, and 04/19/00 aggregated over one second is shown in Figs. 3.1(a-d). The horizontal axis is the time in seconds, the vertical axis is the packet count $y_k$. 
Fig 3.1a: Time Series (feb 16)

Fig 3.1b: Time Series (mar 01)

Fig 3.1c: Time Series (mar 15)

Fig 3.1d: Time Series (apr 19)
3.3 Recursive Estimation of the Mean and Variance

Let $y_k$ represent the number of HTTP packets arriving in the $k^{th}$ time interval. The standard approach of using the arithmetic mean as an estimate yields for example a value of 392 packets for the time-series having time duration of 21600 seconds for the 04/19/00 data set. The expression for the arithmetic mean may be rearranged to yield a recursive algorithm for updating the mean with arrival of new information. The derivation of this expression is presented in Appendix II. If $\hat{a}_k$ represents the estimate of the mean $a_k$ using $k$ observations then

$$\hat{a}_k = \hat{a}_{k-1} - \frac{P_{k-1}}{1 + P_{k-1}} [\hat{a}_{k-1} - y_k]$$  \hspace{1cm} (3.1a)

where $P_k$ represent the weighting coefficients which are updated as,

$$P_k = \frac{P_{k-1}^2}{1.0 + P_{k-1}}$$  \hspace{1cm} (3.1b)

Eq. (3.1) is equivalent to the expression for the sample mean and in each new estimate includes the information from all of the previous samples. To determine if the mean is changing with time, Eqns. (3.1) can be modified by shaping the memory of the estimator. In this case, an exponential window is used to weight the past samples, so that the estimator has a finite memory. As derived in Appendix II, the resulting equations for the estimate and the weighting coefficients at the $k^{th}$ sample are
\[
\hat{a}_k = \hat{a}_{k-1} - \frac{P_{k-1}}{\alpha + P_{k-1}} \left[ \hat{a}_{k-1} - y_k \right]
\]  
\[P_k = \left[ \frac{P_{k-1}}{\alpha + P_{k-1}} \right]\]

(3.2a)

(3.2b)

where \(0 < \alpha < 1\) represents the effect of exponential windowing and may be obtained by matching to the desired exponential time constant \(T_e\) as \(\alpha = e^{-\frac{T_s}{T_e}}\) where \(T_s\) is the sampling interval. \(T_s\) is set to one second in this analysis. Both Eqns. (3.1) and (3.2) offer a recursive approach for estimating the sample mean of the observed process. The term \(\varepsilon_k = \hat{a}_{k-1} - y_k\) gives the residual error in the estimation.

The optimum value of \(\alpha\) both minimizes the error variance and leads to uncorrelated time series of \(\varepsilon\). The hypothesis is that long-range correlation in the autocovariance function of \(y_k\) is due to slow variation in the mean and possibly the variance of \(y_k\). The selection of appropriate value of \(\alpha\) should serve to filter out the mean trend at the characteristic time scale \(\Delta t\). The residual process is obtained by subtracting the time-varying mean from \(y_k\) and should be minimally correlated. This feature will be used to select the magnitude of \(\alpha\).

Fig. 3.2 depicts the result of using Eqns. 3.2 and result of applying Eqns. 3.1 to estimate the mean recursively. The horizontal axis is the time in seconds. The vertical axis depicts the estimate \(\hat{a}_k\) obtained by applying Eqns. 3.1. The horizontal line is the mean computed using all samples. The solid line shows the mean calculated using Eqn 3.1. The estimated mean at each observation instant clearly shows a trend with values that
show significant variation about the final estimate.

![Graph with time and mean values]

**Fig 3.2:** Recursive estimate of mean under stationary assumption (08/19/00).

The estimate of the mean using exponential windowing with Eqns. (3.2) is considered next. First a range of values of $0.9 < \alpha < 0.99$ were considered in increments of 0.0063. For each such value of $\alpha$, the residual errors $\varepsilon_k(\alpha)$ was determined. The errors are zero-mean processes but typically have some degree of residual correlation due to the short time-scale dynamics.

The normalized autocovariance function (NACF) $R_l(\varepsilon(\alpha))$ was calculated for each $\varepsilon(\alpha)$ and the criteria for selecting $\alpha$ was based on minimizing the sum $S_\varepsilon(\alpha) = \sum_{l=2}^{100} R_l(\varepsilon(\alpha))$. These results are tabulated in Table 3.3. Too large a value of $\alpha$ leads to positive correlations in the NACF of the residual error whereas too small a value
of $\alpha$ leads to negative correlations. For values close to unity, the residuals exhibit strong positive correlations, since the effective window is larger than the time-scale at which the mean varies. For, $\alpha$ too small, the effective window is short and serves to act as a moving average filter of random variations in the residuals. The initial conditions $\hat{a}_1 = y_1$ and $p_1 = 1$ were considered in applying the recursive algorithm.

<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>$S_e(\alpha)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.9013333</td>
<td>-0.4792018</td>
</tr>
<tr>
<td>0.9076667</td>
<td>-0.4740541</td>
</tr>
<tr>
<td>0.9140000</td>
<td>-0.4670250</td>
</tr>
<tr>
<td>0.9203333</td>
<td>-0.4573216</td>
</tr>
<tr>
<td>0.9266667</td>
<td>-0.4437207</td>
</tr>
<tr>
<td>0.9330000</td>
<td>-0.4243196</td>
</tr>
<tr>
<td>0.9393333</td>
<td>-0.3960580</td>
</tr>
<tr>
<td>0.9456667</td>
<td>-0.3539028</td>
</tr>
<tr>
<td>0.9520000</td>
<td>-0.2893358</td>
</tr>
<tr>
<td>0.9583334</td>
<td>-0.1875905</td>
</tr>
<tr>
<td>0.9646667</td>
<td>-2.2316054E-2</td>
</tr>
<tr>
<td>0.9710000</td>
<td>0.2550669</td>
</tr>
<tr>
<td>0.9773334</td>
<td>0.7383472</td>
</tr>
<tr>
<td>0.9836667</td>
<td>1.623114</td>
</tr>
<tr>
<td>0.9900000</td>
<td>3.524564</td>
</tr>
</tbody>
</table>

**Table 3.1:** Mean and Variance of the residual error (04/19/00).

It is observed from Table 3.1 that the NACF sum crosses the zero axis for values of $\alpha$ between 0.964 and 0.971. The minimum was found by bracketing $\alpha$ between these values and using a step size of 0.0007. The optimim value of $\alpha$ that minimized the correlations was 0.9647. For this value of $\alpha$ the effective time constant can be calculated using,

$$T_e = -\frac{T_s}{\ln(\alpha)}$$
For $T_s = 1$ second and $\alpha = 0.9647$, $T_e = 27.82$ seconds as the characteristic time scale for variation of the mean. The estimate of the characteristic time scale here is close to the 30 second value qualitatively obtained by Olowoyeye [17]. The result of applying the algorithm using Eqns. (3.2) and $\alpha = 0.9647$ is shown in Figs 3.3a. Figs. 3.3(b-d) show the estimate of the mean packet count process calculated recursively. The horizontal axis is the time in seconds, the vertical axis is the mean packet count. Comparison of Fig. 3.2 and Fig. 3.3a shows the vast difference in the estimator under the windowing operation. The procedure of exponential windowing is seen to effectively track the average drift in the HTTP packet count process that characterizes the original packet count time series in Fig. 3.1.

![Fig 3.3a: Recursive estimate with exponential windowing (04/19/00).](image)
Fig 3.3b: Recursive estimate with exponential windowing (02/16/00).

Fig 3.3c: Recursive estimate with exponential windowing (03/01/00).

Fig 3.3d: Recursive estimate with exponential windowing (03/15/00).

The residual error process for $\alpha = 0.9647$ is plotted in Fig. 3.4(a). Also plotted in Fig 3.4(b) is the cumulative distribution function (cdf) of the errors, and the cdf of a hy-
pothesized Gaussian distribution.

**Fig 3.4a:** Residual error of the recursive mean process

**Fig 3.4b:** CDF of the residual and Gaussian processes.
The residual errors are tested for Gaussian hypothesis, using the Kolmogorov-Smirnov (K-S) test\textsuperscript{[25]}. The K-S test, provides a test for the null hypothesis that the error residuals are Gaussian distributed. The maximum difference between the two distributions is $D = 0.016$ and this value, the D-statistic is the criteria used for the K-S test. For a class size of 30, the value of $D$ is found to be less than the critical value $D_c = 0.19$ at the 0.20 significance level. Therefore the null hypothesis that the residuals are Gaussian can be accepted. The mean and variance of the residual process were estimated to be 0.02432 and 2826 respectively.

![Fig 3.5: NACF of the residual error.](image)

The NACF of the residual errors is shown in Fig. 3.5. The horizontal axis represents the lags and the vertical axis is the NACF. It is observed that the residuals retain
correlations that arise from the short-time scale packet dynamics. Olowoyeye [17] modeled the combined short and long-time scale correlations using a finite state Markov chain for the mean variation and a state dependent autoregressive process for the short time scales. The Markov model for the mean function was obtained by partitioning the exponentially smoothed time-series into a finite number of states using a rate quantization process. Use of the Markov chain to modulate the faster varying autoregressive model yielded a good approximation of the traffic dynamics. This was verified by comparison of the queueing delays and losses in finite and infinite buffer queues respectively. The influence of the long-time scale mean variation is further examined here.

The recursively estimated mean function was first aggregated on a thirty second time scale using 30 samples of the estimated mean. The resulting mean estimate \( \hat{\mu} \) aggregated over 30 seconds is shown in Fig. 3.6.
The aggregated estimate on a 30 second time scale consists of 726 samples. The result shows values of the aggregated mean estimate both below and above the sample average value (which is shown by the dotted line). This variation in the process causes increased queuing delays and losses relative to a process with a constant mean value. The variation in $\hat{a}$ ranges in a continuous manner between values 6000 – 15000. This process is quantized to approximate the variations by a finite set of levels. A quantization level having a step size of 1000 packets is used. The quantization typically yielded 8-10 states for the discrete time Markov chain. The number of unique amplitudes equals the number of states characterizing the Markov chain. For the trace measured on 04/19/00 10 states were used in the model. Given the quantized process, the transition probabilities are estimated from the data, using the relation,
The Markov probability transition matrix was estimated from the data and scaled by 0.03 to correspond to a one-second time scale. For example, for the trace measured on 04/19/00, the empirically estimated probability transition matrix was found to be,

\[
P = \begin{bmatrix}
0.97 & 0 & 0.03 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.03 & 0.97 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0.97 & 0 & 0.03 & 0 & 0 & 0 & 0 & 0 \\
0 & 0.01 & 0 & 0.983 & 0.007 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0.02 & 0.987 & 0.001 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0.004 & 0.987 & 0.008 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0.009 & 0.988 & 0.14 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0.006 & 0.007 & 0.989 & 0.002 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.996 & 0.004 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.01 & 0.99 & 0 \\
\end{bmatrix}
\]

with a corresponding rate vector

\[
R = \begin{bmatrix}
200 \\
233.33 \\
266.66 \\
300 \\
333.33 \\
366.66 \\
400 \\
433.33 \\
466.66 \\
500 \\
\end{bmatrix}
\]

resulting in a mean arrival rate ranging from 200 to 500 packets per second. The diagonally dominant probabilities indicate a high degree of correlation in the mean process.

The influence of time-variation of the mean process on the delay distribution is ex-
A single server infinite buffer queue is considered. The queue is serviced at a constant service rate of \( C \) packets per second. The number of packets waiting for service or the buffer occupancy is represented by an integer random variable \( B \) with range \( 0 \leq b < \infty \). The HTTP packet arrival process consists of a sequence of measured interarrival times. The probability density function (pdf) \( Pr(B=b), \ b = 0, 1, \ldots \) is estimated through simulation. Approximately 8.5 million packets were generated during the six hour duration (12:00 - 18:00). The pdf was summed to obtain the cumulative distribution and its complementary value \( Pr \left( B > b \right) \). This result is compared with the buffer occupancy distribution generated by the finite state Markov chain representing the mean process. Here a fluid buffer analysis \cite{26} was used to numerically estimate the buffer occupancy statistics. Figs. 3.6(a-d) depict a comparison of the simulated buffer occupancy distribution generated by the interarrival times and that generated by the Markov chain model of the mean variation. A plot of the complimentary distribution function using the fluid buffer analysis and a simulation of the delay using a infinite buffer queue is shown in Figs 3.7 (a-d). The horizontal axis represents the packet delays \( \tau_b \) in milliseconds obtained by scaling the buffer occupancy variable \( b \) by the packet service time \( 1/C \).
Fig 3.7a: ICDF (apr 19).
Fig 3.7b: ICDF (feb 16).
Fig 3.7c: ICDF (mar 01).
Fig 3.7d: ICDF (mar 15).
The simulation results show that the complementary delay distribution is characterized by two asymptotes, one in the limit as \( \tau_b \to 0 \) and the other as \( \tau_b \to \infty \). The asymptote in the limit of large delays exhibits approximately the same slope as produced by the Markov chain. The influence of long-range correlations in the mean value of the arrival process is therefore observed in the large buffer dynamics of the queue. If one is to control the queueing performance, it is this feature that must be addressed. The next chapter considers methods for using the recursively estimated mean function for estimating buffer occupancy and for controlling queue lengths at routers using adaptive routing.
Chapter 4

Application of Kalman Estimators for Adaptive Routing

4.1 Introduction

The analysis of traffic in Chapter 3 showed that the HTTP client packet arrival process exhibits a time-varying mean function. The mean value exhibited changes on timescales of $10^{-30}$ seconds. This time-scale can be regarded as a long-time scale relative to the packet inter-arrival times which typically occur on the millisecond time-scale. The sojourn of the mean process at different amplitude levels has been shown to be modeled using discrete time Markov chains or first-order autoregressive processes. Network design and capacity allocation generally take into account time-dependent usage patterns at the longer hourly time-scales. In this process, the network is engineered to sustain the load generated during peak hours of usage. The analysis of Internet traffic measurements in this work shows that there is a significant variation in the average arrival rate at shorter time-scales which causes the links to experience higher than expected utilization levels for a relatively long duration of time over which many packets may experience congestion. To alleviate congested links due to nonstationarity in the mean packet arrival rate, adaptive routing may be considered. This will require that one track and predict the mean level expected ahead of time and take control actions to minimize congestion. The approach proposed here formulates the problem using a state-space approach. In particular,
the states of each network node will represent the expected arrival rate and buffer occupancy. The system dynamics are obtained using the prior knowledge of the model for the mean arrival rate and its parameters. The potential for congestion is determined by evaluating the buffer occupancy state.

Section 4.2 develops the state space model. Models for the mean arrival process and the buffer occupancy are given. In section 4.3 the Kalman filter is implemented for state estimation. The application of the one-step ahead estimation of the mean arrival rate and buffer occupancy in adaptive routing is discussed in section 4.4.

### 4.2 State Space Model

The average arrival rate $A_i$ at node $i$, during time interval $k$ is given by

$$A_i(k) = \alpha_i A_i(k - 1) + \mu_i (1 - \alpha_i) + g_i(k)$$

where $A_i(k)$ represents a first order autoregressive process (AR) with a mean $\mu_i$, the AR coefficient $\alpha_i$, where $0 < \alpha_i < 1$ and $g_i$ represents an iid Gaussian noise process. The noise process has a zero mean and a variance denoted by $\sigma_i^2$. The nodes are assumed to be modeled by infinite buffer queues. In this case, the buffer occupancy or number of packets queued at time $k$ is
\[ B_i(k) = (B_i(k - 1) - C_{ij})^+ + A_i(k) \]  
\[ = B_i(k - 1) - C_{ij} + A_i(k) \quad B_i(k - 1) > C_{ij} \]  
\[ = A_i(k) \quad B_i(k - 1) \leq C_{ij} \]  
where \( C_{ij} \) is the capacity of the link between nodes \( i \) and \( j \). Substituting Eqn. 4.1 in 4.2

\[ B_i(k) = B_i(k - 1) + \alpha_i A_i(k - 1) + \mu_i (1 - \alpha_i) - C_{ij} + g_i(k) \]  
\[ A_i(k) = \alpha_i A_i(k - 1) + \mu_i (1 - \alpha_i) + g_i(k) \]  
Eqns 4.3 and 4.4 can be represented in vector or state space form using a state vector \(^{27}\) as

\[ X(k) = \begin{bmatrix} B(k) \\ A(k) \end{bmatrix} \]

The indices are ignored in the following development of the model. Denoting \( \hat{\mu} = \mu (1 - \alpha) \),

\[ X(k) = \begin{bmatrix} 1 & \alpha \\ 0 & \alpha \end{bmatrix} X(k - 1) + \begin{bmatrix} \hat{\mu} - C \\ \hat{\mu} (1 - \alpha) \end{bmatrix} + \begin{bmatrix} g(k) \\ g(k) \end{bmatrix} \quad B(k - 1) > C \]  
\[ X(k) = \begin{bmatrix} 0 & \alpha \\ 0 & \alpha \end{bmatrix} X(k - 1) + \begin{bmatrix} \hat{\mu} \\ \hat{\mu} \end{bmatrix} + \begin{bmatrix} g(k) \\ g(k) \end{bmatrix} \quad B(k - 1) \leq C \]

Eqns. 4.5 and 4.6 represent a system with time varying state matrices. They can be represented in general as,

\[ X(k) = T_k X(k - 1) + C_k + G_k g(k) \]  
where \( X(k) \) is a \( m \times 1 \) vector, \( T_k \) is a \( m \times m \) matrix, \( \alpha_{t-1} \) is a \( m \times 1 \) vector, \( C_t \) is a \( m \times 1 \) vector, and \( G(k) \) is a \( 1 \times 1 \) vector and \( m = 2 \) refers to the number of state variables. Eqn
4.7 is referred to as the transition equation. The noise process is characterized by a covariance matrix, $Q_k$. The initial state vector $X(0)$, has a mean of $a_0$ and a covariance matrix $P_0$. In addition to the transition equation, an $m \times 1$ observation vector given by $\tilde{y}$ is modeled as,

$$\tilde{y} = ZX + d + \varepsilon$$  \hspace{1cm} (4.8)

where it is assumed in this work that $Z$ is an identity matrix, $d = 0$ and $\varepsilon$ is an $m \times 1$ vector of uncorrelated disturbances of zero mean and covariance matrix $H_k$. Eqn. 4.8 is referred as the measurement equation. The matrices $Z$, $d$, $H_k$, $T_k$, $C_k$, and $Q_k$ are called the system matrices. Given the state transition and measurement equations, the Kalman filter can be applied to derive optimal estimates of the state variables.
4.3 Kalman Filter Estimation

The Kalman filter is a recursive procedure for computing the optimal estimator of the state vector at time $k$, based on information available at time $k^{[23]}$. This information consists of observations up to and including $y_k$, the new observation. Once a model has been put in the state space form, the Kalman filter may be applied and this leads to algorithms for prediction and smoothing $^{[4]}$. Let $X(k-1)$ denote the optimal estimator of $X(k-1)$ based on observations up to and including $y_{k-1}$. Let $P_{k-1}$ denote the $m \times m$ covariance matrix of the estimation error. This is given by

$$P_{k-1} = E[(X(k-1) - \hat{X}(k-1))(X(k-1) - \hat{X}(k-1))^T]$$

Thus given $X(k-1)$ and $P_{k-1}$ the optimal estimator at the instant $k$ of the state vector $X(k)$ is given by

$$\hat{X}_{k|k-1} = T_k \hat{X}_{k-1} + C_k$$

The covariance matrix of the estimation error is

$$P_{k|k-1} = T_k P_{k-1} T_k^T + GQG'$$

where $G$ and $Q$ are time independent. The notation $G'$ indicates the transpose of $G$. Eqns 4.9 and 4.10 are known as the prediction equations. Once the new observation $y_k$ becomes available, the updating equations are

$$\hat{X}_k = \hat{X}_{k|k-1} + P_{k|k-1} Z_k F_k^{-1}(y_k - Z_k \hat{X}_{k|k-1} - d_k)$$

and

$$P_k = P_{k|k-1} - P_{k|k-1} Z_k F_k^{-1} Z_k P_{k|k-1}$$
where

\[ F_k = Z_k P_{k|k-1} Z_k' + H \]

again assuming the time independence of the noise process. The prediction and update equations constitute a recursive least squares algorithm that aims to minimize the estimation error. Using Eqns 4.9 - 4.12 the mean packet count arrival process can be predicted.

In the simulation results described below, the initial conditions for evaluating the algorithm are given by

\[ R_t = \begin{bmatrix} 1 \\ 1 \end{bmatrix}; Q_t = \begin{bmatrix} 2825 & 0 \\ 0 & 2825 \end{bmatrix}; H_t = \begin{bmatrix} 4430 \\ 4430 \end{bmatrix}; a_0 = \begin{bmatrix} 550 \\ 350 \end{bmatrix}; p_0 = \begin{bmatrix} 1500 & 700 \\ 700 & 1500 \end{bmatrix} \]

Using Eqns. 4.11 and 4.12, the mean packet count process \( A_k \) and the buffer occupancy \( B_k \) are estimated.

Fig 4.1 shows the estimated mean packet count process \( A_k \). The horizontal axis is the time \( k \) in seconds and the vertical axis are the estimator of the packet count process \( \hat{A}_k \). This may be compared to the measurement time series shown in Fig 3.1d.

Fig 4.2 shows the estimated buffer occupancy \( \hat{B}_k \) using the Kalman Filter. The accuracy between the exact and estimated buffer occupancy can be viewed using a complementary distribution function of the buffer occupancy. A comparison of the exact and the estimated distribution function is shown in Fig. 4.3. The horizontal axis is the buffer occupancy and the vertical axis is the log \( pr[B > b] \). The estimator is seen to accurately capture this queue statistic.
Fig 4.1: Estimated packet count process using Kalman Process

Fig 4.2: Estimated buffer occupancy using Kalman Process
Fig 4.3: Complimentary distribution function from Kalman estimator and data

Fig 4.4: Error between the model and the estimator for buffer occupancy

Fig 4.4 shows the sequence of the errors between the actual $B_k$ and the estimated $\hat{B}_k$. The Kalman estimator is an unbiased estimator since the expectation of the estimation error is zero. Having shown that the Kalman estimator performs well for estimating the
mean arrival rate and the buffer occupancy, this approach is applied to an adaptive routing problem.

4.4 Adaptive routing

A 3-node network as shown in Fig 4.5 is considered.

Let $A_i(k)$, $i=1,2$ denote the mean arrival rate process at nodes 1 and 2. The objective is to control the delays of $A_1(k)$ at node 1. It is assumed that all of the traffic $A_1(k)$ is destined for node 3. There are two possible routing paths, one a 2-hop route through Node 2 and
another a single hop from Node 1 to Node 3. Each node is assumed to have an infinite buffer size and $B_i(k), i=1,2$ denotes the buffer occupancy. $C_{ij}$ is the link capacity between nodes i and j. Let $R(k)$ denote the fraction of $A_1$ traffic rerouted on $C_{12}$ to minimize delays. The decision to select one of the two routes is based on the cost function defined for each pair of nodes as,

$$S_{13}(k) = \frac{B_1(k)}{C_{13}}$$  \hspace{1cm} (4.13a)

$$S_{12}(k) = \frac{\min(A_1(k), C_{12})}{C_{12}}$$  \hspace{1cm} (4.13b)

$$S_{23}(k) = \frac{B_2(k)}{C_{23}}$$  \hspace{1cm} (4.13c)

Here $S_{ij}(k)$ represents the queueing delays experienced at time $k$. Under the traditional shortest path routing, the arrivals at node 1 would be routed directly to node 3. However using a alternate route 1-2-3 might minimize the delay. The decision criteria is, IF

$$S_{12} + S_{23} < S_{13}$$  \hspace{1cm} (4.14)

then

$$R(k) = \min[A_1(k), C_{12}]$$  \hspace{1cm} (4.15)

else

$$R(k) = 0$$

Using Eqns. (4.1), (4.2), and (4.7), the state vector and system matrices are given by,
\[ X(k) = T_k \alpha(k) - 1 + \mathcal{C}_k + Gg_k \]

where

\[
X(k) = \begin{bmatrix}
A_1(k) \\
B_1(k) \\
A_2(k) \\
B_2(k)
\end{bmatrix}
\]

and

\[
T(k) = \begin{bmatrix}
\alpha_1 & 0 & 0 & 0 \\
\alpha_1 & 1 & 0 & 0 \\
0 & 0 & \alpha_2 & 0 \\
0 & 0 & \alpha_2 & 1
\end{bmatrix}, \quad C(k) = \begin{bmatrix}
\hat{\mu}_1 \\
\hat{\mu}_1 - C_{13} - R(k) \\
\hat{\mu}_2 \\
\hat{\mu}_2 - C_{23} + R(k)
\end{bmatrix}
\]

for \( B_1(k) \geq C_{13}, B_2(k) \geq C_{23} \)

\[
X(k) = \begin{bmatrix}
A_1(k) \\
B_1(k) \\
A_2(k) \\
B_2(k)
\end{bmatrix}, \quad T(k) = \begin{bmatrix}
\alpha_1 & 0 & 0 & 0 \\
\alpha_1 & 1 & 0 & 0 \\
0 & 0 & \alpha_2 & 0 \\
0 & 0 & \alpha_2 & 0
\end{bmatrix}, \quad C(k) = \begin{bmatrix}
\hat{\mu}_1 \\
\hat{\mu}_1 - C_{13} - R(k) \\
\hat{\mu}_2 \\
\hat{\mu}_2 + R(k)
\end{bmatrix}
\]

for \( B_1(k) \geq C_{13}, B_2(k) < C_{23} \)

\[
X(k) = \begin{bmatrix}
A_1(k) \\
B_1(k) \\
A_2(k) \\
B_2(k)
\end{bmatrix}, \quad T(k) = \begin{bmatrix}
\alpha_1 & 0 & 0 & 0 \\
\alpha_1 & 0 & 0 & 0 \\
0 & 0 & \alpha_2 & 0 \\
0 & 0 & \alpha_2 & 1
\end{bmatrix}, \quad C(k) = \begin{bmatrix}
\hat{\mu}_1 \\
\hat{\mu}_1 - R(k) \\
\hat{\mu}_2 \\
\hat{\mu}_2 - C_{23} + R(k)
\end{bmatrix}
\]

for \( B_1(k) < C_{13}, B_2(k) \geq C_{23} \)
\[
X(k) = \begin{bmatrix}
A_{1(k)} \\
B_{1(k)} \\
A_{2(k)} \\
B_{2(k)}
\end{bmatrix}, \quad
T(k) = \begin{bmatrix}
\alpha_1 & 0 & 0 & 0 \\
\alpha_1 & 0 & 0 & 0 \\
0 & 0 & \alpha_2 & 0 \\
0 & 0 & \alpha_2 & 0
\end{bmatrix}, \quad
C(k) = \begin{bmatrix}
\hat{\mu}_1 \\
\hat{\mu}_1 - R(k) \\
\hat{\mu}_2 \\
\hat{\mu}_2 + R(k)
\end{bmatrix}
\]

for \( B_1(k) < C_{13}, B_2(k) < C_{23} \)

The matrix \( G \) and the noise vector \( \bar{g} \) are given by,

\[
G = \begin{bmatrix}
1 & 0 \\
1 & 0 \\
0 & 1 \\
0 & 1
\end{bmatrix}, \quad \bar{g} = \begin{bmatrix}
g_1 \\
g_2
\end{bmatrix}, \quad \text{var}[\bar{g}] = Q
\]

Applying Eqns. (4.9)-(4.12) for Kalman prediction and updating, the buffer occupancy \( B_i, i = 1, 2 \) were estimated. Applying the cost functions and decision criteria given in Eqns. (4.13)-(4.15) the decision to route over the lowest cost route was undertaken. The performance will be a function of the link capacities \( C_{12} \) and \( C_{13} \). Denoting the ratio \( C_f = \frac{C_{12}}{C_{13}} \), the delay performance of \( A_1 \) traffic was examined for successively increasing values of \( C_f \). The results show that the delay performance is in general a function of the degree of dependence that exists in the background traffic on node 2. In particular, as the magnitude of \( \alpha_2 \) becomes greater than or equal to \( \alpha_1 \) the benefits of rerouting over the alternate route will be reduced. If \( \alpha_2 < \alpha_1 \), a finite region of \( C_f \) exists where a considerable
reduction takes place in the delays experienced by $A_1$.

The effect of adaptive routing on the delay performance of buffer at node 1 is demonstrated in Fig. 4.6(a) and (b) where the 90th percentile of the delays is plotted on the vertical axis as a function of $C_f$, which varies from $0.1 - 0.9$. In the simulation results, $\alpha_1 = 0.95$ and $\alpha_2$ is set to values of $0.95, 0.85$ and $0.75$. The results depicted in Fig. 4.6(a) show that when $\alpha_2 < \alpha_1$ there can be improvement in delay performance relative to direct routing for the entire range of $C_f$ considered. However if $\alpha_2$ is comparable to $\alpha_1$, the performance improvement arises in a small region around $C_f = 0.2$ where a local minima is observed. In Fig. 4.6(b) the effect of changing the variance of the traffic at Node 2 is studied. Keeping $\alpha_2 = 0.85$ the variance of the process is changed from $\text{var}1 = 1700$ which is equal to that of the traffic arrivals at Node 1 to values of $\text{var}2 = 1200$ and $\text{var}3 = 700$. Decrease in the variability of Node 2 is also shown to positively impact the performance of the delays of $A_1$. In calculating the delay values of $A_1$ both queuing delays at Nodes 1 and 2 were incorporated.
Fig 4.6a: Delay Performance of Buffer 1 for different $\alpha_2$

Fig 4.6b: Delay Performance of Buffer 1 for different $\sigma_2^2$

Next Fig. 4.7 depicts the fraction of traffic rerouted for the parameters shown in Fig. 4.6(b). As the value of $C_f$ is increased, an increased fraction tends to take the alter-
nate route since for a fixed link capacity of $C_{13}$ this leads to smaller costs for routing over the link with increasing capacity $C_{12}$. However as more packets take the alternate route, they add to the queueing lengths at Node 2, leading to transient increase in the load at Node 2.

**Fig. 4.7**: Fraction of traffic routed on alternate link

Fig. 4.8 shows the complementary delay distributions at node one using direct and adaptive routing and the delay distribution at node two under adaptive routing. The performance improvement in node 1 arises in the region around small delays. However the tail of the delay distribution is extended under adaptive routing due to the burst level de-
lays experienced by $A_1$ traffic at node 2. Therefore the tradeoff is that while over 90% of the packet arrivals experience lower delays, there is a small fraction that experience higher delays than in the direct routing situation.

![Controlled Buffer Dynamics](image)

**Fig. 4.8**: Comparison of delay distributions under direct and adaptive routing

### 4.5 Summary

The description of packet arrival process using an AR process model allows the implementation of Kalman filters for one-step ahead estimation and prediction of the arrival process and buffer occupancy at each node of a data network. The procedure was demon-
strated for a three-node network using routing on an alternate link based on minimizing a delay based cost function. The results show that improved performance can be obtained by selecting optimal values of the alternate path capacity, provided the dependence and variability of the arrival process at alternate nodes are subdominant relative to the controlled traffic.
Chapter 5

CONCLUSIONS

The objective of this thesis was to characterize HTTP client traffic, develop quantitative models for the mean arrival process and implement these models in an adaptive routing application. The traffic characterization was carried out on four different sets of data traffic and a number of similarities were identified.

A visualization study of some of long-term traffic features that characterize Internet related traffic and statistical analysis of flow and packet time-scale features were presented. The visualization of upstream and downstream traffic showed that the TCP protocol was the dominant traffic generator at all times of the day. Among the TCP applications, HTTP, FTP, and SMTP generate over seventy percent of the hourly byte and packet volume. The distribution of packet size over time was found to be a predictable component for both inbound and outbound directions. This study also examined the relationship between applications and the packet sizes they generated. A strong correlation was shown to exist between the dominant applications in each direction and the packet size. On an hourly time scale, over 90% of traffic in the outbound direction was in the 0-100 byte range and was generated by client traffic from web and email related applications. In the inbound direction, over 75% of the traffic was in the 576 and 1500 byte range and arose from server responses to the outbound requests. These dependent relationships between
packet size and applications were found to be consistent features in the day to day patterns of bi-directional traffic between the campus network and the Internet.

A flow characterization study was undertaken and a quantitative model for the cumulative byte count as a function of number of flows was derived. Parametric invariance was shown in four days of traces when HTTP flows were considered, rather than the mix of all applications. The use of this model for determining the number of flow states as a function of time duration was discussed. Finally the dependence features of HTTP packet interarrival times were analyzed using the k-interval coefficient of variation metric. It was observed that the aggregation of flows at the network level leads to a renewal process behavior at short time scales ranging from 1 to 100 milliseconds. However the aggregation of flows at the subnet level did not yield such a feature. This may be attributed to the sparseness of the number of host level flows, typically in the 10 – 20 range. The host level flow characteristics were shown to exhibit approximate renewal features but had a high interarrival time variance.

The mean packet count process was estimated recursively. The packet count process could be well modeled by a first order non-zero mean autoregressive process. The fluid buffer analysis and the simulation of an infinite buffer queue showed that the recursively estimated mean process was sufficient to accurately estimate the delay. Using Kalman Filters a one-step ahead estimation and prediction of the mean arrival process and buffer occupancy was carried out. It was shown using a three-node network example
that model based estimates of the mean arrival process and the buffer occupancy could be used for making decisions on routing traffic on an alternate link to minimize the queueing delays experienced. The Kalman filter was implemented to effectively estimate and predict the state vectors comprising the mean arrival process and buffer occupancy.
APPENDIX I

Packet Networking Fundamentals

The Open Systems Interconnection (OSI) Layer 7 model shown in Figure A1.1, was proposed in the late 1970’s for packet transmission on data networks \[28\]. The four uppermost layers (application, presentation, session and transport) support end-to-end protocols that are network independent while the three lowest layers (network, data link, and physical) support protocols that are network dependent. A message from the source computer flows vertically down until it reaches the physical layer. The message flow takes place in the opposite direction at the destination network.

Fig A1.1. OSI Layer 7 Model
Layer one, referred to as the physical layer, is responsible for transmission of bits over a physical channel. This layer supports functions related to signaling, modulation, and is responsible for converting digital signals to analog. Examples of standards for physical layer include RS-232-C, RS-449, and X.21 for wired networks. Physical layer standards for local area networks include IEEE 802.3, IEEE 802.4 and IEEE 802.5. The next higher layer is the data link layer, which is responsible for the framing of information from higher layers. Examples of link layers are Ethernet, FDDI, ATM etc. Figure A1.2 depicts an Ethernet frame. A 32 bit field cyclic redundancy check (CRC) is allocated for error detection and recovery, the length of the header is not smaller than 64 bytes or larger than 1518 bytes. Each frame can carry information up to 1500 bytes in length. Examples of standards at this layer include higher level data link layer (HDLC). Layer three is the network layer, which packetizes higher layer information and is referred to as the IP layer.

<table>
<thead>
<tr>
<th>Preamble</th>
<th>Dest Addr</th>
<th>Source Addr</th>
<th>Type Field</th>
<th>Data Field</th>
<th>CRC</th>
</tr>
</thead>
<tbody>
<tr>
<td>64</td>
<td>48</td>
<td>48</td>
<td>16</td>
<td>46-1500</td>
<td>32</td>
</tr>
</tbody>
</table>

**Figure A1.2.** Ethernet IEEE 802.3 Frame
Figure A1.3 shows the IP version 6 header format. The length of the IPv6 header is fixed and is 40 bytes unlike IPv4 header which is variable. The header stores the information necessary to route and deliver packets to their destination. The first 4-bit field, version, indicates the version of the Internet Protocol being used. The traffic class field is used to provide differentiated services based on the data being transmitted. The 24 bit, flow label field in combination with source and destination address can uniquely identify a flow that requires special handling by routers. Once per flow handling has been set up the processing of subsequent packets belonging to that flow can be shorter than processing individual packets. The 16-bit payload length field specifies the number of bytes carried in the datagram excluding the header. Thus IPv6 can contain 64K bytes of data. The 8 bit next header field is used to indicate the next header following the IPv6 header. The intended use of this field is identical to the protocol field in the IPv4 header. The hop limit field can be used to limit the number of routers a packet is allowed to visit, which can prevent packets
from being routed circularly. The address field length is 128 bits, in comparison to the 32 bits in IPv4 [19].

Layer 4, the transport layer provides a mechanism for reliable exchange of data between end systems. It ensures that the data are delivered error-free, in sequence with no losses or duplications. Transport protocols are typically the connection-oriented Transmission Control Protocol (TCP) and the connectionless User Datagram Protocol (UDP).

An example of Transmission Control Protocol (TCP) providing a connection-oriented reliable end-to-end service for file transfer using the File Transfer protocol (FTP) is described. When a packet from the Ethernet driver arrives at the IP layer it queues the packet and invokes the IP process indicating that a datagram has arrived. When the IP process has no packets to handle, it accepts the packet waiting in the queue. Once the IP process accepts an incoming datagram, it must decide where to send it for further processing. After IP determines that the data carries a TCP segment it deposits segments in
the port and TCP retrieves them. A port is a finite queue of messages plus two semaphores that control access \[^{29}\]. Once TCP receives the segment, it uses the TCP protocol port numbers to find the connection to which the segment belongs. If the segment contains data, TCP will add the data to a buffer associated with the connection and transmit an acknowledgement to the packet source. TCP implements a number of protocol timers to ensure reliable and synchronized communication between the two end systems \[^{30}\]. The session layer deals with the management of interactions between the two end users. Each session connection has a corresponding transport connection. Should the connection fail, the session layer can re-establish the connection without end user intervention. Similarly, once an end user has completed a session, the session layer may choose to keep the transport connection active initiating another session across the same one. The presentation layer is responsible for representation and transformation of the data between application layer entities. In the presentation layer protocol, internal data is converted into a structure consisting of a type identifier (for example, Boolean, integer or floating point value), an optional length indicator and the converted value. The presentation layer can also be responsible for the encryption and decryption of information as well as file compression. The application layer is the uppermost layer of the OSI reference model. Some of the common application services include file transfer protocols, electronic mails, telnet\[^{31}\].

Packets are transmitted using either the *datagram* or *virtual-circuit* approach. In virtual circuit, a preplanned route is established before any packets are sent. Since the
route is fixed for the duration of a logical connection, it is similar to a circuit in a circuit-switching network, and is called virtual circuit. The node makes a routing decision only once for all packets using the virtual circuit. A delay is induced due to the call setup as well as transmission of packets. If two stations wish to exchange data over an extended period of time, virtual circuit approach becomes efficient for the transmission of packets. In virtual circuit packets are stored until delivered.

In datagram approach, each packet is treated independently, with no reference to packets that have gone before. On each packet the node makes a routing decision. A route is established for each packet. One advantage of using datagram approach over virtual circuit is that the call setup phase is avoided. Thus if a station wishes to send only one or a few packets datagram delivery will be quicker. If congestion develops in one part of the network, incoming datagrams can be routed away from congestion. With the use of virtual circuits, packets follow a predefined route and it thus becomes more difficult for the network to adapt to congestion. Datagram delivery is more reliable compared to virtual circuit. With the use of virtual circuits if a node fails, all virtual circuits that pass through that node are lost. With datagram delivery, if a node fails, subsequent packets may find an alternate route that bypasses that node.
APPENDIX II

RECURSIVE ESTIMATION OF THE MEAN OF A RANDOM VARIABLE

The methodology for recursively estimating the mean value of a random variable is presented in this Appendix. The procedure follows that presented in Young [24]. Let $a_k$ and $\hat{a}_k$ denote the sample mean and its estimate respectively at the $k^{th}$ observation $y_k$. The unbiased and consistent estimate of the mean at the $k^{th}$ sample is given by

$$\hat{a}_k = \frac{1}{k} \sum_{i=1}^{k} y_i$$  \hspace{1cm} (A2.1)

A recursive equivalent of Eq. (A2.1) can be obtained by rearranging the equation in the form,

$$k\hat{a}_k = \sum_{i=1}^{k-1} y_i + y_k$$

and dividing by $k - 1$,

$$\frac{k}{k-1} \hat{a}_k = \hat{a}_{k-1} + \frac{1}{k-1} y_k$$

which leads to,

$$\hat{a}_k = \hat{a}_{k-1} + \frac{1}{k} (y_k - \hat{a}_{k-1})$$  \hspace{1cm} (A2.2)

Eq. (A2.2) may be used to recursively estimate the mean from new observations. The second term is noted to be in the form of a scaled error between the observation and the mean estimate. In the stationary case where $\hat{a}_k$ converges to a constant value in the limit as $k \to \infty$, the second term serves to discount the new information generated by the ob-
The expression in Eq. (A2.2) may also be interpreted as the result of least squares estimation of \( \hat{a} \) through minimization of a cost function proportional to the error term.

To obtain a least squares solution to the problem, let the instantaneous cost function at the \( k^{th} \) observation be given by \( J_k = (y_k - \hat{a}_k)^2 \). The least squares estimate \( \hat{a}_k \) is obtained by setting the gradient of \( J_k \) with respect to \( \hat{a}_k \) equal to zero. The gradient operation yields a term proportional to \( -(y_k - \hat{a}_k) \), which is seen to be a factor in the second term in Eq. (A2.2). The positive sign in Eq. (A2.2) arises due to updates made in the direction of the negative gradient. The multiplicative term \( \frac{1}{k} \) represents the weighting factor for the updates and determines the rate of convergence. The expression for the mean in Eq. (A2.1) can be represented in general form as

\[
a_k = p_k b_k \quad \text{(A2.3a)}
\]

where \( b_k \) are the updates recursively estimated as,

\[
b_k = b_{k-1} + y_k \quad \text{(A2.3b)}
\]

and \( p_k \) are the weights given by

\[
p_k = \frac{1}{k} \quad \text{(A2.3c)}
\]

Substituting Eqs. (A2.3b-c) in Eq. (A2.3a) yields Eqn A2-2.

\[
\hat{a}_k = \hat{a}_{k-1} + p_k (y_k - \hat{a}_{k-1}) \quad \text{(A2.3d)}
\]

Eqns. (A2.3c-d) may be jointly used to recursively estimate the mean with the arrival of new observations.
The generalization of Eq. (A2.3d) may be further generalized for the case where the mean may exhibit a time-varying behavior is considered next. In such a case, the estimation is carried out by shaping the memory of the estimator using a window of selected shape. A window that is exponentially weighted into the past is considered here. The estimates can be obtained by considering the cost function for \( k \) samples modified to the following form,

\[
J = \sum_{i=1}^{k} [\hat{a}_i - y_i] \alpha^{k-i} \tag{A2.4}
\]

where \( 0 < \alpha < 1.0 \) is a constant that can be obtained with the knowledge of the observation sampling interval \( T_s \) and an exponential time constant \( T_e \). From these parameters, \( \alpha = e^{-T_s/T_e} \). The negative gradient of \( J \) with respect to \( \hat{a}_k \) yields the following estimator,

\[
\hat{a}_k = \frac{\sum_{i=1}^{k} y_i \alpha^{k-i}}{\sum_{i=1}^{k} \alpha^{k-i}} = p_k b_k \tag{A2.5}
\]

The weighting function and updates are now modified as \( p_k = \left[ \sum_{i=1}^{k} \alpha^{k-i} \right]^{-1} \) and \( b_k = \sum_{i=1}^{k} y_i \alpha^{k-i} \). The recursive estimates of these variables are given by,

\[
b_k = \alpha \sum_{i=1}^{k} y_i \alpha^{k-1-i} \tag{A2.6a}
\]

\[
= \alpha b_{k-1} + y_k
\]

and
\[ p_k^{-1} = \alpha \sum_{i=1}^{k} \alpha^{k-1-i} \]  
\[ = \alpha p_{k-1}^{-1} + 1 \]  

Substituting Eqs. (A2.6) in Eq. (A2.5) yields,

\[ \hat{a}_k = \frac{p_{k-1}}{\alpha + p_{k-1}} \{ \alpha \hat{b}_{k-1} + y_k \} \]

\[ = \frac{\alpha}{\alpha + p_{k-1}} \hat{a}_{k-1} + \frac{p_{k-1}}{\alpha + p_{k-1}} y_k \]

using the relation \( \hat{a}_{k-1} = p_{k-1} \hat{b}_{k-1} \). Adding and subtracting the term \( \frac{p_{k-1}}{\alpha + p_{k-1}} \hat{a}_{k-1} \) yields the following recursive equations for the time-varying mean,

\[ \hat{a}_k = \hat{a}_{k-1} - \frac{p_{k-1}}{\alpha + p_{k-1}} \left[ a_{k-1} - y_k \right] \]  
\[ (A2.7) \]

\[ p_k^{-1} = \alpha p_{k-1}^{-1} + 1 \]  
\[ (A2.8) \]
LITERATURE CITED


Biography

Mital Parikh was born on October 25 1976 in Bombay, India. He earned his Bachelor of Science degree in Electrical Engineering from University of Bombay, India and Master of Science degree in Electrical Engineering from the University of Massachusetts Lowell in the years 1998 and 2000 respectively. During his tenure at the University of Massachusetts Lowell, he served as a research assistant at the Center for Advanced Computation and Telecommunications where he co-authored a publication. Mital’s current research interests include traffic engineering, and IP routing. He is also a member of Tau Beta Pi Engineering Honor Society and the Sigma Xi Research Honor society.