EMPIRICAL MODELING OF END-TO-END DELAY DYNAMICS IN BEST-EFFORT NETWORKS

A Thesis

by

SRIKAR DODDI

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

May 2004

Major Subject: Mechanical Engineering

EMPIRICAL MODELING OF END-TO-END DELAY DYNAMICS IN BEST-EFFORT NETWORKS

A Thesis

by

SRIKAR DODDI

Submitted to Texas A&M University
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

Approved as to style and content by:

Alexander Parlos
(Chair of Committee)

Won-jong Kim
(Member)

A. L. N. Reddy
(Member)

Dennis L. O'Neal
(Interim Head of Department)

May 2004

Major Subject: Mechanical Engineering

ABSTRACT

Empirical Modeling of End-to-end Delay Dynamics in
Best-Effort Networks. (May 2004)

Srikar Doddi, B.E., Osmania University, Hyderabad, India
Chair of Advisory Committee: Dr. Alexander Parlos

Quality of Service (QoS) is the ability to guarantee that data sent across a network will be recieved by the desination within some constraints. For many advanced applications, such as real-time multimedia QoS is determined by four parameters-end-to-end delay, delay jitter, available bandwidth or throughput, and packet drop or loss rate. It is interesting to study and be able to predict the behavior of end-to-end packet delays in a Wide area network (WAN) because it directly affects the QoS of real-time distributed applications. In the current work, a time-series representation of end-to-end packet delay dynamics transported over standard IP networks has been considered. As it is of interest to model the open loop delay dynamics of an IP WAN, the UDP is used for transport purposes. This research aims at developing models for single-step-ahead and multi-step-ahead prediction of moving average, one-way end-to-end delays in standard IP WAN's.

The data used in this research has been obtained from simulations performed using the widely used simulator ns-2. Simulation conditions have been tuned to enable some matching of the end-to-end delay profiles with real traffic data. This has been accomplished through the use of delay autocorrelation profiles. The linear system identification models Auto-Regressive eXogenous (AR) and Auto-Regressive Moving Average with eXtra / eXternal (ARMA) and non-linear models like the Feedforwad Multi-layer Perceptron (FMLP) have been found to perform accurate single-step-ahead predictions under varying conditions of cross-traffic flow and source send rates. However as expected, as the multi-step-ahead prediction horizon is increased, the models do not perform as accurately as the single-step-ahead prediction models. Acceptable multi-step-ahead predictions for up to 500 msec horizon have been obtained.

To parents and sister

ACKNOWLEDGMENTS

I am grateful to my advisor Dr. Alexander Parlos, who has introduced me to the field of Neural Networks and System Identification. His friendly and patient attitude has always encouraged me during the perpetual cycles of bug-fixing. He provided constant encouragement, sound advice, good teaching, and good company throughout my research under him. I also owe my thanks to Dr. Won-jong Kim and Dr. A. L. N. Reddy for their time. Thanks are also due to my friends Aninda, Parasuram, Tolis and Vivek. I am grateful to the faculty members of the Department of Mechanical Engineering at Texas A&M University. I also thank my grandparents Dr. V. Nandi and Dr. Sheila Nandi for their constant encouragement. Lastly, and most importantly, I wish to thank my parents, D. Satyanarayana and D. Sarala who raised me, supported me, taught me, loved me, and gave me wise advice. To them I dedicate this thesis.

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

LIST OF TABLES

TABLE Page

TABLE Page

LIST OF FIGURES

FIGURE Page

FIGURE Page

CHAPTER I

INTRODUCTION

The objective of this research is to develop single-step and multiple-step ahead predictors for forecasting the per-flow end-to-end delays in best-effort wide area networks (WAN). The best example of a best-effort WAN is the Internet. The dynamic behavior of a best-effort network can be characterized by end-to-end delays, end-to-end delay variation or jitter, packet losses, and throughput measurements of the various flows. These variables have a direct impact on the Quality of Service (QoS) delivered to the users. Hence, it is essential to develop predictive models of these variables using available measurements. Per-flow one-way end-to-end delays exhibit high degree of long-term dependency and non-stationarity nonlinearity due to the nonlinear dynamic behavior of the underlying network dynamics. Predicting this variable would be beneficial for addressing a number of technical issues, such as network provisoning, congestion avoidance/control and application QoS control, to name a few.

A. Motivation and Objectives

The Internet has developed over the years into a large and complex interconnection of networks and it is growing exponentially. As time progresses it will be very difficult to understand the behavior of such a WAN and to model its behavior would be a difficult task. If we look at the Internet today as an example of a WAN, there are millions of users who send and receive traffic across it. As of today more than 80% of the Internet resources are used by traffic generated from applications using TCP [1]. UDP-based applications use the remaining resources. Table I gives a summary

The journal model is *IEEE Transactions on Automatic Control*.

Table I. Internet Application Protocols [2].

Application	Application Layer Protocol	Transport Protocol
Electronic Mail	SMTP	TCP
Remote Terminal Access	Telnet	TCP
Web	HTTP	TCP
File Transfer	FTP	TCP
Remote File Server	NFS	typically UDP
Streaming Multimedia	proprietary	typically UDP
Internet Telephony	proprietary	typically UDP
Network Management	SNMP	typically UDP
Routing Protocol	RIP	typically UDP
Name Translation	DNS	typically UDP

of different applications using TCP and UDP as the transport protocol. The difficulties associated with the characterization and simulation of the Internet have been clearly dealt by Paxson and Floyd [3]. In this paper it is mentioned that the Internet has three key properties associated with it. These are technical and administrative heterogeneity, rapid growth over time, and immense changes over time. They have highlighted the importance of simulation-based studies in Internet characterization, but at the same time measurement-based studies are essential for sanity check. An interesting area in this field is the end-to-end behavior of a best-effort WAN which can be characterized using forward delays, reverse delays and delay variation. There are many other variables that influence the end-to-end behavior of a best-effort WAN such as packet losses, throughput, and duplication of packets. It is an interesting challenge to measure and study the end-to-end delays on a best-effort WAN in order to characterize its end-to-end performance.

The immediate motivation for this research is that the dynamic behavior of the end-to-end delays of a best-effort WAN as measured by arriving packets directly influence the QoS for real-time multimedia applications. A delay-based approach as proposed in [4] has several advantages, for example more efficient congestion control mechanisms. Once a delay-based congestion control mechanism is developed, one could anticipate packet losses which are typically but not always caused by increased packet delays (except in wireless networks). Such anticipatory schemes could be used to reduce packet losses. A lot of researchers in this area have done end-to-end measurement-based studies [5, 6, 7]. This research has demonstrated that the Internet is very dynamic, i.e., it exhibits non-equilibrium traffic conditions. In [8], it has also been demonstrated that WAN traffic is non-equilibrium, i.e. it is bursty and heavy-tailed, violating the Poisson distribution assumption embedded in queueing theory. This dynamic behavior of the Internet is one of the main reasons that queuing theory and associated statistical analysis approaches have not been very successful. In [9], Ohsaki proposed a new approach for investigating end-to-end packet delay dynamics. In this approach end-to-end packet delay dynamics has been modeled using system identification or empirical modeling techniques. This approach assumes the Internet to be a "black-box", as seen by the source and the destination, and the packet delays obtained from simulation are modeled using a Single-Input and Single-Output (SISO) system based on an Auto-Regressive eXogenous (AR) model. This approach does not take the nonlinearities of the Internet into account and it has shown limited effectiveness. Preliminary evidence suggests the feasibility of the approach in this thesis [10], where the original "black-box" model [4, 9, 11] as seen by the source and destination is being retained, but artificial neural networks (ANN) are used to model the delay dynamics of a single packet flow.

In this study, empirical models for the multi-step prediction of one-way end-to-end delays of UDP packets have been developed. The UDP end-to-end packet dynamics represent the open-loop dynamics of a best-effort network as shown in the Figure 1. The UDP open-loop dynamics can also be referred as IP-level dynamics since the delay associated with sending packets from application layer to transport layer is negligible and also there is no transport layer feedback from the destination back to the source. Figure 1 depicts a dashed feedback signal which is occasionally used by some real-time applications, such as streaming. In this research no such feedback is utilized. The TCP end-to-end dynamics can be considered as closed-loop dynamics since there is a delay or loss-based feedback from the destination back to the transport layer of the source as shown in the Figure 2. The implicit congestion feedback signal starting from the transport layer of the destination and going back to the transport layer of the source can be clearly seen. Thus, studying the open-loop dynamics of a best-effort network can only be accomplished by using UDP as the transport protocol, as showin in Figure 1. Such a modeling effort can eventually be useful for either UDP or TCP based applications.

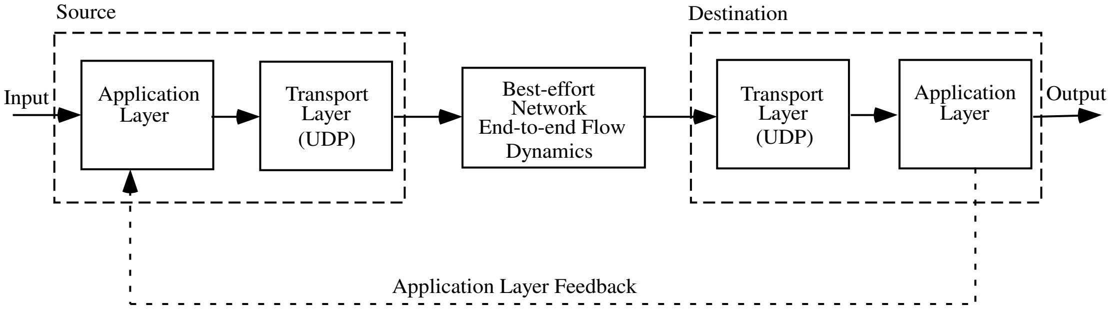

Fig. 1. End-to-end Block Diagram for Applications Using UDP.

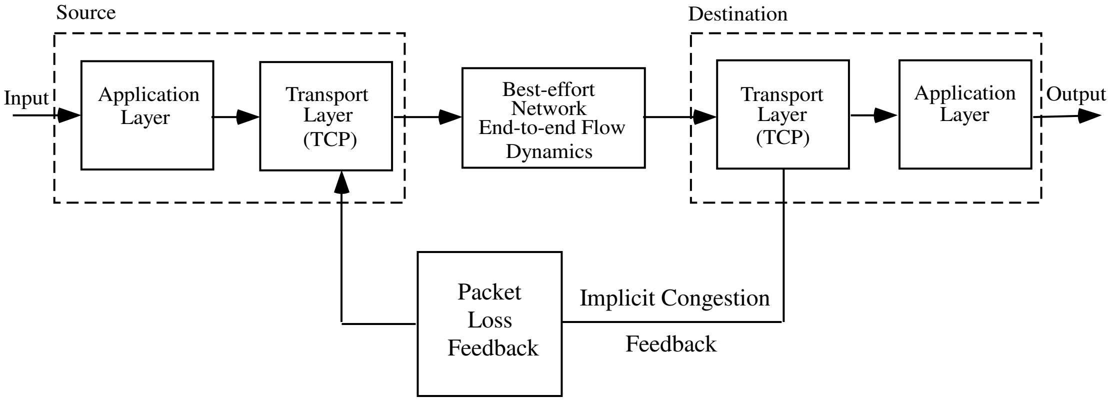

Fig. 2. End-to-end Block Diagram for Applications Using TCP.

B. Literature Review

1. Research in End-to-end Delay Measurements

There are many tools that are available for characterizing and measuring end-to-end delay behavior in WANs. Basically these tools can be classified as simulators and real-time tools. The first simulator that has been developed is REAL [12]; a packet level simulator. It is actually intended for analyzing the dynamic behavior of flow and congestion control mechanisms in packet-switched networks. It is primarily used for emulating a variety of transport protocols like TCP. Another simulator developed later, called x-sim [13], is useful only for studying network topologies and protocols. In recent times, Network Simulator (ns-2) [14] has become the most widely used simulation tool. Ns-2 is an event-based tool which is very useful in characterizing end-to-end delays. None of these tools can accurately capture the precise behavior of a WAN taking the non-equilibrium conditions into consideration.

There are many active and passive measurement techniques available on the Internet for characterization purposes but most of the data available are for round-trip times. Furthermore, the temporal aspects of data are not well-documented. Only aggregate delay information is made available. This study is mainly focussed on analysis of one-way trip times i.e., both forward and reverse delays.

Furthermore, there are several real-time tools available for studying the network characteristics from an end-to-end point of view. Most of these tools measure throughput, latency, and routing information as aggregates. This research requires tools which allow the measurement of one way end-to-end delays and losses as a function of time. In 1998, Ikjun Yeom developed the UPBAT and TPBAT tools [15]. These tools allow one to measure both forward and reverse delays between a source and destination thereby allowing one to obtain delays as a function of send time by affecting several parameters like the packet size and inter-departure time. There are other real time tools like SPAND [16], which measure available bandwidth and packet loss rate. Réseaux IP Européens (RIPE) [17] is an organization of individuals interested in wide area IP networks in Europe and some parts of Asia. One of it's area of interest is the RIPE Network Coordination Center (NCC) Test Traffic Measurements (TTM) which measures several parameters that affect the Internet i.e., one-way delay and packet loss, instantaneous packet delay variation (IPDV), bandwidth, and several other measurements. They use dedicated test boxes which allow them to continuously monitor the network connectivity to the other parts of the Internet. The data can be obtained in several useful ways like plots showing the delay and packet-loss during a day, week, and month, and plots for arbitrary time intervals. Unfortunately, to obtain this data, a test box is needed. In other words, one must become part of the RIPE network. The measurements obtained from RIPE NCC have been analyzed in [18]. The tools developed by Yeom - TPBAT and UPBAT, have been used for measuring one-way end-to-end delays of TCP and UDP flows. The simulated data used in the current study have been obtained from ns-2 simulator.

2. Research in End-to-end Delay Estimation

As compared to tools used for measurement of end-to-end delays, the tools available for delay estimation are very few if any. Network Weather Service [19] is a distributed system which monitors certain network conditions and then forecasts them for a given time frame. This forecasting tool measures end-to-end performance metrics like bandwidth and latency for TCP/IP networks. Modeling the Internet traffic has always been approached from a statistical point-of-view [20]. Lot of queueing theory has also been applied for such modeling purposes [21]. More recently, a dynamic "black-box" based approach has been proposed in [9, 11]. In this approach the end-to-end packet delay dynamics are modeled using empirical modeling tools from System Identification. The approach is quite innovative even though the network used a 100 Mbps LAN. The authors have used an AR model in identifying the system. This approach is based on “treating the Internet as seen by the source and the destination hosts as a black-box where the end-to-end packet delay dynamics are modeled as a Single-Input and Single-Output (SISO) system” [11]. The input to this black-box model is the packet inter-departure time from the source host and the output is the packet delay variation measured by the destination host. Though AR was able to predict the end-to-end delay dynamics of the simulated network, the LAN network size is relatively small when compared to the actual size of a WAN or the Internet. One of the main reasons that this model might fail in predicting the actual end-to-end packet delay dynamics is because the Internet is nonlinear in nature. Traditional linear models like AR and ARMA might not approximate the delay dynamics accurately, especially when considering WAN dynamics of single flows

3. Research in Using System Identification Techniques for Empirical Modeling

System identification techniques can be used to solve complex problems. These problems can be broadly classified as linear and nonlinear problems. Linear system identification methods like AR and ARMA can be used to model simple linear problems. The linear models AR and ARMA come under the class of linear input-output models which model the dynamic relationship between system inputs and outputs in the form of linear regression. However, complex problems involving nonlinearities may not be solved accurately using linear methods. Neural network based nonlinear methods have to be used in place of linear tools like AR, ARMA, etc.

Artificial Neural Networks (ANNs) have been shown particularly useful in predicting the dynamics of nonlinear systems [22]. It has been shown in [23] that ANNs can be used for single and multi-step-ahead prediction of frames/VOP sizes of MPEG-coded real-time video streams. One of the tool used in this research for estimating the delays is based on ANNs. Hence, it is important to know about the function offered by ANNs. An ANN can be defined as "a system of simple processing elements, the equivalent of neurons, that are connected into a network by a set of synaptic weights" [24].

The reason for considering ANNs for estimating the delays is because of the nonlinearity involved in the measured delays. The basic unit of an ANN, a node, is a nonlinear device. Consequently, an ANN is made up of multiple interconnected nodes [25]. These nodes can also be considered as interconnected processors or parallel distributed processors [25]. Each node is connected to another node through a weighted link. A single node may receive more than one input signal but it produces only one output signal. The function of an ANN is dependent on the network architecture, the weight magnitudes and the processing elements' mode of operation.

There are numerous types of ANNs, but the most effective category in addressing signal processing type of problems is the multilayer perceptron. A multilayer perceptron network can be further classified into two groups based on how it processes the data:

(a) Feedforward Multi-layer Perceptron (FMLP), and (b) Recurrent Multi-layer Perceptron (RMLP)

FMLP networks are very good at approximating memory-less nonlinear functions, whereas RMLP networks have inherent memory. RMLP networks are useful in approximating dynamic nonlinear systems. A feed forward multilayered perceptron typically has one or more hidden layers. Typically such a network consists of 3 layers: an input layer of source nodes, a series of hidden or middle layers, and an output layer of computational nodes. The training of multilayer perceptrons is typically done using the backpropagation algorithm. A RMLP network is similar to an FMLP except that it contains feedback loops in its hidden layers. Similarly to FMLP, an RMLP is trained using versions of the backpropagation algorithm appropriate for recurrent networks.

In this thesis predictors based on linear methods such as AR and ARMA, and nonlinear methods, such as FMLPs, have been developed for performing single-step-ahead (SSP) and multi-step-ahead (MSP) prediction of moving average one-way end-to-end delays in best-effort networks.

C. Proposed Solution

The above sections explain the need for a method to empirically model the end-to-end packet delay dynamics of a best-effort network like the Internet. The present research does not utilize real traffic data. However, a best-effort WAN network is simulated in ns-2. The main reason for using ns-2 to generate date is to enable utilization of a controlled test bed. Thus the data traces used for empirical modeling in this research are obtained from ns-2. In this research, three predictors have been developed for modeling purposes. Each of these predictors are based on AR, ARMA and FMLP respectively. Thus, the use of linear methods such as AR, and ARMA and ANNs in modeling the end-to-end delays in best-effort WANs has been demonstrated in this thesis. Furthermore, these predictors have been used for SSP and MSP.

D. Contributions of This Research

The area of modeling the Internet dynamics, and in general those of a best-effort WAN, is relatively new and very few researchers have attempted to model such systems. Simulation studies are more widespread. The ns-2 simulator has been used to generate best-effort WAN traffic with several intermediate source and destination nodes acting as cross-traffic. The data collected from these ns-2 simulations has been used in this study instead of real traffic. The proposed modeling approach would be considered as contribution in a number of ways.

- This research makes a bold and honest attempt to develop empirical models for the Internet end-to-end delay dynamics with some preliminary success.

- It could be used to improve the QoS for “hard” real-time multimedia applications on the current Internet. Real-time multimedia applications with ”hard” constraints cannot utilize extensive buffering or caching to solve QoS related problems.

- It could pave the way for much more efficient congestion avoidance/control mechanisms. An effective congestion avoidance/control mechanism based on end-to-end delay prediction would anticipate packet losses through ”increases in end-to-end delays” and the probability of a packet loss through actively controlling packet flows.

E. Thesis Overview

Chapter II presents a qualitative discussion of delays in best-effort networks. Chapter III outlines various system identification methods and algorithms for modeling network delay dynamics. Measurement and analysis of end-to-end packet delays is discussed in Chapter IV. Single-step-ahead prediction results of moving average one-way end-to-end delays are presented in Chapter V and multi-step-ahead prediction results of the same are presented in Chapter VI. Chapter VII deals with the thesis summary and provides some conclusions. It also includes recommendations for future work in this area.

CHAPTER II

END-TO-END NETWORK DELAY: A QUALITATIVE DISCUSSION

A. Delays in Best-effort Networks

Data sent over the Internet is split into segments called data packets. These packets are then directed to their destination by routers over different paths, in general. Once these packets reach their destination, they are reassembled. The time taken by these data packets to reach their destination application is called end-to-end delay. The delay which this research deals with is the end-to-end one-way delay. The end-to-end delay is defined more specifically as the difference in time when a data packet leaves the application layer at the source and the time when it reaches the application layer at the destination. The end-to-end packet delay can be clearly understood from Figure 3 [10]. Ideally, the end-to-end delay must remain constant over time. This does not happen due to several factors that influence the end-to-end delays. Hence, end-to-end packet delays are said to be time varying time delays [26].

1. Causes of Delays in Networks

The end-to-end delay can be split into two components, a fairly constant component and a variable component. The former component of the delay includes transmission delay at nodes and propagation delay on the links through which the data packet traverses before it reaches the destination. The latter component of the delay is the queueing delay mainly caused by the cross-traffic and processing delays. The following is a brief explanation of some of the important components of the end-to-end delays [27].

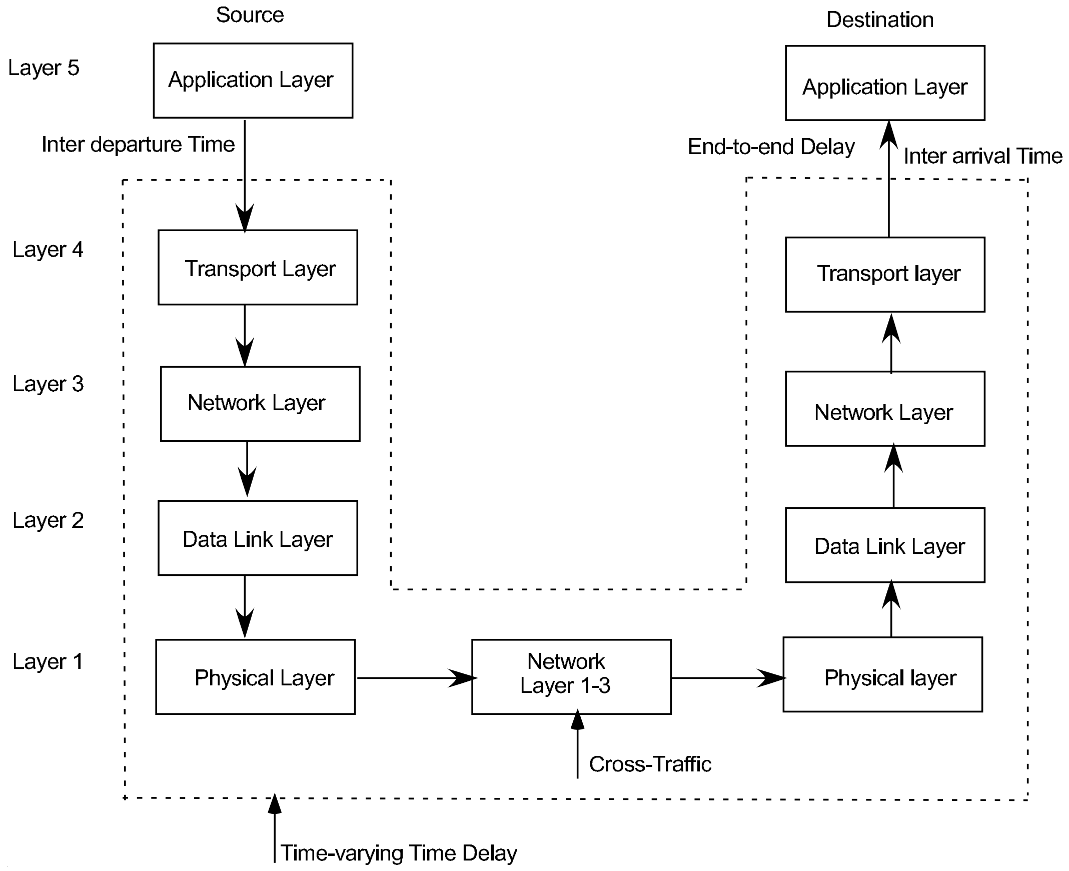

Fig. 3. Schematic Diagram of End-to-end Delay

- **Transmission Delay**: This is defined as the time taken to transmit all the bits of a packet into the link. It is basically the time required by a node to push the packet onto the link.It can be obtained by dividing the length of the packet by the transmission rate of the link. For WANs, this is generally on the order of microseconds.

- **Propagation Delay**: This is defined as the time required by a bit to propagate from the beginning of the link to the end of the link. The speed at which the bit propagates is the propagation speed of the link. This type of delay depends on the physical medium of the link. Propagation delay between two nodes is the distance between the two nodes divided by the propagation speed. In WANs, this is generally on the order of msec.

- **Processing Delay**: This is defined as the time associated with checking of the bit-level errors in the packet and in examining the packet's header to determine where to direct the packet. In WANs, the processing delay is generally on the order of microseconds.
- **Queuing Delay**: A packet is said to experience queueing delay as it waits in the queue to be transmitted onto the link. This delay depends on packets that are queued before the specific packet and which are already in the queue waiting to be transmitted. Hence this delay could vary significantly from one packet to another and it could range from a few msec to hundreds of msec.

2. Causes of Packet Losses in Networks

The performance of a best-effort network is not only measured by packet delays but also by packet losses. Packet losses occur when the router queues are full. Since the queue limit or the queue capacity of a router is always finite, incoming packets sometimes do not find place and the router drops these packet. This is more often called congestion. A router generally has incoming interface buffers, system and outgoing interface buffers. Depending upon where the packet is lost, the drop is said to be an input drop or an output drop [28]. Packet losses at input interface buffer occur when the router cannot process incoming packets fast enough. Packet losses at output interface buffer occur when the outgoing link is very busy. Suggestions have been made that sometimes packets should be dropped even before the queue becomes full as in Random Early Detection (RED). This warns TCP to back-off so as to avoid congestion. Though congestion is the most common cause of a packet loss, there are several other causes for packet losses. Packet losses can also occur due to transmission errors and sometimes packets are dropped if the checksum on the packet fails. Additional modes of packet losses exist in wireless networks, e.g. channel fading.

3. Causes of Delay Variations in Networks

Delay variations or delay jitter in best-effort networks can occur due to several reasons. They are [5, 28]

1. **Faulty Clocks**: Calibration or synchronization of clocks plays a vital role in the measurement of delays and subsequently, in the delay variations. Dealing with clock accuracy is the first and foremost step to be addressed as it has a direct effect on the observed state of a network. It is absolutely necessary to synchronize the clocks of the two end-to-end nodes in which we are interested, in order to remove the time drifts and clock skews.

2. **Queues**: Queuing is one of the main causes of jitter. Queueing is said to occur if the total packet input rate is more than the bandwidth or capacity of the output link. If consecutive packets experience different waiting periods at a particular queue, then jitter comes into picture. Generally, queues build up at switches and routers. More often, queues are a result of two or more flows competing for the same output link capacity. Jitter between two consecutive packets can be very high. Two packets, one following the other, are subjected to different queuing delays during transmission from source to destination as a result of varying cross-traffic flow.This creates jitter. Since there are several hops between a particular source and destination, these queueing delays can

have a cumulative effect and result in large delay variations.

3. **Bursty Traffic**: Jitter can occur due to sudden bursts of traffic. If a burst of traffic is about to enter a link and if the capacity or the bandwidth of the outgoing link is less than the input rate of the burst, then the queue builds up. This queue lasts till the end of the burst. Each packet, according to its position in the burst, experiences a delay different than other packets in the burst, thus causing jitter. This becomes even more complex when several bursty flows compete for the limited bandwidth available.

4. **Reordering of Packets**: A packet is said to be reordered if its sequence number is less than the sequence number of any packet of the same flow that arrived before it. This may happen either due to route changes, multiple buffers at routers or due to the retransmission algorithms present in reliable protocols, in the case of TCP. Reordering of packets will directly affect delay variation.

5. **Route Changes**: The stability of a route taken by a stream of packets have a significant effect on the delay variation. Route changes occur due to router failures, sudden recovery of failed routers, changes in routing algorithms, and sometimes while balancing network load. The propagation delay on these routes may not be the same, and hence route changes cause delay variations. Also the queuing and congestion delays may be different at different routers. All these factors contribute to variations in end-to-end packet delays.

4. Causes of Throughput Variations in Network

Throughput of a flow is defined as the rate at which data is received at the destination. It is directly related to the rate at which data is pushed into the network which is otherwise called send rate. Assuming a single flow, this send rate approaches the minimum capacity of the end-to-end path, the queues start building up causing increased queueing delays which finally decreases the throughput. As the queues reach their queue limit, packet losses occur resulting in zero throughput or no throughput at all. Huge variations in throughput are sometimes caused by bursty nature of best-effort traffic.

B. Impact of QoS Metrics in Real-time Applications

Delay and delay jitter in the network are very important parameters for QoS for real-time multimedia applications/services. Delay jitter can be defined as the change in delay between consecutive packets of the same flow. This variable plays a major role especially in voice and video applications. For non-real-time applications like ftp and telnet, delay jitter has a negligible impact on QoS. Delay jitter can be both negative as well as positive. It is negative if there is network congestion and consecutive packet losses and it is positive if there is significant delay in the network. The effect of jitter in multimedia video streams can be removed by a playback buffer at the receiver end at the expense of additional delay. Throughput, another major factor impacting Application QoS. Data intensive and multimedia applications need high throughput. Basically, real-time applications can be divided into following classes:

- **Interactive multimedia applications**: A very good example of this type of applications is **Voice over IP (VoIP)** and video conferencing . VoIP can be defined as the ability to deliver voice packets over IP-based data networks in real-time. An important design consideration in implementing VoIP is to minimize the end-to-end delay. A similar design consideration can be applied to video over IP. Voice and video services have a strict QoS requirement on delay and delay jitter compared to other applications, which involve data transfer. As the packets are transported on first-come, first-served (FIFO) basis, large data file transfers take advantage of large packet size. This in return introduces large delays and delay jitters to packets, which affect the QoS for voice and video over the Internet. The most common problems caused by these delays are echo and overlap that can be seen in VoIP.

Another example for this type of applications is **network games**. Today the traffic due to games on the Internet is increasing at a fast pace. These games use UDP as transport protocol to transmit data. The effect of traffic due to games on the TCP throughput is shown in [29]. The report also highlights the reasons for using UDP in games. Some of the main reasons are as follows.

1. Real-time applications like games need immediate response times for comfortable playing. TCP adjusts and readjusts itself based on the congestion in the network which is not suitable for games.
2. Retransmission of lost packets in TCP introduces jitter resulting in the loss of quality of the game. Though TCP guarantees complete information to the player, the player can see that the game pauses whenever the TCP goes into the recovery mode.
3. Processor overhead for TCP is more than UDP, since TCP uses processing time for setting up connections, and checking for errors.
4. The header size of UDP is less than the header size of TCP and in a way UDP sends smaller packets into the network and this causes smaller end-to-end delays.

- **Non-interactive multimedia applications**: Applications like news on demand, video on demand (VoD), audio and video broadcasts, and distance learning are part of this class of applications. These applications use UDP as the transport protocol. This class of applications pose a challenge in the area of multimedia networking. The video stream in VoD is variable bit rate and hence it requires variable bandwidth. The transmission of this video stream across the Internet with a guaranteed QoS is essential. The inter-arrival times or rather the jitter must lie in a specified bound in order for the frame to have desired quality. The effect of delay jitter can be minimized or can be removed completely using a playback buffer at the receiver side in these class of applications.

C. Chapter Summary

Delay jitter is more critical than packet loss as it has direct effect on the QoS guarantees for many multimedia applications, like video conferencing, voice over IP (VOIP), and several other interactive multimedia applications. The typical problem involving transmission of voice and video packets is the lack of QoS guarantee. UDP is more aggressive and is suitable for real-time applications. The effect of delays in non-interactive applications can be removed by using a playback buffer at the receiver side. Basically, the effect of delays in conversational and interactive multimedia applications must be given more consideration than non-interactive applications. There is a need for predictive estimation of end-to-end delays in the network in order to better manage application QoS over the Internet.

CHAPTER III

METHODS AND ALGORITHMS FOR EMPIRICAL MODELING OF NETWORK DELAY DYNAMICS

A. Introduction

The estimation problem in this research is the prediction of end-to-end delays in best-effort networks. Prediction in the context of system identification is referred to as estimation of a variable of interest at a future point in time given the measured data up until and including the present time itself [30]. There are two types of models in the context of system identification, physical models and empirical models. In physical models, the mathematical expressions describe the relationships between the system variables. Empirical models are those models which are derived from the observed data of the system. The models developed in this research for the network end-to-end delay dynamics can be called empirical models. Empirical models are also called as "black-box" models.

System identification (SI) aims at developing an empirical model from the observed data of a particular system under consideration. Several models have been developed using system identification in the fields of business, medicine, process control, and computer engineering. This type of modeling is necessary as the dynamics of a system becomes increasingly complex and uncertain limiting further mathematical analysis of the system. The modeling of a system is the core step of any control problem. An accurate model representing the dynamics of the system is necessary to build an efficient controller with desired performance.

The current chapter is organized as follows: The next section deals with the system identification procedures. It is followed by a section each on an overview of some of the linear and nonlinear methods used for estimation. This chapter is extensively based on the basic definitions and principles of system identification available in [25, 30, 31].

B. System Identification Procedure

SI in itself is a culmination of a set of sequential procedures as shown in the figure 4. The following are the various steps involved in the system identification process [30].

1. **Experiment design and data collection**: This is the first and foremost step in the system identification procedure. The main purpose of this step is to collect input-output data in such a way that the important dynamics are captured using persistent excitation. The experiments must be conducted in such a fashion that the input signal must be sufficiently rich to extract all of the important dynamics of the system. The data thus collected must be informative and must be analyzed rigorously. After the data is analyzed, correlation tests must be performed to make sure that the data being used are suitable for system identification. If the data collected are not suitable for this purpose, then the experiments must be performed again, or if necessary the experiment must redesigned. This step needs to be repeated till the collected data are suitable for use in system identification.

2. **Model structure selection**: After the collection of appropriate data for identification, a model structure must be assumed. To select an appropriate model structure, apriori knowledge about the system is useful. Some intuition can also be used for model structure selection. Model structures are generally classified into two classes, input-output and state space model structures. In this research, the model structures selected is of input-output type as the system

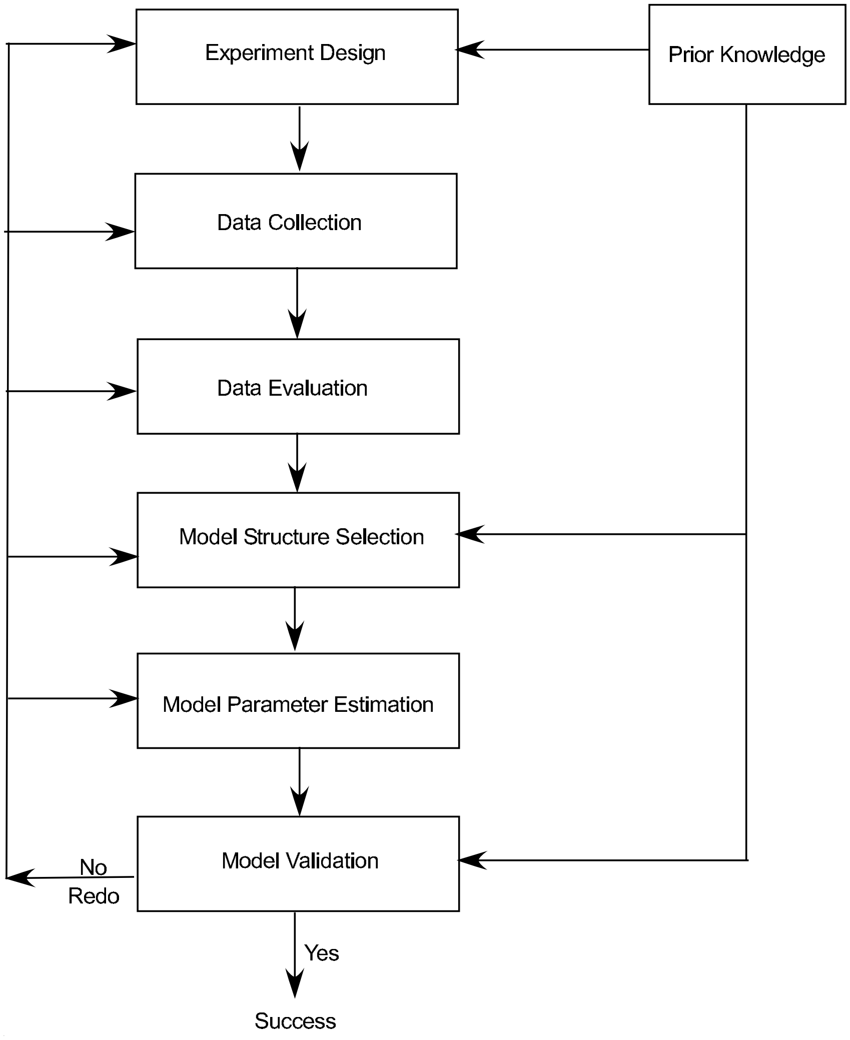

Fig. 4. Block Diagram for the System Identification Process.

under consideration is end-to-end network delays and no state-space is easily defined.

3. **Model parameter estimation**: This step is actually an outcome of the first two steps. The data collected in the first step and the structure selected in the second step are used to solve an optimization problem using any of the available techniques. The numerical values obtained from this optimization step are the model parameters that completely describe the model structure selected in the previous step.

4. **Model validation**: This is the final step of the system identification procedure. The model developed from the previous three steps must be tested and be proven to work in the operating range of interest. The model validity is generally ascertained by taking completely new data sets which have never used in the previous steps and by testing the model using these data sets. The previous steps must be repeated if this step invalidates the developed model.

The above system identification procedure can be summarized as, given a finite set of input observations {u(1),....,u(T)} and the corresponding output observations {y(1),....,y(T)}, the aim of the above procedure is to obtain the free parameters θ of a function $\mathcal{F}()$ such that the one-step prediction $\widehat{y}(t|t-1)$ is expressed as :

$$\widehat{\mathbf{y}}(t|t-1) = \mathcal{F}\Big(\mathcal{U}(t-1); \theta\Big) \tag{3.1}$$

where, in general, the regression vector $\mathcal{U}(t)$ is a combination of the actual system inputs u(.), the system outputs y(.), the past predictions $\widehat{\mathbf{y}}(.|.)$, and/or some other indicators resulting from the combination of inputs and outputs.

C. Linear Methods

The primary assumption on which linear estimation algorithms are based is that the system under consideration can be represented by a linear model. Linear models can be classified into two types, linear input-output models and linear state-space models. Input-output models are basically used to model the relationship between the inputs and the outputs in the form of a linear regression whereas in state-space models, the system is modeled through intermediate variables called states. Through out this research we use input-output models. This is because the system under consideration is the per-flow end-to-end delay dynamics of a best-effort network, modeled on the basis of either an input-output model or rather a time-series model. Moreover, the system under consideration is a single input single output (SISO) system. The following sections provide a brief overview of some of the linear model structures commonly utilized.

1. Auto-Regressive Exogenous Model Structure

This is the simplest model used in system identification. The AR in the ARX model refers to the autoregressive part and X to the extra input called the exogenous variable. The general SISO ARX model is represented by the following equation:

$$\begin{gathered} y(t+1) = a_1 y(t) + \ldots + a_{n_y} y(t - n_y + 1) \\ +b_1 u(t) + \ldots + b_{n_u} u(t - n_u + 1) \end{gathered} \tag{3.2}$$

where y(t) is the output of the SISO ARX model, n_y is the number of past outputs more commonly called the lag terms of the model, u(t) is the input to the ARX model, n_u is the number of past inputs used in the model. The coefficients a_1,. . . ,a_{n_y} and b_1,. . . ,b_{n_u} are assumed to be known.

From the SISO ARX model , represented by the Equation 3.2, the following SSP of the system output, $\widehat{y}(t+1|t)$, is given by the following equation:

$$\begin{aligned}\widehat{\mathbf{y}}(t+1|t) = a_1 y(t) + \ldots + a_{n_y} y(t-n_y+1) \\ +b_1 u(t) + \ldots + b_{n_u} u(t-n_u+1)\end{aligned} \tag{3.3}$$

This form of equation, an SSP, is also called as the predictor form of the SISO ARX model.

Similarly, the MSP SISO ARX predictor is expressed as:

$$\begin{aligned}\widehat{\mathbf{y}}(t+1|t-p+1) = a_1 \widehat{y}(t|t-p+1) + \ldots + a_{n_y} \widehat{y}(t-n_y+1|t-p+1) \\ +b_1 u(t) + \ldots + b_{n_u} u(t-n_u+1)\end{aligned} \tag{3.4}$$

2. Auto-Regressive Exogenous Parameter Estimation

The predictor forms of ARX model are discussed in the previous section. The parameters of the ARX model, $a_1, \ldots, a_{n_y}, b_1, \ldots, b_{n_u}$ are unknown. These parameters must be determined from the measurement data, $y(t)$ and $u(t)$. Parameter estimation for the ARX model is now presented.

The ARX predictor form can be also written in another form as:

$$\widehat{y}(t+1|t) = \varphi^T(t+1)\theta, \tag{3.5}$$

where,

$\varphi(t+1) = [y(t), \ldots, y(t-n_y+1), u(t), \ldots, u(t-n_u+1)]^T$,

$\theta = [a_1, \ldots, a_{n_y}, b_1, \ldots, b_{n_u}]^T$.

The ARX predictor form is written as a scalar product between the data vector $\varphi(t+1)$ and the parameter vector θ. This is in the form of a linear regression with the parameter vector θ as the regression vector and hence least-squares method can be used to solve for θ [30].

In order to solve for the parameters of the ARX predictor using least-squares method, the mean-square of the prediction error, $V_N(\theta, Z^N)$ must be defined as:

$$V_N(\theta, Z^N) = \frac{1}{N}\sum_{t=1}^{N}[y(t) - \widehat{y}(t|t-1;\theta)]^2, \tag{3.6}$$

where Z^N is the data set of N input-output samples $u(t), y(t)$ for $t = 1, \ldots, N$. The above equation can also be called as the objective function of the least-squares problem. The objective function, $V_N(\theta, Z^N)$, is minimized with respect to θ. The solution to the least-squares problem is the value of $\widehat{\theta}_N$ that minimizes $V_N(\theta, Z^N)$, which is as follows:

$$\widehat{\theta}_N = [\sum_{t=1}^{N}\varphi(t)\varphi^T(t)]^{-1}\sum_{t=1}^{N}\varphi(t)y(t). \tag{3.7}$$

3. Auto-Regressive Moving Average Exogenous Model Structure

This is a more general input-output model. The disadvantage of the ARX model, failure in taking the disturbance dynamics into account. This is overcome by adding dynamics to the disturbance term. Additional flexibility is added to the disturbance term by formulating it as a moving average of a white noise process. The AR in the ARMAX model is autoregressive part as in ARX , and the MA is the moving average and X corresponds to the extra input called the exogenous variable as in ARX. The SISO ARMAX model is represented by the following equation:

$$\begin{gathered} y(t+1) = a_1 y(t) + \ldots + a_{n_y} y(t - n_y + 1) \\ +b_1 u(t) + \ldots + b_{n_u} u(t - n_u + 1) + \\ c_0 e(t) + c_1 e(t-1) + \ldots + c_{n_e} e(t - n_e) \end{gathered} \tag{3.8}$$

where n_e is the number of past noise terms (lags), and the other variables are the same as in the ARX model structure.

From the SISO ARMAX model, the following SSP of the system output, $\widehat{y}(t{+}1|t)$, is given in the predictor form by the following equation:

$$\begin{aligned}\widehat{\mathbf{y}}(t+1|t) = a_1 y(t) + \ldots + a_{n_y} y(t-n_y+1)\\ +b_1 u(t) + \ldots + b_{n_u} u(t-n_u+1)\\ +c_0 e(t) + c_1 e(t-1) + \ldots + c_{n_e} e(t-n_e)\end{aligned} \tag{3.9}$$

where $y(t)$ and $u(t)$ are the outputs and inputs (measurements), respectively, of the process system, $e(t) = y(t) - \widehat{y}(t|t-1)$ is the prediction error or residual term.

This form of equation, an SSP, is also called as the predictor form of the SISO ARMAX model.

Similarly, the MSP SISO ARMAX predictor, can be shown in the same form as the MSP ARX predictor and is expressed as:

$$\begin{aligned}\widehat{\mathbf{y}}(t+1|t-p+1) = a_1 \widehat{y}(t) + \ldots + a_{n_y} \widehat{y}(t-n_y+1)\\ +b_1 u(t) + \ldots + b_{n_u} u(t-n_u+1)\\ +c_0 e(t) + c_1 e(t-1) + \ldots + c_{n_e} e(t-n_e)\end{aligned} \tag{3.10}$$

4. Auto-Regressive Moving Average Exogenous Parameter Estimation

The parameters of the ARMAX model, $a_1, \ldots, a_{n_y}, b_1, \ldots, b_{n_u}, c_1, \ldots, c_{n_e}$ are to be determined from the measurement data, $y(t)$ and $u(t)$. The ARMAX predictor form represented by the Equation 3.9 is rewritten as:

$$\widehat{y}(t+1|t;\theta) = \varphi^T(t+1;\theta)\theta, \tag{3.11}$$

where,

$\varphi(t+1;\theta) = [y(t), \ldots, y(t-n_y+1), u(t), \ldots, u(t-n_u+1), e(t;\theta), \ldots, e(t-n_e+1;\theta)]^T$,

$\theta = [a_1, \ldots, a_{n_y}, b_1, \ldots, b_{n_u}, c_1, \ldots, c_{n_e}]^T$, and where,

$e(t,\theta) = y(t) - \widehat(y)(t|t-1;\theta)$.

ARMAX predictor is written as a scalar product between the data vector $\varphi(t+1;\theta)$ and the parameter vector θ. The data vector $\varphi(t+1;\theta)$ depends on the parameter vector θ. The Equation 3.11 is in the form of a pseudo-linear regression with the parameter vector θ as the regression vector. The least squares method is used to solve for θ. This is similar to ARX predictor, except that closed-form solutions are not possible [30].

D. Neural Network Based Nonlinear Methods

The end-to-end delay dynamics can be characterized as a complex system with the following characteristics [32]:

1. Dynamic behavior, because significant memory is exhibited by the measured response.

2. Nonlinear behavior, linear superposition principle cannot be applied because the measured response exhibits strong nonlinearities.

3. Stochastic behavior, because the measured response has a significant stochastic component, and,

4. Poorly understood behavior, that is uncertain and time varying because the processes governing the behavior of the system are difficult to precisely express and there are no validated process models.

The complexity of a system is inversely proportional to one's ability to model it, which means that if the complexity of a system increases then it becomes difficult to model such a system. Hence, this research also uses ANNs to model such a complex system. ANNs are very useful in predicting the dynamics of nonlinear systems [22].

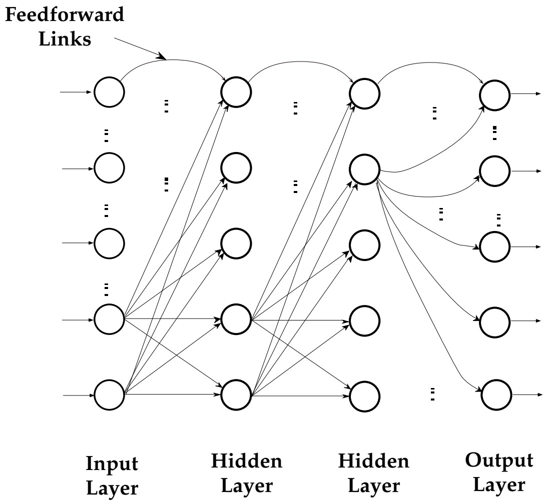

Fig. 5. Schematic Diagram of the FMLP Network. [23].

1. FMLP Network

Figure 5 shows a typical FMLP network. It is composed of an input layer, multiple hidden layers, and an output layer. This type of network can be considered as a nonlinear input-output model structure. The signals in this type of network are transmitted from one node to all other nodes in the next layer. The input and the output layers have linear discriminatory functions and the hidden layer has a sigmoid-type discriminatory function. An FMLP network with appropriate signals in the input layer can be very good at approximating static nonlinearities, i.e. memoryless nonlinear functions. Each processing element in an FMLP network is governed by the following equation:

$$x_{[l,i]} = \sigma_{[l,i]}\Big(\sum_{j=1}^{N_{[l-1]}} w_{[l-1,j][l,i]}\, x_{[l-1,j]} + b_{[l,i]} \Big), \tag{3.12}$$

for $i = 1, \ldots, N_{[l]}$ (the node index), and $l = 1, \ldots, \mathcal{L}$ (the layer index), where $x_{[l,i]}$ is the ith node output of the lth layer for sample k, $w_{[l-1,j][l,i]}$ is the weight which is a adjustable parameter connecting the jth node of the $(l-1)$th layer to the ith node of the lth layer, $b_{[l,i]}$ is the bias which is another adjustable parameter, of the ith node in the lth layer, and $\sigma_{[l,i]}(\cdot)$ is the discriminatory function of the i-th node in the l-th layer. The adjustable parameters, that is the weights and the biases are estimated iteratively using the available training algorithms.

2. FMLP Predictor Formulation

As mentioned earlier, prediction is defined as estimation of a variable of interest at a future point in time given the measured data up until and including the present time [30]. There are two different types of predictions that can be performed using neural networks, single-step-ahead prediction (SSP) and multi-step-ahead prediction (MSP). SSP refers to the prediction of the output of a given system at time $t+1$, given the inputs and outputs until time t, whereas MSP refers to prediction of the output of a given system at time $t+1$, given the inputs and outputs until $t-p+1$, where p is a positive integer greater than 1.

An FMLP can be used as a nonlinear predictor by configuring the input and output layers to represent the measured and predicted system inputs and outputs. If the input layer is set to

$$x_{[1]}(t+1) = \mathcal{U}(t), \tag{3.13}$$

then, the SSP and MSP can be represented by the following equations:

$$\widehat{\mathbf{y}}_{\mathbf{NN}}(t+1|t;\mathcal{W}) = \mathcal{F}\Big(\mathcal{U}(t);\mathcal{W}\Big), \tag{3.14}$$

$$\widehat{\mathbf{y}}_{\mathbf{NN}}(t+1|t-p+1;\mathcal{W}) = \mathcal{F}\Big(\widehat{\mathcal{U}}(t-p+1);\mathcal{W}\Big), \tag{3.15}$$

respectively, where p is a positive integer greater than one, $\mathcal{W}$ is determined by the learning algorithm, $\mathcal{F}$ denotes the nonlinear transformation between the network inputs and outputs.

3. FMLP Learning Algorithm

The main objective of SI is to determine an appropriate algorithm, which changes the parameters of the model structure, based on a given set of observations. This is called parameter estimation, but in the ANN literature, it is called learning. Learning actually means adjusting the weights and biases of the network to get the most suitable approximation between the input-output data. Backpropagation (BP) learning is the most commonly used learning method for FMLP.

There are several variations of the BP algorithm. The BP algorithm used in this research utilizes teacher forcing (TF). The learning algorithm for using TF is the gradient descent algorithm minimizing a mean-squared-error (MSE), which is the objective function. The primary mechanism of this learning method is the adjustment of the network weights and bias terms, until the MSE between the prediction and the observation is less than a prescribed tolerance.

In this thesis, the off-line derivation of the algorithm for the TF case is presented for the TF case. Let us consider an estimation data set which is expressed as follows:

$$S \equiv (u_n(t), y_m(t)), \forall t = 1, \ldots, NP; n = 1, \ldots, N_{[1]}; m = 1, \ldots, N_{[\ell]}, \tag{3.16}$$

where NP is the total number of data pairs in the estimation data set, $N_{[\ell]}$ is the number of nodes in the layer ℓ.

The objective of the learning algorithm is to determine the change in the network weights $\omega_{[\ell-1,i]}$, $\omega_{[\ell,j]}$ and the bias terms $b_{[\ell,i]}$ for all i,j and ℓ, such that the following error function:

$$E \equiv \frac{1}{2}\sum_{t=1}^{NP} E(t+1) \equiv \frac{1}{2}\sum_{t=1}^{NP}\sum_{t=1}^{N_{[\ell]}}[y_{N\hat{N},j}(t+1|t) - y_j(t+1)]^2, \tag{3.17}$$

is minimized. This error function is used when ANN predictors are developed.

The NP pairs of input-output data, is used for training. This data is used by the neural network again and again, till it reproduces them to within a prescribed tolerance. During this training, the network weights and bias terms are updated using the following gradient descent rules:

$$\Delta\omega_{[\ell-1,j][\ell,i]} = -\eta\sum_{t=0}^{K}(\frac{\partial E(t+1)}{\partial\omega_{[\ell-1,j][\ell,i]}}), \tag{3.18}$$

$$\Delta\omega_{[\ell,j][\ell,i]} = -\eta\sum_{t=0}^{K}(\frac{\partial E(t+1)}{\partial\omega_{[\ell,j][\ell,i]}}), \tag{3.19}$$

$$\Delta b_{[\ell,i]} = -\eta\sum_{t=0}^{K}(\frac{\partial E(t+1)}{\partial b_{[\ell,i]}}), \tag{3.20}$$

where K is set to 1 for individual update and NP for batch update, η is the learning rate, which is set interactively by the user to allow for proper convergence.

E. Chapter Summary

In this chapter, the linear and neural network based nonlinear methods for SSP and MSP have been discussed. The linear tools provide accurate linear system models, but occasionally fail to model nonlinear systems or systems having complex dynamics. The need for neural network based estimation arises because such model structures can model certain systems very effectively, as the system complexity increases. The equations presented in this chapter have been taken as it is from [30] to retain their originality.

CHAPTER IV

MEASUREMENT AND ANALYSIS OF END-TO-END PACKET DELAY

A. Introduction

The first step for modeling the end-to-end packet delays is to obtain sufficient amount of data necessary for developing predictors so as to predict the delay associated with each packet or a group of packets ahead of time. A lot of research has been done in area of packet measurements, but with the aim of modeling the behavior of Internet delays statistically, rather than modeling them empirically. This chapter provides information regarding the data collection process used to collect simulated traffic data as well as real traffic data. A brief analysis on such data is also presented.

B. Assumptions

The various assumptions made during the collection of data from ns-2 are mentioned in this section. The most important assumption is that all packets sent from the source take the same unique path to reach their destination. In the collection of real traffic, routes may change very abruptly or sometimes may take months for a change to occur. Since different routes have different propagation delays associated with them, consecutive packets may arrive out of sequence. This is most commonly termed as re-ordering of packets. In this research, packet re-ordering is not considered in the predictor development process.

C. Collection of Simulated Data

This section deals with the generation of artificial or simulated TCP and UDP traces from the ns-2. The data generated by ns-2 will now be called as simulated data.

This section provides details about the topologies used for the simulated traffic data and their respective analysis. Figure 6 shows the basic topology used for collecting simulated data from ns-2 using both UDP and TCP.

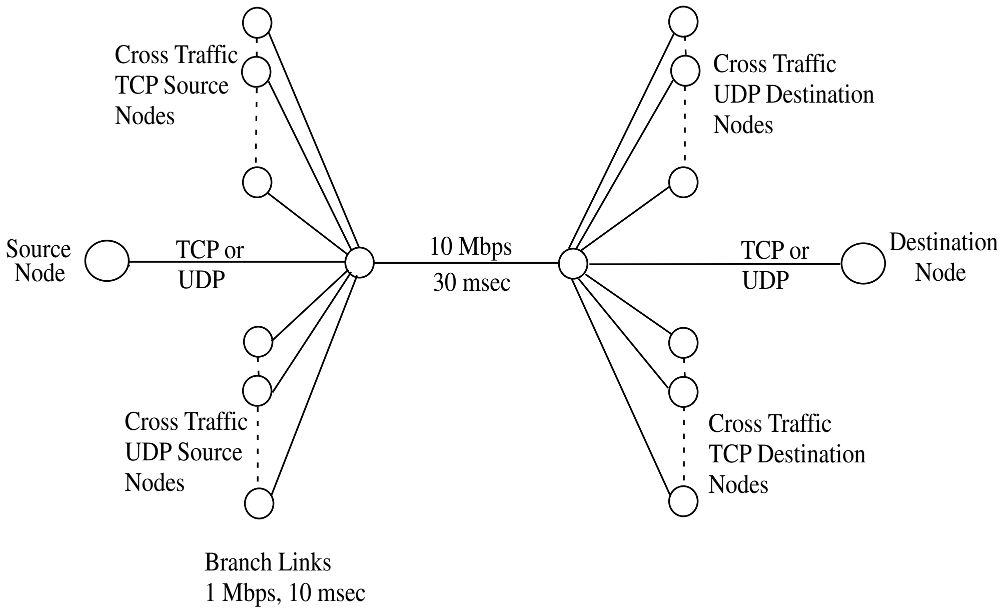

Fig. 6. Network Topology for Simulated Data.

1. Network Simulations for Modeling a UDP Flow

The model corresponding to the UDP has the end-to-end source-destination nodes using UDP for transport. The model considered has the following network architecture:

1. There are 100 TCP nodes in the network. Each node acts as a source as well as sink to allow two-way flow in the network. Each TCP source node has FTP as the application sending packets of size 512 bytes.

2. There are 10 UDP nodes in the network which also act as sources and sinks to

allow two-way flow of data in the network. The UDP source sends a constant bit rate. The nominal UDP packet size is 512 bytes and the nominal inter-departure time is 5 msecs.

3. There are 2 UDP end-to-end nodes which send the traffic flow being modeled in this research. The remainder nodes create the cross-flow traffic and they are not being observed.

4. The 100 TCP nodes along with the 12 UDP nodes are connected to a bottleneck link that has 10 mbps bandwidth and a propagation delay of 30 ms.

5. Each of these 112 nodes are connected to the bottleneck link that has a bandwidth of 1 mbps and a propagation delay of 10 ms.

2. Types of Traces

The traces collected from the ns-2 simulator can be classified as those of a TCP and a UDP flow. In each of these cases two categories, there are again two different classes of test cases which are classified as send rate test cases and cross-traffic test cases. A *Baseline* test case is defined as the test case which uses 100 TCP and 10 UDP nodes for cross-traffic and a constant send rate of 100 kbps for the flow of interest. Eight different traces have been collected by varying the cross-traffic and six different traces have been obtained by varying the send rate. In the variable cross-traffic scenarios, the cross-traffic is varied from 150% increase to 90% decrease from the baseline test case. Figure 7 shows an example UDP trace generated from the ns-2 simulator where the send rate is kept at a constant 400 kbps. The simulation scenario involves baseline cross-traffic. In the send rate test case scenarios, several traces have been generated using constant send rate ranging from 50 kbps to 800 kbps. Losses for each of the test cases and the training set are provided in terms of percentage in Appendix A.

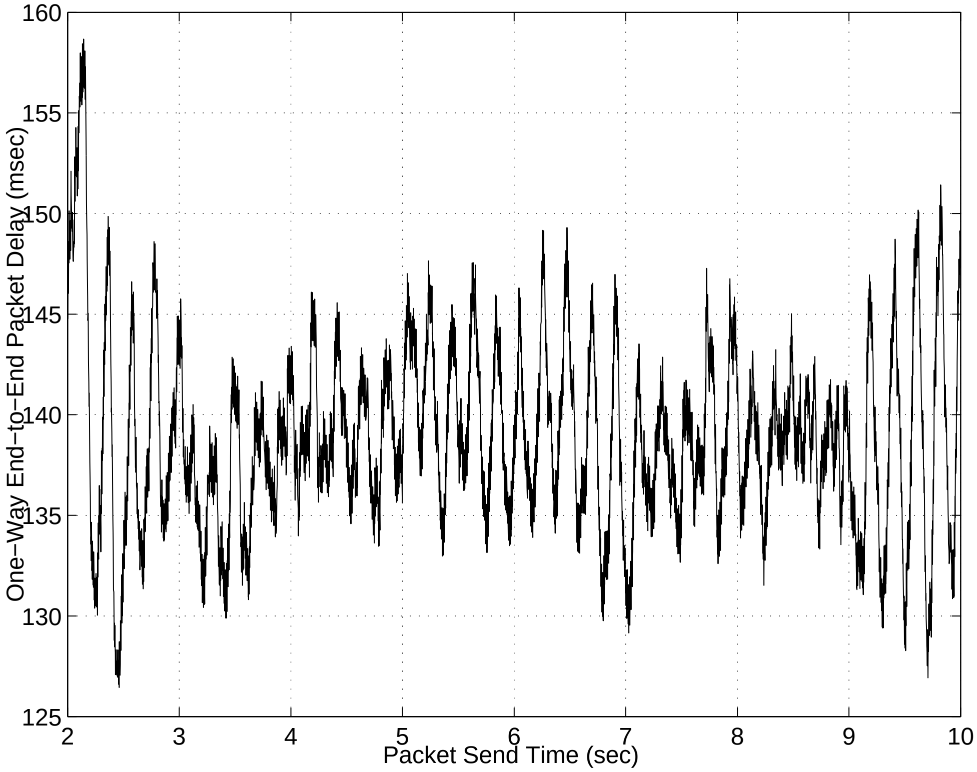

Fig. 7. One-Way End-to-end UDP Delays; Constant Send Rate of 400 Kbps with Baseline Cross-traffic.

The next step is to re-sample the data based on the principle of moving average window. The Moving average window principle is based on computing the average of the data in a particular time interval and then moving this window forward in small time steps. This principle is applied in this research. The moving window is set at 100 msec and the window is moved in steps of 50 msec. According to this, an average of all the delays in a window of 100 msec is computed and is represented at the leading edge of the window. This window is now moved by a time interval of 50 msec and the moving average is computed for the time interval of another 100 msec and this is continued over the entire data. Thus at the end of this sampling, the data obtained is a moving average time-series of one-way end-to-end delays that will be used for prediction purposes.

The important thing to be noted while applying the moving average principle is that some of these windows might contain packet losses. These packet losses have been modeled in three different ways in this study as follows:

- **No Loss Model**: In this case, the packet losses have been identified as zero delays. This means that packet losses have been neglected while calculating the moving average of the one-way end-to-end delays.

- **Interpolated Delay Loss Model**: This model involves calculating a linearly interpolated value of the delay for a lost packet based on the slope of previous delays.

- **High Delay Loss Model**: In this method, every packet loss is attributed a high delay value. The high delay value used in this research is 10% over the maximum delay observed over the complete sample of data collected. The maximum delays is over the packets that successfully reach their destinations.

Figure 8 shows the moving average of one-way end-to-end delays using UDP for a constant send rate of 400 kbps and baseline cross-traffic.

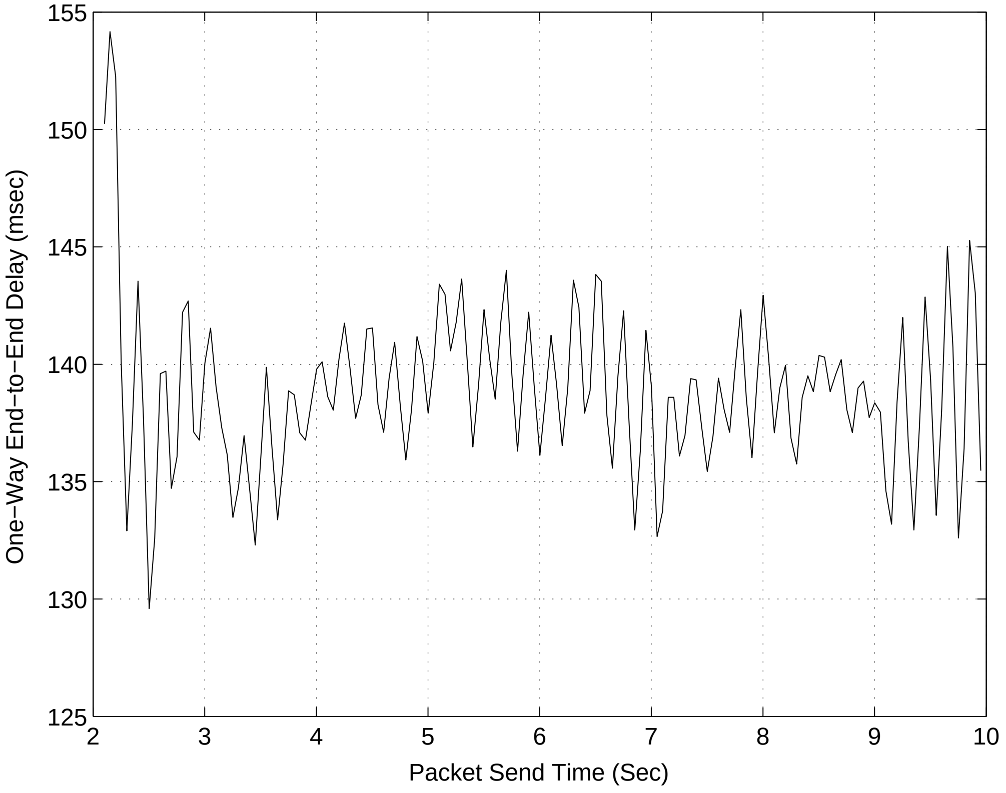

Fig. 8. Moving Average of One-Way End-to-end UDP Delays; Constant Send Rate of 400 Kbps with Baseline Cross-traffic.

3. Auto-correlation of the Simulated Traffic Data

Figure 9 represents the normalized auto-correlation function of moving average one-way end-to-end delays using UDP for a constant send rate of 400 Kbps and baseline cross-traffic. The auto-correlation coefficients are plotted for 20 lags. Figure 10 represents the normalized auto-correlation function of the same for 100 lags. From the figures, it can be observed that the autocorrelation function with a lag of 20 delays, does not drop below 0.86, whereas for 100 lags it can be seen that the autocorrelation function reaches 0.38. A preliminary study on autocorrelation functions of various UDP traces is necessary as it helps in choosing the order of the linear predictive models like AR and ARMA.

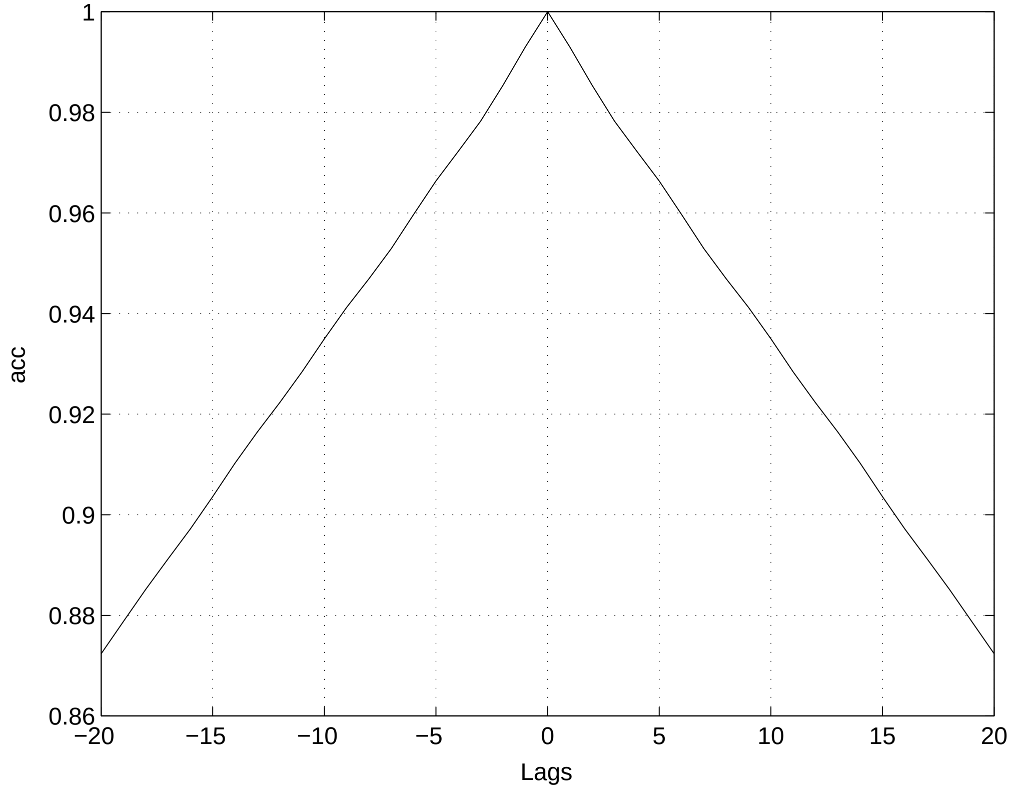

Fig. 9. Auto-Correlation Function of Moving Average One-Way End-to-end UDP Delays for 20 lags; Constant Send Rate of 400 Kbps with Baseline Cross-traffic.

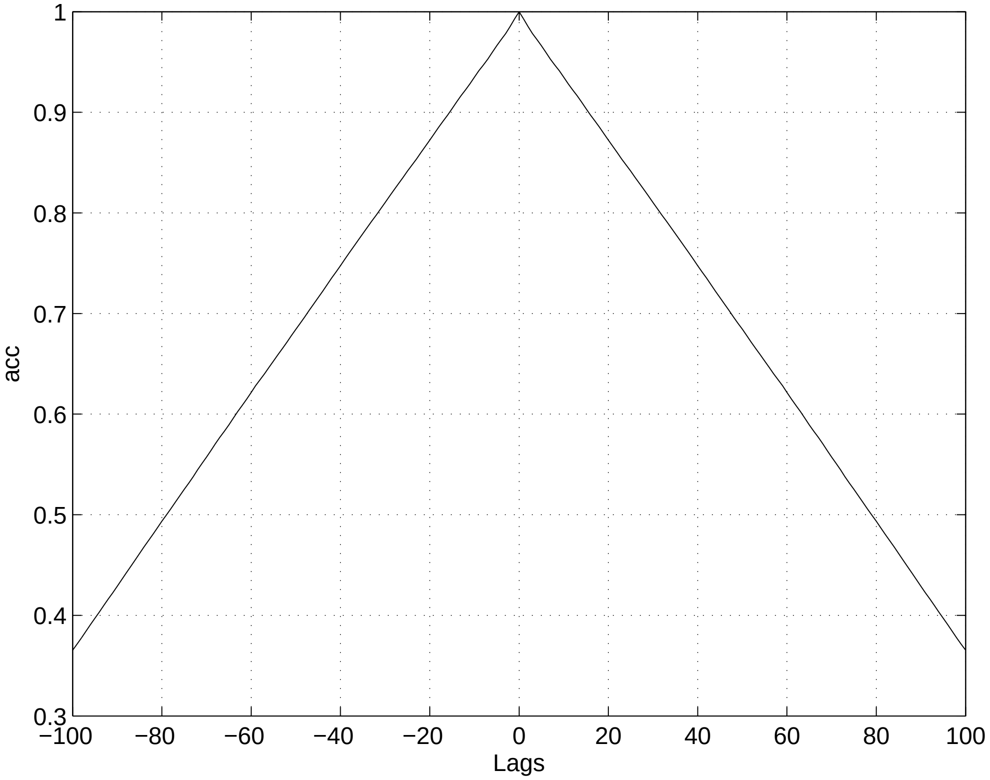

Fig. 10. Auto-Correlation Function of Moving Average One-Way End-to-end UDP Delays for 100 lags; Constant Send Rate of 400 Kbps with Baseline Cross-traffic.

D. Data Collection of Real Traffic Data

This section describes the generation of real TCP and UDP traffic traces using the tools TPBAT and UPBAT developed by Yeom [15]. The data generated by these tools will be called as real traffic data. This section explains the various methods used to collect the real traffic data. The initial motivation for the collection of the real traffic data is to compose them to the simulated data through the auto-correlation functions. Eventually as the proposed approach reaches maturity real traffic data must be used for predictor validation. Figure 11 is the basic topology used for collecting the real traffic data using the UDP Packet Behavior Analyzing Tool (UPBAT) and TCP Packet Behavior Analyzing Tool (TPBAT). These tools require two nodes, a source node for sending and a destination node for receiving packets. A node at Texas A & M University (TAMU) is used as the source host and another node at Boston University (BU) is used as the destination host. University of New Mexico (UNM) is also used as the destination host for some data. The significant factor in using these tools is that the delay measurements obtained using these tools are the delays measured from application layer of the source host to the application layer of the destination host.

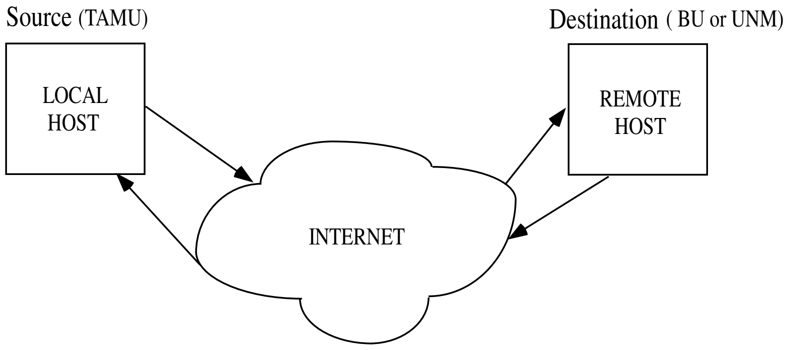

Fig. 11. Network Setup for Real Traffic Data Collection.

1. Experiment Setup for UDP

A measurement tool by the name UDP Packet Behavior Analyzing Tool (UPBAT) has been developed by Yeom [15] for measuring one-way delays of UDP packets. This tool uses two separate threads for sending and receiving UDP data packets. A server program is started on the remote host or the destination host and a client program is started on the local host or the source host. Packets are sent from the source host to the destination host and once the packets reach the destination host, they are echoed back to the source host with the destination time-stamp on it. Thus this tool generates both forward and reverse delays for UDP packets. This tool has been originally written to send UDP packets with a constant send rate or at constant interdeparture rate. This research is also interested in studying the effects of variable send rate on the network. Thus this tool has been modified to allow for both constant and variable bit rates.

2. Types of Traces

The traces generated from ns-2 simulator have a special advantage over the traces generated from the above tools. In ns-2, both the send rate and the cross-traffic can be controlled. Although send rate can be varied in a real network, the cross-traffic cannot be influenced in a controlled manner, other than performing the experiments at different times of the day. The cross-traffic is not under the control of the user. Cross-traffic can affect the data collected depending upon the time of the day as the traffic is high during day time as compared to night time. Figure 12 shows a representative UDP trace from a node at TAMU to another node at BU. The send rate used is a variable bit rate whose average is 60 kbps.

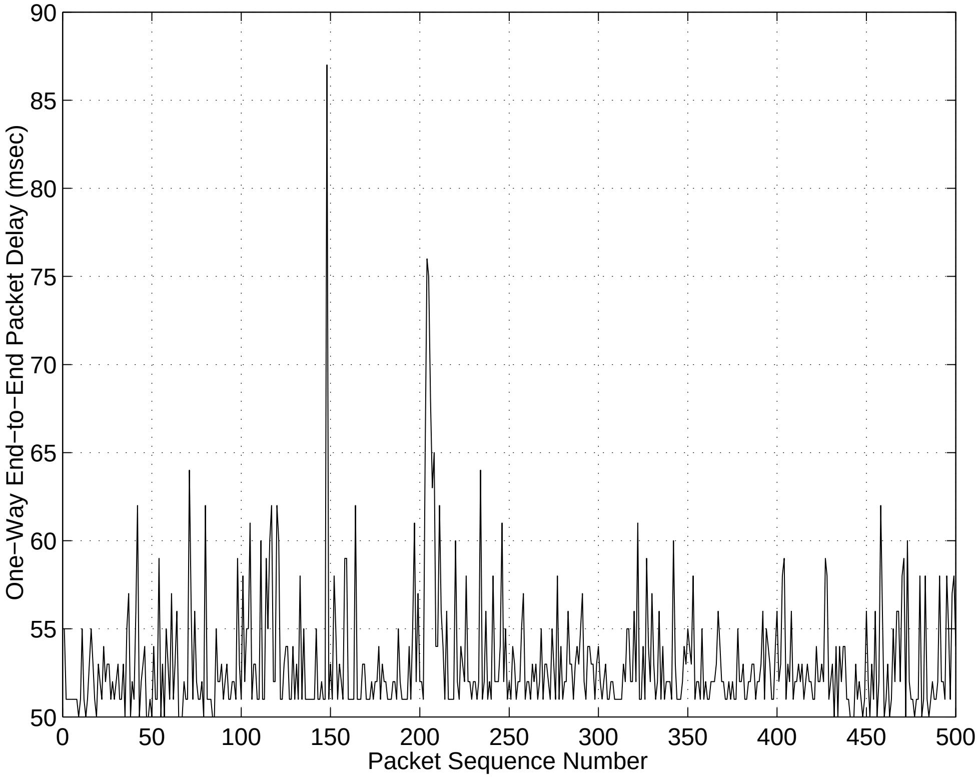

Fig. 12. Typical UDP Trace Between TAMU and BU; Average Send Rate of Approximately 60 Kbps.

3. Auto-correlation of Real Traffic Data

Figures 13 and 14 represent the normalized auto-correlation functions of one-way end-to-end delays of the UDP trace shown in the Figure 12. From the Figures 13 and 14, it can be observed that the auto-correlation coefficients do not go below 0.99 for 20 lags and 0.982 for 100 lags. This decay rate indicates a long range temporal dependency among the one-way delays of UDP. The direct practical implication of this observation is that it is difficult to obtain an empirical model for these data used in this research. The data exhibits self-similar patterns as with any network traffic indicating that it has similar statistical properties at different time scales [33] but has long range temporal dependency of a very high order.

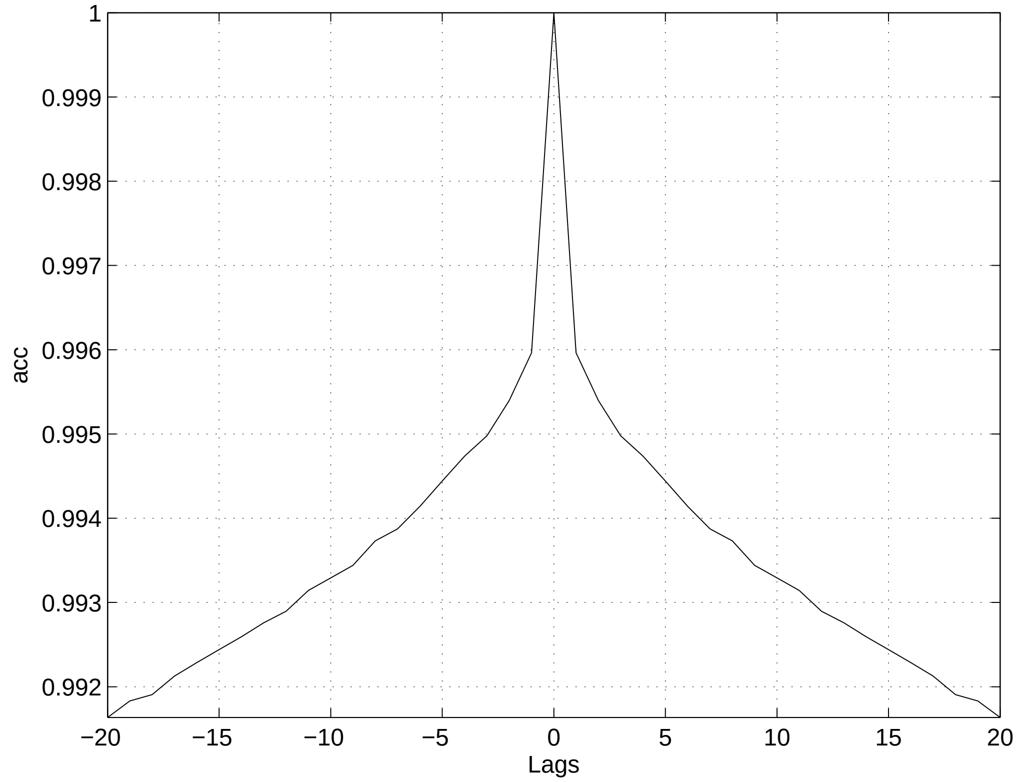

Fig. 13. Auto-Correlation Function of a Typical UDP Trace Between TAMU and BU for 20 lags; Average Send Rate of Approximately 60 Kbps.

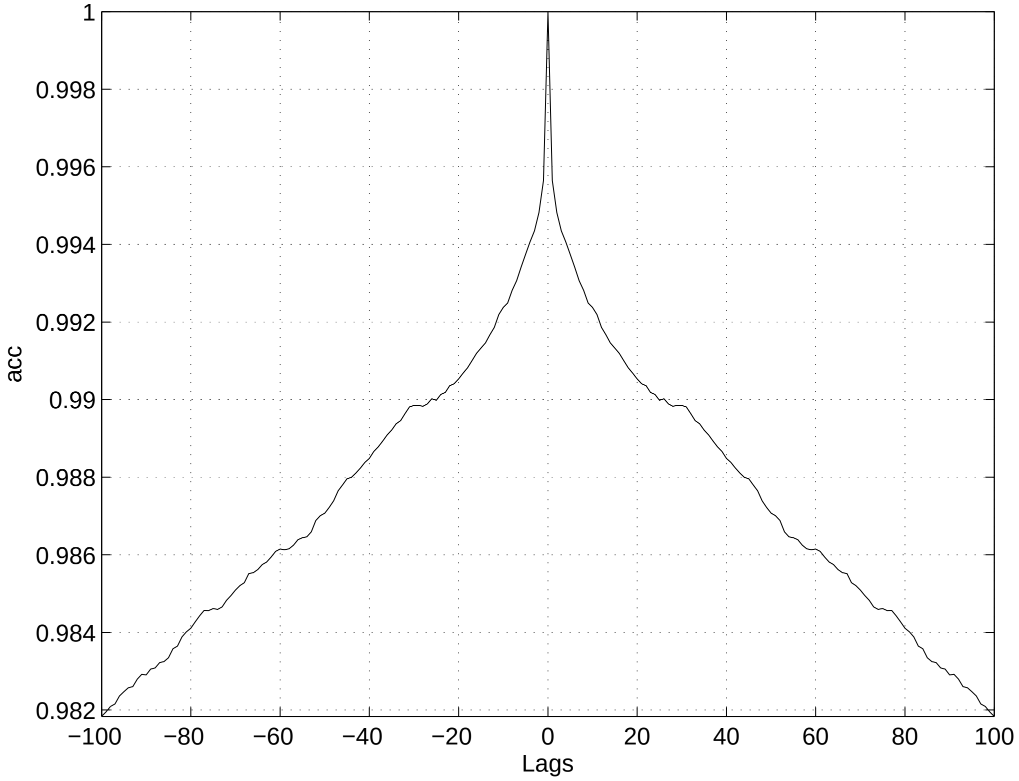

Fig. 14. Auto-Correlation Function of a Typical UDP Trace Between TAMU and BU for 100 lags; Average Send Rate of Approximately 60 Kbps.

E. Chapter Overview

This chapter provides a detailed description of the generation of UDP and TCP traffic traces. The traces have been generated from ns-2 simulations, as well as from real traffic experiments. The concept and the need to compute the moving average of the original time-series is also explained in this chapter. Finally each of these traces has been represented as a moving average time-series of one-way end-to-end delays. The next chapter presents empirical models for SSP of the moving average end-to-end delays.

CHAPTER V

END-TO-END DELAY SINGLE-STEP-AHEAD PREDICTOR DEVELOPMENT AND TESTING

A. Introduction

In this chapter, the linear and nonlinear empirical models for the moving average end-to-end delays in a best-effort network are developed and tested. The empirical modeling in both linear and nonlinear cases is done assuming that the network is a "black-box" and the delay data is modeled as a time-series. SI techniques such as AR, ARMA and neural network based techniques such as FMLP have been employed for modeling. The main objective of this chapter is to design and develop an empirical model capable of performing accurate single-step-ahead prediction.

The linear methods AR and ARMA were presented in some detail in Chapter III. The neural network based nonlinear model FMLP and RMLP was also discussed in chapter III. The current chapter is organized in the following way. A description of the performance metrics used in this research is given in the next section and the subsequent sections deal extensively with the SSP based on linear and nonlinear models. Section 5 presents a comparative study of the various predictors used for single-step-ahead prediction. Section 6 is the chapter summary.

B. Performance Metrics

In this research, three different types of errors are used as performance metrics for the predictors developed. The first performance metric is called *Mean Square Error* (MSE_1). It is the ratio between the sum of the square of the prediction error and the sum of the square of the input data, as follows: Equation 5.1.

$$MSE_1 = \frac{\sum_{k=1}^{N}\left(x(k) - \widehat{x}(k|k-1)\right)^2}{\sum_{k=1}^{N} x(k)^2} \times 100 \qquad (5.1)$$

where N is the total number of data points, $x(k)$ is the observation, and $\widehat{x}(k|k-1)$ is the prediction data. Since MSE is the inverse of *Signal-To-Noise Ratio*(SNR), it can be considered as one of the best performance metric which gives a good picture on the quality of the predictor.

The second performance metric is called *Maximum Absolute Error* (MAE). It is the maximum prediction error or in other words the maximum error between the observation and the prediction and it is defined as

$$MAE = \max_{1 \le k \le N} |x(k) - \widehat{x}(k|k-1)| \qquad (5.2)$$

This error is useful for identifying the regions where the predictor fails.

The third metric used in this research is another variant of Mean Square Error (MSE$_2$), but calculated in a different way. It is the ratio of the sum of the square of the prediction error and sum of the square of the input data from which the mean has been removed. It is defined by

$$MSE_2 = \frac{\sum_{k=1}^{N}\left(x(k) - \widehat{x}(k|k-1)\right)^2}{\sum_{k=1}^{N}\left(x(k) - \bar{x}\right)^2} \times 100 \qquad (5.3)$$

where $\bar{x}$ is the arithmetic mean of the observation $x(k)$. This metric is sometimes used as it might give a better indication of the prediction as compared to earlier discussed MSE$_1$ for signal with large variations.

C. Description on Training, Testing and Validation Data Sets

Data is needed for training or model parameter estimation purposes in system identification methods. A simulation using ns-2 is performed with the baseline cross-traffic of 100 TCP and 10 UDP nodes providing the training data. This cross-traffic scenario is used as the standard baseline test set and then it is compared with other test cases in this research. The validation of the predictive models is done by varying the cross-traffic about the baseline cross-traffic. The source bit rate used for generating the data is also varied from 120 kbps to 800 kbps. The variable bit rate is obtained by keeping the packet size constant and varying the inter-departure time of the send packets. The simulation is performed for 100 seconds and the data is post-processed. Which means a time-series of the moving average of the end-to-end delay using the *no loss delay model* and the *interpolated loss delay model* is obtained for system identification purpose. The data based on *no loss delay model* is used for linear modeling. For nonlinear modeling, data based on *interpolated loss delay model* is used for training the neural network. The data thus obtained is divided into 3 sets namely training data, testing data and validation data as shown in Figure 15.

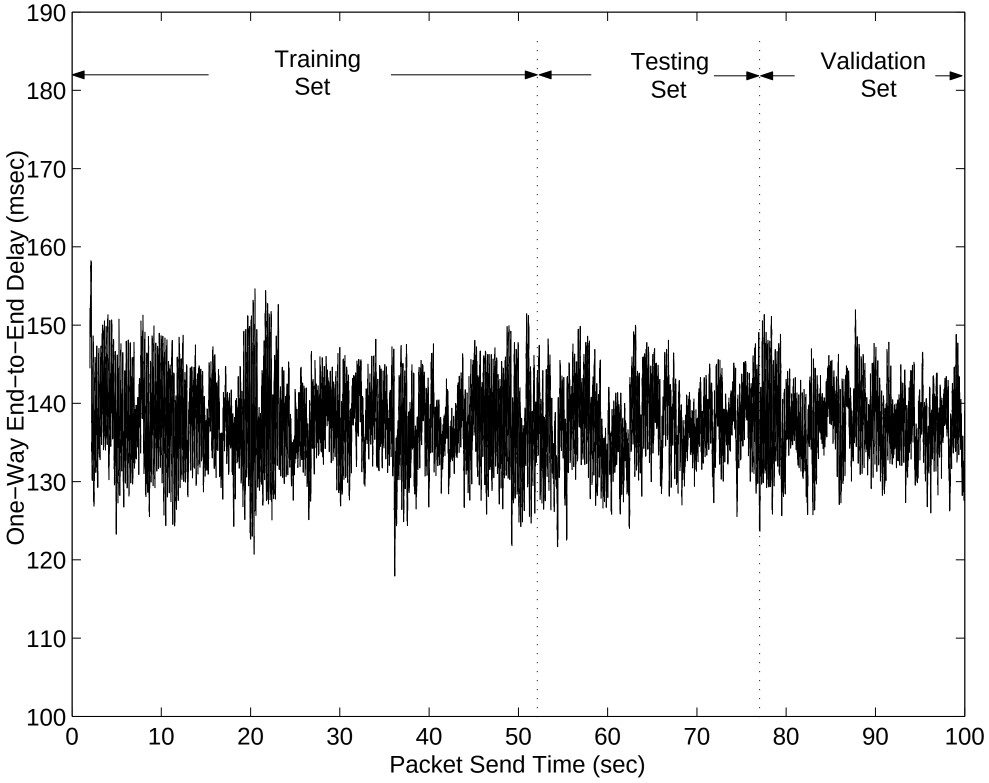

Fig. 15. Representation of Training, Testing and Validation Data Sets.

Apart from the traffic data used for training, a set of 14 different test cases have been simulated in ns-2 for validation purposes. These can broadly classified as send rate test sets and cross-traffic test sets. In send rate test sets, 6 different data sets have been generated. In each of these cases, the cross-traffic is set to 100 TCP and 10 UDP nodes, whereas the bit-rate is different for each case. The different bit-rates used are 50 kbps, 100 kbps, 200 kbps, 400 kbps, 500 kbps and 800 kbps. In cross-traffic test cases, 8 different data sets have been generated. The cross-traffic scenarios explored in this research are vary from 150% increase to 90% decrease from the standard baseline cross-traffic. Table II shows the number of TCP and UDP nodes used for cross-traffic in terms of percentage increase or decrease from the baseline test set.

Table II. Cross-traffic Test Cases.

Cross-traffic (percentage)	Cross-traffic (number of nodes)
90% decrease	10 TCP + 1 UDP
75% decrease	25 TCP + 3 UDP
50% decrease	50 TCP + 5 UDP
25% decrease	75 TCP + 8 UDP
Baseline	100 TCP + 10 UDP
25% increase	125 TCP+ 12 UDP
50% increase	150 TCP+ 15 UDP
100% increase	200 TCP+ 20 UDP
150% increase	250 TCP+ 25 UDP

Apart from the above described test cases, a special test case is also simulated in ns-2 to take into effect, the real traffic conditions in the Internet. In this special test case, the cross-traffic is varied from 75 % decrease in cross-traffic to baseline cross-traffic and then to 50 % increase in cross-traffic. This variation can be clearly seen in the Figure 16. In this figure, the cross-traffic which is represented as a fraction of cross-traffic over baseline cross-traffic is plotted against packet send time. The send rate used in this test case is a constant 100 Kbps.

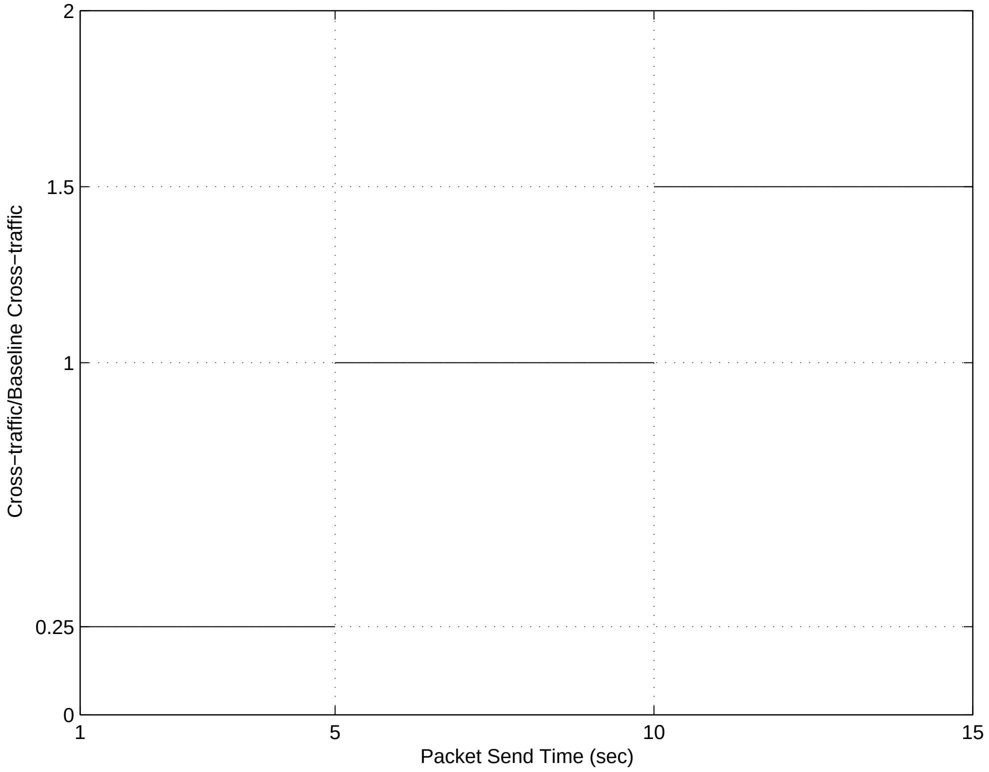

Fig. 16. Variation of Cross-traffic in a Special Test Case

D. Development of Linear Predictors and Training, Testing and Validation Results

After generating the desired data, the next step is to use system identification techniques to obtain the best model. A SSP in this study means a 50 msec ahead prediction of the moving average one-way end-to-end packet delay. An AR predictor with model structure {4} and ARMA with model structure {4 6} gave the best fit for the traffic data. Tables III and IV summarize the different errors discussed in the section 5.2 for the SSP obtained from the AR and ARMA predictors respectively.

Table III. Single-Step-Ahead Prediction of Training, Testing and Validation Data Using AR.

Data Set	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
Training Set	0.009	4.9	12.2
Testing Set	0.008	5.1	13.1
Validation Set	0.007	4.2	12.9

Table IV. Single-Step-Ahead Prediction of Training, Testing and Validation Data Using ARMA.

Data Set	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
Training Set	0.008	4.8	11.8
Testing Set	0.008	4.9	12.9
Validation Set	0.006	4.1	13.1

Tables III and IV show the various errors MSE_1, MAE, and MSE_2 for training, testing and validation data sets. It can be observed from the above tables that the ARMA model gives the least training MSE_2 error of 11.8% compared to 12.2% for AR. The testing MSE_2 error for ARMA is also less compared to AR, however, the validation error is lower in the case of AR which is 12.9%. However, these errors are close to each other and hence cannot be used as a criteria for picking the best model. The MAE for both the predictors seems to be around 5 msecs for all the data sets used.

E. Performance Evaluation of Linear Predictors

Performance evaluation of the two linear predictors AR and ARMA is now presented. This is done by testing each of these models with 6 different send rate test cases and 8 cross-traffic test cases. Figure 17 shows the SSP of moving average one-way end-to-end delay using the AR model. It depicts the measured and the predicted moving average delays for a constant send rate of 100 Kbps with the baseline cross-traffic. Figure 18 depicts the predicted moving average delays and the actual UDP packet delays. The figure shows that the predictor can capture the dynamics of the network by predicting the moving average one-way end-to-end delays. It should be noted that the maximum prediction error with respect to the actual packet delays is approximately 10 msec. The figure shows the prediction given by the AR is good enough when compared with the observed packet delays. Figure 19 shows the single-step-ahead prediction of moving average one-way end-to-end delay using the ARMA model. It depicts the measured and the predicted moving average delays for a constant send rate of 100 Kbps with 25% decrease in baseline cross-traffic. Figure 20 depicts the predicted moving average delays and the actual UDP packet delays. This indicates that the ARMA model can capture the end-to-end delay dynamics of the network as well under changing cross-traffic conditions. Figure 21 and Figure 22 depict the prediction of moving average one-way end-to-end delays using the AR model for the special test case of variable cross-traffic.

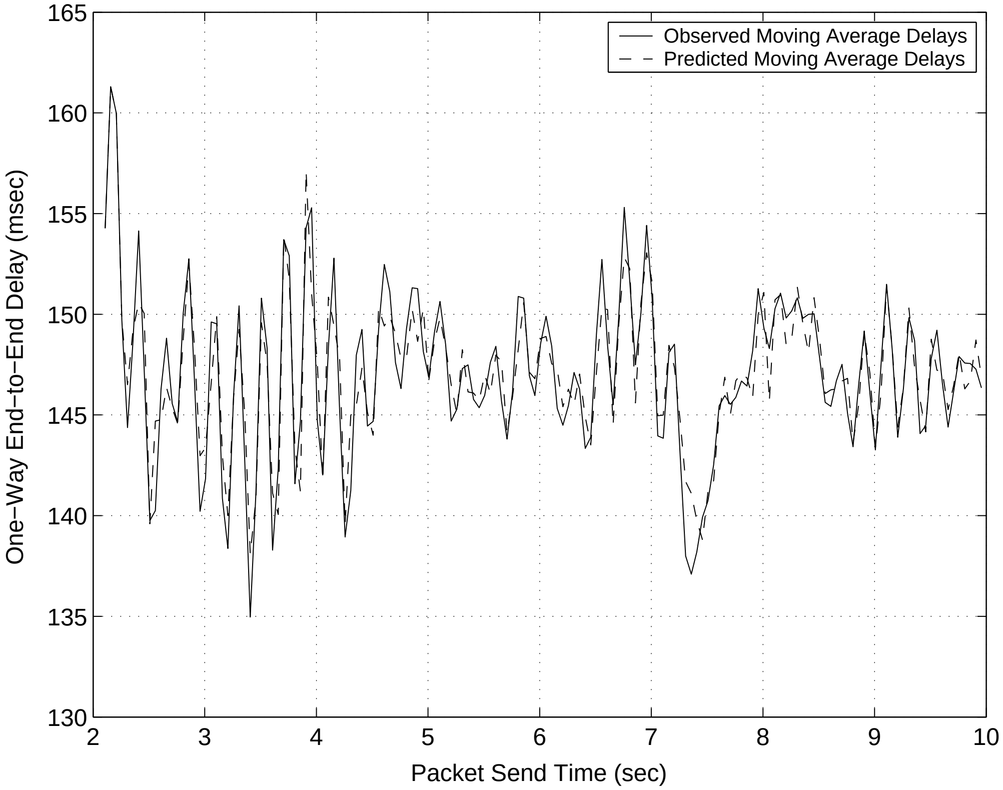

Fig. 17. Single-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the AR Model; Constant Send Rate of 100 Kbps with Baseline Cross-traffic Showing Moving Averaged Delays.

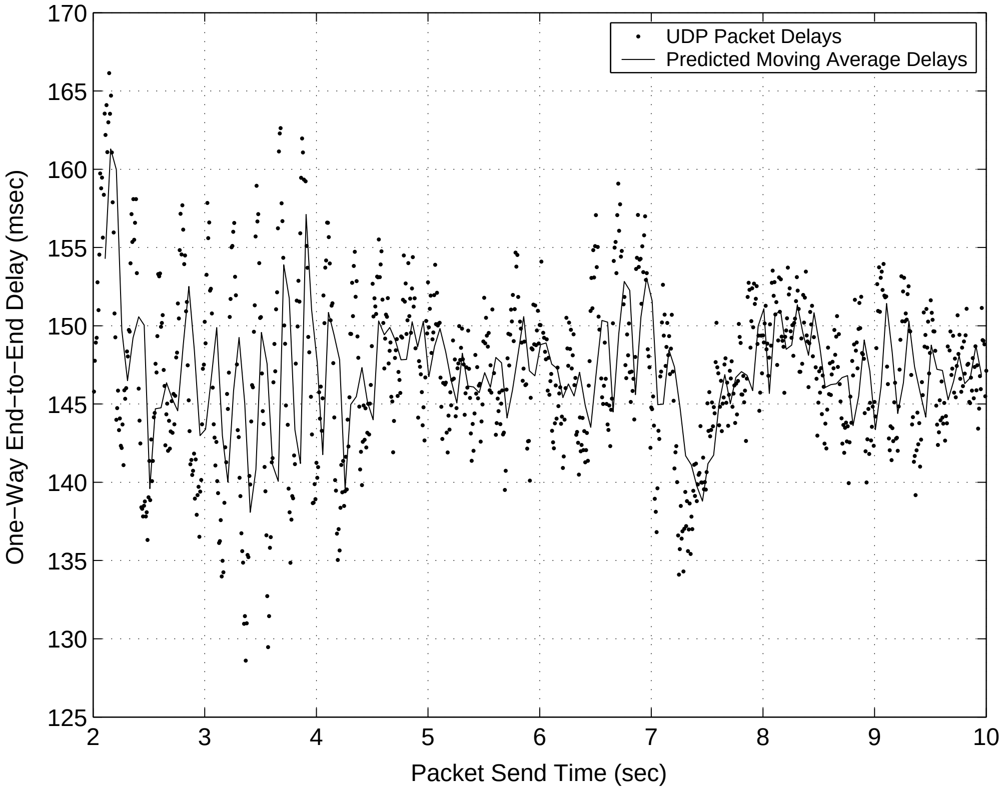

Fig. 18. Single-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the AR Model; Constant Send Rate of 100 Kbps with Baseline Cross-traffic Showing Actual Delays.

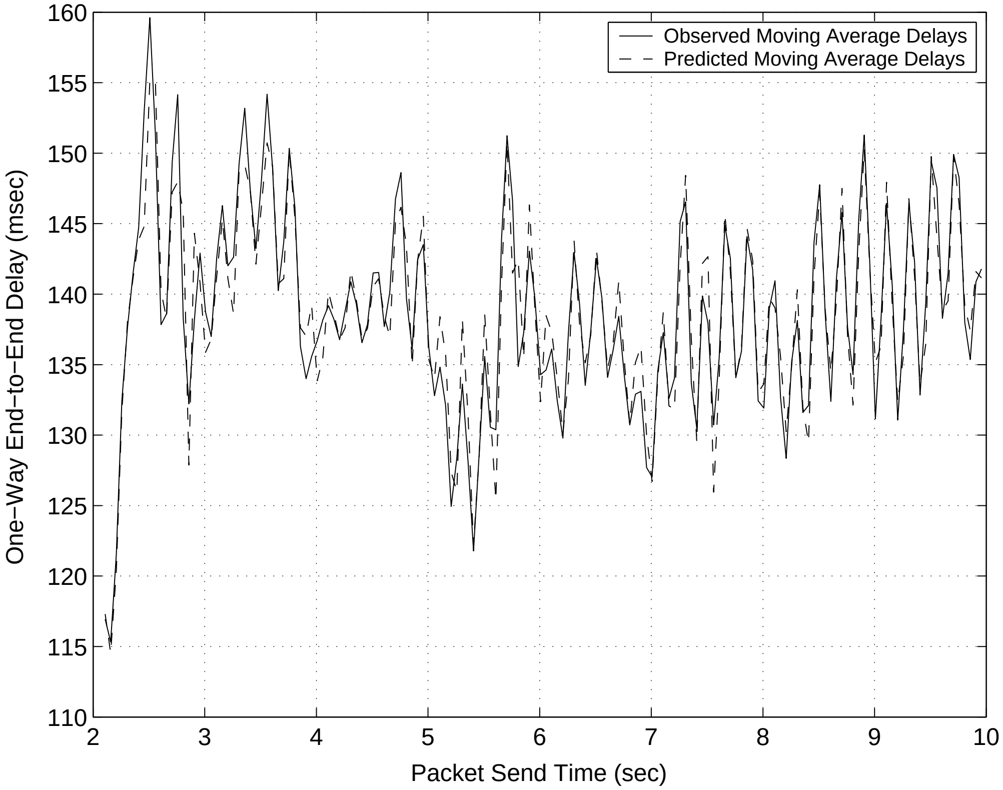

Fig. 19. Single-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the ARMA Model; Constant Send Rate of 100 Kbps with 25% Decrease in Baseline Cross-traffic Showing Moving Averaged Delays.

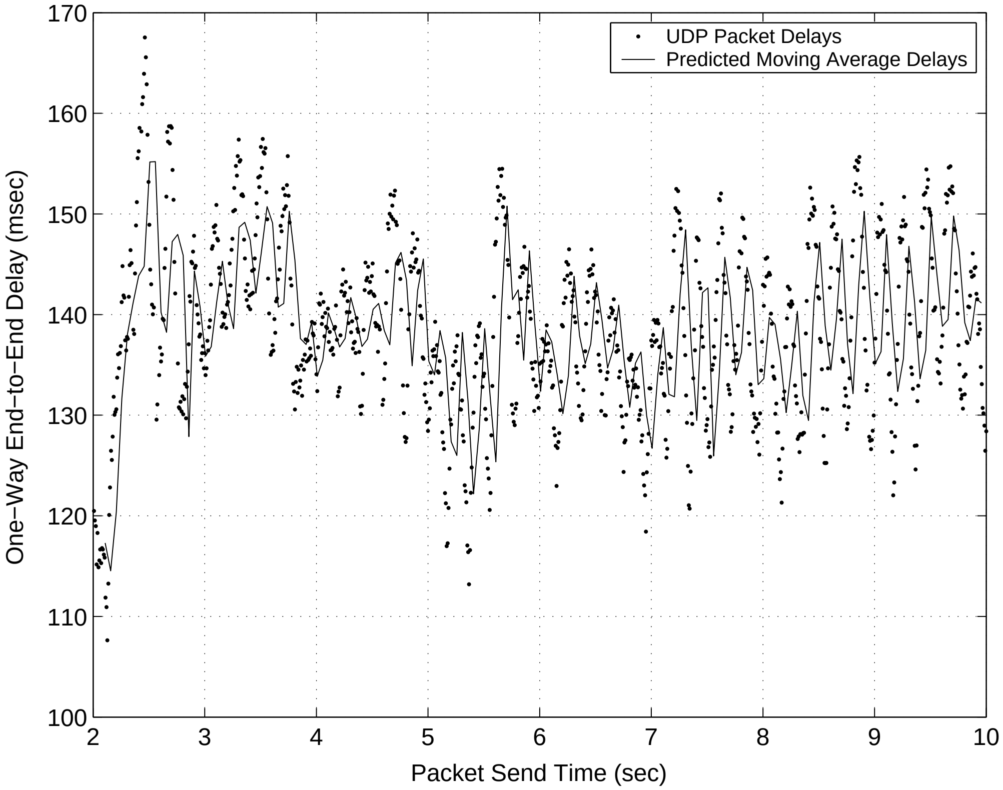

Fig. 20. Single-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the ARMA Model; Constant Send Rate of 100 Kbps with 25% Decrease in Baseline Cross-traffic Showing Actual Delays.

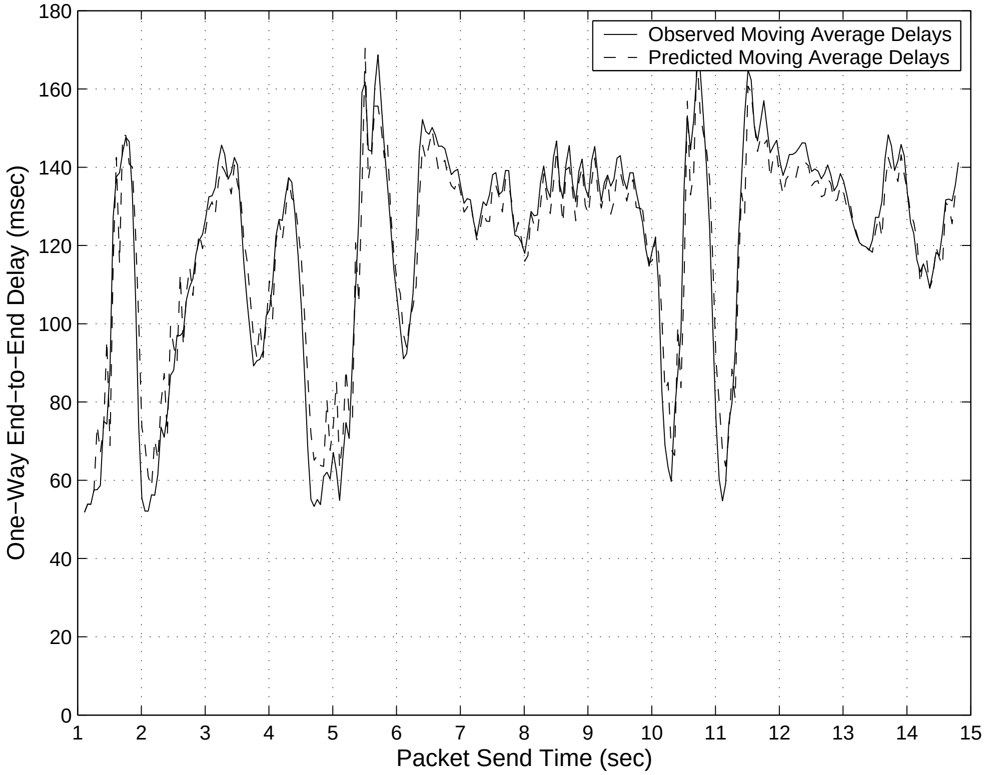

Fig. 21. Single-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the AR Model; Constant Send Rate of 100 Kbps with Variable Cross-traffic Showing Moving Averaged Delays.

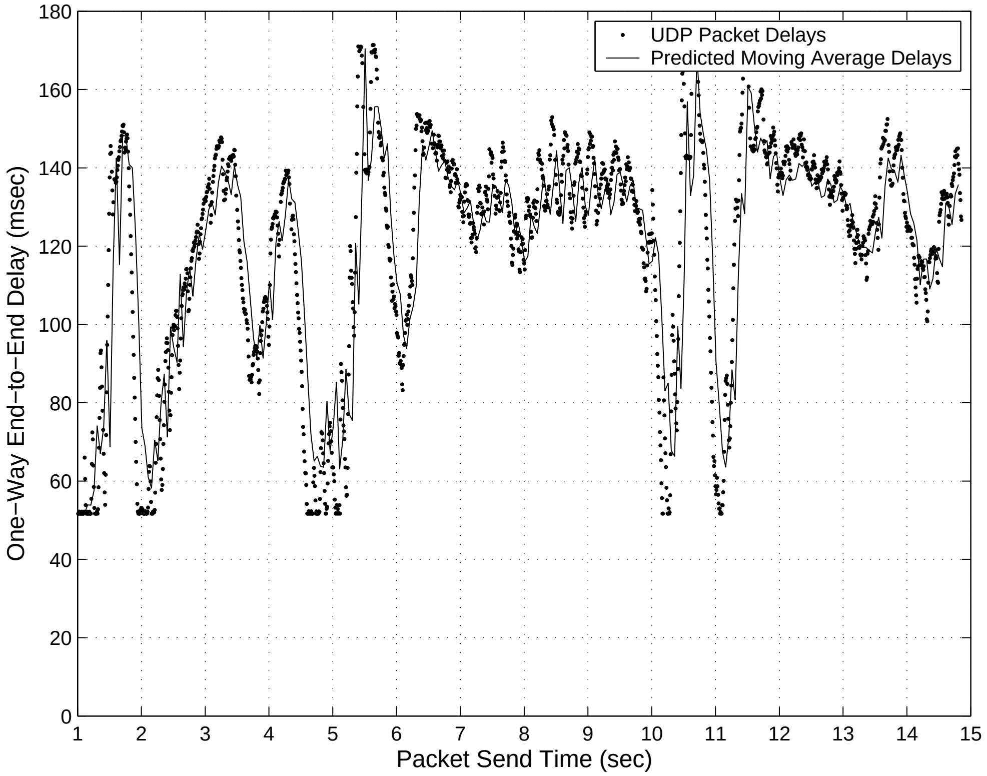

Fig. 22. Single-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the AR Model; Constant Send Rate of 100 Kbps with Variable Cross-traffic Showing Actual Delays.

F. Development of Nonlinear Predictors and Training, Testing and Validation Results

The development of predictors based on neural networks is different than linear predictor development. In this research, FMLP networks are trained for SSP. The predictors are developed using teacher forcing (TF). There are no inputs for the predictor as the cross-traffic, which has the highest impact on the delays, cannot be measured and it is considered a disturbance of the model.

During the training process the performance of the predictor is determined using the MSE_1 error. The trained predictors are evaluated in terms of this error on a validation data which is part of the data but not used in the estimation of the weights. The best model is obtained after an extensive search over several possible FMLP architectures. In this case an FMLP with model structure $\{4\ 7\ 1\}$ which translates into 4 input layer nodes, 7 hidden layer nodes and 1 output layer node has given the best SSP. The performance results of the FMLP predictor on the training, testing and validation sets are summarized in Table V. The SSP obtained through this model is comparable to those obtained from AR and ARMA, but the validation error is a little higher compared to the error obtained using the linear predictors. The MAE which was around 5 msecs for the AR and ARMA models is around 13 msecs for the FMLP model.

G. Performance Evaluation of the Nonlinear Predictor

Six different send rate test cases and eight different cross-traffic test cases are used for validating the FMLP predictor with model structure $\{4\ 7\ 1\}$. Figure 23 shows the SSP of moving average one-way end-to-end delay using the AR model. It depicts the measured and the predicted moving average delays for a constant send rate of

Table V. Single-Step-Ahead Prediction of Training, Testing and Validation Data Using FMLP.

Data Set	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
Training Set	0.010	13.1	13.8
Testing Set	0.009	4.2	13.0
Validation Set	0.011	12.5	21.3

100 kbps with the 50% increase in the baseline cross-traffic. Figure 24 depicts the predicted moving average delays and the actual UDP packet delays. It can be seen that the predictor suceeds in capturing the dynamics of the network by predicting the moving average one-way end-to-end delays. The longest prediction error is no more than 10 msec. The figure also shows a good prediction of the model for a 50 percent increase in cross-traffic. This implies that the developed model can predict delays well even if there are variations in the cross-traffic.

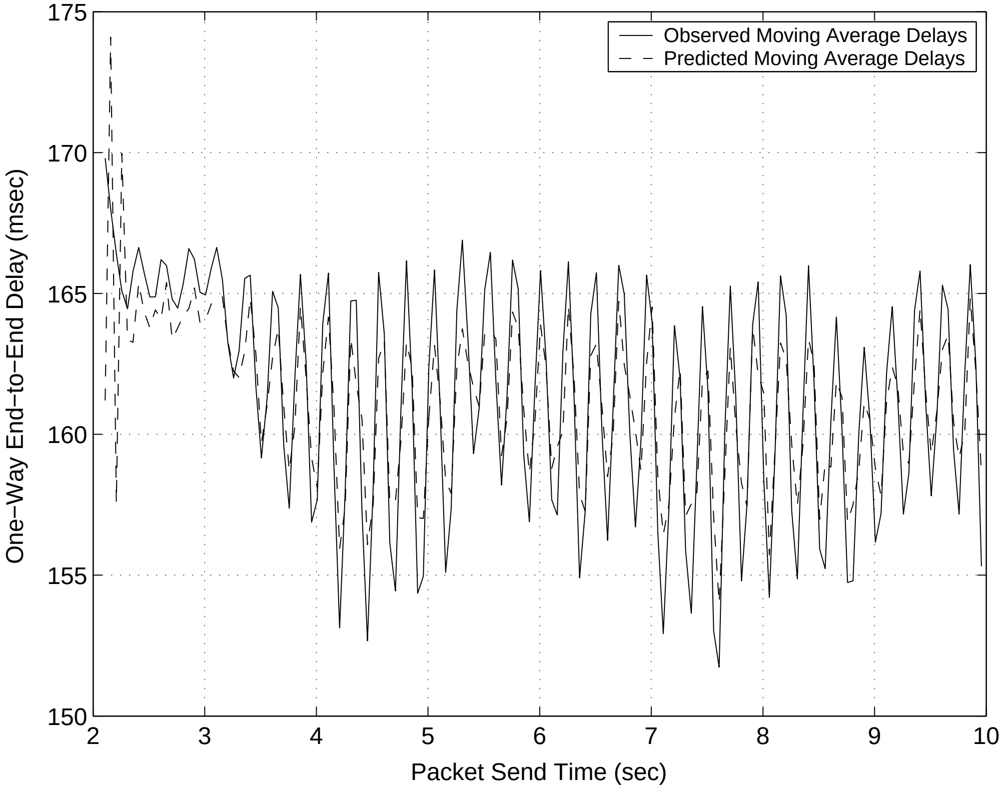

Fig. 23. Single-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the FMLP Model; Constant Send Rate of 100 Kbps with 50% Increase in Baseline Cross-traffic Showing Moving Averaged Delays.

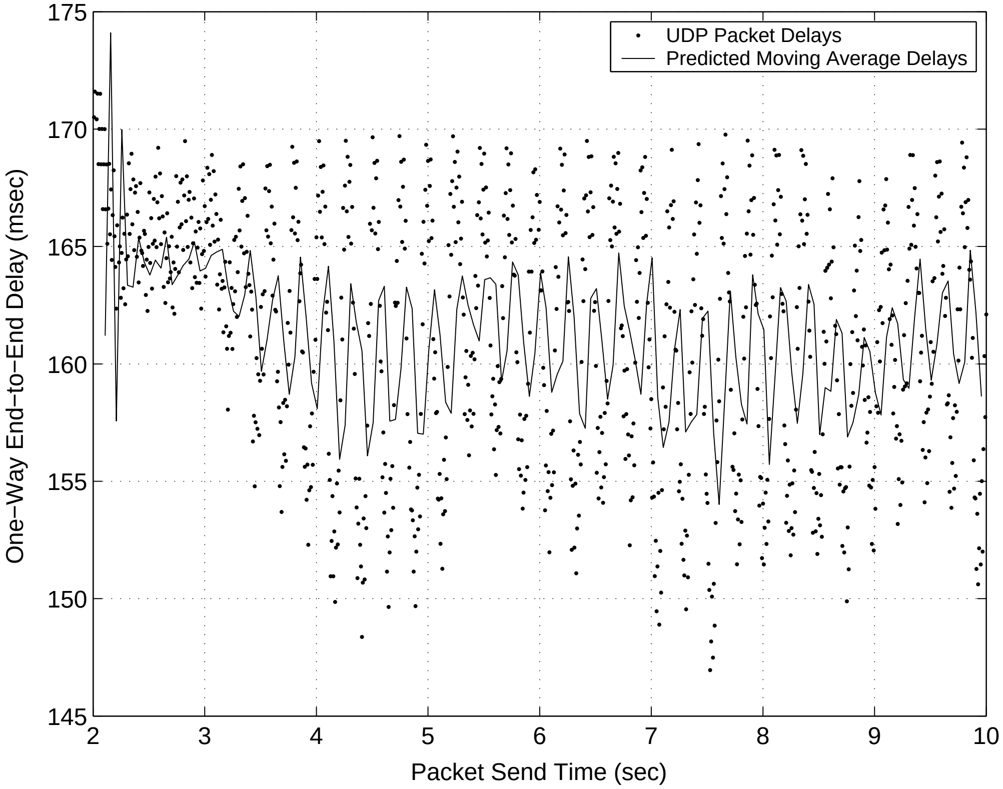

Fig. 24. Single-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the FMLP Model; Constant Send Rate of 100 Kbps with 50% Increase in Baseline Cross-traffic Showing Actual Delays.

H. Comparison of Single-Step-Ahead Predictor Performance

The results of the SSP on all the 14 test cases using AR, ARMA and FMLP are tabulated in this section. Tables VI and VII show the performance evaluation results of the AR predictor on the various send rate and cross-traffic test cases in terms of the three performance indicators MSE_1, MAE, and MSE_2. It can be seen from Table VI that the AR model gives the best prediction for the bit-rate of 500 kbps and the 50 kbps test set has the highest error among all the bit-rate test cases. From the Table VI, it can be seen that predictor based on AR model is able to give an accurate prediction for the send rate test cases. The MAE in the 50 kbps send rate test case is 7.3 msec which is more than the MAE of the baseline test case (around 5.0 msec). It can also be deduced from the Table VII that the model fails for the cross-traffic test case which has 10 TCP nodes and 1 UDP node or 90% decrease. MSE_1 does not give a correct indication of the prediction in this case because MSE_1 is very small compared to MSE_2. Moreover MSE_2 gives consistent results for all the test cases. Table VII shows that the developed AR predictor works fine around +150% to -75% of the baseline cross-traffic with the highest MAE being 16.4 msec for the test case with 75% decrease from the baseline cross-traffic. Another deduction that can be drawn is that MSE_2 increases with increase in the cross-traffic.

Tables VIII and IX present the ARMA predictor results. The developed predictor performs well within a send rate range of 50 kbps to 800 kbps. It also performs well in the cross-traffic range of +150% increase to -75% decrease of the standard baseline cross-traffic. From the Table IX, it can be observed that the MAE of 16.1 msec is the highest for 25% decrease in cross-traffic, whereas the corresponding MAE in AR based prediction is 7.8 msec. MSE_2 increases with the increase in cross-traffic and decreases with increasing send rate.

Table VI. AR Single-Step-Ahead Predictions for Send Rate Test Cases.

Send Rate	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
800 Kbps	0.012	4.7	9.7
500 Kbps	0.000	3.9	6.9
400 Kbps	0.005	3.2	9.1
200 Kbps	0.009	4.8	7.7
100 Kbps	0.011	4.4	15.0
50 Kbps	0.017	7.3	21.3
Baseline	0.007	4.2	12.9

Table VII. AR Single-Step-Ahead Predictions for Cross-traffic Test Cases.

Cross-traffic	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
90% decrease	0.002	0.6	207.4
75% decrease	0.354	16.4	7.3
50% decrease	0.126	12.8	6.7
25% decrease	0.033	7.8	12.6
25% increase	0.011	3.8	17.1
50% increase	0.014	4.1	21.2
100% increase	0.015	5.8	26.7
150% increase	0.008	4.8	24.2
Baseline	0.007	4.2	12.9

Table VIII. ARMA Single-Step-Ahead Predictions for Send Rate Test Cases.

Send Rate	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
800 Kbps	0.012	4.3	9.5
500 Kbps	0.009	3.9	6.6
400 Kbps	0.006	2.8	9.9
200 Kbps	0.009	4.9	7.7
100 Kbps	0.011	4.0	14.2
50 Kbps	0.017	6.6	21.1
Baseline	0.006	4.1	13.1

Table IX. ARMA Single-Step-Ahead Predictions for Cross-traffic Test Cases.

Cross-traffic	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
90% decrease	0.002	0.7	255.4
75% decrease	0.369	16.0	7.6
50% decrease	0.130	13.4	6.9
25% decrease	0.030	16.0	11.4
25% increase	0.010	4.2	16.0
50% increase	0.011	3.9	18.3
100% increase	0.014	5.4	24.0
150% increase	0.007	4.3	21.6
Baseline	0.006	4.1	13.1

Tables X and XI present the FMLP results of SSP of moving average one-way end-to-end delays. It can be concluded from the results that the validation errors on all the test cases is definitely higher than the errors obtained from linear models. The MAE is very high for the -75% and -50% cross-traffic test cases indicating an inaccurate prediction at certain send times.

From the above tables, it can be safely concluded that ARMA with model structure {4 6} performs accurate SSP followed by AR with model structure {4} and FMLP with model structure {4 7 1}.

Table X. FMLP Single-Step-Ahead Predictions for Send Rate Test Cases.

Send Rate	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
800 Kbps	0.034	13.6	26.6
500 Kbps	0.026	13.0	18.1
400 Kbps	0.016	11.6	26.5
200 Kbps	0.015	9.2	11.7
100 Kbps	0.014	7.3	18.2
50 Kbps	0.019	7.4	23.1
Baseline	0.011	12.6	21.4

Table XI. FMLP Single-Step-Ahead Predictions for Cross-traffic Test Cases.

Cross-traffic	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
90% decrease	0.003	0.8	320.1
75% decrease	0.799	55.3	16.5
50% decrease	0.689	72.7	36.6
25% decrease	0.067	21.6	25.2
25% increase	0.018	12.8	29.1
50% increase	0.018	8.8	28.8
100% increase	0.019	8.1	33.5
150% increase	0.011	6.7	32.0
Baseline	0.011	12.6	21.3

The special test case is developed to test the AR predictor. The data for the test case is obtained from the running a simulation in ns-2 by varying the cross-traffic. The MSE_1 for this case is 0.42 % and the MAE is 27.2 ms. The MSE_2 is 7.6 %. The MAE is high compared to other single-step-ahead predictions from the AR predictor.

I. Chapter Overview

This chapter explains the training, testing and validation of the linear and nonlinear predictors. The various performance metrics used for the evaluation of SSP are also discussed in this chapter. The chapter provides the results of SSP using both linear and nonlinear models on several different test cases. The models give accurate SSP on most of the test cases.

CHAPTER VI

END-TO-END DELAY MULTI-STEP-AHEAD PREDICTORS DEVELOPMENT AND TESTING

A. Introduction

The previous chapter dealt with the development and validation of SSP predictors. The current chapter is an extension of the previous chapter. In this chapter, the various procedures followed in the development and evaluation of MSP predictors are presented. A delay prediction must be in a finite future horizon and beyond the next packet so that the prediction can actually be used in congestion control/avoidance algorithms. The delay estimate obtained from SSP is useful in predicting the delay associated with the next packet. But as the prediction horizon grows, SSP becomes meaningless and there is a need for MSP. Summary of the MSP results of the moving average one-way end-to-end packet delays are presented in this chapter.

B. Performance Metrics

Similar to section 5.2, three different types of errors are used as performance metrics for the predictors developed to perform MSP. The first performance metric is called *Mean Square Error* (MSE$_1$). It is the ratio between the sum of the square of the prediction error and the sum of the square of the input data and is defined as follows:

$$MSE_1 = \frac{\sum_{k=1}^{N}\left(x(k)-\widehat{x}(k|k-p)\right)^2}{\sum_{k=1}^{N}x(k)^2}\times 100 \tag{6.1}$$

where N is the total number of data points, $x(k)$ is the observation, and $\widehat{x}(k|k-1)$ is the prediction, and p is the number of time steps representing MSP. Since MSE is the inverse of *Signal-To-Noise Ratio* (SNR).

The second performance metric is called *Maximum Absolute Error* (MAE). It is the maximum prediction error or in other words the maximum error between the observation and the prediction and it is defined by

$$MAE = \max_{1 \leq k \leq N} |x(k) - \widehat{x}(k|k-p)| \tag{6.2}$$

This error is useful for identifying the regions where the predictor fails.

The third metric used in this research is another variant of Mean Square Error (MSE$_2$). It is the ratio of the sum of the square of the prediction error and sum of the square of the input data from which the mean has been removed and it is given by

$$MSE_2 \quad = \quad \frac{\sum_{k=1}^{N} (x(k) - \widehat{x}(k|k-p))^2}{\sum_{k=1}^{N} (x(k) - \bar{x})^2} \times 100 \tag{6.3}$$

where $\bar{x}$ is the arithmetic mean of the observation $x(k)$. This metric is gives a better indication of the prediction as compared to earlier MSE$_1$ for large signal variations.

C. Description on Training, Testing and Validation Data Sets

The data used for training, testing and validation of MSP predictors is same as the data used for training SSP schemes. The data used for training AR and ARMA predictors is based on *no loss delay model* whereas the data used for training FMLP network is based on *interpolated loss delay model.* The source send rate used to generate the delay measurements is varied from 120 kbps to 800 kbps. The ns-2 simulation is run for 100 seconds, of this the first 52 seconds of data is used for training, next 25 seconds of data for testing and the remaining data is used for validation purpose. The data thus, is divided into 3 sets namely training data set, testing data set and validation data set.

D. Development of Linear Predictors and Training, Testing and Validation Results

Linear models AR and ARMA are used to develop MSP predictors. These methods are implemented in Matlab SI toolbox which is used for this research for empirical modeling purposes. The different time steps considered for this research are two-step, three-step, four-step and ten-step. Each time-step is 50 msec. A two-step-ahead prediction means a 100 msec ahead prediction of moving average of one-way end-to-end packet delays. Similarly, 150 msec, 200 msec, 500 msec ahead prediction correspond to three-step, four-step and ten-step-ahead prediction, respectively. The AR predictor with model structure {4} and ARMA predictor with model structure {4 6} gave the best fit for this data. From the Tables XII and XIII, it can be deduced that both AR and ARMA predictor performances are similar in nature. MSE_2 of two-step-ahead prediction on training data is 40% when compared to single-step-ahead prediction which is around 12%. The MSE_2 for three-step-ahead and four-step-ahead predictors is around 50%. For ten-step-ahead prediction, MSE_2 is around 70%. The following subsections present the training, testing and validation results in terms of various performance indicators.

1. Two-Step-Ahead Prediction

Tables XII and XIII summarize the training, testing and validation errors from two-step-ahead prediction using AR and ARMA predictors. In Table XII, the performance indicators are MSE_1, MAE, and MSE_2. The MSE_2 error is around 50% on both testing and validation data sets and the MAE on the validation set is no more than

10.3 msecs. The Table XIII shows that validation MSE_2 of ARMA is more than that obtained from AR model. However, selection of a predictor cannot be done based on these errors.

Table XII. Two-Step-Ahead Prediction of Training, Testing and Validation Data Using AR.

Data Set	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
Training Set	0.031	13.2	41.6
Testing Set	0.033	8.1	50.4
Validation Set	0.027	10.3	50.9

Table XIII. Two-Step-Ahead Prediction of Training, Testing and Validation Data Using ARMA.

Data Set	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
Training Set	0.030	13.2	40.8
Testing Set	0.032	7.9	48.3
Validation Set	0.027	10.3	51.1

2. Three-Step-Ahead Prediction

A three-step-ahead prediction gives prediction information every 150 msecs in the future prediction horizon. Tables XIV and XV summarize the training, testing and validation errors from three-step-ahead prediction using AR and ARMA predictors. As seen in the Table XIV, the MSE_2 on the validation data set is around 63% which is high compared to two-step-ahead prediction. Another important observation that can be made from the Tables XIV and XV is that the MAE is 10.274 msecs. This is the same for both AR and ARMA predictions. The possible reason that can be attributed to this observation is that the initial prediction is the same for both the models. It can also be deduced that the errors increase as the prediction horizon increases. In other words, as the time-step increases, the predictors start losing their accuracy.

Table XIV. Three-Step-Ahead Prediction of Training, Testing and Validation Data Using AR.

Data Set	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
Training Set	0.038	16.5	51.9
Testing Set	0.042	10.2	64.9
Validation Set	0.033	10.3	62.8

Table XV. Three-Step-Ahead Prediction of Training, Testing and Validation Data Using ARMA.

Data Set	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
Training Set	0.038	16.5	51.6
Testing Set	0.041	10.4	62.9
Validation Set	0.033	10.3	63.1

3. Four-Step-Ahead Prediction.

A four-step-ahead prediction means prediction information is obtained every 200 msecs in the future prediction horizon. Tables XVI and XVII summarize the training, testing and validation errors from four-step-ahead prediction using AR and ARMA predictors. From the Tables XVI and XVII, it can be observed that the MAE's on the data sets remain the same owing to the reason mentioned in the previous section. Also the MSE_2 does not degenerate to higher values as compared to MSE_2 obtained from three-step-ahead prediction.

Table XVI. Four-Step-Ahead Prediction of Training, Testing and Validation Data Using AR.

Data Set	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
Training Set	0.040	16.5	53.7
Testing Set	0.043	10.3	65.8
Validation Set	0.033	10.3	63.2

Table XVII. Four-Step-Ahead Prediction of Training, Testing and Validation Data Using ARMA.

Data Set	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
Training Set	0.039	16.5	53.3
Testing Set	0.042	10.5	63.8
Validation Set	0.033	10.3	63.5

4. Ten-Step-Ahead Prediction

A ten-step-ahead prediction means prediction information is obtained every 500 msecs which is half a second in future prediction horizon. Tables XVIII and XIX summarize the training, testing and validation errors from ten-step-ahead prediction using AR and ARMA predictors. The MSE_2 is around 88% for the AR based ten-step-ahead prediction and is around 89% for the ARMA based prediction. It can be seen that the errors increased to a very high value when compared to SSP.

Table XVIII. Ten-Step-Ahead Prediction of Training, Testing and Validation Data Using AR.

Data Set	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
Training Set	0.054	16.5	73.0
Testing Set	0.059	11.9	89.6
Validation Set	0.046	10.9	88.1

Table XIX. Ten-Step-Ahead Prediction of Training, Testing and Validation Data Using ARMA.

Data Set	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
Training Set	0.053	16.5	71.5
Testing Set	0.057	12.2	87.6
Validation Set	0.047	11.0	89.0

E. Performance Evaluation of Linear Predictors

The developed predictors have been trained and tested after performing an extensive search through several model structures. The 2 linear predictors AR with model structure {4} and ARMA with model structure {4 6} are found to give the best MSP performance. The next step is to validate these predictors on several different test cases. The test cases which are used for validating the SSP predictors will be used for validating the MSP predictors. This will be useful in comparing the various time-step-ahead predictors on a common scale. Two test cases are presented in this section for discussion. In the first test case, the source send rate is a constant bit-rate of 200 Kbps with baseline cross-traffic and in the second test case, the source send rate is a constant 50 kbps bit-rate with baseline cross-traffic. The following subsections deal with MSP using AR and ARMA predictors.

1. Two-Step-Ahead Prediction

Figure 25 depicts the two-step-ahead prediction of moving average one-way end-to-end delays using AR predictor. It shows the measured and predicted moving average delays for a constant send rate of 200 kbps with baseline cross-traffic. Figure 25 also shows a good two-step-ahead prediction is achieved though the MSE_2 for this test case is 27.6%. Figure 26 shows the predicted moving average delays and the actual UDP packet delays.

Figure 27 depicts the two-step-ahead prediction of moving average one-way end-to-end delays using ARMA predictor. It shows the measured and predicted moving average delays for a constant send rate of 50 Kbps with baseline cross-traffic. Figure 27 shows a reasonable good two-step-ahead prediction. The MSE_2 for this test case is 54.1%. Figure 28 shows the predicted moving average delays and the actual UDP packet delays.

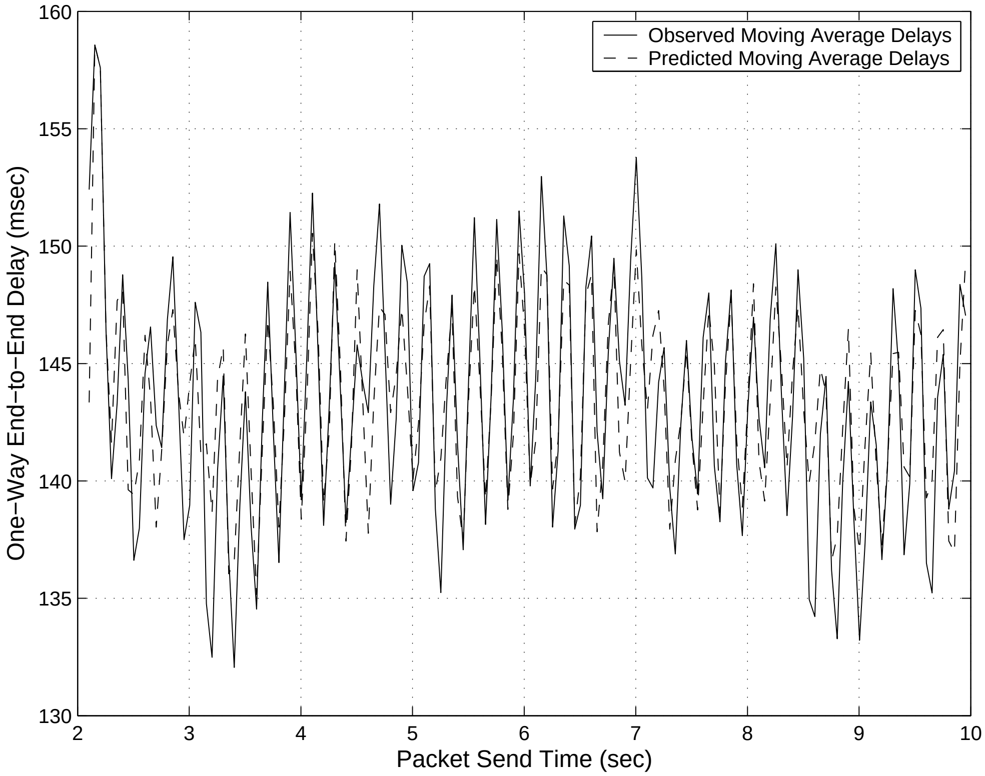

Fig. 25. Two-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the AR Model; Constant Send Rate of 200 Kbps with Baseline Cross-traffic Showing Moving Averaged Delays.

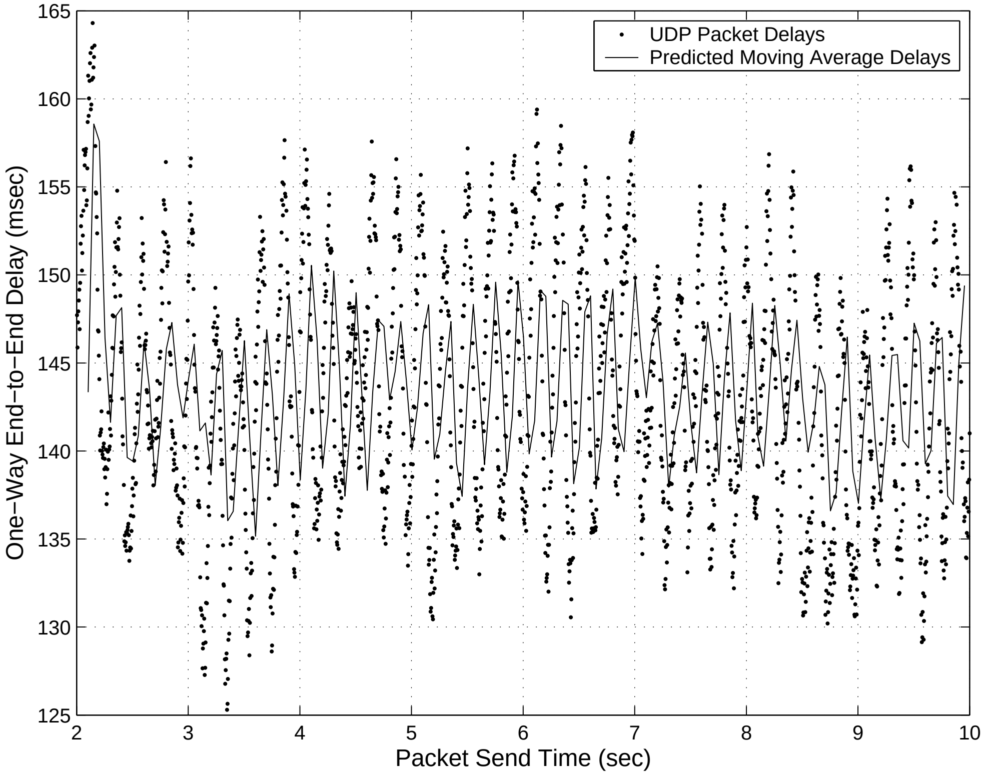

Fig. 26. Two-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the AR Model; Constant Send Rate of 200 Kbps with Baseline Cross-traffic Showing Actual Delays.

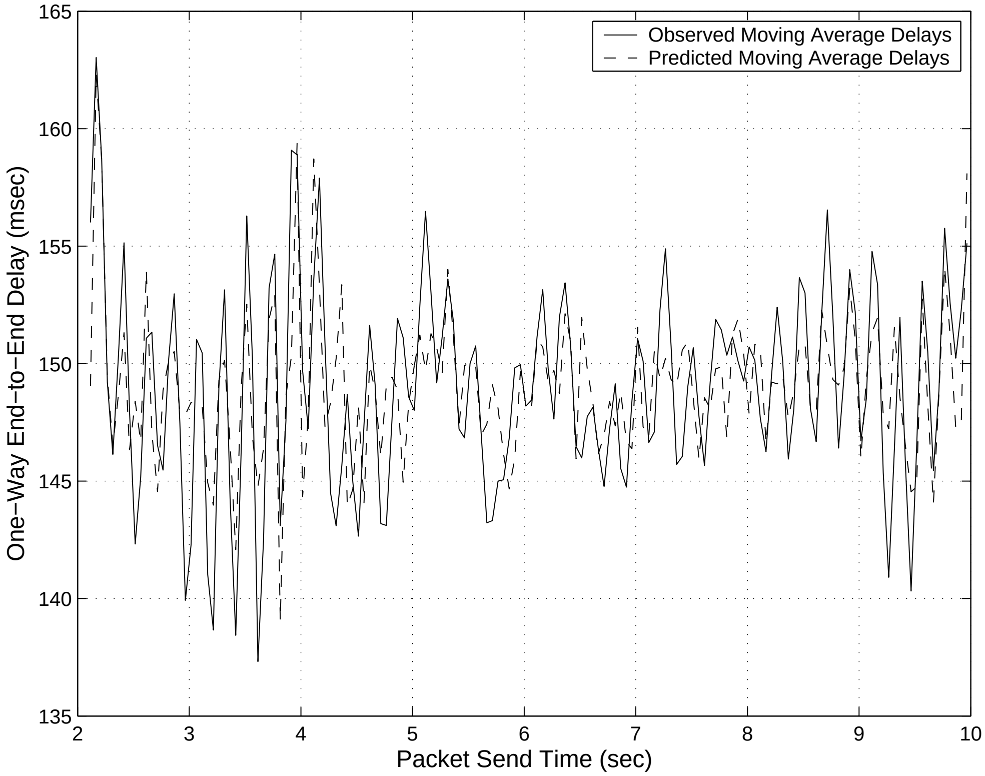

Fig. 27. Two-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the ARMA Model; Constant Send Rate of 50 Kbps with Baseline Cross-traffic Showing Moving Averaged Delays.

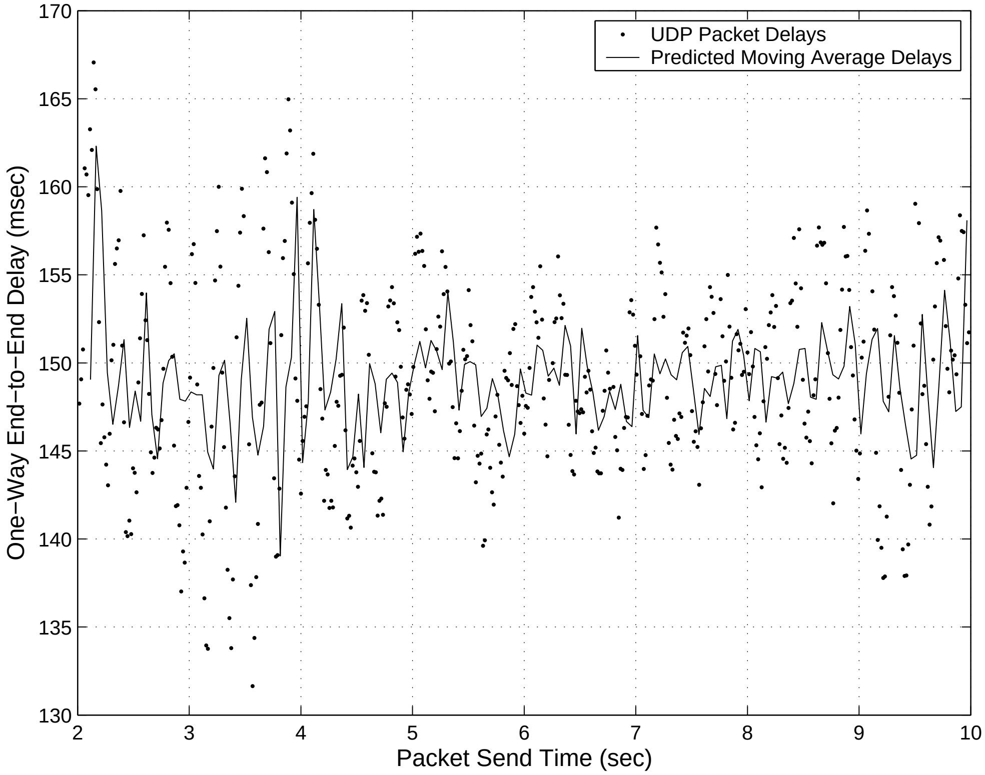

Fig. 28. Two-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the ARMA Model; Constant Send Rate of 50 Kbps with Baseline Cross-traffic Showing Actual Delays.

2. Three-Step-Ahead Prediction

Figure 29 depicts the three-step-ahead prediction of moving average one-way end-to-end delays using AR predictor. It shows the measured and predicted moving average delays for a constant send rate of 200 kbps with baseline cross-traffic. Figure 29 shows a good three-step-ahead prediction is achieved. MSE_2 for this test case is 36.2%. Figure 30 shows the predicted moving average delays and the actual UDP packet delays.

Figure 31 depicts the three-step-ahead prediction of moving average one-way end-to-end delays using ARMA predictor. It shows the measured and predicted moving average delays for a constant send rate of 50 kbps with baseline cross-traffic. Figure 31 shows a reasonable good two-step-ahead prediction. Figure 32 shows the predicted moving average delays and the actual UDP packet delays. MSE_2 for this test case is 57.7% which is close to MSE_2 obtained in two-step-ahead prediction of this test case.

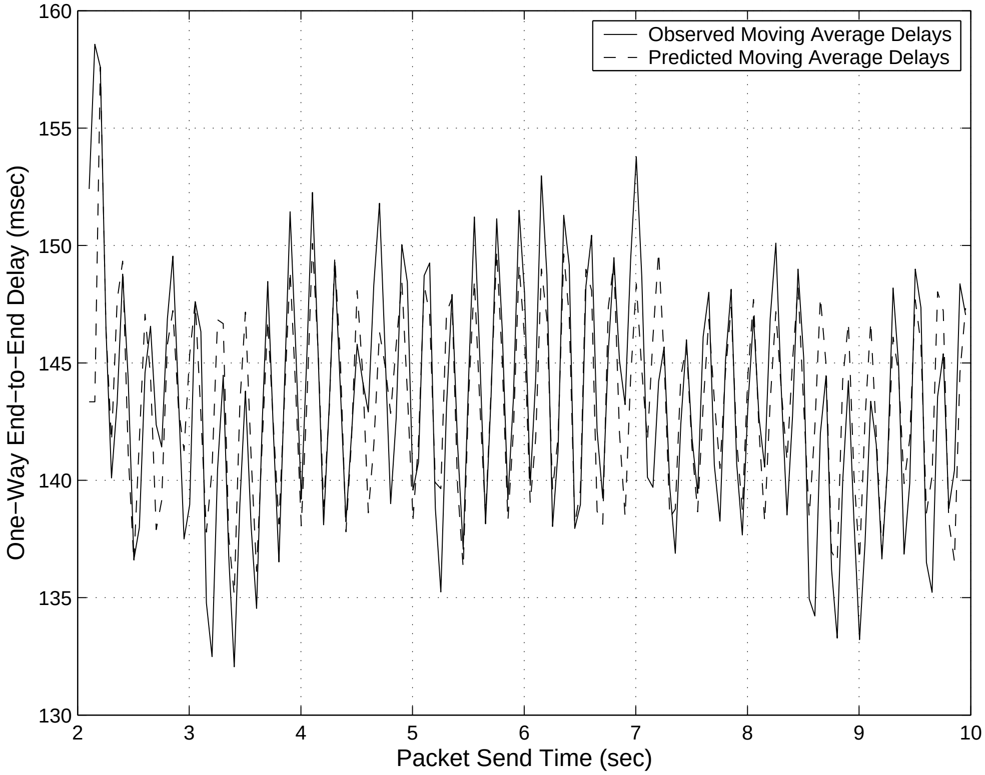

Fig. 29. Three-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the AR Model; Constant Send Rate of 200 Kbps with Baseline Cross-traffic Showing Moving Averaged Delays.

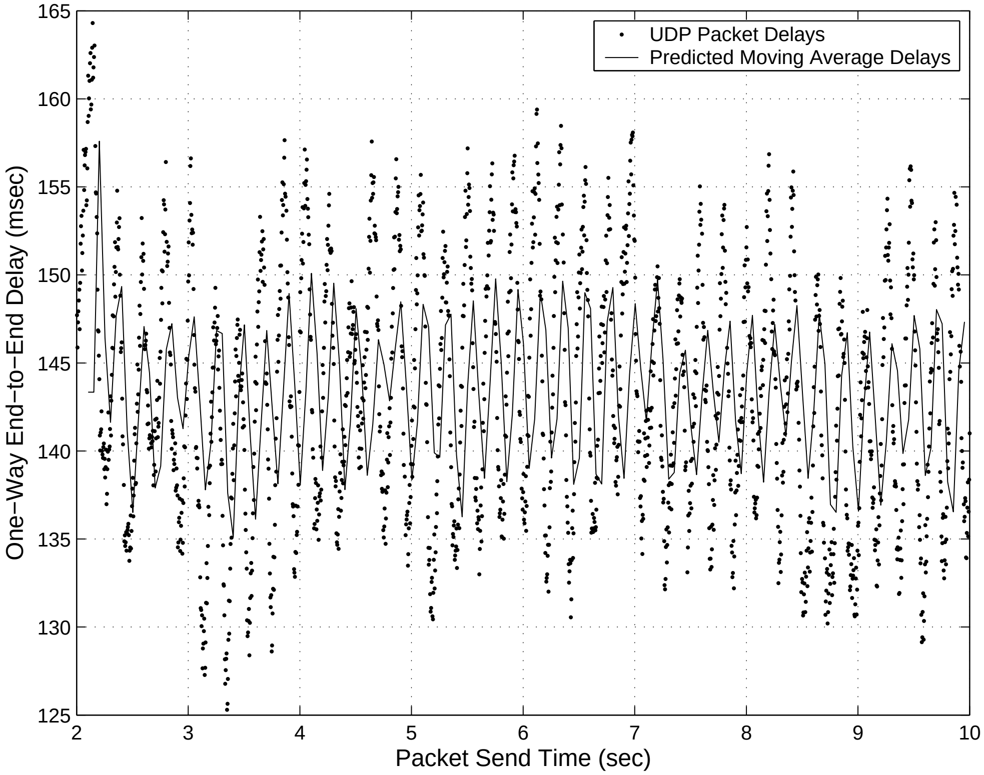

Fig. 30. Three-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the AR Model; Constant Send Rate of 200 Kbps with Baseline Cross-traffic Showing Actual Delays.

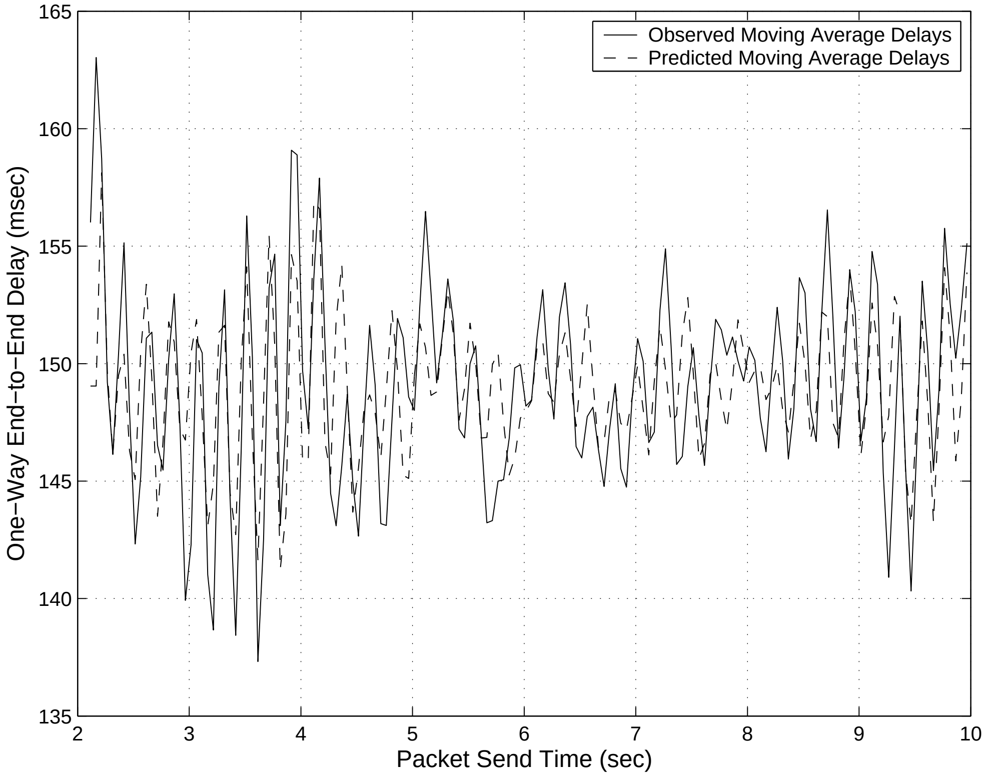

Fig. 31. Three-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the ARMA Model; Constant Send Rate of 50 Kbps with Baseline Cross-traffic Showing Moving Averaged Delays.

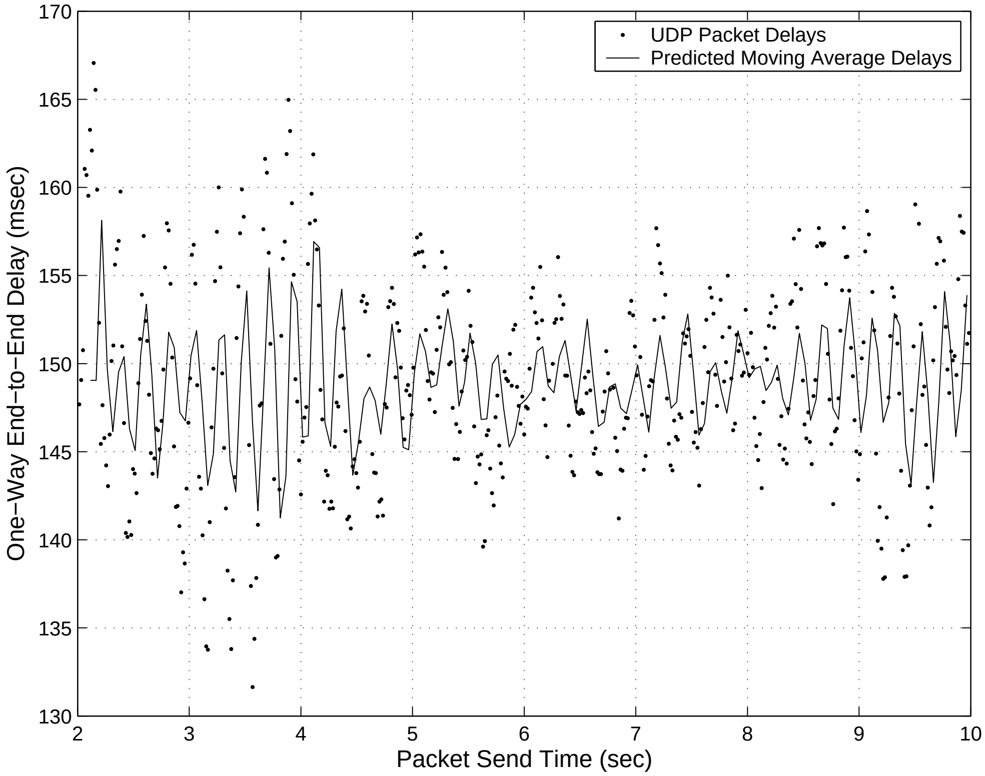

Fig. 32. Three-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the ARMA Model; Constant Send Rate of 50 Kbps with Baseline Cross-traffic Showing Actual Delays.

3. Four-Step-Ahead Prediction

Figure 33 depicts the four-step-ahead prediction of moving average one-way end-to-end delays using AR predictor. It shows the measured and predicted moving average delays for a constant send rate of 200 Kbps with baseline cross-traffic. Figure 33 shows a good four-step-ahead prediction is achieved. The MSE_2 for this test case is 40.5%. The MSE_2 is increasing as the time step used for prediction is increased. Figure 34 shows the predicted moving average delays and the actual UDP packet delays.

Figure 35 depicts the four-step-ahead prediction of moving average one-way end-to-end delays using ARMA predictor. It shows the measured and predicted moving average delays for a constant send rate of 50 kbps with baseline cross-traffic. Figure 35 shows a reasonable good two-step-ahead prediction. The Figure 36 shows the predicted moving average delays and the actual UDP packet delays. MSE_2 for this test case is 59.3% which is close to MSE_2 obtained in two-step-ahead prediction and three-step-ahead prediction indicating that the developed predictor is not performing well on this test case with send rate 50 kbps.

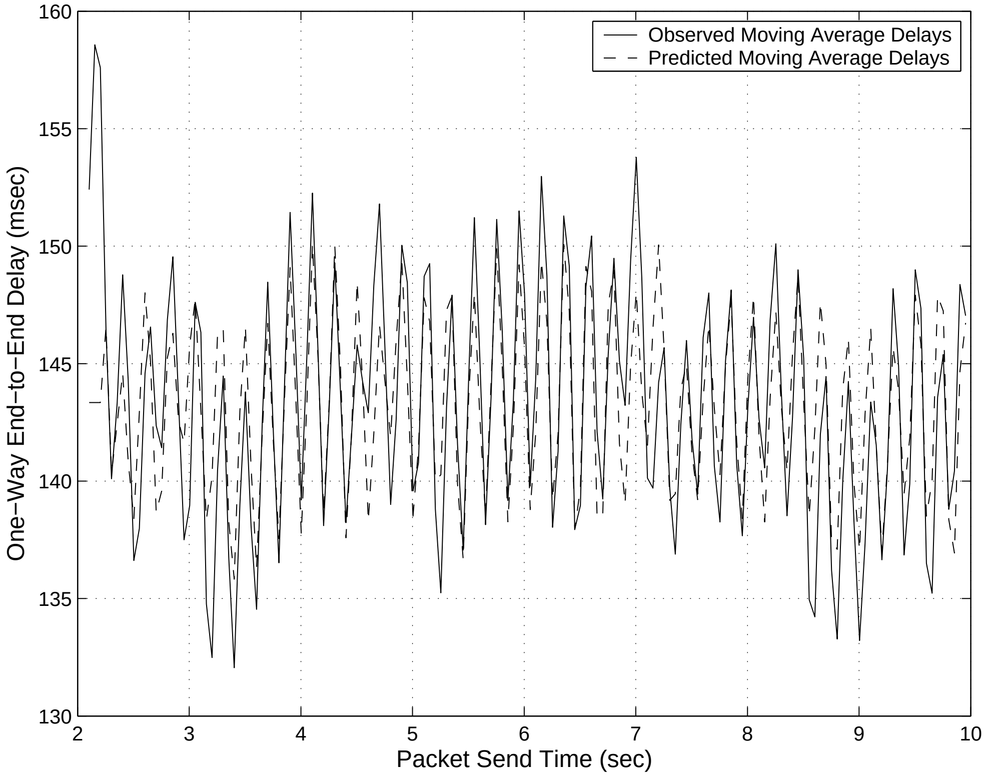

Fig. 33. Four-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the AR Model; Constant Send Rate of 200 Kbps with Baseline Cross-traffic. Showing Moving Averaged Delays

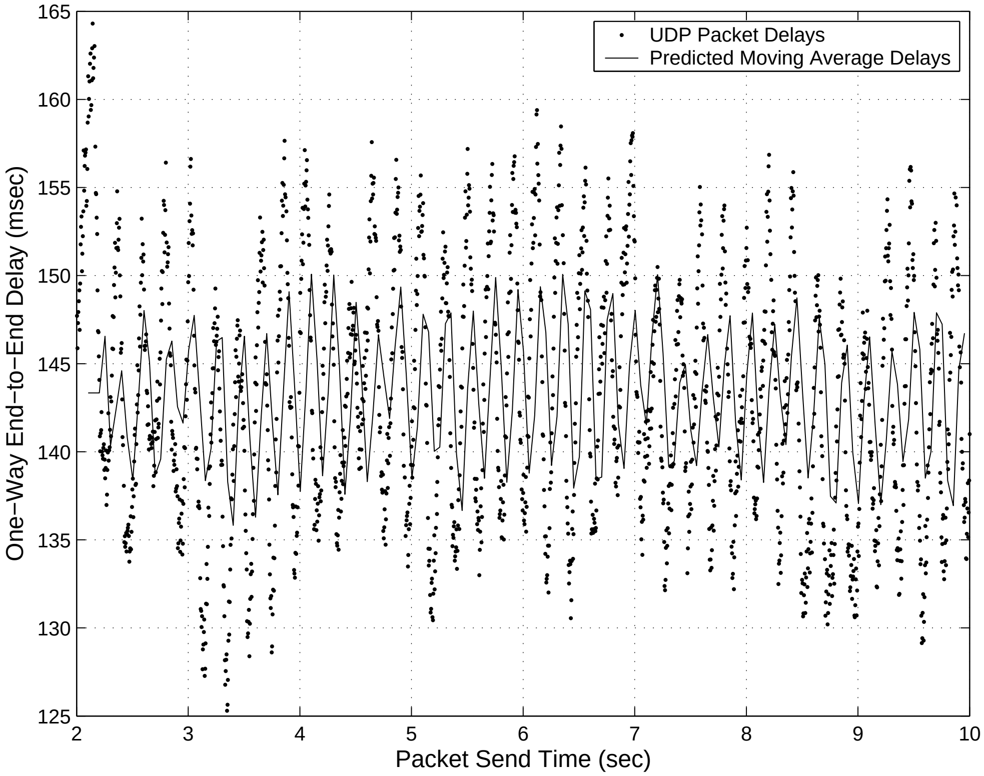

Fig. 34. Four-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the AR Model; Constant Send Rate of 200 Kbps with Baseline Cross-traffic Showing Actual Delays.

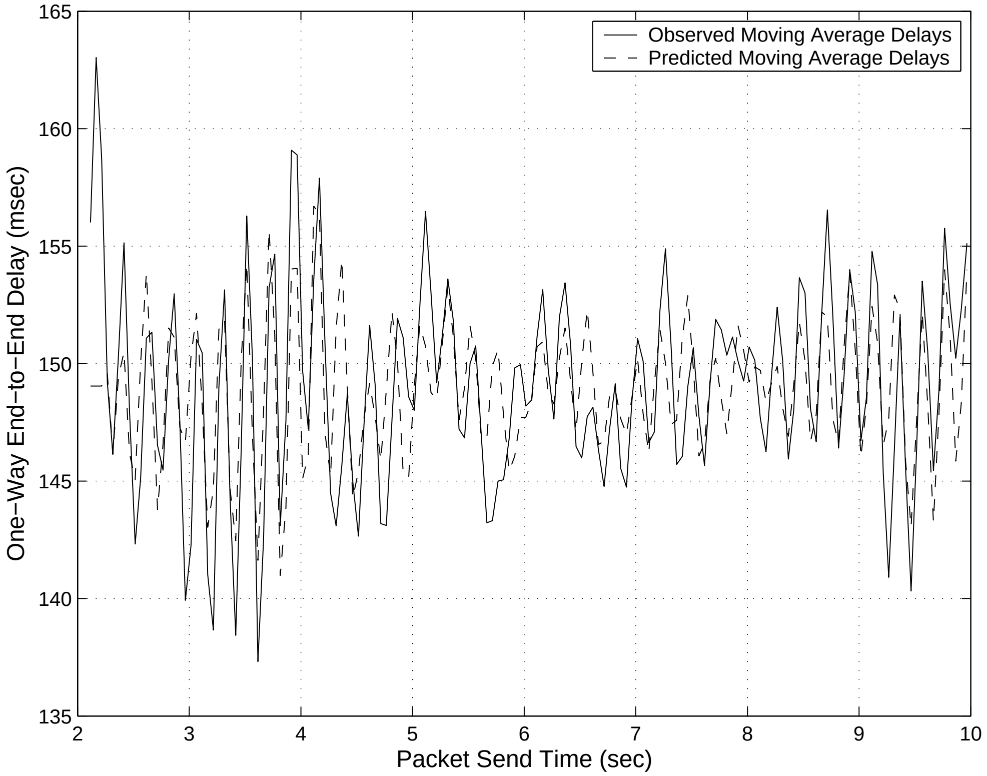

Fig. 35. Four-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the ARMA Model; Constant Send Rate of 50 Kbps with Baseline Cross-traffic Showing Moving Averaged Delays.

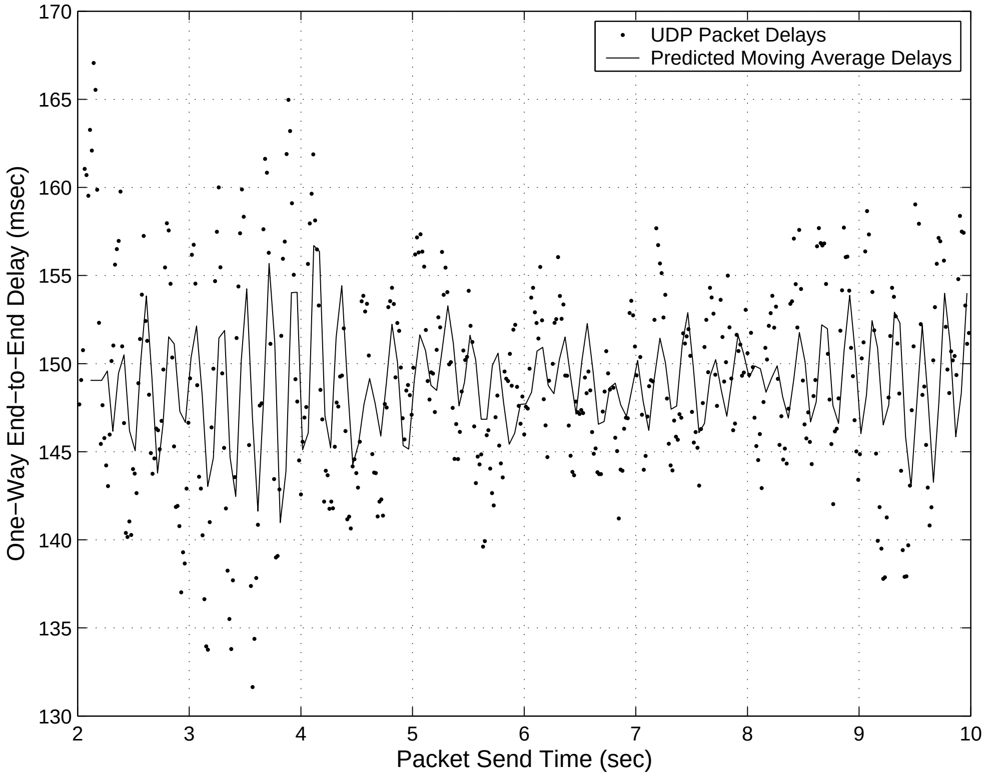

Fig. 36. Four-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the ARMA Model; Constant Send Rate of 50 Kbps with Baseline Cross-traffic Showing Actual Delays.

4. Ten-Step-Ahead Prediction

Figure 37 depicts the ten-step-ahead prediction of moving average one-way end-to-end delays using AR predictor. It shows the measured and predicted moving average delays for a constant send rate of 200 kbps with baseline cross-traffic. Figure 38 shows the predicted moving average delays and the actual UDP packet delays. MSE_2 for this test case is 62.8%.

Figure 39 depicts the ten-step-ahead prediction of moving average one-way end-to-end delays using ARMA predictor. It shows the measured and predicted moving average delays for a constant send rate of 50 kbps with baseline cross-traffic. Figure 40 shows the predicted moving average delays and the actual UDP packet delays. MSE_2 for this test case is a very high value of 105.2% indicating the poor performance of the predictor.

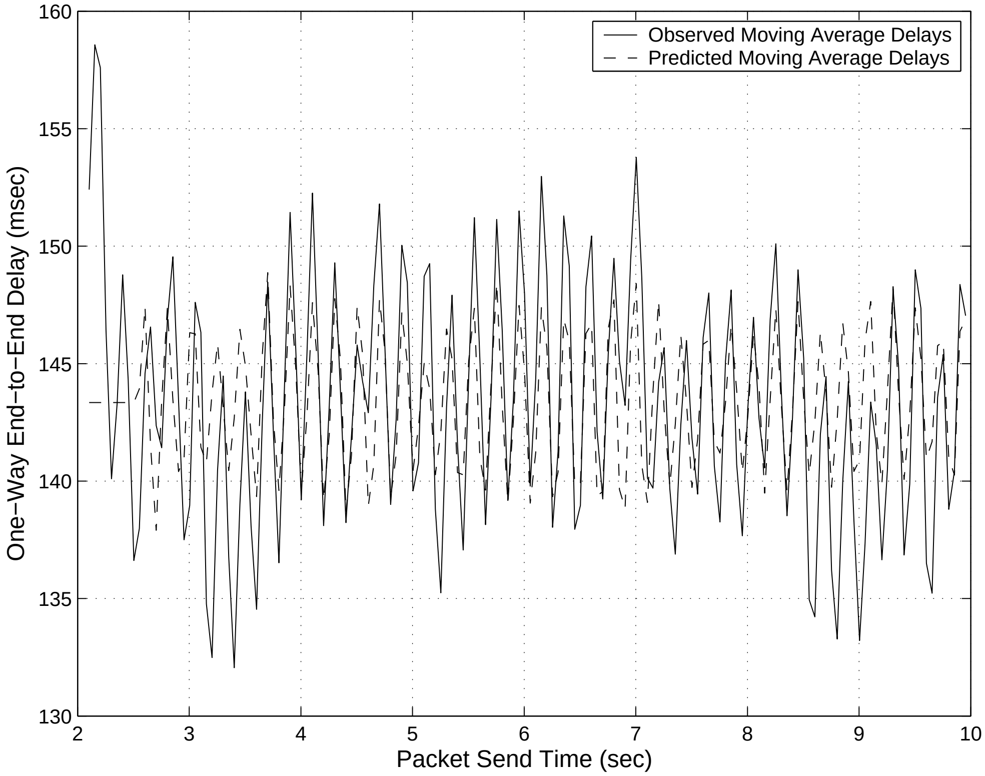

Fig. 37. Ten-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the AR Model; Constant Send Rate of 200 Kbps with Baseline Cross-traffic Showing Moving Averaged Delays.

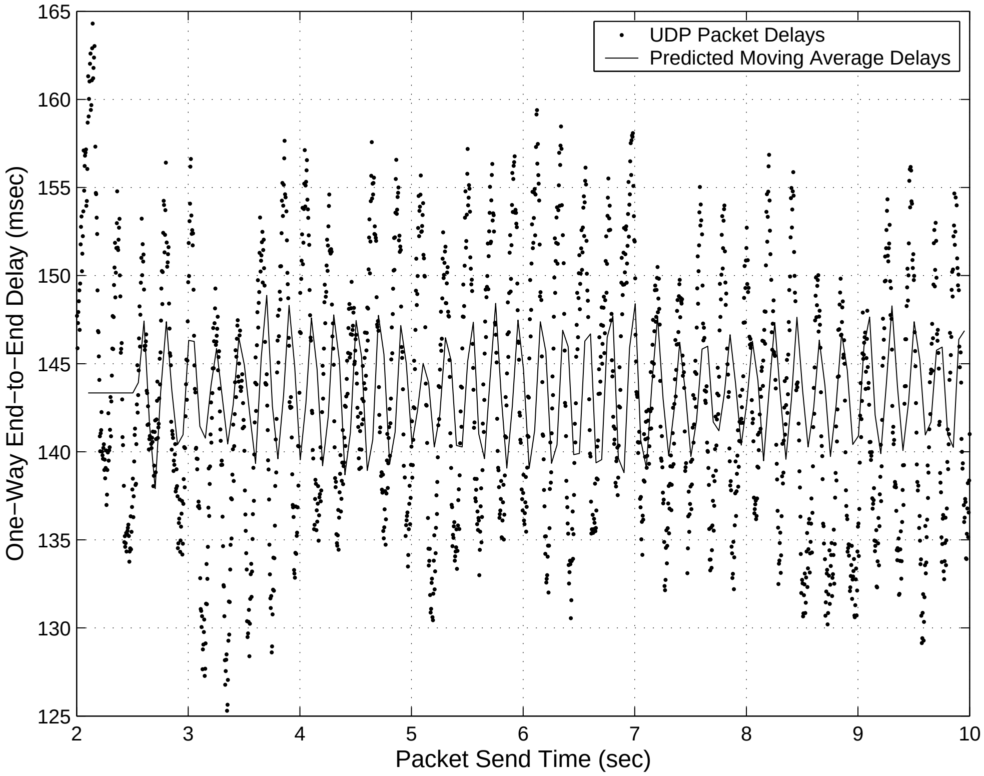

Fig. 38. Ten-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the AR Model; Constant Send Rate of 200 Kbps with Baseline Cross-traffic Showing Actual Delays.

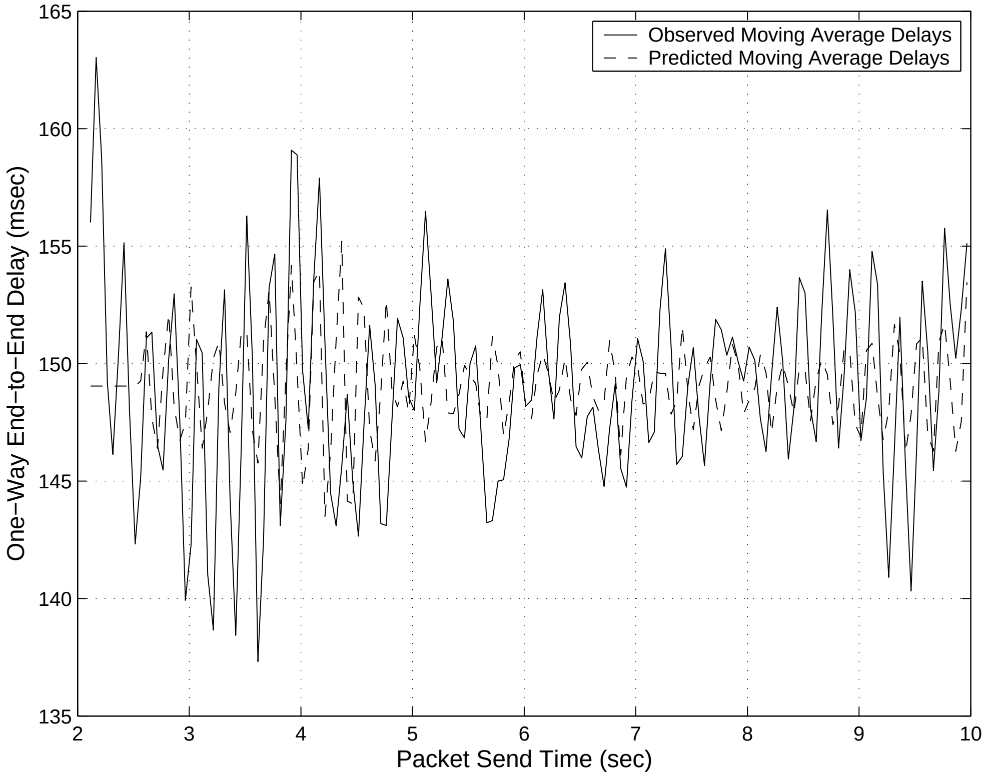

Fig. 39. Ten-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the ARMA Model; Constant Send Rate of 50 Kbps with Baseline Cross-traffic Showing Moving Averaged Delays.

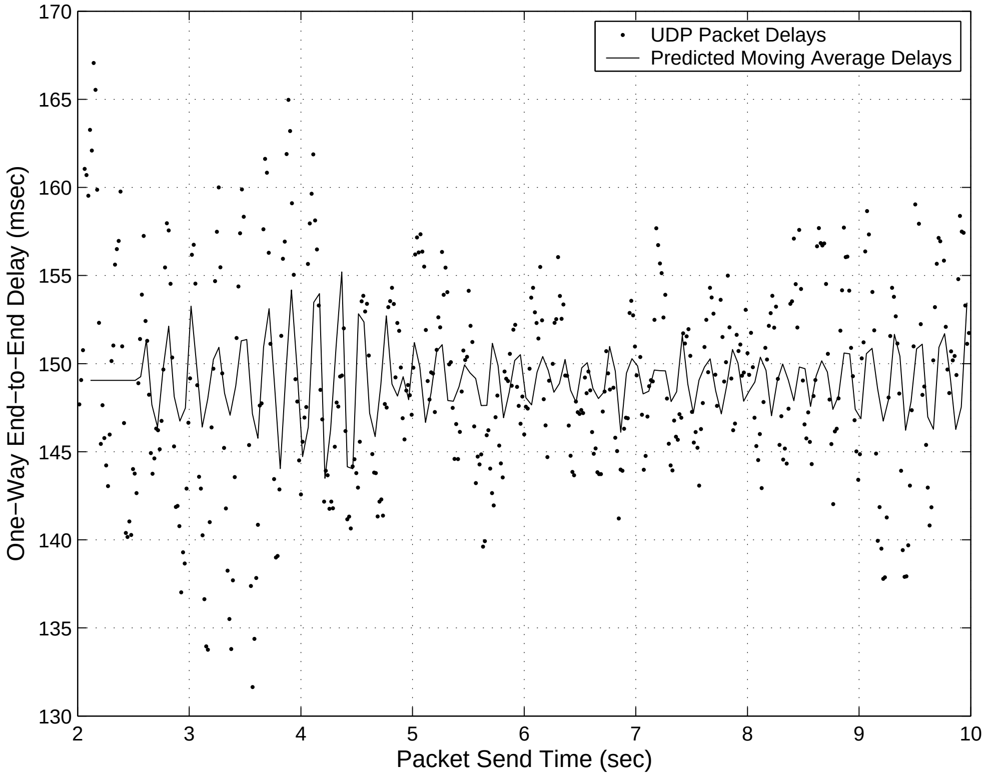

Fig. 40. Ten-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the ARMA Model; Constant Send Rate of 50 Kbps with Baseline Cross-traffic Showing Actual Delays.

5. AR Predictor Performance on the Special Test Case

The special test case was specially designed to test the AR predictor performance in real traffic conditions in the Internet. AR predictor is chosen as it is the most simple and the best predictor obtained in this research. The multi-step ahead predictions are presented in the Table XX. It can be seen that, as the prediction horizon increases the MSE$_1$, MAE, and the MSE$_2$ also increase. It can be observed that the MAE is quite high which is around 70 msec compared to other test cases. However this MAE occurs for a single prediction in the entire sample. Hence the overall predictor performance cannot be judged based on MAE.

Table XX. AR Multi-Step-Ahead Predictions for the Variable Cross-traffic Special Test Case.

Number of Steps Ahead	MSE$_1$ (%)	MAE (in msec)	MSE$_2$ (%)
Two-step	2.11	68.2	37.6
Three-step	3.17	73.3	56.6
Four-step	3.41	73.9	60.93
Five-step	4.2	72.1	75.1
Six-step	5.4	71.1	96.2
Seven-step	5.7	72.4	102.3
Eight-step	5.5	81.3	98.4
Nine-step	5.5	81.4	99.0
Ten-step	5.59	75.2	100.02

F. Development of Nonlinear Predictors and Training, Testing and Validation Results

A predictor based on FMLP network is developed for the moving average one-way end-to-end delay. The predictor is developed using teacher forcing (TF) algorithm. There are no inputs to this predictor because the biggest indicator which influences the one-way end-to-end delays is the cross-traffic. It cannot be measured and hence it is considered as a disturbance of the model. The training is performed till the MSP error is minimized. The error used for determining the performance of the predictor is the mean square error (MSE_1). The trained predictor is evaluated in terms of its performance for MSP on a validation data set which is part of the training set but not used in the weight estimation.

After an extensive search over possible FMLP architectures, the FMLP network with 4 input layer nodes, 7 hidden layer nodes and 1 output layer node is found to give the best MSP performance. The time steps used for prediction are two-step, three-step and four-step. Ten step ahead prediction is not possible with FMLP because it has only 4 input layers. The performance results of this predictor on the training, testing and validation data sets are summarized in the Tables XXI, XXII, and XXIII.

1. Two-Step-Ahead Prediction

Table XXI summarizes the performance results of the FMLP predictor on the training, testing and validation data sets in terms of the various performance metrics. MSE_2 on the validation set is 59.2% which is around 8% more compared to the MSE_2 error obtained from linear models. However, the MAE is 11.2 msecs. This is almost same as that of MAE obtained from AR and ARMA.

Table XXI. Two-Step-Ahead Prediction of Training, Testing and Validation Data Using FMLP

Data Set	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
Training Set	0.035	17.5	47.4
Testing Set	0.034	8.2	53.0
Validation Set	0.031	11.2	59.2

2. Three-Step-Ahead Prediction

The Table XXII summarizes the performance results of the FMLP predictor on the training, testing and validation data sets in terms of the various performance metrics. The MAE on the validation data set is around 10 msecs. The MSE_2 is 65.6% as compared to 62.8% obtained using the AR and 63.1% using the ARMA predictors respectively.

Table XXII. Three-Step-Ahead Prediction of Training, Testing and Validation Data Using FMLP

Data Set	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
Training Set	0.041	16.7	56.5
Testing Set	0.044	10.1	67.1
Validation Set	0.035	10.1	65.6

3. Four-Step-Ahead Prediction

The Table XXIII summarizes the performance results of the FMLP predictor on the training, testing and validation data sets in terms of the performance metrics. The errors in four-step-ahead prediction do not deteriorate farther from three-step-ahead prediction as can be seen from the Table XXIII.

Table XXIII. Four-Step-Ahead Prediction of Training, Testing and Validation Data Using FMLP

Data Set	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
Training Set	0.041	16.7	56.5
Testing Set	0.044	9.9	66.9
Validation Set	0.035	10.1	65.6

G. Performance Evaluation of Nonlinear Predictors

Performance evaluation of nonlinear predictors is done by validating the developed predictors on 6 different send rate test cases and 8 different cross-traffic test cases. Following subsections will present a representative test case whose send rate is a constant 100 kbps bit-rate with 75% decrease in baseline cross-traffic. From the Figures 41, 43 and 45, it can be observed that as the time-steps increase, the accuracy in the prediction decreases.

1. Two-Step-Ahead Prediction

Figure 41 depicts the four-step-ahead prediction of moving average one-way end-to-end delays using the FMLP predictor. It shows the measured and predicted moving average delays for a constant send rate of 100 Kbps with 75% decrease in cross-traffic from baseline cross-traffic. Figure 42 shows the predicted moving average delays and the actual UDP packet delays. Figure 41 shows an inaccurate two-step-ahead prediction. MSE_2 for this test case is 48.5%. The MAE for this test case is 64.8 msecs which is a very high value indicating that the FMLP predictor is not effective when there is a 75% decrease in cross-traffic from the baseline cross-traffic.

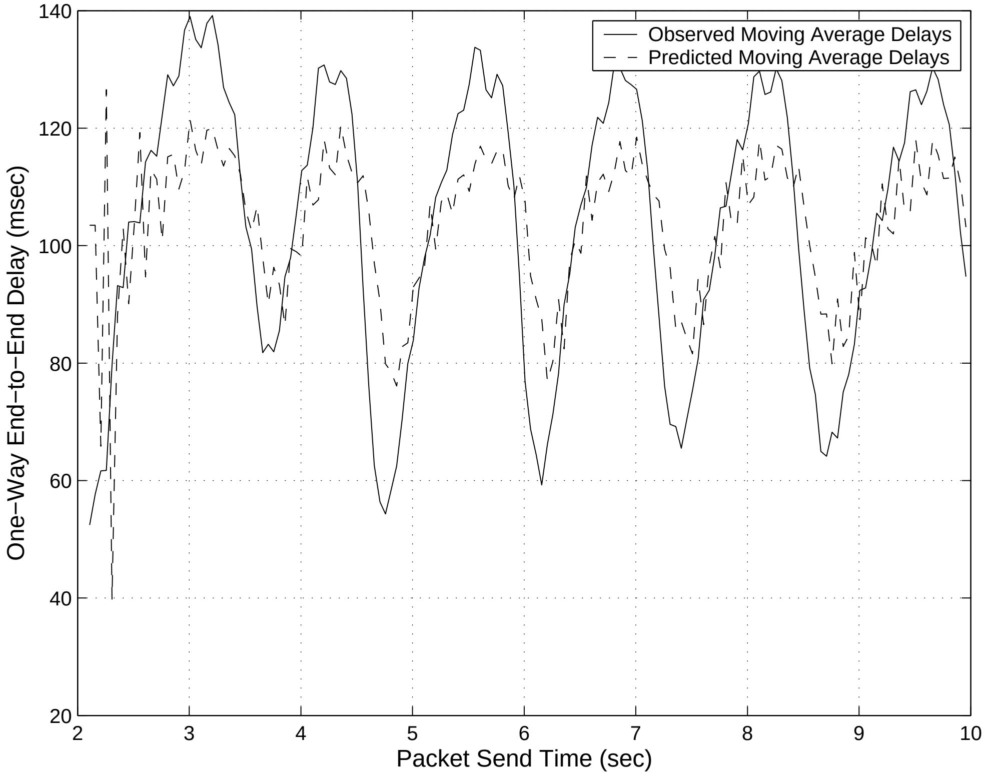

Fig. 41. Two-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the FMLP Model; Constant Send Rate of 100 Kbps with 75% Decrease in Baseline Cross-traffic Showing Moving Averaged Delays.

2. Three-Step-Ahead Prediction

Figure 43 depicts the four-step-ahead prediction of moving average one-way end-to-end delays using the FMLP predictor. It shows the measured and predicted moving average delays for a constant send rate of 100 kbps with 75% decrease in cross-traffic from baseline cross-traffic. Figure 44 shows the predicted moving average delays and

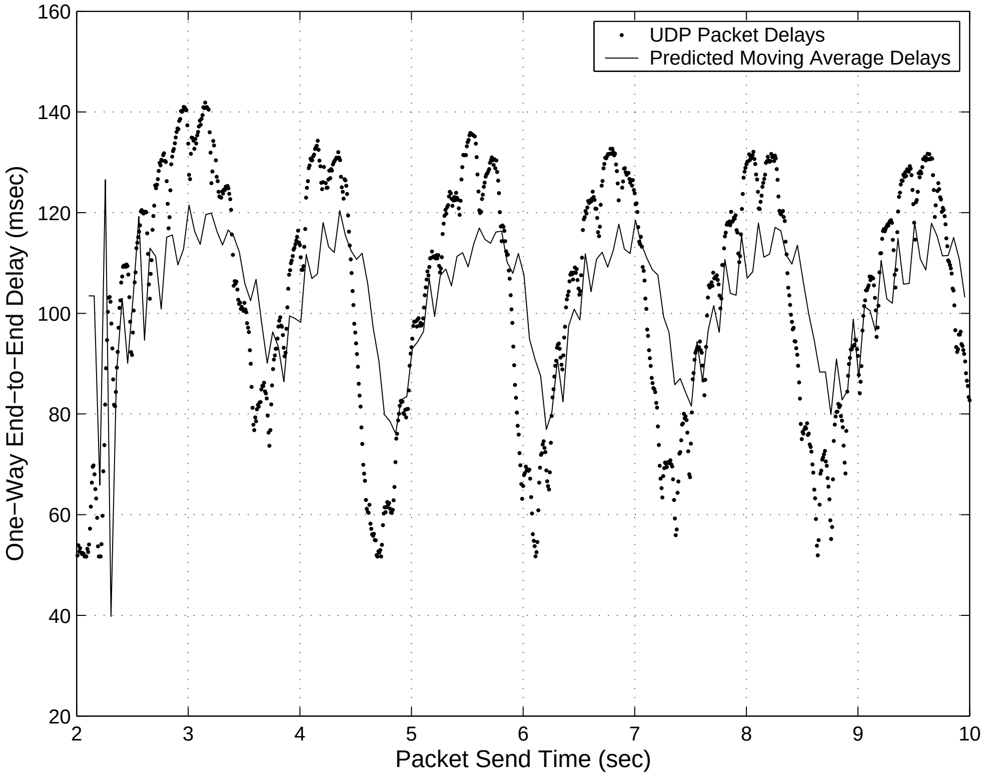

Fig. 42. Two-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the FMLP Model; Constant Send Rate of 100 Kbps with 75% Decrease in Baseline Cross-traffic Showing Actual Delays.

the actual UDP packet delays. Figure 43 shows a poor three-step-ahead prediction. MSE_2 for this test case is 64.1% but the MAE for this test case came down to 51.0 msecs from 64.8 msecs as obtained in the two-step-ahead prediction.

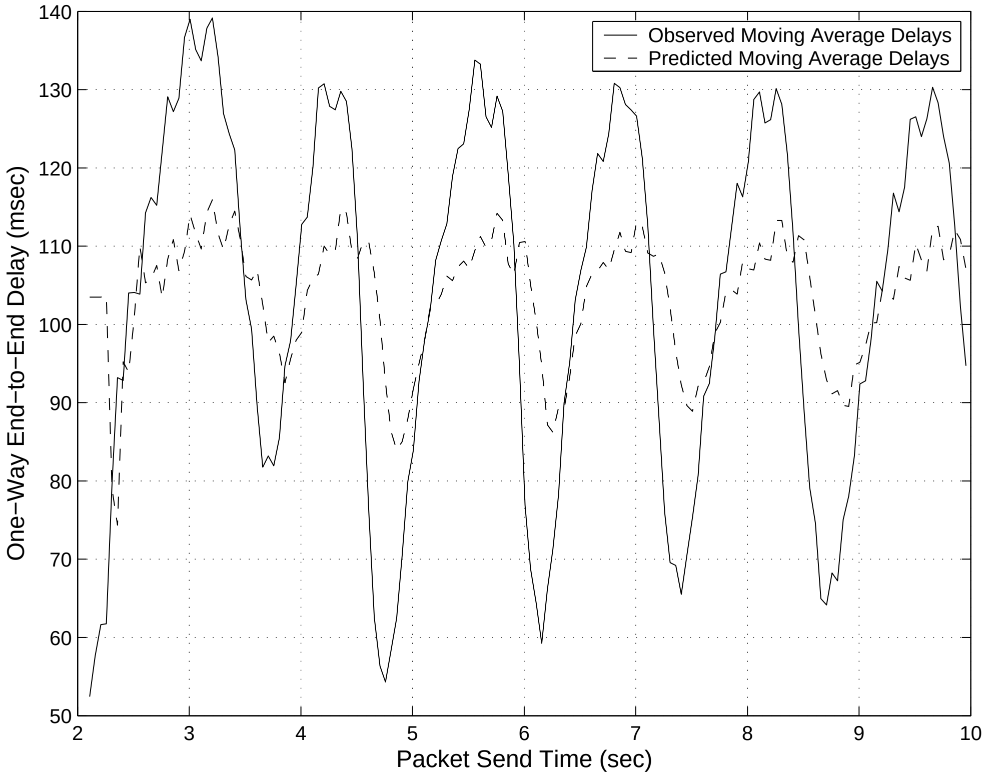

Fig. 43. Three-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the FMLP Model; Constant Send Rate of 100 Kbps with 75% Decrease in Baseline Cross-traffic Showing Moving Averaged Delays.

3. Four-Step-Ahead Prediction

Figure 45 depicts the four-step-ahead prediction of moving average one-way end-to-end delays using the FMLP predictor. It shows the measured and predicted moving average delays for a constant send rate of 100 kbps with 75% decrease in cross-traffic

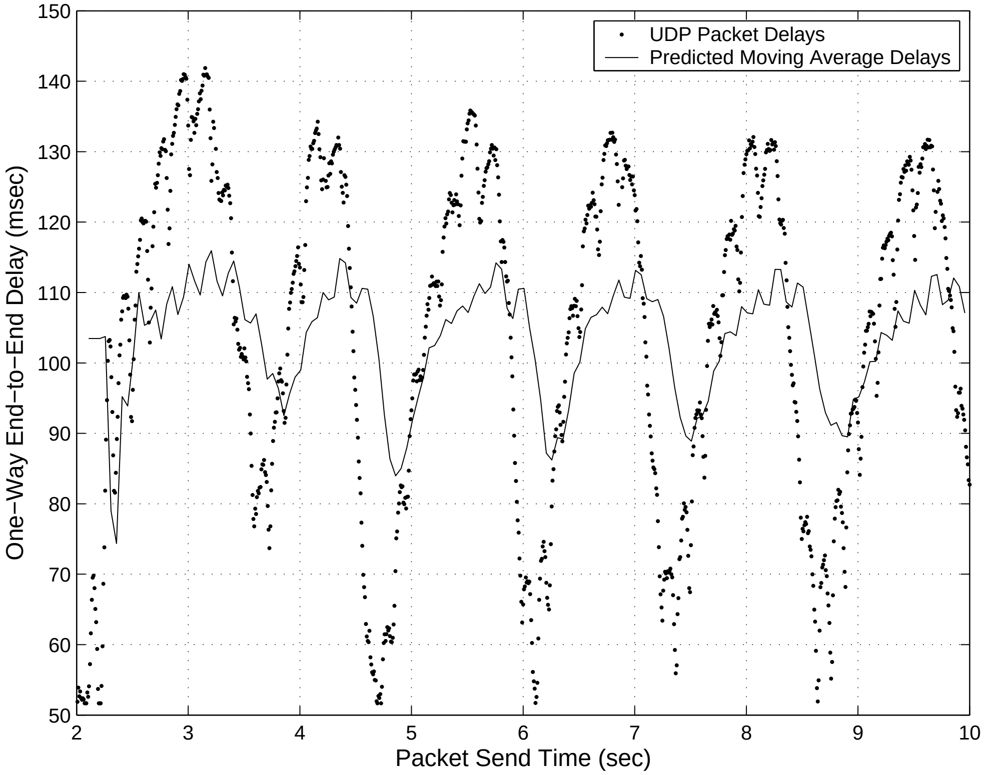

Fig. 44. Three-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the FMLP Model; Constant Send Rate of 100 Kbps with 75% Decrease in Baseline Cross-traffic Showing Actual Delays.

from baseline cross-traffic. Figure 46 shows the predicted moving average delays and the actual UDP packet delays. MSE_2 for this test case is 63.9% which is almost same as that obtained from three-step-ahead prediction. The MAE for this test case decreased to 51.0 msecs from 64.8 msecs as obtained in the two-step-ahead prediction.

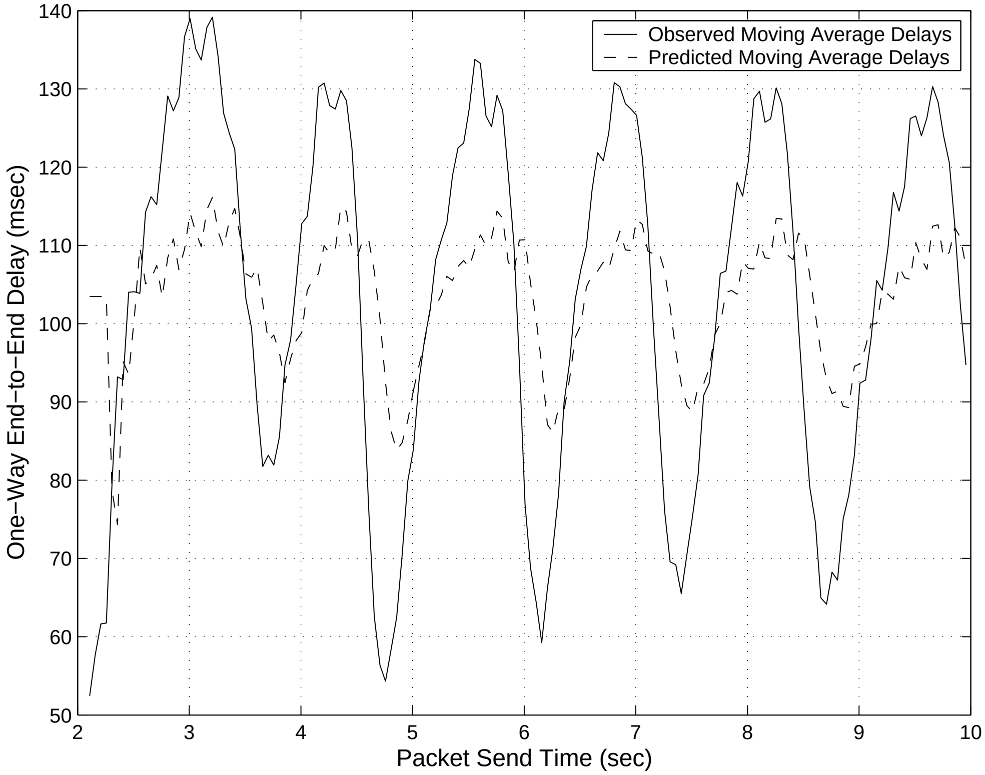

Fig. 45. Four-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the FMLP Model; Constant Send Rate of 100 Kbps with 75% Decrease in Baseline Cross-traffic Showing Moving Averaged Delays.

H. Comparison of Multi-Step-Ahead Predictor Performance

The results of the MSP on all the 14 different test cases using AR, ARMA and FMLP predictors are tabulated in this section. This section is split into 4 subsections. Each

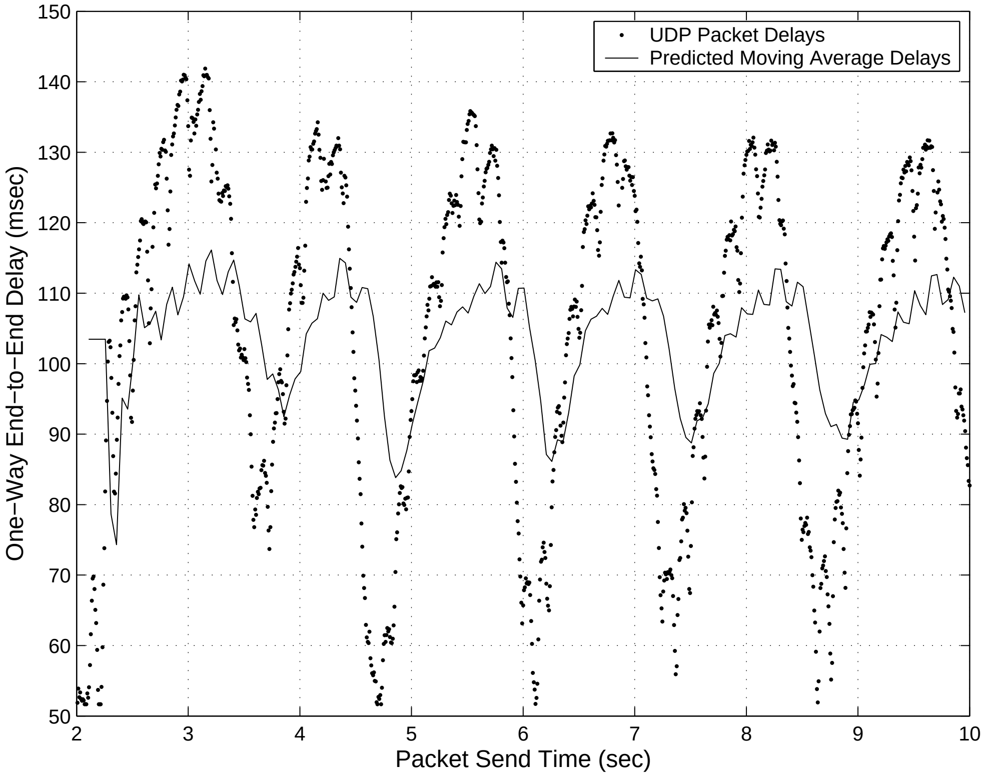

Fig. 46. Four-Step-Ahead Prediction of Moving Average One-Way End-to-end Delay Using the FMLP Model; Constant Send Rate of 100 Kbps with 75% Decrease in Baseline Cross-traffic Showing Actual Delays.

subsection provides performance evaluation results on all the 14 test cases in terms of the performance indicators discussed previously. In addition, the tables also contain the performance evaluation results of the validation data set which is used as baseline to compare the results and evaluate the performance of each of these predictors.

1. Two-Step-Ahead Prediction

This section presents the performance results of two-step-ahead predictions using AR, ARMA and FMLP predictors. There are two tables depicting the results of each predictor. The first table presents the prediction results on the different source send rate test cases. The second table presents the prediction results on the eight different cross-traffic test cases. Each table has the results of the baseline test set obtained from validation during the training process. This helps in comparing the test results with that obtained from training.

Tables XXIV and XXV present the performance results of the AR predictor. It can be seen from the Table XXIV that the MAE is around 10 msecs for all the test cases and the MSE_2 of all the test cases is less than that of the baseline validation set except for the 50 Kbps send rate test case. This shows that the predictor performs well between the range of 50 kbps and 800 kbps. Table XXV shows the performance results for the different cross-traffic test cases. It can observed that the AR predictor fails in quite a few test cases. A very high MSE_2 can be observed on the test cases with 90% decrease from baseline cross-traffic, 50% increase, 100% increase and 150% increase from baseline cross-traffic. It can be concluded that the developed predictor works well within the range of 75% decrease to 25% increase from baseline cross-traffic.

The Tables XXVI and XXVII present the performance results of the ARMA predictor. The results on the send rate test cases are similar to that of the AR

Table XXIV. AR Two-Step-Ahead Predictions for Send Rate Test Cases.

Send Rate	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
800 Kbps	0.047	12.6	36.6
500 Kbps	0.042	12.9	29.7
400 Kbps	0.023	11.4	37.7
200 Kbps	0.036	9.1	27.6
100 Kbps	0.035	9.8	45.4
50 Kbps	0.043	8.8	52.7
Baseline	0.027	10.3	50.9

Table XXV. AR Two-Step-Ahead Predictions for Cross-traffic Test Cases.

Cross-traffic	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
90% decrease	0.003	0.9	311.2
75% decrease	1.710	51.4	35.3
50% decrease	0.758	68.3	40.3
25% decrease	0.120	21.8	45.3
25% increase	0.040	7.8	63.7
50% increase	0.055	8.9	85.8
100% increase	0.057	9.2	100.6
150% increase	0.032	9.2	94.9
Baseline	0.027	10.3	50.9

predictor but there is a slight improvement in the cross-traffic test results. The 50% increase, 100% increase and 150% increase cross-traffic test cases show better results than those obtained from the AR predictor.

Table XXVI. ARMA Two-Step-Ahead Predictions for Send Rate Test Cases.

Send Rate	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
800 Kbps	0.046	12.6	35.9
500 Kbps	0.041	12.9	28.8
400 Kbps	0.024	11.4	40.3
200 Kbps	0.036	9.0	28.2
100 Kbps	0.036	9.5	46.6
50 Kbps	0.044	8.8	54.1
Baseline	0.027	10.3	51.1

The Tables XXVIII and XXIX present the performance results of FMLP predictor. It can be seen from the Table XXVIII that the MAE is around 18 msecs for all the test cases and the MSE_2 of all the test cases is more than that of the baseline validation set except for the 200 kbps send rate test case which is 38.4%. The results shows that the FMLP predictor performs well between the range of 50 kbps and 800 kbps. Table XXIX shows the performance results for the different cross-traffic test cases. It can observed that the FMLP predictor fails on the same test cases for which AR and ARMA predictors failed. A very high MSE_2 can be observed for the test cases with 90% decrease from baseline cross-traffic, 50% increase, 100% increase and 150% increase from baseline cross-traffic. It can also be seen that the test case with 75% decrease from baseline cross-traffic is the only cross-traffic test case which showed good results.

Table XXVII. ARMA Two-Step-Ahead Predictions for Cross-traffic Test Cases.

Cross-traffic	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
90% decrease	0.004	0.9	386.0
75% decrease	1.728	51.4	35.7
50% decrease	0.760	68.3	40.5
25% decrease	0.113	21.8	42.9
25% increase	0.038	7.8	60.5
50% increase	0.048	8.8	74.9
100% increase	0.050	9.0	89.2
150% increase	0.028	8.9	82.6
Baseline	0.027	10.3	51.1

The predictors completely fail for the test case with 90% decrease from baseline cross-traffic. This can be attributed to very less cross-traffic affecting the end-to-end delays. The cross-traffic is an important indicator affecting the end-to-end delays in a network. The predictors were developed by modeling this cross-traffic as a disturbance. The predictors seem to work in a particular range of cross-traffic.

2. Three-Step-Ahead Prediction

This section presents the performance results of three-step-ahead predictions using AR, ARMA and FMLP predictors. Table XXX shows the prediction results for the different send rate test cases. The MSE_2 on the baseline test case is 62.8%. MSE_2 on all the send rate test cases is lower than that of the baseline test set indicating a good three-step-ahead prediction on the send rate test cases. However, the MAE is around 15 msecs which indicates that the maximum difference between the prediction and the

Table XXVIII. FMLP Two-Step-Ahead Predictions for Send Rate Test Cases.

Send Rate	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
800 Kbps	0.068	19.9	53.5
500 Kbps	0.066	18.8	46.6
400 Kbps	0.042	17.0	69.9
200 Kbps	0.050	15.7	38.5
100 Kbps	0.045	14.6	57.8
50 Kbps	0.050	14.4	60.5
Baseline	0.031	11.2	59.2

Table XXIX. FMLP Two-Step-Ahead Predictions for Cross-traffic Test Cases.

Cross-traffic	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
90% decrease	0.009	1.3	999.9
75% decrease	2.347	64.8	48.5
50% decrease	1.379	79.9	73.4
25% decrease	0.160	22.9	60.4
25% increase	0.047	15.0	74.9
50% increase	0.058	8.9	90.4
100% increase	0.058	8.7	103.6
150% increase	0.034	8.8	99.9
Baseline	0.031	11.2	59.2

observation is around 15 msecs. Table XXXI presents the performance results on the different cross-traffic test cases compared to the baseline validation set. The predictor performs well on the test cases with cross-traffic in the range of 75% decrease to 25% increase from the baseline cross-traffic and fails on all the remaining test cases.

Table XXX. AR Three-Step-Ahead Predictions for Send Rate Test Cases.

Send Rate	MSE$_1$ (%)	MAE (in msec)	MSE$_2$ (%)
800 Kbps	0.052	12.6	40.7
500 Kbps	0.057	15.3	40.1
400 Kbps	0.031	15.3	50.9
200 Kbps	0.047	15.2	36.2
100 Kbps	0.044	14.2	55.8
50 Kbps	0.046	13.9	56.7
Baseline	0.033	10.3	62.8

Tables XXXII and XXXIII provide performance results of three-step-ahead prediction using the ARMA predictor. From Table XXXII, it can be seen that the developed ARMA predictor performs well. The MSE$_2$ on the test cases is lesser than that obtained on the baseline validation data set. The test case with 800 kbps send rate has the least MAE of 12.564 msecs. Table XXXIII summarizes the performance results on the cross-traffic test cases. The predictor performs well on the test cases with cross-traffic in the range of 75% decrease to 25% increase from the baseline cross-traffic and fails on all the remaining test cases.

Tables XXXIV and XXXV provide performance results of three-step-ahead prediction using the FMLP predictor. Table XXXIV shows that the MSE$_2$ for 400 kbps

Table XXXI. AR Three-Step-Ahead Predictions for Cross-traffic Test Cases.

Cross-traffic	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
90% decrease	0.001	0.7	143.4
75% decrease	2.563	51.4	52.9
50% decrease	1.151	68.3	61.3
25% decrease	0.160	23.4	60.5
25% increase	0.041	7.8	64.8
50% increase	0.057	9.4	88.9
100% increase	0.056	8.1	99.2
150% increase	0.033	7.9	97.1
Baseline	0.033	10.3	62.8

Table XXXII. ARMA Three-Step-Ahead Predictions for Send Rate Test Cases.

Send Rate	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
800 Kbps	0.051	12.6	40.1
500 Kbps	0.055	15.3	38.9
400 Kbps	0.032	15.3	53.2
200 Kbps	0.047	15.2	36.9
100 Kbps	0.045	14.2	58.0
50 Kbps	0.047	13.9	57.7
Baseline	0.033	10.3	63.1

Table XXXIII. ARMA Three-Step-Ahead Predictions for Cross-traffic Test Cases.

Cross-traffic	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
90% decrease	0.001	0.7	150.7
75% decrease	2.658	51.4	54.9
50% decrease	1.153	68.3	61.4
25% decrease	0.152	23.4	57.6
25% increase	0.039	7.8	61.9
50% increase	0.050	9.3	78.7
100% increase	0.050	8.2	88.0
150% increase	0.029	7.8	84.8
Baseline	0.033	10.3	63.1

send rate test case is 74.2. This is more than that of the baseline validation data set. The errors, in general, are more than those obtained from AR and ARMA predictions. Table XXXV presents the performance results of FMLP predictor on various cross-traffic test cases. The test cases with 75% decrease and 50% decrease from baseline cross-traffic have the highest MAE's of 51.0 and 67.9 msecs respectively.

3. Four-Step-Ahead Prediction

This section presents the performance results of four-step-ahead predictions using AR, ARMA and FMLP predictors. Tables XXXVI and XXXVII provide performance results of four-step-ahead prediction using the AR predictor. In general the prediction errors do not deviate much from those obtained in three-step-ahead prediction. From Table XXXVI, it can be seen that the MSE_2 error on all the test cases is lesser than the MSE_2 on the baseline validation test set. The MAE seems to increase with

Table XXXIV. FMLP Three-Step-Ahead Predictions for Send Rate Test Cases.

Send Rate	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
800 Kbps	0.060	12.9	46.9
500 Kbps	0.072	15.8	50.9
400 Kbps	0.045	15.8	74.2
200 Kbps	0.057	15.6	44.2
100 Kbps	0.051	14.6	65.1
50 Kbps	0.050	14.4	60.9
Baseline	0.035	10.1	65.6

Table XXXV. FMLP Three-Step-Ahead Predictions for Cross-traffic Test Cases.

Cross-traffic	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
90% decrease	0.008	1.1	829.0
75% decrease	3.101	51.0	64.0
50% decrease	0.521	67.9	80.9
25% decrease	0.183	22.9	69.4
25% increase	0.044	9.8	69.3
50% increase	0.055	8.9	85.6
100% increase	0.053	7.8	93.9
150% increase	0.032	7.5	93.8
Baseline	0.035	10.1	65.6

increase in the bit-rate but is the least for the maximum send rate of 800 Kbps. Table XXXVII provides the performance results on the cross-traffic test cases. The AR predictor gives good results for the test case with 25% increase from the baseline cross-traffic. Both the MAE and MSE_2 are less and are comparable to that of the baseline set.

Table XXXVI. AR Four-Step-Ahead Predictions for Send Rate Test Cases.

Send Rate	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
800 Kbps	0.053	12.6	41.9
500 Kbps	0.062	15.4	44.1
400 Kbps	0.036	15.3	59.8
200 Kbps	0.052	15.2	40.5
100 Kbps	0.047	14.2	60.8
50 Kbps	0.048	13.9	58.3
Baseline	0.033	10.3	63.2

Tables XXXVIII and XXXIX provide performance results of four-step-ahead prediction using the ARMA predictor. Table XXXVIII shows that the MAE increases with increase in the bit-rate. However 800 kbps has the least MAE of 12.6 msecs. The predictor works well in the operating range. Table XXXIX shows that the test case with 50% decrease from the baseline cross-traffic has the highest MAE of 68.3 msecs and the corresponding MSE_2 of 69.3%. If the test case with 90% decrease is considered as a failed prediction case, the test case with 25% increase from baseline cross-traffic has the least MAE of 7.8 msecs.

Tables XL and XLI provide performance results of four-step-ahead prediction using the FMLP predictor. From Table XL, it can be seen that the errors obtained

Table XXXVII. AR Four-Step-Ahead Predictions for Cross-traffic Test Cases.

Cross-traffic	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
90% decrease	0.001	0.7	148.3
75% decrease	2.782	51.4	57.5
50% decrease	1.305	68.3	69.5
25% decrease	0.170	23.4	64.3
25% increase	0.040	7.8	63.2
50% increase	0.054	9.1	84.0
100% increase	0.052	7.9	92.7
150% increase	0.031	7.7	91.6
Baseline	0.033	10.3	63.2

Table XXXVIII. ARMA Four-Step-Ahead Predictions for Send Rate Test Cases.

Send Rate	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
800 Kbps	0.053	12.6	41.5
500 Kbps	0.061	15.4	43.0
400 Kbps	0.037	15.3	61.9
200 Kbps	0.053	15.2	41.1
100 Kbps	0.049	14.2	62.9
50 Kbps	0.049	13.9	59.3
Baseline	0.033	10.3	63.5

Table XXXIX. ARMA Four-Step-Ahead Predictions for Cross-traffic Test Cases.

Cross-traffic	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
90% decrease	0.001	0.7	153.9
75% decrease	2.875	51.4	59.4
50% decrease	1.302	68.3	69.3
25% decrease	0.162	23.4	61.3
25% increase	0.038	7.8	60.5
50% increase	0.048	8.9	74.5
100% increase	0.046	8.1	82.5
150% increase	0.027	7.6	80.3
Baseline	0.033	10.3	63.5

are higher when compared to errors from AR and ARMA predictions. Table XLI has the same characteristics as those of the other predictors, mainly the test case with 50% decrease in cross-traficc has the highest MAE. The FMLP predictor does not perform well as the time-step increases gradually.

4. Ten-Step-Ahead Prediction

This section presents the performance results of ten-step-ahead predictions using AR and ARMA predictors. Tables XLII and XLIII provide performance results of ten-step-ahead prediction using the AR predictor. A ten-step-ahead prediction gives half a second of prediction information in the future horizon. Table XLII shows that the developed predictor gives the least MSE_2 for the test case with 200 kbps send rate. Table XLIII shows that the predictor fails on most of the cross-traffic test cases. Similarly, Tables XLIV and XLV provide performance results of ten-step-ahead

Table XL. FMLP Four-Step-Ahead Predictions for Send Rate Test Cases.

Send Rate	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
800 Kbps	0.060	12.9	46.7
500 Kbps	0.072	15.8	50.7
400 Kbps	0.045	15.8	74.5
200 Kbps	0.058	15.7	44.8
100 Kbps	0.051	14.6	65.7
50 Kbps	0.051	14.4	61.7
Baseline	0.035	10.1	65.6

Table XLI. FMLP Four-Step-Ahead Predictions for Cross-traffic Test Cases.

Cross-traffic	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
90% decrease	0.008	1.1	852.3
75% decrease	3.095	51.0	63.9
50% decrease	1.520	67.9	80.9
25% decrease	0.183	22.9	69.4
25% increase	0.044	9.8	70.1
50% increase	0.056	8.9	86.7
100% increase	0.054	7.8	95.5
150% increase	0.032	7.6	95.0
Baseline	0.035	10.1	65.6

prediction using the ARMA predictor. The results indicate a poor performance of the ARMA predictor in both the send rate test case scenario as well as the cross-traffic test case scenario.

Table XLII. AR Ten-Step-Ahead Predictions for Send Rate Test Cases.

Send Rate	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
800 Kbps	0.092	12.6	72.1
500 Kbps	0.102	15.3	71.9
400 Kbps	0.058	15.3	96.1
200 Kbps	0.081	15.2	62.8
100 Kbps	0.083	14.2	106.1
50 Kbps	0.085	13.9	103.5
Baseline	0.046	10.9	88.1

Table XLIII. AR Ten-Step-Ahead Predictions for Cross-traffic Test Cases.

Cross-traffic	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
90% decrease	0.001	0.5	156.5
75% decrease	4.500	51.4	92.9
50% decrease	1.846	68.3	98.3
25% decrease	0.230	23.4	86.9
25% increase	0.096	12.1	152.9
50% increase	0.114	12.7	177.9
100% increase	0.107	12.7	190.6
150% increase	0.060	11.9	179.3
Baseline	0.046	10.9	88.1

Table XLIV. ARMA Ten-Step-Ahead Predictions for Send Rate Test Cases.

Send Rate	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
800 Kbps	0.089	12.6	69.7
500 Kbps	0.099	15.3	69.7
400 Kbps	0.058	15.3	97.1
200 Kbps	0.084	15.2	65.5
100 Kbps	0.084	14.2	107.8
50 Kbps	0.086	13.9	105.2
Baseline	0.047	11.0	89.0

Table XLV. ARMA Ten-Step-Ahead Predictions for Cross-traffic Test Cases.

Cross-traffic	MSE_1 (%)	MAE (in msec)	MSE_2 (%)
90% decrease	0.001	0.5	156.5
75% decrease	4.500	51.4	92.9
50% decrease	1.846	68.3	98.3
25% decrease	0.230	23.4	86.9
25% increase	0.096	12.0	152.9
50% increase	0.114	12.7	177.9
100% increase	0.107	12.7	190.7
150% increase	0.060	11.9	179.3
Baseline	0.047	11.0	89.0

I. Chapter Overview

This chapter dealt with the training, testing and validation of the linear and nonlinear predictors for the MSP of moving average one-way end-to-end delays. The different time steps used in this research are two-step, three-step, four-step and ten-step. The various performance metrics used for the evaluation of MSP are discussed in this chapter. The chapter provides the results of MSP using both linear and nonlinear models on the various test cases. The models gave a good prediction on most of the test cases but MSP is not as accurate as SSP and fails on several test cases as the prediction horizon is increased gradually.

CHAPTER VII

SUMMARY AND CONCLUSIONS

A. Summary

The objective of this research study is to develop predictors for end-to-end packet delays in a best-effort network capable of performing accurate single-step-ahead prediction (SSP) and multi-step-ahead prediction (MSP). The proposed predictors are tested on simulated data generated from a widely used simulator called network simulator (ns-2). An empirical model with the above mentioned capabilities has many applications in the field of much rapidly growing Internet applications and services. Due to the highly dynamic behavior of best-effort networks, it is very difficult to apply traditional methods such as queuing theory and other statistical methods for modeling such systems. The Wide area network (WAN) traffic is often bursty and heavy tailed leading to the non-equilibrium nature of the end-to-end packet delays that were analyzed in this study. In this research, the end-to-end packet delays have been modeled using system identification (SI) techniques involving both linear models as well as neural network based nonlinear models. The linear methods used for modeling are Auto-Regressive eXogenous (AR) and Auto-Regressive Moving Average eXogenous (ARMA), whereas the nonlinear method used in this study is a Feedforward Multilayered Perceptron (FMLP).

In Chapter I, a detailed review of literature used for the research is presented. The literature covers most of the work done in this area including some recent advances made in this field. This chapter provides information on research done in end-to-end packet delay measurements, delay estimation, and the use of system identification and artificial neural networks (ANN's) for empirical modeling of network delay dynamics.

A qualitative discussion on end-to-end packet delays in best-effort networks has been presented in Chapter II. The first half of this chapter mainly deals with the causes of delays, delay variations, packet losses, and throughput in the network. These metrics have a direct impact on Quality of Service (QoS) and, hence, the impact of these metrics on real-time application QoS has been addressed in the second half of this chapter.

Chapter III gives a detailed description of SI. In this chapter, the linear methods AR and ARMA model structures have been explained along with their mathematical equations used. The algorithms for training SSP and MSP developed in this research are explained in some detail in this chapter.

Chapter IV describes the measurement and analysis techniques used for collecting the data required for this research. It is divided into two main parts, one dealing with the collection of simulated data and the other deals with the collection of real traffic data. Each of these parts explains in detail the type of network topology and setup, bit rate, and the various types of traces collected for this study. Furthermore, these traces are analyzed using their auto-correlation to check for long range and short range dependencies.

In Chapter V, the various performance metrics used as performance indicators for the prediction are discussed and the results of single-step-ahead predictions SSP are presented. The results obtained from all the three predictors using AR, ARMA and FMLP are compared. SSP of moving average time series of end-to-end packet delays is quite accurate. The predictors gave good predictions on most of the cases used for testing. The quality of SSP paved way for the development of MSP predictors for moving average time series of end-to-end packet delays. These results for which are presented in Chapter VI.

Chapter VI discusses the results of MSP. In this chapter, results from predictors based on linear methods AR , ARMA as well as predictions from FMLP are presented and compared. These predictions yielded good results, though not as accurate as the SSP. Improvement is needed in four-step and ten-step-ahead prediction.

B. Conclusions and Recommendations

The proposed approach in this study has a direct impact on the end-to-end network delay dynamics, though not much can be done in changing the behavior of the delay dynamics. Empirical models like these can be used in developing effective bandwidth allocation and network control strategies. An effective delay-based bandwidth allocation scheme and a prediction based congestion control scheme can lead to improved QoS of non-interactive and interactive real-time multimedia applications.

The following are the conclusions drawn from this study:

1. The use of linear system identification techniques and neural networks as nonlinear model structures to identify the end-to-end delay dynamics of a best-effort network, such as an Internet, seems possible. Network measurements can be used to obtain empirical models to predict the network delay behavior.

2. It is observed in this study that SSP is more accurate than MSP. In reality MSP is needed to produce delay predictions within a finite future prediction horizon. This is because of the element of delay involved in the network while collecting delay measurements itself.

3. The developed predictors perform accurately over a wide range of different network conditions designed for this study. The conditions involve changes in the source send rate and the network cross-traffic. This indicates that the developed predictors are able to identify the IP-level open-loop network dynamics of an application which uses UDP as the transport protocol.

The following recommendations are proposed for further research in this area:

1. Applicability of the proposed approach to real traffic data measurements is an issue, as it has been shown in this study that real traffic delay measurements contain long range temporal dependencies and nonlinearities.

2. An accurate test bed needs to be set up for collecting both active and passive delay measurements through the Internet.

3. Further optimizing the MSP results is a necessity as MSP is more needed than SSP to produce a delay prediction within a finite future prediction horizon.

4. Empirical models have to be developed to model closed-loop network dynamics of applications using TCP as the transport protocol.

REFERENCES

[1] H. Sawashima , Y. Hori , and H. Sunahara , “Characteristics of UDP packet loss: Effect of TCP traffic,”, in *Proceedings of the 1997 INET*, Kuala Lumpur, Malaysia, June, 1997.

[2] Connectionless Transport: UDP, *Available http://www-net.cs.umass.edu/kurose/transport/UDP.html, Accessed on March 2001.*

[3] V. Paxson and S. Floyd, “Why we don’t know how to simulate the internet,” in *Proceedings of the 1997 Winter Simulation Conference*, Atlanta, Georgia, December 1997.

[4] R. Jain, “A delay-based approach for congestion avoidance in interconnected heterogenous computer networks,” *ACM Computer Communication Review*, vol. 19, pp.56-71, October 1989.

[5] V. Paxson, “Measurements and analysis of end-to-end internet dynamics,” Ph.D. Dissertation, University of California, Berkeley, April 1997.

[6] V. Paxson, “End-to-end internet packet dynamics,” in *Proceedings of ACM SIGCOMM* San Francisco, California, vol. 7, no. 3, pp. 139–152, 1997.

[7] J-C. Bolot, “End-to-end packet delay and loss behaviour in the internet,” in *Proceedings of SIGCOMM’93*, San Francisco, California, pp.289–298, September 1993.

[8] V. Paxson and S. Floyd , “Wide-area traffic: The failure of poisson modeling,” in *IEEE/ACM Transactions on Networking*, vol. 3, no. 3, pp. 226–244, 1995.

[9] H. Ohsaki, M. Murata and H. Miyahara, "Modeling end-to-end packet delay dynamics of the Internet using system identification," *presented at International TeleTraffic Congress*, pp. 1027–1038, December 2001.

[10] A. G. Parlos, "Identification of the Internet end-to-end delay dynamics using multi-step neuro-predictors," in *Proceedings of the 2002 International Joint Conference on Neural Networks*, vol.3, pp. 2460–2465, May 2002.

[11] H. Ohsaki, M. Morita, and M. Murata, "On modeling round-trip time dynamics of the internet using system identification," *IEICE Trans. COMMUN.*, vol. E85-B, no. 1, January 2002.

[12] S. Keshav, "REAL: A network simulator," Technical Report 88/472, Department of Computer Science, University of California, Berkeley, 1988.

[13] L. S. Brakmo and L. L. Peterson, "Experiences with network simulation," in *Proceedings of the ACM SIGMETRICS*,pp. 80–90, Philadelphia, PA, 1996.

[14] NS, UCB/LBNL/VINT Network Simulator - ns (version 2), *Available http://www.isi.edu/nsnam/ns/, Accessed on March 2001.*

[15] I. Yeom, "ENDE: An end-to-end network delay emulator ," M.S. Thesis, Texas A&M University, College Station, August 1998.

[16] S. Sesham, M. Stemm, and R. Katz, "SPAND: Shared passive network performance discovery," in *Proceedings of the USENIX Symposium on Internet Technologies and Systems*, pp. 135–146, December 1997.

[17] RIPE, Réseaux IP Européens, *Available http://www.ripe.net, Accessed on January 2001.*

[18] C. J. Bovy, H. T. Mertodimedjo, G. Hooghiemstra, H. Uijterwaal, and P. Van Mieghem, “Analysis of end-to-end delay measurements in Internet,” in *Proceedings of the Passive and Active Measurements Workshop, PAM2002*, pp. 26–33, Ft.Collins, Colorado, March 2002.

[19] R. Wolski, N. T. Spring and J. Hayes, “The Network Weather Service: a distributed resource performance forecasting service for metacomputing,” *Journal of Future Generation Computing Systems*, vol. 15, no. 5-6, pp. 757-768, October 1999.

[20] C. You and K. Chandra, “Time series models for Internet data traffic,” in *Proceedings of 24th conference on Local Computer Networks, LCN-99*, pp. 164-171, October 1999.

[21] L. Kulkarni and S. Li, “Measurement-based traffic modeling: Capturing important statistics,” *Journal of Stochastic Model*, vol. 14, no. 5, 1998.

[22] A. G. Parlos, O. T. Rais, and A. F. Atiya, “Multi-step-ahead prediction using dynamic recurrent neural networks,”*Neural Networks*, vol. 13, pp. 765–786, September, 2000.

[23] A. Bhattacharya, “Prediction of MPEG-coded video source traffic using neural networks,” M.S. Thesis, Texas A&M University, College Station, December, 2002.

[24] M. Norgaard, O. Ravn, N.K. Poulsen, and L.K. Hansen, *Neural networks for modeling and control of dyanamic systems - A practitioner’s handbook*, Springer Verlag, London, 2000.

[25] S. Haykin, *Neural networks: A comprehensive foundation*, 2nd ed., Prentice Hall, New Jersey, 1999.

[26] J. W. Mangan III, “Adaptive control of media applications in best-effort networks using reliable transport protocols ,” M.S. Thesis, Texas A&M University, College Station, August 2002.

[27] J. F. Kurose and K. W. Ross, *Computer networking: A top down approach featuring the Internet*, Addison Wesley, New Delhi, India, 2001.

[28] U. Schwantag, “An analysis of the applicability of RSVP,” Diploma Thesis, University of Oregon, Oregon and Institute of Telematics, University of Karlsruhe, July 1997.

[29] S. Joyce, “Traffic on the Internet - a study of Internet games traffic,” BCMS 420 Honours project report, University of Waikato, Hamilton, New Zealand, October 2000.

[30] A. G. Parlos, “Lecture notes: Neural networks for intelligent estimation and control,” Texas A&M University, College Station, December 2002.

[31] L. Ljung, *System identification theory for the user*, Prentice Hall, Englewood Cliffs, New Jersey, 1987.

[32] J. W. Forrester, *Urban dynamics*, MIT Press, Cambridge, Massachusetts, 1969.

[33] M. S. Borella, “On estimating long range dependence of network delay,” *International Journal of Chaos Theory and Applications*, vol. 6, no. 4, December 2001.

APPENDIX A

DATA SETS USED IN THIS STUDY.

Baseline Data	Avg. Loss (%)	Avg. $E2e^*$ Delay (in msec)	$E2e^*$ Variance
Training Set	3.3	136.4	14.1
Testing Set	2.5	137.4	12.4
Validation Set	3.5	137.5	10.0
Send Rate Test Cases			
800 Kbps	2.7	132.2	25.7
500 Kbps	3.3	136.2	25.2
400 Kbps	1.5	138.9	11.0
200 Kbps	1.7	143.4	22.2
100 Kbps	1.5	147.1	13.0
50 Kbps	2.2	149.1	14.7
Cross-traffic Test Cases			
90% decrease	0.6	52.0	0.1
75% decrease	0.6	103.6	427.1
50% decrease	1.5	123.9	255.6
25% decrease	2.0	138.6	46.2
25% increase	1.5	157.8	9.8
50% increase	0.6	161.4	9.0
100% increase	0.7	162.2	7.8
150% increase	0.3	163.8	4.9

* where E2e means End-to-end.

VITA

Srikar Doddi was born in Vijayawada, Andhra Pradesh, India on the 27th of March, 1979. He received his B.E. degree in Mechanical Engineering from Osmania University, Hyderabad, India, in 2000. In August 2000, he joined the master's degree program in Mechanical Engineering at Texas A&M University.

Permanent Address:
1-1-276, R. K. Apartments,
Apt 302, Chikkadapally,
Hyderabad, India 500020.
Contact email address: srikar@neo.tamu.edu

The typist for this thesis was Srikar Doddi.