Design and Performance Analysis of Link-by-Link Congestion Avoidance Algorithm for the Multiplexed Traffic in the Internet

Sneha Thombre

Certificate

This is to certify that the thesis entitled " **Design and Performance Analysis of Link-by-Link Congestion Avoidance Algorithm for the Multiplexed Traffic in the Internet**", submitted by Sneha Kiran Thombre (Roll No.10610203), a research scholar in the Department of *Electronics and Electrical Engineering, Indian Institute of Technology Guwahati*, for the award of the degree of **Doctor of Philosophy**, is a record of original work carried out by her under my supervision and guidance. The thesis has fulfilled all requirements as per the regulations of the Institute and in my opinion has reached the standard needed for submission. The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

Prof. Lalit Mohan Patnaik
Consciousness Studies Program
National Institute of Advanced Studies
Indian Institute of Sciences Campus,
Bangalore- 560012, Karnataka, India.

Sign
August 2020

Prof. Sanjay Kumar Bose
Department of Electronics and Electrical Engineering
Indian Institute of Technology Guwahati,
Guwahati- 781039, Assam, India.

Sign
August 2020

Design and Performance Analysis of Link-by-Link Congestion Avoidance Algorithm for the Multiplexed Traffic in the Internet

A
Thesis Submitted
in Partial Fulfillment of the Requirements
for the Degree of
DOCTOR OF PHILOSOPHY
By

SNEHA KIRAN THOMBRE
Under the Supervision of
PROF. LALIT MOHAN PATNAIK
PROF. SANJAY KUMAR BOSE

Department of Electronics and Electrical Engineering
Indian Institute of Technology Guwahati
June, 2020

Dedicated to the loving memory of my in-laws,
[Late Hari Thombre and Late Shashikala Thombre],
You are gone but your belief in me has made this journey possible and my parents, [Hemant Yeolekar and Jayashri Yeolekar] who always believed in my ability to be successful in the academic arena.

Acknowledgements

This thesis is the culmination of my journey of Ph.D. which was just like climbing a high peak step by step accompanied by encouragement, hardship, trust, and frustration. When I found myself at top experiencing the feeling of fulfillment, I realized though only my name appears on the cover of this dissertation, a great many people including my family members, well-wishers, my friends, colleagues and various institutions have contributed to accomplishing this task. At this moment of accomplishment, I am greatly indebted to my research guide, Prof. L. M. Patnaik, who offered me his mentorship. My earnest thanks to Prof. Sanjay Bose, for accepting me as his student and supporting this work. I am grateful for his valuable academic advice and timely administrative support. No research is possible without infrastructure and requisite materials and resources. For this, I extend thanks to Dr. Madhuri Khambete, Principal, Cummins College of Engineering for Women, Pune for providing me with the requisite institutional facilities throughout my research tenure. I greatly appreciate and acknowledge the support received from her and the Institute. This work would not have been complete without the guidance of Prof. Anil Tavildar. I will remain indebited to him always. I would like to mention my sincere thanks to my doctoral committee members: Prof. Ratnajeet Bhattacharjee, Prof. Praveen Kumar, and Dr. A Rajesh for evaluating the progress of my work. Their periodic suggestions were always very valuable. I extend my thanks to the EEE office and the Academic section for all sorts of administrative help. It's my fortune to gratefully acknowledge the support of my friends, Pratik Rathod and Arjit for their support and generous care throughout the research tenure. Finally, I acknowledge the people who mean a lot to me, my parents, Aai, Baba, and sisters for showing faith in me.

I owe thanks to a very special person, my husband, Kiran for his continued and unfailing love, support and understanding during my pursuit of a Ph.D degree that made the completion of the thesis possible. You were always around at times I thought that it is impossible to continue, you helped me to keep things in perspective. I greatly value his contribution and deeply appreciate his belief in me. I appreciate my son Saket for abiding my ignorance and the patience he showed during my Ph.D. tenure. Words would never say how grateful I am to both of you. I consider myself the luckiest in the world to have such a lovely and caring family, standing beside me with their love and unconditional support.

I thank the Almighty for giving me the strength and patience to work through all these years so that today I can stand proudly with my head held high.

Sneha Kiran Thombre

Abstract

The performance of the Internet is closely linked to the dominant TCP protocol and the performance of the TCP is associated with its congestion control mechanism. With the increasing usage of TCP protocol, the worsening of end-to-end delay and jitter performance is a serious concern affecting QoS requirements for Internet communications, particularly for the real-time applications. Jitter is particularly important to manage the QoS of the real-time applications. Queuing delay and jitter are clearly related to congestion control and occur at the network layer; therefore, the queuing delay and jitter are analyzed at the network layer in this thesis work. Datagrams at the routers are from the number of multiplexed flows and constitute a stochastic process. To quantitatively analyze the effect of TCP multiplexing initially, arrival and service processes for the multiplexed TCP and UDP datagrams at the congested router output are modeled. The model accounts for the fraction of TCP and UDP datagrams (as they contend for resources), the arrival and service distributions of TCP and UDP with their respective datagram sizes. Mean queuing delay, average instantaneous queuing delay, and jitter are quantified using queuing theory and the arrival and service process model. The findings of the analytical predictions are validated using NS2 simulations. The interesting observation is that multiplexing of the TCP flows hurts the performance of fellow flows. Delay and jitter of the tagged flow are adversely affected by the fraction of TCP in the background traffic. This is particularly important because TCP dominates the overall traffic. The simulation results fairly agree with the analytical predictions. The degradation of the average mean delay for TCP datagrams at the router, which has the highest proportion of the TCP in the background is as large as 400 %. For jitter, the degradation for datagrams of a typical flow is more than 3 times, for the highest proportion of TCP in the background traffic. These conclusions are true for Cubic, Reno, and Compound TCP flavors also. Therefore, it is evident that a new TCP variant is not the solution for TCP delay friendliness issues. Further, in this thesis work, a new approach, namely the Link-by-link Congestion Avoidance (LbLCA) algorithm, which works at the network layer, has been proposed. The LbLCA is a proactive congestion avoidance algorithm. It uses explicit feedback to prevent congestion to happen in the first place. The novelty of the LbLCA is that no per-flow information is required, which makes it more scalable. Based on the design philosophy and equations, the sizing of various router buffers has

been arrived at for typical network topologies. Buffer sizes in the LbLCA depend upon the mean arrival rate at router input and outgoing link capacities and are independent of round trip time (RTT) and the number of flows passing through the router. The buffer sizes determined using LbLCA design are validated using extensive NS2 simulations. The performance evaluation has been carried out using NS2 simulations on the typical network topologies. The performance comparison between the TCP and the LbLCA reveals that the proposed LbLCA algorithm gives improved performance for the end-to-end delay and packet delivery ratio. The LbLCA is impartial to all flows, as the LbLCA works at the network layer and therefore, cannot differentiate between flows. Furthermore, the issue of the router buffer size design is significant as it is closely associated with the performance of the Internet. In the Internet, router buffer design is related to the congestion control mechanism of TCP which is the most commonly used protocol. Many rules have been presented by researchers for Internet router buffer design. However, there is no agreement on the buffer rule to be followed in the Internet router buffer sizing. This thesis also addresses the issue of buffer size design in LbLCA using linear multiple regression and proposes that it is possible to predict the buffer size value for any core and edge router with multiple input/output ports, making use of the linear multiple regression technique. The buffer values predicted using multiple linear regression have small standard error and feasible coefficient of determination (R^2) equal to 0.8301. A coefficient of determination realized equal to 0.8301 indicates that about 83% of the variation in the router buffer size can be explained by the arrival rates at the router ports following LbLCA. This would be contemplated a good fit under the assumption that it would substantially help predict buffer sizing in the routers following LbLCA. Initially, the estimated router buffer size values using LbLCA are used to train the model. After training the model, buffer size values can be predicted using the regression equation for the given network scenario for the LbLCA router. The LbLCA is implemented in the NS2 simulator with buffer values predicted by linear multiple regression. Thereafter, a performance comparison between TCP with buffer sizes following small buffer rule and LbLCA with buffer sizes predicted by linear multiple regression reveals that the LbLCA algorithm gives improved performance for the end-to-end delay, throughput, packet delivery ratio and jitter. The lower network delay and the lower jitter (peak-to-peak) values prove LbLCA an option to TCP, especially for the delay-sensitive applications. Though the preliminary results of LbLCA are encouraging, there is a need to really ensure that LbLCA is indeed a viable solution; this can be taken up as continuing research activity.

Contents

List of Figures

List of Tables

Glossary

TCP	Transmission Control Protocol
UDP	User Datagram Protocol
QoS	Quality of Service
E2ED	End-to-End Delay
PDR	Packet delivery ratio
MAC	Medium Access Control
OSI	Open Systems Interconnections
cwnd	Congestion Window
rwnd	Receiver Window
AIMD	Additive increase Multiplicative Decrease phase
SS	Slow Start phase
LbLCA	Link-by-Link Congestion Avoidance Algorithm
RED	Random Early Detection
RTP	Real Time Protocol
RTCP	Real Time Control protocol
Ri	R_i^{th} upstream router
Ri+1.	R_{i+1}^{th} downstream router
API	Application Program Interface Access
HTTP	Hyper Text Transfer Protocol
HTTPS	Secure Hyper Text Transfer Protocol
FTP	File Transfer Protocol

Chapter 1

Introduction

THE Internet has revolutionized human life. While the use of the Internet was originally thought of for elastic kind of applications, in reality, the Internet has been extensively used for both the elastic, as well as, the non-elastic i.e. delay-sensitive or real-time applications. The Internet makes extensive use of the TCP/IP protocol suite, which works on the end-to-end argument principle and the best effort mechanism, with no admission control policy for datagrams and no guarantee for their delivery. The performance of the Internet has been closely linked to the TCP protocol, a large portion of the Internet traffic has been carried over this protocol. HTTP, HTTPS, and FTP are the widely used application layer protocols which in turn use TCP as their underlying transport protocol. The performance of TCP depends upon the congestion control mechanism built into the protocol. The congestion control algorithm does not merely avoid congestion collapse, but also acts as a resource allocation mechanism [1, 2]. Therefore, this makes TCP slightly unsuitable for delay-sensitive applications. Some researchers have suggested that the UDP may perform better for such applications as UDP has been a simple request-response protocol. This is because UDP minimizes the delays and therefore is a preferred protocol for real-time applications.

However, IETF RFC 8085 forbids using UDP to transport data, unless it has been accompanied by congestion control. For real-time applications, delay jitter remains one of the most pertinent parameters of quality-of-service (QoS). Complying to the stringent ITU G.114 standard of one-way delay to be 150 msec, still appears to be a challenge. With the increasing usage of TCP protocol, the worsening of end-to-end delay and delay jitter performance is a serious concern affecting QoS requirements for Internet communications, particularly for real-time applications. Queuing delay and jitter are much more difficult to analyze or evaluate. Jitter is particularly important to manage the QoS of real-time applications. For most applications, the jitter at the receiver side is adjusted by using a de-jitter (playback) buffer to provide a regular packet stream for the application. Estimating and controlling the jitter is important to avoid both, a buffer underflow when the application does not receive packets for some time and buffer overflow, which causes packet loss. In either case, the users experience a degraded QoS. De-jitter buffer is the most widely used method to control jitter.

The minimum value of congestion and flow control has been selected as window size (cwnd) by TCP protocol. Each flow has been subjected individually to congestion control and flow control mechanism. Datagrams at the routers are from number of multiplexed flows which constitute a stochastic process. Datagrams from these multiplexed flows share resources and therefore strongly affect and get affected by fellow datagrams [2]. Further, these flows may have radically different performance requirements depending upon the applications they support, like some flows may be delay sensitive and the rest may be elastic. Furthermore, congestion control is related to buffer size and therefore directly impacts queuing delay. The behavior of TCP flows when they interact has been quantitatively analyzed in this thesis work. A new model has been proposed for arrival and service distribution of datagrams which constitute TCP flows at the router.

The number of flows M(n) generated at the transport layer is a dynamic process. To analyze quantitatively the impact of multiplexing of flows with regards to the delay performance is the motive of this thesis work. The thesis work attempts to gain insights into the questions like do the multiplexed flows influence each other in terms of QoS if so, then do they collaborate or harm each other? The queue modeled in this thesis work is to be considered at the network layer and is to be applied to the datagrams of a congested

router. Even though the sending rates of the source flows are decided by the transport layer which understands congestion and flow control, when datagrams of these flows first hit the edge router and are buffered, the outgoing rates are changed. Then onward, incoming and outgoing rates at the subsequent routers are decided by router buffering and bear hardly any correlation to the original sending source rates. Buffering changes the timing information between the rates decided by source flows at the transport layer and actual outgoing rates followed by the routers. To understand the effect of bursty arrivals and the role of buffering in communication networks, datagram arrivals at router links can be modeled as random processes. In this thesis work, the arrival and service rates have been modeled at a typical congested router considering buffering and then queuing delay performance has been analyzed. Even though TCP is the most studied and widely used protocols and precincts of TCP are known, researchers have suggested some advances may be possible through the investigations of delay sensitivity of TCP [3]. The end-to-end delay has components like propagation delay, transmission delay, processing delay which are all fixed, predictable components for a given network environment and have been adequately characterized. The one random component is queuing delay and the end-to-end delay is affected by the queuing delay [4, 5]. The datagrams at the router queues (TCP and UDP) share resources and therefore impact and get impacted by fellow datagrams and also by the proportion of TCP in the Internet traffic. The fraction of TCP has crossed 90 % mark in the Internet traffic. Therefore, evaluation of the performance due to interaction between datagrams of TCP and also between TCP and UDP is imperative. The rationale of this thesis work is also to analyze and quantify the impact of varying TCP proportion in the background traffic on, mean delay of datagrams using queuing theory and on average instantaneous delay of the datagrams of a tagged flow using statistical analysis of a discrete queue. The study related to increased queuing delay with an increased proportion of TCP in the background traffic has not been exhaustively addressed so far.

Further, to evaluate the impact of TCP flows on UDP such as on performance parameters like throughput has also been the motivation of this thesis work. This research work proposes an algorithmic model for UDP throughput in the presence of TCP.

Internet routers are packet-switched and provide buffering of the packets during times of

congestion. Possibly, router buffers are the single main contributor to the delay uncertainty on the Internet. Router buffers cause variable queuing delay and introduce delay jitter [6]. The router buffer design is crucial to the delay performance of the Internet. Smaller router buffers cause the packets to be lost which adversely impacts the packet delivery of the system. On the other hand, large buffers introduce delays and jitters. Given the significance of the buffer, the dynamics and sizing of the router buffers need to be investigated for minimizing the delay and maximizing the packet delivery ratio. The router buffer design plays a very significant role related to congestion control and thereby for queuing delay. The motivation of this thesis work is to propose the design of a new, simple, but novel congestion control algorithm. The design goals are, firstly, the recommended algorithm should be independent of Additive Increase Multiplicative decrease (AIMD) and use explicit feedback from neighbors. Secondly, the proposed algorithm is expected to work at the network layer where congestion occurs and therefore, should not distinguish between flows because fairness in handling flows is a crucial issue. In this thesis work, using the proposed algorithm buffer size of the router has been determined, which directly controls the variable queuing delay. The outcome of this thesis work is that the proposed algorithm provides better performance (minimum end-to-end delay and maximum packet delivery ratio) which can coexist with the existing TCP. Though, the initial results of LbLCA are encouraging further continued research is required to prove LblCA as a viable option to avoid congestion in the Internet.

LbLCA is a network layer algorithm. It can be used along with transport layer protocols like UDP, RTCP etc. instead of TCP. The simulations carried out in the thesis indicate that, with increase in the TCP fraction in the overall traffic, the end-to-end delay increases substantially. This can be a major challenge particularly for the delay sensitive applications. Most of the solutions suggested or applied on the Internet are implemented at the application layer or by implementing different TCP variants. Therefore, individual applications implement their own mechanism to address the challenge of maintaining low end-to-end delay using TCP. The thesis, therefore, has proposed that LbLCA may be used in the network layer as congestion occurs at the network layer. The LbLCA may be considered as possible alternative to meet the QoS requirements specifically for the delay sensitive applications. This, however, would need comprehensive experimentation

in the real Internet environment. Similar cross layer techniques have been successfully explored earlier also. The major contribution of this thesis work is the proposition of a different alternate to the widely used TCP protocol. The design philosophy of the proposed LbLCA protocol has been based on the buffer sizing considerations of the router. The delay performance of any congestion control protocol has been directly related to the size of the router buffer. Therefore, buffer sizes are determined using the LbLCA algorithm. The performance of the proposed LbLCA algorithm has been compared with normal TCP (with AIMD congestion control). Further, the estimated buffer sizes are used to predict the router buffer size using multiple linear regression and followed by the implementation of the proposed algorithm in NS2. Use of machine learning method like multiple linear regression for prediction of router buffer sizes is innovative. It has been observed that LbLCA provides improved End-to-End Delay and Packet Delivery Ratio (PDR) performance, compared to the normal TCP based traffic flows, which suffer from excessive end-to-end delays, compromising on QoS parameters.

1.1 Motivation and objectives

The network delay has been an important parameter adversely affecting the Quality-of-Service (QoS) for Internet communications, particularly for real-time applications. The solutions that have been attempted are either by adding newer TCP flavors or some proprietary mechanisms to the individual applications. Numerous models are available for a mixture of TCP and UDP, but none of them specifically quantify the effect on queuing delay performance due to an increased proportion of TCP in the Internet traffic. Therefore, the need for a new model does exist.

Due to the concern that queuing delay has been on the rise and TCP is the dominant protocol, the investigation of the effect of multiplexing of the TCP flows at the transport layer on fellow TCP and UDP is the primary motivation of this thesis. A flow represents a set of datagrams possessing common properties and typically holds aggregated statistics about the datagrams corresponding to the flow. An application user flow has been qualified by a tuple with attributes such as source IP address, source port number, destinations IP address and destination port number. This thesis work analyzes the

variable queuing delay by modeling the arrival and service processes as random processes and by accounting, various proportions of TCP and UDP with their respective datagram sizes. Next, the average instantaneous delay for the flow of interest has been derived using discrete transient queue analysis. To ascertain the values of the Internet traffic, the dataset used in this paper has been generated by collecting real-time data of traffic measurements done at a leading academic Institute in India[1]. The curve fitting procedure has been used on the Internet traffic data collected for various parameters like arrival process, varying datagram sizes and inter-arrival times of TCP, UDP and TCP plus UDP together for statistical characterization of the traffic. The distributions and the parameters of the distribution for the flows of TCP, UDP and TCP plus UDP from the statistical characterization results of the real time traffic are used in the analytical model. Next, the mean queuing delay, mean instantaneous values of queuing delay and jitter are plotted to obtain analytical results. NS2 simulations have been carried out for different network configuration scenarios to verify the results predicted by the analytical approaches. In this thesis work, an attempt has been made to propose a new approach refereed to as Link-by-Link Congestion Avoidance (LbLCA) algorithm. LbLCA is a simple feedback-based protocol which tries to prevent congestion before it occurs. The network delay has been an important parameter adversely affecting the Quality-of-Service (QoS) for the Internet communications, particularly for real time applications and solutions have been attempted either by adding newer TCP flavors or at the application layer (layer 5) by proprietary individual applications. Numerous models are available for mixture of TCP and UDP, but none of them specifically addresses to quantify the effect on queuing delay performance due to increased proportion of TCP in the Internet traffic. Therefore, need for a new model does exist.

The objectives of this thesis work are:

1. Investigating the effect of multiplexing of TCP flows at the transport layer on fellow TCP and UDP and thereby analyzing whether TCP flows collaborate or hurt each other.

[1]Cummins College of Engineering for Women, Pune

2. Motivated by the concern that queuing delay has been on the rise and TCP being the dominant protocol, the objective of this thesis work has also been to analyze the variable queuing delay by modeling the arrival and service processes as random processes and by accounting various proportions of TCP and UDP with their respective datagram sizes to evaluate the mean delay.

3. The next objective has been to analyze the average instantaneous delay and jitter for the flow of interest using discrete transient queue analysis. Here the impact of background TCP flows affecting the Quality-of-Service (QoS) of a typical TCP flow and the UDP flow has been investigated.

4. To generate dataset by collecting real-time data of traffic measurements at an academic institute to implement the proposed model.

5. Another objective was to explicitly analyze the impact of multiplexing of TCP flows on UDP throughput. This thesis work proposes the algorithmic model for UDP throughput in the presence of TCP.

6. Depending on the results of the above objectives, suggesting suitable modifications to improve the performance of Internet communications.

7. To propose alternative to existing TCP, namely, the Link-by-link congestion avoidance algorithm.

8. To carry out buffer design of the Link-by-Link congestion avoidance algorithm using linear multiple regression.

1.2 Thesis contribution

1. Mathematical modeling of arrival and service processes at the congested routers in the network layer has been done using the contributions of TCP and UDP. The model also includes the length of the datagram, which also significantly affects the delay performance. Based on the above formulations, the mean queuing delay has been estimated using the parameters of traffic distribution.

2. This analytical approach itself is the unique feature of the thesis work. Further, quantification of mean instantaneous delay has been attempted using a discrete statistical queue model, with respect to the arrival sequence of datagrams for various values of α, the fraction of TCP traffic in the background. Furthermore, jitter quantification has also been carried out using a discrete statistical queue model.

3. Using real-time traffic, measurements were carried out to ascertain the arrival, service and inter-arrival distributions of TCP, UDP and TCP plus UDP. Further, the parameters of the distribution of TCP, UDP and TCP plus UDP were determined. These real values of traffic distributions were substituted in the proposed arrival and service model. Further, analysis was carried out, to obtain the mean queuing delay, average instantaneous delay and jitter. For validation of the above mathematical approach, actual estimation of average values of end-to-end delay based on exhaustive NS2 simulations, under various conditions has been carried out. Simulation results clearly validate trends predicated using analytical approaches.

4. Further, in this thesis work, UDP throughput is algorithmically modeled for the first time in the presence of TCP. The algorithm is implemented based on the data traces using measurement data and is validated using NS2 simulations.

5. In this work, a simple but new approach, namely Link-by-link Congestion Avoidance (LbLCA) algorithm, which works at the network layer, has been proposed. The buffer design is proposed for LbLCA. Further, the performance evaluation has been done using NS2 simulations on the typical network topologies for comparing LbLCA with TCP.

1.3 Thesis organization

This thesis is organized as follows.

Chapter 1 states the problems studied in this thesis, the motivation, and the objectives behind studying these problems. This chapter provides the introduction to the known, practical, but complex issue of delay performance of TCP on the Internet. This chapter also presents the objectives of the thesis and its contributions.

Chapter 2 reviews the current state of the art, literature of the work done in the areas relevant to this thesis and associated topics. The topics are different types of TCP variants, different efforts to improve delay performance typically for individual delay-sensitive applications and the Internet buffer design.

Chapter 3 proposes a new model for arrival and service processes at the router. Secondly, statistical characterization of the network traffic from an academic Institution (Cummins Engineering College for Women, Pune India) is presented. The feature of self-similarity of the measurement data is in conformation to the standard data used in the literature. The proposed model for arrival and service processes at the router and the measurement data is used in the analysis of the QoS parameters raised in the next two chapters.

In **Chapter 4**, we present the effect of multiplexing TCP flows on delay sensitivity in the Internet communications. The three parameters, mean queuing delay, mean instantaneous delay both with finite buffer size and average values of delay jitter for a typical flow, are quantitatively analyzed using the proposed model for arrival process (A) and effective service (S) process. The analytically obtained values are then corroborated using NS2 simulations. Different TCP variants, the effect of packet size and scalability are also accounted for in the NS2 simulations.

In **Chapter 5** an algorithmic model for UDP throughput in the presence of TCP is proposed. The model uses the arrival (A) and service (S) process distributions proposed in chapter 3. The proposed algorithmic model for UDP throughput is implemented with the values obtained from the measurement data and validated using NS2 simulations. This chapter therefore, analyzes the effect of multiplexing TCP flows on UDP throughput.

Chapter 6 proposes an alternative to TCP. A simple but innovative approach, namely Link-by-link Congestion Avoidance (LbLCA) algorithm, which works at the network layer, has been proposed. LbLCA is a proactive congestion avoidance algorithm, that uses explicit feedback to prevent congestion to happen. Based on the design philosophy and equations of LbLCA, the size of various router buffers has been arrived at for different typical network topologies. Buffer sizes depend upon the mean arrival rate at router input and outgoing link capacities and are independent of round trip time (RTT) and the

number of flows passing through the router. The buffer sizes determined using LbLCA design are validated using extensive NS2 simulations. Further, a new buffer size design mechanism using multiple linear regression is proposed in this chapter. The performance evaluation has been done using NS2 simulations on the typical network topologies. The results of the performance comparison between TCP and LbLCA are presented in this chapter.

Chapter 7 concludes this thesis with a summary of the work done. It also proposes some suggestions that may be investigated in future.

Figure1.1 depicts the organizational mind-map of thesis.

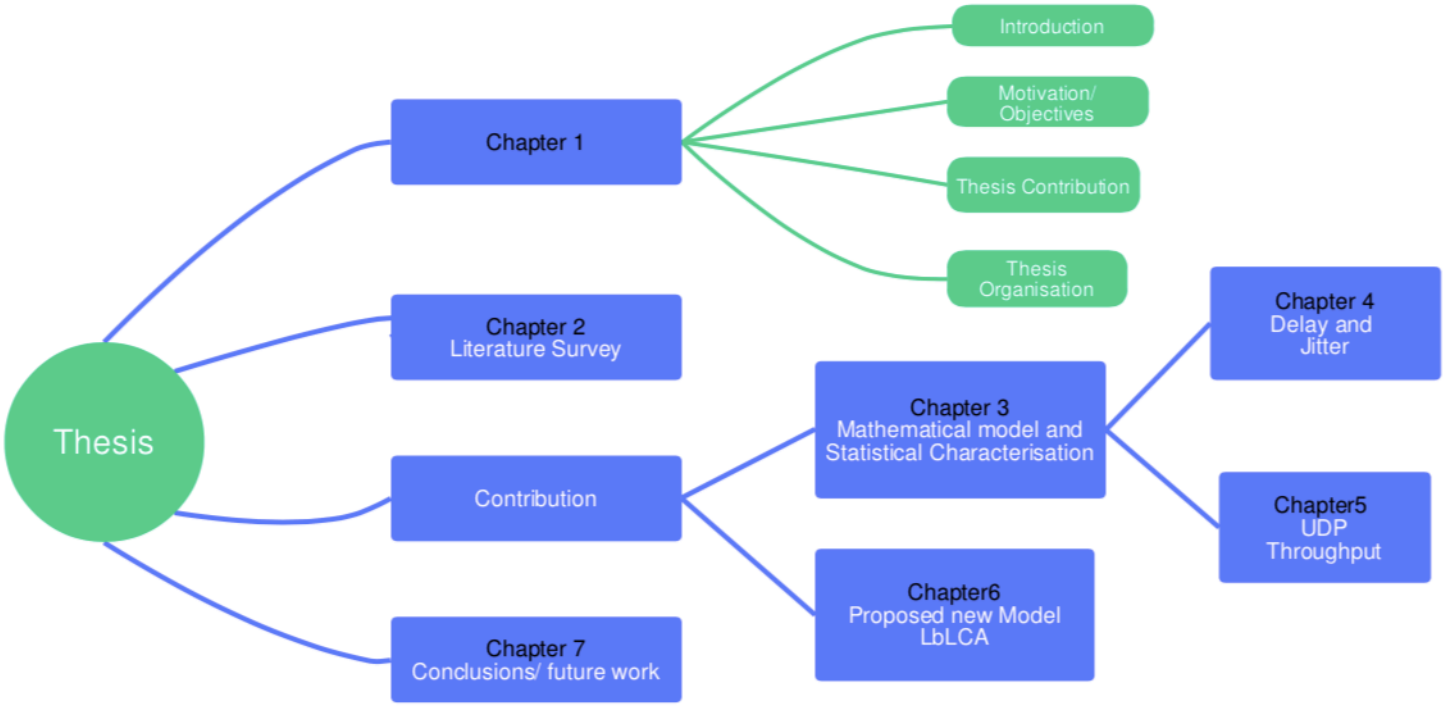


FIGURE 1.1: Mind-map of the thesis

Chapter 2
Literature Survey

LAST 15-20 years have witnessed extensive investigation of TCP protocol by researchers, both in academia and industry. The following review of related work is carried out considering different aspects of TCP protocol, its usage, and its characteristics.

2.1 Self-Similarity and significance of variable segment size in TCP

The TCP traffic flow originates at the application layer based on user sessions which translate into a single flow or multiple flows. These flows have been observed to be bursty in nature and network traffic has been considered aggregations of bursts of single or many flows and is called bursty traffic. The burstiness of arrivals becomes the distinct feature of self- similarity [7, 8]. Empirical measurements were done and observations made, like self-similarity in arrival statistics, datagram sizes and dominance of TCP in overall traffic have been largely consistent with inferences drawn in the research studies. The statistical features of aggregate TCP arrival processes and inter-arrival times have been shown to have a long-range dependency [9–11]. The approaches used for evaluating self-similarity have been based on percentile-percentile plots, curve fitting, curvature tests,

complementary cumulative distribution function (CCDF) tests and tools developed by Crovella [8, 12]. Mirza *et al.* [13] too have modeled TCP traffic using long term statistics, experimental results and newer ways of deep packet inspection coupled with machine learning algorithms, and found it to be self-similar.

2.2 Relation of queuing delay and buffer size

Buffer size and queuing delay are closely related. Raina *et al.* [14] examines the issue of design of router buffer size and prove that small buffers [10-20 packets] actually promote desynchronization of TCP flows and therefore the stability of the network. Wischik *et al.* [15] suggested that very large buffer size [e.g. 1000,000 packets] following thumb rule in the Internet backbone can be reduced to the small buffer size [10,000 packets] without any packet losses. Further, the buffer size can be reduced to a very small size [10-20 packets] with a small cost of bandwidth utilization. Enachescu *et al.* [16] argue that buffers smaller than 10-20 packets are sufficient if sources are not bursty for high throughput. Buffer sizes O (log W) where W is the window size of TCP are sufficient for high throughput. In an interesting work, Rouskas *et al.* [17] have evaluated the relationship between buffer sizes of core routers and throughput of TCP. Rouskas *et al.* [17] discuss the anomalous behavior of TCP real-time and TCP closed-loop traffic. It has been mentioned that for a specific buffer range, real-time traffic losses increase, as the buffer size increases; basically, due to buffer sharing dynamics at routers between real-time and elastic traffic. Further, this anomalous behavior has been linked to various factors like the nature of real-time traffic, a mixture of long-lived and short-lived TCP flows and varying packet sizes. Furthermore, it has been observed that these factors also impact the severity of the anomaly which is related to the interaction between TCP and real-time traffic. Huang *et al.* [18] have observed that buffer occupancy is, in fact, the primary state variable, that an Adaptive Bitrate (ABR) algorithm should control. Further, Huang *et al.* [18] have proposed a design that directly chooses the video rate, according to the current router buffer occupancy, and uses simple capacity estimation only when necessary. It means that capacity estimation is unnecessary in steady-state and capacity estimation is only important during the startup phase when the router

buffer occupancy is growing from empty. Lokshina *et al.* [19] advocated an approach to evaluate the probability of buffer overflow in self-similar queuing networks with finite buffer capacity. Typically for video streaming applications, existing adaptive bit rate (ABR) algorithms face a challenge in estimating the future capacity over time. Lokshina *et al.* [19] have proposed new methods to measure effectively the probabilities of buffer overflow in high-speed communication networks.

Buffer size requirement at the routers has been an implicit function of the TCP congestion window. Imputato *et al.* [20] have suggested that to deal with bufferbloat problem, focus should be on traffic control infrastructure and network device drivers to reduce queuing time. The results of experiments conducted by Imputato *et al.* [20] have been helpful to understand the impact of network device buffer (size of the buffer) on performance. Taht *et al.* [21] have proposed FQ-CoDel, the Fair Queue Controlled Delay Packet Scheduler, which combines fair queuing with CO-AQM, as a tool for mitigating bufferbloat and minimizing delay. FQ-CoDel has been used as a multiplexer of packets from multiple flows to reduce the impact of head-of-line blocking from bursty traffic. Araldo *et al.* [22] have proposed a methodology to gauge bufferbloat (the queuing delay) on the Internet using passive measurements of TCP traffic. The Araldo *et al.* [22] have concluded that there appears to be no correlation between ISP traffic load and queuing delay. However, it has been mentioned that the queuing delay may be related to the traffic of a single user and can be affected by applications generating cross-traffic (background traffic).

2.3 Dominance of TCP and different TCP types

It may be readily observed from the sample anonymized Internet Trace of 2018 Dataset from the center for Applied Internet Data Analysis' (CAIDA) website that TCP seems to be the dominant protocol and more than 90% traffic on the Internet constitutes TCP. The surveys for various TCP congestion control approaches that rely on implicit and explicit signaling from the network highlight some issues [23, 24]. Afanasyev *et al.* [23] have discussed the evolutionary graph of TCP congestion control mechanisms and all major versions of TCP with their merits and demerits. Afanasyev *et al.* [23] have remarked that there appears no consensus in the research community on the superiority

of any specific TCP congestion control mechanism, over the others. Lar *et al.* [25] have classified bibliography on various TCP types and concluded that researchers have shifted their attention from the basic problem of removing the congestion collapse phenomenon to issues of use of available network resources effectively in different types of environments. Datagrams, which get inordinately delayed, would be effectively considered as lost in real-time communications, though buffering can handle the delay variations up to two seconds for storing multimedia communications [26]. Detailed information about TCP heterogeneous congestion avoidance algorithm deployment patterns in the current Internet is provided by Yang *et al.* [27]. It is evident that TCP Cubic (46.92 %) is widely used, followed by TCP Compound (25.62 %) and then TCP Reno (14.46%).

Alrshah *et al.* [28] have compared, high-speed Linux TCP Variants over High-BDP Networks and inferred that all TCP flavors have a fatal problem which is known as the burst loss which happens at the end of the initial stage of Slow Start phase. Some researchers Wang *et al.* [29] have studied TCP for data center applications and proved that DC-Vegas is the most suited TCP type for data center needs. Patel *et al.* [30] have evaluated the end-to-end congestion control algorithms, e.g., TCP Tahoe, TCP Reno, TCP Newreno, TCP Veno, etc. and compared them on various performance parameters like throughput, queuing delay, goodput, etc. Further, a new performance parameter is proposed to measure network performance. Bisoy *et al.* [31] have evaluated the inter-protocol compatibility between TCP Reno and TCP Vegas, both in wired and wireless communications and have inferred that inter-protocol fairness issues are present between TCP flavors of Reno and Vegas. Wang *et al.* [29] have compared datacenter TCP flavors (DCTCP, D^2TCP and D^3) for typical data center issues like soft real-time delay and incast throughput issues. Further, they have proposed a new TCP type for data center applications; namely, DC-Vegas to deal with soft real-time delays and incast throughput issues with minimal modifications. Kushwaha *et al.* [32] have exhaustively studied the two congestion approaches, namely source-based and router-based on high-speed networks. Further, Kushwaha *et al.* [32] have summed up their work with interaction and challenges in both approaches with regard to congestion control. Xu *et al.* [33] proposed a new TCP variant, namely, (CMT-NC) Concurrent Transfer Multipath for improving

the concurrent multipath transfer performance using Network Coding in wireless networks. Further, the Xu *et al.* [33] claimed CMT-NC improved performance in data reordering due to path asymmetry and packet loss due to wireless channel unreliability. Yadav *et al.* [34] proposed a new MultiPath TCP (MPTCP) algorithm that predicts the data segment to be transmitted for a given subflow based on round-trip-time and present congestion state of the subflow. Further, the Yadav *et al.* [34] concluded that their simulations provided improved performance. MP-TCP is being explored in wireless networks, concurrent multipath, for newer test-beds, for improved performance and some new variants [35–40]. Arokkiam *et al.* [41] have experimentally evaluated TCP (Reno, Cubic and B-TCP) performance over 10 Gb/s passive optical networks (XG-PON). They have concluded that Reno, Cubic and B-TCP are not able to achieve high average throughput, close to the effective bandwidth for higher RTT/BDP values and observed fluctuating convergence when new TCP flows were introduced in existing TCP flows.

2.4 TCP and QoS for real-time applications

It has been reported that the modern-day traffic suffers from the issue of TCP delay affecting the stringent QoS requirement particularly, for real-time applications [42]. Ciucu *et al.* [43] have analyzed queuing behavior subject to multiplexing a stochastic process M(n) of flows; then considering the case M (n) as iid, it has been shown that multiplexing of flow results in the increased queue size distribution. Some of the noteworthy solutions proposed to deal with TCP delay QoS issues have been briefly reviewed in the following paragraphs. XU *et al.* [44] has carried out a measurement study for video telephony applications (Google+, ichat, and Skype) to obtain insights into the architectural design of these applications and the design impact on QoS (Quality-of-Service) and (Quality-of-experience). A comparison of parameters like throughput, fairness, and friendliness of Linux TCP variants has been done by Callegar *et al.* [45]. Casas *et al.* [42] have developed an online QoE monitoring system to understand degradation and then provisioning resources on IP by service providers. They have also made an important observation that YouTube and other Content Distribution Networks (CDN) bring the content closer to the end-user so as to minimize propagation delay. Similar strategies have been adopted

by Google and other CDNs. So far to date, HTTP/HTTPS and therefore, TCP do not differentiate the divergent QoS needs of dissimilar applications in Cloud IoT traffic. Therefore, when content is brought near to the end-user of the application in consideration then propagation delay component of end-to-end is addressed and not the queuing delay.

Shailendra *et al.* [46] have compared the mean queuing delay incurred in link-by-link error control with end-to-end mechanisms. Shailendra *et al.* [46] have inferred that link-by-link mechanism results in much lesser mean queuing delay, compared to the end-to-end approach. Wu *et al.* [47] evaluated streaming high-quality mobile video with multipath TCP in heterogeneous wireless networks with regards to path asymmetry in access networks. Further, the Wu *et al.* [47] have proposed a new TCP variant quality-Driven MultIpath TCP (ADMIT). Rajaboina *et al.* [48] have proposed a TCP variant namely, TCP friendly rate control (TFRC) for the transfer of multimedia applications, because of its smooth throughput. Rajaboina *et al.* [48] have compared the performance of TCP, UDP and TFRC in static wireless environment. Cardwell *et al.* [49] have proposed a BBR model congestion control model, which is based on measuring the Bottleneck Bandwidth and Round-trip (BBR) time and has been deployed on Google's B4 backbone. According to the Cardwell *et al.* [49], compared to TCP CUBIC, BBR shows improved throughput by orders of magnitude. BBR also has been implemented by Google and YouTube for substantially reducing latency. Xu *et al.* [50] have presented a review of congestion control solutions for multipath transport protocols and have explored the multipath congestion control design in order to address the need for some useful attributes including load balancing, TCP-friendliness, Pareto optimality, and stability. Ferlin *et al.* [51] have proposed a new TCP variant, namely Shared Bottleneck Detection Multipath TCP (MPTCP-SBD) and have compared it with MPTCP for throughput performance. Further, the Ferlin *et al.* have concluded that MPTCP-SBD works better even when paths do not share bottleneck nodes. Molnar *et al.* [52] have proposed a protocol namely, Digital Fountain based communication protocol (DFCP) which relies on fountain code-based data transfer. Further, Molnar *et al.* [52] have advocated that, instead of controlling the congestion as it is applied on the Internet by the TCP, congestion can been utilized and control algorithms in TCP can be neglected. Therefore,

in DFCP congestion in the network is actually utilized. TCP variants and DFCP have been compared for performance by the Molnar *et al.* [52]. Perkins *et al.* [53] has not proposed any congestion control algorithm. Instead they have defined a minimal set of RTP circuit breaker conditions under which an RTP sender is supposed to impede the transmitting media data to safeguard the network from excessive congestion. Deng *et al.* [54] has proposed a new TCP variant for data centers where the deadline of application flows is important. The proposed TCP variant has an improved slow start phase of TCP. The Deng *et al.* [54] have claimed that the delay sensitivity has been improved for data center applications. Bisoy *et al.* [55] have proposed a Stable Active Queue Management controller named SAQM. The SAQM has been designed to control the congestion in the network and improve the stability of the queue at the Internet router under a dynamic environment (number of TCP connections, bottleneck bandwidth, round trip time (RTT), and target queue length changing). Owens *et al.* [56] provide research contribution in providing end-to-end QoS for video applications. The Owens *et al.* [56] have proposed Video over Software Defined Networks (VSDN), which is an architecture and protocol capable of making optimal path selection utilizing a global network view. Further Owens *et al.* [56] have developed the prototype for quantitatively evaluating the behavior of VSDN. Nan *et al.* [57] have investigated multimedia cloud QoS, particularly the mechanism for provisioning QoS in multimedia live streaming by capturing the queue model. The Nan *et al.* [57] then provide a relationship between service response time and resource allocation in a cloud-based environment. Zhou *et.al* [58] have suggested a design for a new media-aware security framework for typical multimedia traffic in IoT, where multimedia IoT traffic has been identified and divided into three categories: communication, computation, and services. The Zhou *et.al* [58] then provide a context-aware multimedia framework and have suggested the possibility of using Quality-of-Experience (QoE) as an important parameter for developing cloud services with regard to mobile access technologies.

Internet Engineering Task Force (IETF) has defined two models namely Integrated services (Inetserv) and Differentiated Services (Diffserv) for facilitating end-to-end QoS to deal with real-time Internet communications. Inetserv provides services on a per-flow basis. IntServ routers must maintain per-flow state information and have features of

reserved resources and call set-up. All devices en-route destination need to be aware of the reservation. Routers need to maintain state for each reservation along the path, which can be a complex issue and memory requirements would increase too. Resource Reservation Protocol (RSVP) tries to signal QoS requirements but can be fulfilled only if resources are available. The resources are soft meaning they need to be refreshed periodically. New traffic addition occurs continuously and therefore due to the dynamic conditions of traffic, the probability of reserved packets being timed out and lost increases. Some proprietary enhancements like 'RSVP Refresh Reduction and Reliable Messaging' and 'RSVP scalability Enhancements' are added. Still, Insetserv focuses on per-flow basis end-to-end QoS. So Insetserv suffers from scalability. The Diffserv approach also aims to provide QoS in-network by providing scalable service differentiated on the Internet that is used to permit differentiated pricing of internet services. In DiffServ, flows are aggregated into classes that receive "treatment" by class. More complex operations are pushed out to edge routers and simpler operations done by core routers. The system is based on differentiated pricing for different classes of Internet services. It is a packet forwarding model separate from the routing model. Diffserv enables scalable and coarse-grained QoS throughout the network. Unlike RSVP, DiffServ needs to be provisioned for resources in routers. This is done by having different classes of networks for different applications. So application discovery and profiling becomes essential which can be time-consuming. Further, security concerns can also become an important issue. Management in terms of billing and monitoring becomes difficult.

The biggest disadvantage of Inetserv and Differsv is that the signaling/ provisioning happens independent of routing and congestion control algorithms. True end-to-end QoS with maximum network utilization is best possible with the combination of traditional QoS and routing. Liers *et al.* [59], have explicitly mentioned the drawbacks of Inetserv and Diffserv and have provided a new architecture which handles inetsev and Diffserv in an integrated way. Guck *et al.* [60] have proposed an alternative model to achieve end-to-end QoS using Software Defined Networking. This has been done by integrating routing and access control to provide predefined end-to-end QoS to flow. Flavius *et al.* [61] have illustrated some important distinctiveness of the existing service delivery models like Multi Protocol Label Switching (MPLS), Inertsev, Diffserv and best effort

mechanisms. Flavius *et al.* [61] have highlighted open issues for guaranteeing Internet QoS. The Flavius *et al.* [61] have emphasized the need for research on the measurability of the Resource Reservation Protocol (RSVP). Malik *et al.* [62] have explained a new approach that can simplify network configuration and operation by defining a limited common set of Diffserv Per-Hop Behaviors (PHBs) and Diffserv Code Points (DSCPs) to be applied to connections of two separate autonomous networks.

It is observed that by using Inetserv and Diffserv, the basic principle of best-effort delivery on the Internet i.e. not to differentiate or prioritize the applications/flows, may not get followed. Further Diffserv and Inetserv are essentially QoS facilitating mechanisms and not routing or flow control and congestion control mechanisms. Therefore, some resource reservation and resource management arrangements become essential for implementing these models to really achieve any QoS improvements. However, the actual impact on QoS parameters needs to be evaluated. Furthermore, this may add some more issues like impact on security, network scalability, etc. which need to be studied separately. To improve QoS, modern-day applications prefer TCP multiple sessions, TCP multi-path sessions, dynamic use of ephemeral port numbers greater than 1024 not registered with the Internet Assigned Numbers Authority (IANA) and multiple content servers at different locations with HTTP redirect to minimize delay. None of them has explored the linkage of QoS to TCP congestion control and the effect of fellow TCP flows on a typical TCP flow or on UDP flow. The need to address the delay concerns has been recognized since long. However, many of the researchers have come up with newer types of TCP to deal with the problem, which may not be helpful in controlling the delay [23, 63].

Some modern and different ways to deal with the delay friendliness of TCP to provide better QoS for real-time applications reported in the literature have been reviewed below.

Transparent computing has become a new paradigm, where stream execution can provide improved Quality-of-experience (QoE). Wang *et al.* [64] have analyzed utility function for clients' requirement of bandwidth under fixed QoS. A resource allocation scheme on double-sided combinational auctions was then proposed by the Wang *et al.* [64]. Li *et al.* [65] have considered the resource allocation problem in real-time (non-elastic) traffic for multipath networks. Further, the resource allocation problem has been converted

to the strictly convex problem and optimal solution satisfying the Karush-Kuhn-Tucker conditions have been analyzed. Li *et al.* [65] have inferred that the Dynamic Queue limit (DQL) algorithm has been essential for Active Queue Management (AQM) algorithms to be effective to minimize latency. Raina *et al.* [66] have studied Compound TCP (default in Windows) to derive conditions for local stability. The Raina *et al.* [66] have also shown that variation in system parameters can lead to Hopf bifurcation and the role of buffer size has become crucial to ensure stability. Abbas *et al.* [67] have presented a new taxonomy of classifying fairness-driven queue management schemes. The Abbas *et al.* [67] have also discussed design approaches and key features of these schemes and have provided their comparison and analysis. Yang *et al.* [68] have designed a routing mechanism for green Internet. The Yang *et al.* [68] have also explored the relationship between QoS requirements and reduced energy consumption. Further, Yang *et al.* [68] have suggested that using hop-by-hop in line cards of routers would save 50% on power. A different perspective has been provided by Ballan *et al.* [69]. The Ballan *et al.* [69] have analyzed the feasibility at IP-level and hosts exchange reachability constraints on different destinations to defend compromise and resource exhaustion attacks. Gholipur *et al.* [70] have proposed hop-by-hop congestion avoidance in wireless sensor networks using support vector machines (SVM) to improve throughput significantly. The Gholipur *et al.* [70] have also calculated buffer occupancy and have estimated the congestion of the downstream node. This information then has been conveyed to the upstream node to adjust its sending rate to mitigate congestion. Similar work without the use of SVM is seen in [70]

2.5 Buffer size design

Ganjali *et al.* [71] have briefly compared different results of the proposed rules for buffer size design (rule of thumb, small buffer rule, drop buffer rule and tiny buffer rule) in the Internet routers and have not provided any preference for any rule. In fact, Ganjali *et al.* [71] have indicated that more analysis is required for the changed Internet circumstances. Vishwanath *et al.* [72] have presented a comprehensive survey of results in the area of router buffer sizing and have classified the buffer sizing rules and explained the merits

and demerits of each rule. Vishwanath *et al.* [73] have examined the dynamics of TCP and UDP interaction at a bottleneck-link router equipped with very small buffers. Vishwanath *et al.* [73] have observed a strange phenomenon. Losses for real sensitive traffic do not fall monotonically with the size of the router buffer. Alternatively, there exists an inflection point beyond which loss increases with increasing buffer size. This indicates the existence of anomalous loss behavior. Allman *et al.* [74] have addressed the issue of bufferbloat typical of deep buffers. Allman *et al.* [74] have concluded that bufferbloat is more pronounced in residential areas but how often long queues build up in the buffers needs to be analyzed. Gharakheili *et al.* [75] have compared host pacing and edge pacing in the context of small buffer networks. Comparison is carried out exhaustively for different scenarios like bottleneck and non-bottleneck core nodes, short-lived and long-lived flows, different variants of TCP, high-speed and low-speed link, etc. Gharakheili *et al.* [75] have concluded that edge pacing is definitely better than host pacing in terms of link utilization (TCP throughputs) and average per-flow goodputs. In the LbLCA, similarity with an edge pacing is observed where traffic is seen paced by the edge node that connects into the core network. Ghosh *et al.* [76] have carried out performance comparison of compound TCP for single-bottleneck node and multi-bottleneck node under different scenarios. The motivation of the work was, a rise in increased queuing delays in the Internet. Detailed analysis carried out in Ghosh *et al.* [76] concludes that small buffers favor stability whereas large buffers in addition to increasing latency, are prone to inducing limit cycles in the system dynamics, via a Hopf bifurcation.

2.6 UDP as transport?

It is clear that transport layer in Internet is ossified and further development looks arduous. The primary reason attributed is the wide use of middleboxes like firewalls and NATs. These middleboxes have made tough the deployment of newer transport mechanisms. Augmenting to middleboxes is limited openness and pliability of application programming interfaces (APIs) typically at application layer. The improvements come in working around transport and application layers. Therefore the solutions emanated

are for specific applications and point needs rather than catering to many facets of the problem [77, 78]. The exhaustive survey of ossification issue and the solutions proposed is carried out by Pasterigous (et al.) [78] and they conclude that none of the proposed solutions has aided evolution of transport layer. Some newer protocols which use UDP as base are Google's Quick UDP Internet Connections (QUIC) protocol, SPDY (pronounced "speedy"), HTTP/2, Adobe's Real Time Media Flow Protocol (RTMFP), Multipath Real-Time Transport Protocol (MPRTP), DTLS , uTorrent Transport Protocol (uTP), BitTorrent, UDP-based Data Transfer (UDT) protocol, Structured Stream Transport (SST) protocol etc [78]. Some newer architectures like NEAT are also proposed in [79].

2.7 Conclusions

The above review suggests that the network delay is an important parameter adversely affecting the QoS for Internet communications, particularly for real-time applications and solutions that have been attempted either by adding newer TCP flavors at the application layer (layer 5) or by proprietary individual applications. Numerous models are available for a mixture of TCP and UDP, but none of them specifically addresses quantitatively the effect on queuing delay performance due to an increased proportion of TCP in the Internet traffic. Therefore, the need for a new model does exist. Motivated by the concern that queuing delay is on the rise and TCP is the dominant protocol, this thesis work attempts to analyze the variable queuing delay. The mean delay analysis is carried out by modeling the arrival and service processes as random processes and by accounting various proportions of TCP and UDP with their respective datagram sizes. Next, the average instantaneous delay and jitter is derived from the flow of interest using discrete transient queue analysis. Here the impact of background TCP flows affecting the QoS of a typical TCP flow and the UDP flow has been investigated. To ascertain the values of the Internet traffic, the dataset used in this thesis work has been generated by collecting real-time data of traffic measurements done at well-known academic institute in Pune, India.[1] The curve fitting procedure has been used for data collected in the arrival process, varying packet sizes and inter-arrival times of TCP, UDP, and TCP

[1]Cummins College of Engineering for Women

plus UDP together. The analytical results from the queuing delay making use of measured values have been plotted. The analysis of instantaneous values of queuing delay using a discrete statistical queue model has also been carried out. NS2 simulations have also been implemented for estimating average end-to-end delays for different scenarios to verify the results predicted by analytical approaches. The conclusion of the investigation of effect of TCP multiplexing on the Internet flows is that TCP flows hurt each other and are not friendly. The solution is that, a novel approach has been proposed for congestion avoidance in this thesis work at the network layer for core and edge routers, using explicit feedback (about the state of the router) without any windowing mechanism. This approach is link-by-link based, i.e., every router participates in congestion avoidance. Every router acts as a source router for the downstream router and acts as the destination router for the upstream one. Flow control is delinked from congestion control. Further flow control can be applied need-based (full, partial, none) and may be done at the transport or even at the application layer. The results are improved delay performance and better packet delivery ratio in LbLCA compared to existing AIMD based TCP protocol. This delinking of congestion and flow control does not violate the end-to-end argument design principle of the Internet. A typical router buffer size design methodology has also been proposed. Using these buffer sizes, exhaustive simulations have been done for commonly used network topologies and performance comparisons for different protocols have also been attempted in the thesis work. Next chapters discuss the contribution of this thesis work.

Chapter 3

Traffic Characterization and Proposed Mathematical Model

This chapter presents two major aspects of the contribution in the thesis. They are as follows,

1. Statistical characterization of real-time traffic. The distribution parameter values by statistical characterization obtained are then used in analytical derivations of mean queuing delay, average instantaneous delay, and jitter in the subsequent chapters.

2. Mathematical model of arrival and service process distributions of datagrams at router. The model is used to derive mean queuing delay, average instantaneous delay, and jitter in the subsequent chapters.

3.1 Statistical characterization of the traffic

The reasons for using real data for characterization of the Internet traffic are as follows:

1. The traffic characterization is based on measurement data of the Internet traffic so that no assumptions are made on traffic characterization. The characterization aims to describe the real Internet.

2. The parameters of IP datagrams and flows are statistically independent.

3. The methodology for traffic characterization was measure, model and understand.

4. Characterization was done at the bottleneck router for the academic institute as most of the packet loss happens at the bottleneck router compared to the core part of the Internet.

5. The dataset generated is now available with correct figures in the campus. Though the measurements come from the academic institute, the dataset acts as a sample of the Internet traffic.

6. The distribution of the real time traffic at the academic institute modeled with the parameters closely agrees with the results reported by the research community.

7. The proposed protocol in the thesis needs to be tested on the Internet test bed and not on simulations alone.

Generation of the experimental dataset was carried at the academic institute (Cummins Engineering College for Women, Pune India) for various attributes at different timings of one hour each and having definite mean values. Traffic measurements on the experimental data at the edge router were carried out over one week (168 hours) duration and hourly arrival rates, datagram sizes and inter-arrival timings for each 10μ interval were measured using Wireshark. Based on these measurement parameters like arrival rates, datagram sizes and inter-arrival times were worked out. After the curative process, curve fitting was done to obtain the distributions and their parameters. The flows from the obtained dataset were sorted and distributions for TCP only, UDP only and TCP plus UDP together, for tagged traffic were determined.

It was observed that the traffic sorted by three cases as TCP only, UDP only and TCP plus UDP together, follow self similarity in each individual case, resembling two parameter Weibull distribution. It was noted that arrival rates and datagram sizes were

independent of each other in all cases. Further, TCP contributes maximum to overall traffic (around 92%). The Weibull distributed process is heavy-tailed and can model the fixed rate in ON/OFF period lengths, when producing self-similar traffic by multiplexing ON/OFF sources [11]. Weibull distribution has been widely used for modeling Internet traffic by the research community. [80–85, 85–90] The density function of a two parameter Weibull distribution is given as in Eq. (3.1)

$$f(t) = \frac{\beta}{\eta}(\frac{t}{\eta})^{\beta-1} e^{-(\frac{t}{\eta})^{\beta}} \tag{3.1}$$

where parameters $\beta > 0$ and $\eta > 0$ represent the shape and scale parameters respectively. Table. 3.1 shows the values of parameters of the Weibull distribution from experimental dataset where $\alpha = 0.92$. However, for different values of α the shape and size parameters of Weibull distribution were also calculated and used in subsequent analysis.

TABLE 3.1: Experimental data: Parameters for Weibull distribution [$(\alpha) \approx 0.92$]

	TCP only	**UDP only**	**TCP+UDP**
Arrival Rate			
a) Shape Parameter (β)	1.34266	1.29519	1.71667
b) Scale parameter (η)	10.8176	2.72557	15.7024
Datagram Size			
a) Shape parameter (β)	0.811221	1.10141	0.806258
b) Scale parameter (η)	712.344	826.505	641.644
Inter arrival Time			
a) Shape parameter (β)	0.52512	0.61658	0.519675
b) Scale parameter (η)	1.50925	0.00941	1.71158

3.2 Mathematical Modeling for arrival/service processes

Justifications for modeling traffic on the Internet are as follows

1. Numerous models are available for mixture of TCP and UDP, but none of them specifically addresses to quantify the effect on delay performance due to increased fraction of TCP in the overall background Internet traffic.

2. Motivated by the concern that queuing delay is on rise in Internet and the fraction of TCP in overall Internet has increased well beyond 90%, the focus was to analyze queuing delay which is the only random delay component in end-to-end delay.

3. The parameters of IP datagrams and flows are statistically independent

4. The objective of modeling the traffic was to account for different fractions of datagrams from both TCP flows and UDP. Datagrams from TCP flows and UDP contend at the router for resources and are bound to impact and as well get impacted by fellow datagrams. To investigate the effect of multiplexed datagrams of TCP flows and UDP was one of the reasons to model traffic at the network layer.

5. In the thesis, the proposed traffic model for arrival and service processes not only accounts for different fractions of TCP and UDP, but also their respective datagram sizes and arrival rates to understand buffer dynamics at network layer. The variable size of datagrams is one of the significant parameters which affect queuing delay.

6. The network layer does not distinguish between different flows. Buffering at the router changes the timing information between the rates decided by source flows at transport layer and outgoing rates actually followed at the routers. The sending rates of the source flows are decided by transport layer which understands congestion and flow control. When datagrams of these flows first hit the edge router and are buffered, the outgoing rates are changed. Then onwards incoming and outgoing rates at the subsequent routers are decided by router buffer dynamics and bear hardly any correlation to sending source rates. To understand the effect of bursty arrivals and the role of buffering in communication networks, datagram arrivals at router links are modeled as random processes. The arrival and service rates are modeled at the typical congested router considering buffering, and mean queuing delay is analyzed.

Consider a router at network layer where a datagram, originating from a typical source, termed as tagged traffic, gets multiplexed with various background streams consisting of both TCP segments and UDP datagrams. In this situation, we model the arrival and service processes, accounting for individual fractions of TCP and UDP protocols. To appreciate the effect of packet size on the waiting time of a queue, we consider the following M/M/1 example for packet size n, with arrival rate (λ) and mean service time for packet size n ($\frac{1}{\mu_n}$). Then mean delay (T) and utilization (ρ) are given by Eq. (3.2) and Eq. (3.3) respectively [91].

$$T = \frac{\frac{1}{\mu_n}}{(1-\rho)} \tag{3.2}$$

and

$$\rho = \frac{\lambda}{\mu_n} \tag{3.3}$$

Now considering the same model, except the packet size is reduced to half i.e; ($\frac{n}{2}$), packet arrivals are still Poisson, which means arrival rate doubles ($\lambda \rightarrow 2\lambda$). Assuming the service time of a packet is proportional to its size, then the service time is halved ($\frac{1}{2\mu_n}$) or service rate doubles. The result of this change is that mean service time decreases proportionally to the factor by which the packet size is increased since utilization remains the same. The same analogy for modeling service rate is also applied. The sizes of TCP segments and UDP datagrams are accounted for, along with the fraction of TCP in the model. Though the service time for datagrams in the router is constant, the need for service time being proportional to the size of datagrams provides better accuracy and accounts for the dynamic behavior of buffer occupancy.

The arrival process (A) and effective service process (S) can be modeled as:

$$A \stackrel{d}{=} \alpha CP + (1-\alpha)BD \tag{3.4}$$

$$S \stackrel{d}{=} \theta\,(\alpha C + (1-\alpha)B) \tag{3.5}$$

where,

α = fraction of TCP in background traffic and $(1-\alpha)$ = fraction of UDP,

C= a random variable indicating TCP burst (segment) size in bytes,

P= a random variable indicating arrival rate of TCP segments in datagrams/s,

B= a random variable indicating size of UDP datagrams in bytes,

D= a random variable indicating UDP arrival rate in datagrams/s,

θ = fixed service rate at router in datagrams/s both for TCP and UDP, and

$\stackrel{\mathrm{d}}{=}\rightarrow$ represents equality in distribution.

It is assumed that P and C are independent and similarly B and D are independent random variables. The service decision of routing is done at the router by opening the header of datagram and it is independent of the payload size of the datagram in question. The TCP has variable segment sizes; therefore, these datagrams when served at the router queue create space in the buffer according to their respective sizes. This space, therefore, decides whether the newer arriving datagram can be accommodated or discarded. Thus, the above model takes into account the effect of the size of datagrams and continuously changing buffer occupancy, which decides the waiting time of datagram in the queue. While servicing the datagrams, the router will find different sizes of TCP segments and UDP datagrams. For example, a UDP datagram of 210 bytes and a TCP segment of 1500 bytes are served at the same rate. Therefore, if a UDP packet is served at (θ) packets/sec, then the TCP is effectively served at approximately ($\theta/7$ i.e. 1500/7) datagrams/sec.

3.3 Conclusions

It was observed that the traffic sorted by three cases as TCP only, UDP only and TCP plus UDP together, follow self-similarity in each individual case, resembling two-parameter Weibull distribution. It was noted that arrival rates and datagram sizes were independent of each other in all cases. Further, TCP contributes maximum to overall traffic (around 92%). The Weibull distributed process is heavy-tailed and can model the fixed rate in ON/OFF period lengths when producing self-similar traffic by multiplexing ON/OFF sources. The density function of a two-parameter Weibull distribution is given

as in Eq. (3.1) [11]. The statistical characterization of the real time traffic has close resemblance with values available in public domain (CAIDA: Center for Applied Internet Data Analysis) and literature [92–97]. Also the arrival and service process models are unique and account for the fraction of TCP and UDP datagrams along with their respective sizes. The buffer dynamics at the router is also accounted. The statistical characterization of the traffic and arrival and service processes at the router from this chapter are used in the estimation of mean delay, average instantaneous delay and jitter in Chapter 4 and Chapter 5. Next chapter considers the estimation of mean delay, average instantaneous delay and jitter using the proposed arrival process (A) and effective service process (S) model.

Chapter 4

Estimation and Simulations of Mean Delay, Average Instantaneous Delay, and Jitter

To calculate the mean delay, the queue is defined using the arrival and effective service process distributions given in Eq. (3.4) and Eq. (3.5) respectively. The arrival and service of datagrams happen at the network layer, where TCP segments and UDP datagrams contend at the router. Random variables P, C, B and D follow a Weibull distribution with parameters, as indicated in the Table. 3.1 obtained by experimental measurements. In the analytical derivation, the term mean queuing delay indicates the average waiting time of the datagram in the router buffer before being serviced. Whereas the term mean instantaneous delay is the queuing length distribution of two consecutive datagrams emerging from a typical flow (TCP or UDP) at the router output buffer queue. The analytical results represent the mean queuing delay (waiting time) at the single router (propagation, switching and transmission delays are excluded which are deterministic in nature).

4.1 Mean queuing delay

The probability density function of Weibull distribution of independent random variables P and C of TCP only are given by:

$$f_P(t) = \frac{\beta 1}{\eta 1}(\frac{t}{\eta 1})^{\beta 1-1} e^{-(\frac{t}{\eta 1})^{\beta 1}} \tag{4.1}$$

$$f_C(t) = \frac{\beta 2}{\eta 2}(\frac{t}{\eta 2})^{\beta 2-1} e^{-(\frac{t}{\eta 2})^{\beta 2}} \tag{4.2}$$

where parameters $\beta 1 > 0$, $\eta 1 > 0$ and $\beta 2 > 0$, $\eta 2 > 0$ are the shape and scale parameters of random variables P and C corresponding to arrival rate and datagram sizes for TCP part (second column of Table. 3.1).

We calculate the product distribution of α.P.C by Jacobian transformation. Consider, X =αP, Y = C

Let Z = X.Y , W = Y

Therefore, X = Z/W, Y = W and x > 0, y > 0 is equivalent to $z > 0, w > 0$.

Using standard Jacobian (J) transformation ,

$$f_{zw}(ZW) = \frac{f_{XY}(x,y)}{|\partial(z,w)/\partial(x,y)|_{x=z/w,y=w}} \tag{4.3}$$

where the Jacobian of the above equations is defined by standard formulations:

$$J = \frac{\partial(Z,W)}{\partial(X,Y)} = \begin{vmatrix} \frac{\partial z}{\partial x} & \frac{\partial z}{\partial y} \\ \frac{\partial w}{\partial x} & \frac{\partial w}{\partial y} \end{vmatrix} = \begin{vmatrix} y & x \\ 0 & 1 \end{vmatrix} = y \tag{4.4}$$

Thus,

$$f_{Z,W}(z,w) = \frac{f_X(x) f_Y(y)}{w} \tag{4.5}$$

as X and Y are independent. Similar exercise has been carried out for UDP fraction using Jacobin transformation for independent random variables B and D. $f_{(1-\alpha)D}(t)$ and $f_B(t)$

are distributions of B and D respectively. As can be seen from Eq. (3.4), the distribution of arrival process (A) is viewed as distribution of summation of random variables (αPC) and ((1−α)BD). Therefore the distribution of arrival process (A) using Laplace property of pdf, can be written as follows:

$$f_A(s) = f_{\alpha PC}(s)\, f_{(1-\alpha)DB}(s) \tag{4.6}$$

Using effective service process distribution given in Eq. (3.5), the distributions for effective service rate (S) can be determined considering random variables (C) and (B) representing variable datagram sizes for TCP and UDP respectively. The density functions of the Weibull distribution of the independent random variables C and B are given as follows:

$$f_{(\theta\alpha C)}(t) = \frac{\beta 2}{(\theta\alpha\eta 2)}\left(\frac{t}{(\theta\alpha\eta 2)}\right)^{\beta 2-1} e^{-\left(\frac{t}{(\theta\alpha\eta 2)}\right)^{\beta 2}} \tag{4.7}$$

$$f_{(\theta(1-\alpha)B)}(t) = \frac{\beta 4}{(\theta(1-\alpha)\eta 4)}\left(\frac{t}{(\theta(1-\alpha)\eta 4)}\right)^{\beta 4-1} e^{-\left(\frac{t}{(\theta(1-\alpha)\eta 4)}\right)^{\beta 4}} \tag{4.8}$$

where parameters $\beta 2 > 0$, $\eta 2 > 0$ and $\beta 4 > 0$, $\eta 4 > 0$ represent the shape and scale parameters of random variables C and B corresponding TCP and UDP respectively.

Using Eq. (4.7), Eq. (4.8) and using Laplace transform property of distributions, the pdf of effective service rate (S) can now be written as:

$$f_S(s) = f_{\theta\alpha C}(s)\, f_{\theta(1-\alpha)B}(s) \tag{4.9}$$

The notation G/G/1 queue is usually referred to as a single-server queue with first-in-first-out discipline and with a general distribution of the sequences of inter-arrival and service times. It can be noted that the router can be modeled as a G/G/1 queue as an experimental dataset (refer Table 3.1) clearly presents inter-arrival and service distributions as Weibull. The M/M/1 queue discussed earlier is a special case of the

G/G/1 queue. For a G/G/1 queue, when traffic offered to the router is high, the upper bound on the distribution of waiting time $'T'$ can be approximated to an exponentially distributed random variable [91] with the mean value given by

$$T \leq \frac{\lambda \left(\sigma_A^2 + \sigma_S^2\right)}{2(1-\rho)} \tag{4.10}$$

where (σ_A) represents the standard deviation for distribution of arrival process (A) and (σ_S) represents the standard deviation of distribution of effective service rate (S).

The values for Figure 4.1 are acquired as below:
Weibull distribution parameters for arrival rate and datagram size are substituted from Table 3.1. Background traffic Weibull parameters are obtained from mixed traffic (TCP + UDP) and for TCP flow the Weibull parameters are obtained from the same table (with TCP only column). The arrival process distribution and effective service rate distribution are obtained by varying values of (α) from 0.1 to 0.9 with increasing steps of 0.1. Numerically solving Eq. (4.6) and Eq. (4.9) for (σ_A) and (σ_S), the maximum mean delay due to the single congested router has been computed using Eq. (4.10) for various values of α. For comparison, the analysis has been done by considering the size of datagrams as fixed and also by varying the datagram sizes of different fractions of TCP and UDP in the background traffic. The results are plotted using Matlab 2016b.

Figure 4.1 shows the relationship of mean waiting time $'T'$ with a varying fraction of TCP traffic (i.e. α) in the background for both the cases. The first case where the mean delay is calculated considering datagram size as a random variable and the other case considering datagram size as constant (TCP packet size of 1500 bytes, which is Maximum Transfer Unit (MTU) of Ethernet and UDP packet size 210 bytes). The mean queuing delay values in Figure 4.1 are for a single congested router. It can be readily seen that the mean waiting time of the datagrams at the router increases with the increased TCP fraction in the background traffic. The size of datagrams is also a critical parameter for estimating mean waiting time, as it increases considerably with variable datagram sizes. The significant increase in the mean waiting time for variable size datagram traffic, compared to the fixed size datagram, for values of α between 0.6 and 0.7 may be

attributed to parameter values determined from the measurement dataset. The mean queuing delay values due to single router increase to 100 msec, when background traffic has datagrams from TCP contributing more than 90 % (UDP contribution less than 10 %), which is typically the case in the present Internet communications. The next section focuses on statistics of average instantaneous delay and jitter of tagged datagrams, due to multiplexing and buffering of datagrams for tagged traffic against background traffic with finite buffer size.

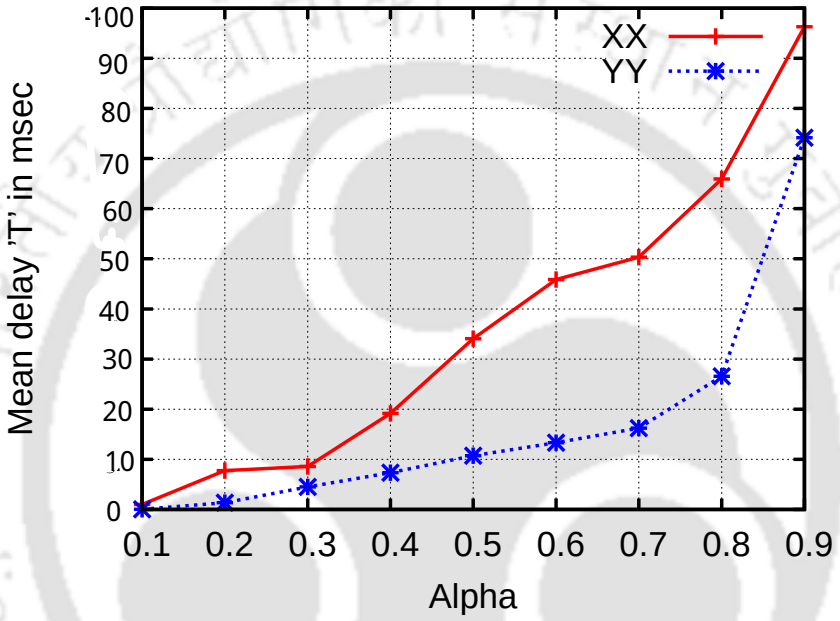


FIGURE 4.1: Mean waiting time for various fractions of TCP where XX: Mean delay considering datagram size as random variable and YY: Mean delay considering datagram size as constant

4.2 Analysis of average values of instantaneous delay for tagged traffic

Each datagram at the router input can be treated independently and hence the router queue can also be modeled as a discrete-time queue, where the time axis can be divided into discrete slots such that only one datagram arrives in a slot and arrival/departure of datagrams happens at the boundary of the slot. The channel time can be slotted with the size equal to the transmission time (t) required by one IP datagram and the processing of discrete-time queue has been assumed to be on FCFS basis. When datagrams are

buffered, information between inter datagram timing is lost. However, buffering is necessary for the contention of resources. The router's output port queue has been modeled as a finite single server queue (independent of the other output ports) and the objective is to estimate the mean instantaneous delay of either TCP or UDP tagged datagrams. The tagged datagrams are essentially from a typical flow of interest and the rest of the datagrams constitute the background traffic, consisting of both TCP and UDP. Further, in the analysis instead of providing the best estimate of results, statistical averaged results are provided for simplicity.

Let the tagged traffic datagrams arrive as n^{th} and $(n+1)^{st}$ arrivals and let there be k slots in between them occupied by background traffic, as shown in Figure 4.2.

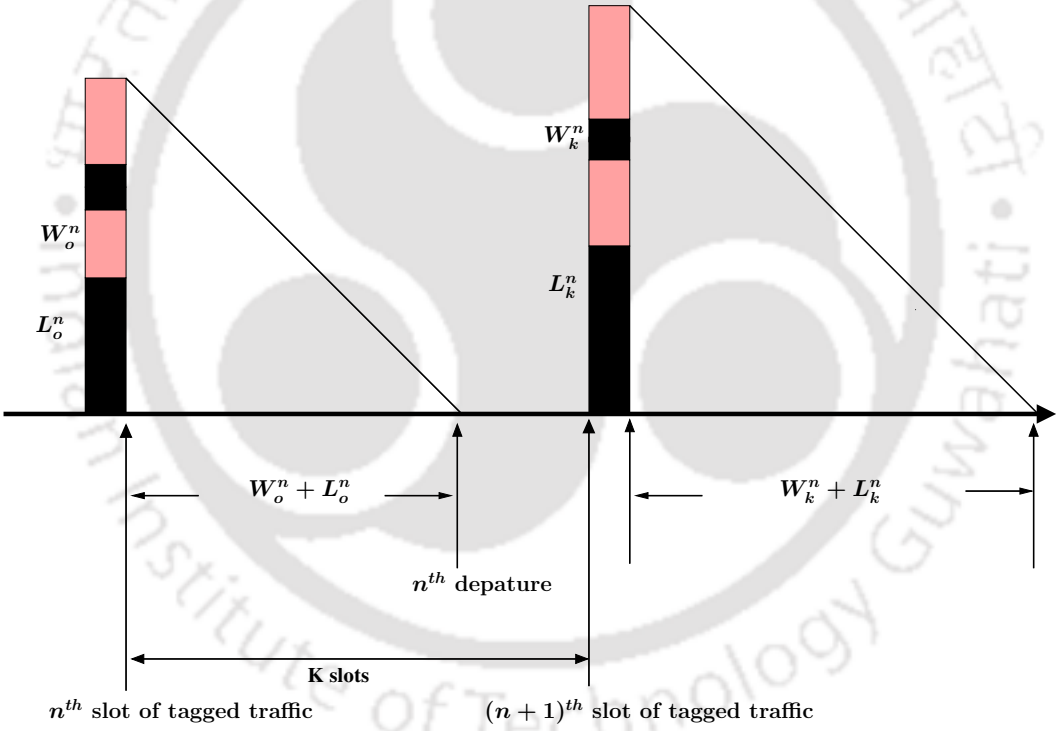


FIGURE 4.2: Overall traffic flow diagram

The distributions of tagged traffic (TCP or UDP) and also, the distributions of inter-arrival times of tagged traffic are indicated in Table 3.1. Following a similar analysis as done for arrival/service process, the probability density function for inter-arrival time slots = k can be considered as discrete Weibull pdf given by,

$$f(k) = \frac{\beta}{\eta}(\frac{k}{\eta})^{\beta-1}e^{-(\frac{k}{\eta})^{\beta}} \tag{4.11}$$

where parameters $\beta > 0$ and $\eta > 0$ represent the shape and scale parameters respectively, where f(k) = Pr (inter-arrival time =k slots) for tagged traffic. The resultant arrival distribution of the background traffic is given by,

$$f_A(t) = f_{\alpha PC}(t) + f_{(1-\alpha)DB}(t) \tag{4.12}$$

where $f_{\alpha PC}(t)$ and $f_{(1-\alpha)DB}(t)$ represent TCP and UDP arrival distributions and are determined as in Eq. (3.4).

As seen from Figure 4.2, L_k^n is a random variable representing the queuing length at the end of the k^{th} time slot after the receipt of n^{th} datagram from the tagged traffic. For k = 0, let L_0^n represent the queue length immediately after the n^{th} datagram arriving from tagged traffic. At the beginning of the first time slot after the receipt of the n^{th} datagram from the tagged traffic, there are L_1^n datagrams in the output queue, where L_1^n includes,

1) number of datagrams already in the buffer immediately before the n^{th} datagram arriving from the tagged traffic, L_0^n
2) one datagram newly arriving from tagged traffic and
3) W_0 datagrams newly arriving from background traffic.

Thus, we can obtain the following equations:

$$L_1^n = L_0^n + 1 + W_0^n \tag{4.13}$$

$$L_{k+1}^n = L_k^n + W_k^n \qquad k = 1, 2, 3, ..., I \tag{4.14}$$

where I represents the maximum number of slots in the output queue. Let $q_k^n(j)$ be the probability density function of L_k^n where j is the discrete random variable, representing the number of datagrams in the output queue at (n,k) slot. Then

$$q_1^n(j) = q_0^n(j-1) + w_0(j-1) \tag{4.15}$$

$$q_{(k+1)}^n(j) = q_k^n(j) + w_k(j) \tag{4.16}$$

Therefore,

$$L_k^n = \max(L_k^n - 1, 0) \tag{4.17}$$

When the $(n+1)^{st}$ datagram from the tagged traffic arrives at the queue and occupies the $(k+1)^{st}$ slot, the corresponding queue length is L_0^{n+1}. Then the probability distribution $q_0^{n+1}(j)$ represents the distribution of queuing length before arrival of $(n+1)^{st}$ datagram from the tagged traffic and can be written as,

$$q_0^{n+1}(j) = \sum_{k=0}^{I} f(k).q_k^n(j) \tag{4.18}$$

Eq. (4.18) requires iterative calculations by setting the initial values to $q_0^1(j)$ with the condition;

$$\sum_{j=0}^{I} q_0^1(j) = 1 \tag{4.19}$$

The delay for the tagged traffic can be estimated using queuing length distribution at the output queue. When the $(n+1)^{st}$ datagram of tagged traffic arrives at the $(k+1)^{st}$ slot, the queuing delay can be estimated based on the length of the queue given by,

$$L_k^n = L_{(k-1)}^n + W_{(k-1)}^n \tag{4.20}$$

where $L^{n}_{(k-1)}$ = datagrams already in the queue and $W^{n}_{(k-1)}$ = newly arriving datagrams of background traffic at the k^{th} slot along with the $(n+1)^{st}$ datagram, but served before $(n+1)^{st}$ datagram from the tagged traffic. So, the waiting time for the $(n+1)^{st}$ datagram includes:

1) time for datagrams in the buffer after arrival of the n^{th} datagram in the $(k-1)^{st}$ slot and

2) the time taken by the background datagrams arrived along with the $(n+1)^{st}$ datagram but served before the $(n+1)^{st}$ datagram.

Thus, the pdf of waiting time of the $(n+1)^{st}$ datagram is simply the convolution of the two pdfs given by:

$$q^{n}_{k} = q^{n}_{(k-1)}(j) \otimes w^{n}_{(k-1)}(j) \tag{4.21}$$

Mean instantaneous delay (M_{id}) can be expressed as

$$M_{id} = \sum_{j} L^{n}_{k}.q^{n}_{k} \tag{4.22}$$

Using Eq. (4.21) the above leads to

$$M_{id} = \sum_{j} L^{n}_{k}[q^{n}_{k-1}(j) \otimes w^{n}_{k-1}(j)] \tag{4.23}$$

Figure 4.3 indicates the mean instantaneous delay for TCP tagged traffic for three chosen values of (α).

The model for instantaneous delay and jitter was implemented in NS2 using the "Delay-Box" node. "DelayBox" (Tmix traffic generator) is a tool developed by the Distributed and Real-Time systems research group at the University of North Carolina at Chapel Hill for NS2 to implement a transport layer delay following a specified distribution in a link [98]. It is placed on the link to introduce a delay to each packet based on a specified

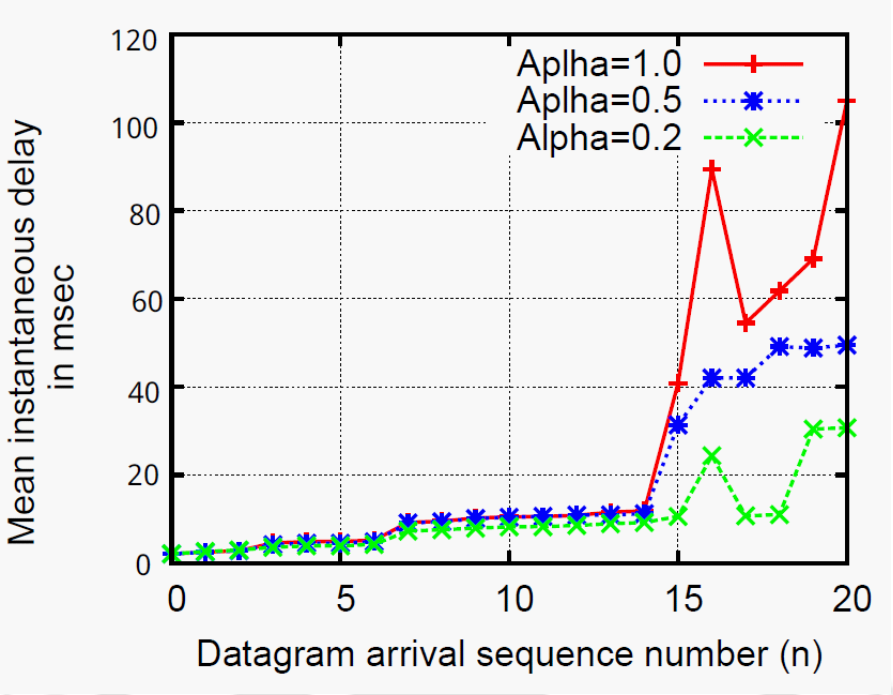


FIGURE 4.3: Mean instantaneous delay for tagged TCP flow for various fractions of TCP in the background traffic

distribution. For this work, the specified distribution is the Weibull distribution according to the parameters mentioned in Table 3.1. This delay box can be treated like regular nodes. The topology is a bottleneck as shown in Figure 4.5.

The mean instantaneous delay has been plotted with respect to the sequence of datagrams (n) in Figure 4.3 for different values of α for TCP tagged traffic. It can be readily seen that the mean instantaneous delay increases sharply as α, increases beyond 0.5. It is also observed that mean instantaneous delay increases considerably when traffic load in the number of datagrams increases beyond n = 15, this may be due to buffer capacity getting fully utilized.

4.3 Analysis for average values of delay jitter for tagged traffic

Let random variable $DJitter_k^n$ represent the delay jitter between n^{th} and $(n+1)^{st}$ datagrams emerging from the same source. Then $DJitter_k^n$ is given by;

$$\mathrm{DJitter}_k^n = \mathrm{L}_k^n + \mathrm{W}_k^n - (\mathrm{L}_k^0 + \mathrm{W}_0^n) \tag{4.24}$$

The distribution of jitter can be calculated by de-conditioning $\mathrm{DJitter}_k^n$ as,

$$\mathrm{djitter}^n(j) = \sum_{k=0}^{K_{max}} f(k).\mathrm{djitter}_k^n(j) \tag{4.25}$$

The steady state value of delay jitter is given by,

$$\mathrm{Djitter}(j) = \lim_{n\to\infty} \mathrm{djitter}^n(j) \tag{4.26}$$

It is evident from Table 4.1 that peak-to-peak jitter values for a typical flow degrade when the fraction of TCP datagrams increase in the background traffic.

TABLE 4.1: Analytical Jitter values

α	**Peak jitter values (ms)**
1.0	3000
0.5	1400
0.2	900

4.4 Average end-to-end delay based on simulations

In a widely accepted conceptual hierarchical internetwork three-layer model [99] as shown in Figure 4.4 with core, aggregation (distribution) and access layers where access layer represents hosts' connection to the network. Distribution is aggregation for all access layer devices and the core layer provides an optimized and reliable transport structure by forwarding traffic at very high speeds. Core layer switches generally work at the fastest speed and are generally implemented using MPLS by the Internet Service Providers. At the edge of the MPLS network, IP headers are still visible where congestion avoidance mechanisms are still implemented. However, within core MPLS, P-routers (Provider routers) typically only look at MPLS headers and therefore cannot implement TCP

congestion avoidance mechanism, but manage congestion based on EXP (Traffic class) values (RFC 5462). Most delay on core networks is caused by speed-of-light propagation over long distances. Many Internet Service Providers (ISP) provide online public access to round trip delay statistics on their networks e.g. AT&T, Global Crossing's network, etc. The "last mile" of the access network, usually operates at speeds significantly slower than that of the core network. Queuing delays add to both the access and core networks. Due to the lower data rates involved, queues in the "last mile" contribute a larger share to the overall network delay than those in the core routers for similar levels of congestion. For instance, the servicing time required for a single 1500-byte packet is 12ms at 1 Mbps, but only 12 μ s at 1 Gbps. Therefore, bottleneck nodes at the edge of the network topology are considered, as the major contributor of the overall delay [around 85 %]. It is the access and aggregation network that contributes maximum to end-to-end delay compared to the core network delay.

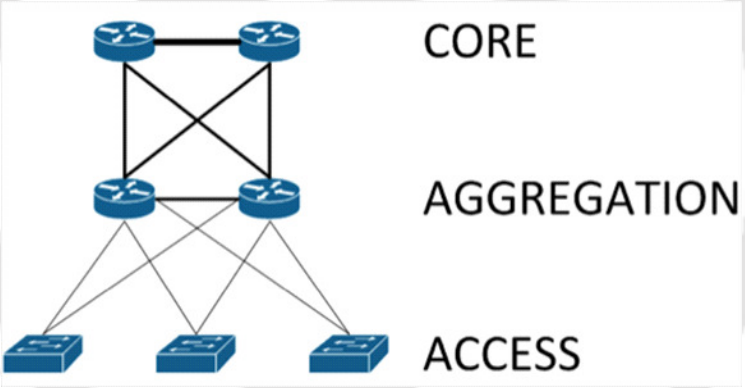


FIGURE 4.4: Hierarchical block diagram

To understand the implicit feedback mechanism in congestion control, performance analysis regrading the delay friendliness of TCP is required. The overall analysis requires an understanding of how the router handles congestion as congestion occurs in the network and the dynamic buffer occupancy mechanism. A network topology which needs to be used in such studies is of several TCP flows sharing a bandwidth-limited bottleneck link with two routers is followed in this work for simulations. The simulation study of queuing behavior due to the multiplexing of the TCP flows is analyzed in this section.

Datagram losses occur very often on the bottleneck router from local area networks (LANs) to wide area networks (WANs). The simulation model consists of the bottleneck

node from LAN to WAN and source-destination pairs. A simple dumble-shaped network topology as shown in Figure 4.5 has been simulated. The simulations have been carried out using the NS2 platform and the plots for the average total end-to-end delay, which includes queuing delay, along with propagation delay, transmission delay, and processing delays, have been plotted in the following subsection. Except for queuing delay and number of retransmissions, the rest of the delay components being deterministic, are the same in all the cases. The retransmission delays due to congestion are also a significant random parameter. Equation for retransmission delay is given as,

$$A_d = \sum (1 + p)\,[X + Y + Z] \tag{4.27}$$

where A_d = Average total end – to – end network delay, p is the number of times the datagram needs to be retransmitted and X, Y and Z are propagation, transmission plus processing and queuing delay components respectively. Clearly p, the number of retransmissions and Z (queuing delay) are random delay components and p=0 represents the UDP case, where there are no retransmissions.

The intention of NS2 simulation has been to understand the effect of quantum of background traffic (TCP flows) on tagged traffic flow at the congested router and measure the variability of queuing delay and number of retransmissions in the controlled environment. Therefore, the parameters of the simulations are chosen accordingly. In the simulated network, the proportions of TCP and UDP flows were varied. Analytical results represent the mean queuing delay (waiting time) at the single router (propagation, switching and transmission delays are excluded which are deterministic). The analytical results indicated clearly that the mean delay of the datagrams at the router, gets degraded by the increased fraction of TCP in the background traffic. Further, the least minimum end-to-end delay encountered is for the UDP flow (which has no retransmissions and acknowledgements). The analytical results of mean queuing delay is for a single congested router whereas, in simulations the performance is obtained for a typical flow (either TCP or UDP) for parameters like average end-to-end delay, instantaneous delay and jitter for the topology in question. Therefore, analytical and simulation results can't be directly compared but validated by the trend (quantitative match is expected). Also, it is that, as

the TCP fraction of the background traffic increases, the number of retransmissions also increase for a tagged TCP flow. Therefore, the random parameter, namely number of retransmissions occurs, only in simulations and not in analytical results. The parameters used in the NS2 simulation have been shown in Table 5.3. As mentioned earlier TCP Cubic has been the most widely used protocol [46.92 %] among the TCP variants [27] in the Internet. Therefore, all simulations have been carried our using TCP Cubic protocol. Further, Random Early Detection (RED) active queue management (AQM) has also been incorporated to reduce overall end-to-end delay during simulations. The simulations have also been done with other TCP flavors, such as TCP Reno, TCP Compound and comments on the comparative delay performance that have also been incorporated in this work.

While computing the average, 15 iterations were carried out in NS2 simulator and end-to-end delays were evaluated for all iterations. The average of all these iterations has been taken as average end-to-end delay for the network. In the Internet environment, both TCP segments and UDP datagrams are multiplexed at routers and contend for bandwidth allocation. Typically, on the Internet, loss of datagrams occurs at the bottleneck router, where datagrams move from LAN to WAN.

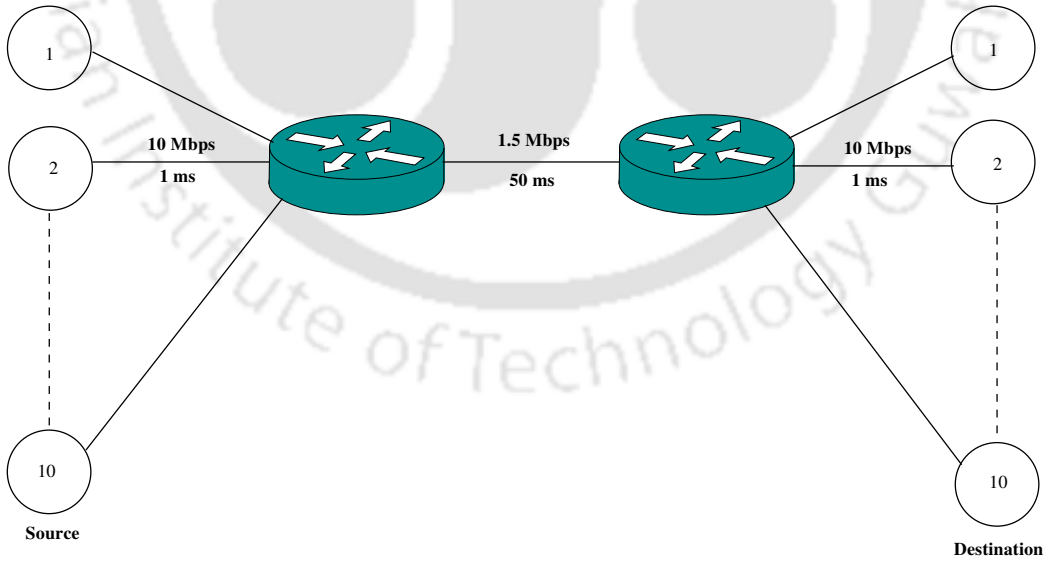


FIGURE 4.5: Network layout for homogeneous scenario

TABLE 4.2: NS2 Simulation parameters

LAN bandwidth	10 Mbps
WAN bandwidth	1.5 Mbps
Access-link delay	1 ms at both sides (source and destination) in homogeneous scenarios, and in case of heterogeneous scenarios 1 ms at source side and pattern of (1,3,..19) ms for the last destination side
Bottleneck-link delay	50 ms
Queue limits	Default
TCP type	Std. Cubic TCP (with built in AIMD mechanism)
TCP's maximum window size	Default (1000 packets)
Packet size	Default (1460 in bytes)
Queue management	RED queue management was used in the bottleneck link and DropTail in the access link

4.4.1 Average end-to-end delay evaluation

For the simple network topology considered for evaluation shown in Figure 4.5, ten sources and ten corresponding destinations have been assumed. Initially, traffic on TCP is taken as 100% and that in UDP as 0%. Gradually the TCP traffic is reduced to 0% by simultaneously increasing the UDP traffic to 100%. Two network scenarios, namely homogeneous and heterogeneous, have been considered. In both the cases, it was assumed that the links are error-free; the link bandwidths are as shown in Figure 4.5 and routers have infinite buffer space. In the homogeneous scenario, delays on the source side and delays on the destination side have been assumed equal; whereas, for the heterogeneous scenario, delays on the source side LAN are fixed and delays on the destination side followed a pattern such as 1 ms, 3 ms...19 ms for the last destination. Infinite FTP sources were attached to generate TCP traffic, while the constant bit rate (CBR) traffic source was used to generate UDP traffic.

For a typical flow, average delay values were determined by varying the proportion of TCP and UDP, total flows remaining constant. For both homogeneous and heterogeneous

cases, values of average end-to-end delay for a typical tagged flow of TCP and UDP, with different proportions of TCP in the background have been presented in Figure 4.6 and Figure 4.7. They show average end-to-end delay for homogeneous and heterogeneous scenarios respectively. For a fair comparison, care has been taken to ensure that the WAN link was never underutilized.

In Figure 4.6 and Figure 4.7 x-axis indicates an increasing number of TCP nodes, which also indicates decreasing number of UDP nodes as the total number of nodes remains constant e.g., 2 TCP nodes means 8 UDP nodes, 5 TCP nodes means 5 UDP nodes and so on (α increasing from 0.2 to 1.0).

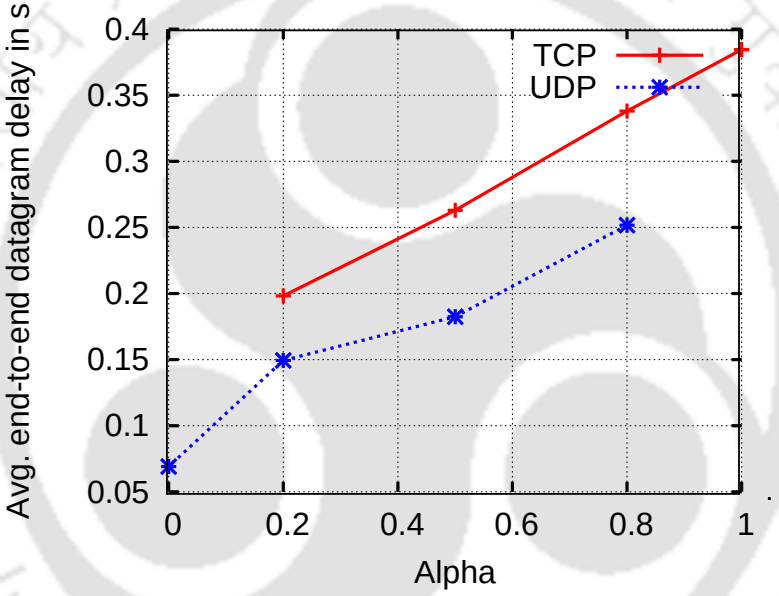


FIGURE 4.6: Average end-to-end delay for a typical TCP and UDP flow for various proportions of TCP background traffic (homogeneous case)

Observations, which may be drawn from Figure 4.6 and Figure 4.7, could be as follows:

1 For all the proportions of TCP in the background traffic and for both cases of homogeneous and heterogeneous network scenarios, the average end-to-end delay for TCP tagged flow is significantly higher compared to UDP tagged flow. The increase of the TCP fraction of the background traffic degrades the average end-to-end delay more for TCP tagged flow compared to UDP tagged flow.

2 When TCP background traffic gradually increases to 100% (UDP background traffic decreases to 0%), the average end-to-end delay for TCP tagged flow increases

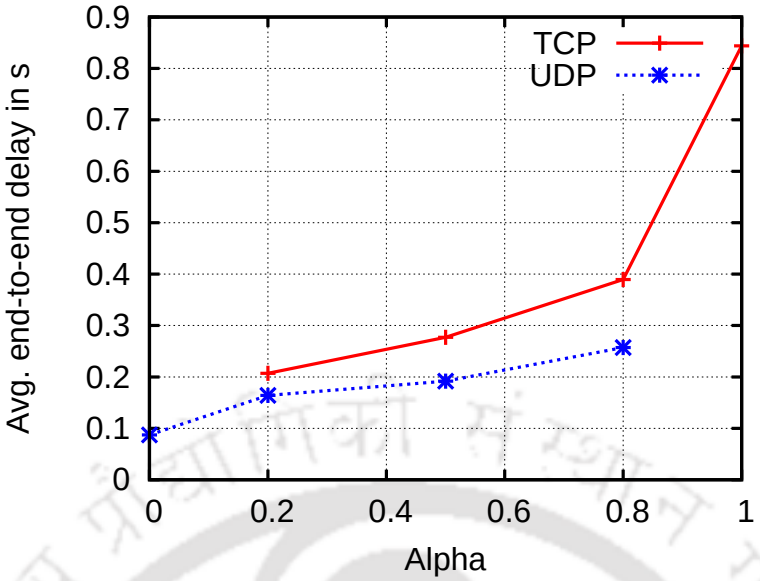


FIGURE 4.7: Average end-to-end a delay for a typical TCP and UDP flow for various proportions of TCP background traffic (heterogeneous case)

considerably. This increase is 200 % for homogeneous cases and 400 % for heterogeneous cases.

3 For 100 % TCP background traffic, the average end-to-end delay for TCP tagged flow in the heterogeneous case is 820msec as compared to 408msec for the homogeneous case.

4 Average end-to-end delay has transmission, propagation, queuing delay components. In homogeneous scenarios, RTT between source-destination nodes is fixed, while in heterogeneous scenarios RTT varies in agreed fashion. In either case, for a typical UDP flow with maximum background TCP traffic, the average total end-to-end delay for homogeneous and heterogeneous scenarios is 120ms and 200ms respectively. Secondly, if compared to a typical TCP flow with maximum TCP in the background, the average total end-to-end delay for the TCP flow with homogeneous and heterogeneous cases is 400ms and 800ms respectively. As UDP does not have retransmissions and congestion control mechanisms, the significant increase in total end-to-end delay for the TCP case can definitely be attributed to TCP congestion control mechanisms and retransmissions.

The observations represented in Figure 4.1 (the analytical queuing values are for single router) and simulation results are for 2 routers. The end-to-end delay is consistent with the observation-based simulation results shown in Figure 4.7, indicating that in a controlled network environment, the variation in average end-to-end delay is mainly due to queuing delay component and number of retransmissions and is more significant for TCP tagged traffic. Considerable (fourfold) increase in queuing delay is observed as the fraction of TCP in the background traffic increases. It is therefore clear that the fraction of TCP in background degrades the average end-to-end delay for both tagged TCP and UDP flows. Secondly, for the NS2 simulations, the datagram sizes of TCP (1000 bytes), TCP ACKs (40 bytes) and UDP (210 bytes) are fixed; therefore, the significant increase can definitely be attributed to TCP congestion control mechanism peculiarly the retransmission aspect of TCP congestion control. Without retransmissions, total end-to-end delay of a flow would be as low as UDP RTT values, which has random queuing delay component, but no retransmissions.

4.4.2 Plots for average instantaneous end-to-end delay

For both homogeneous and heterogeneous cases, average instantaneous end-to-end delays experienced by TCP/UDP tagged traffic have been recorded and plotted for some chosen values of α i.e. percentage of TCP in the background traffic, with respect to datagram arrival sequence. Figure 4.8 and Figure 4.9 show a plot of average instantaneous end-to-end delay of TCP tagged traffic for homogeneous and heterogeneous scenarios respectively. Similarly, Figure 4.10 and Figure 4.11 show the plots of average instantaneous end-to-end delays for UDP tagged traffic. The following observations can be made based on these plots:

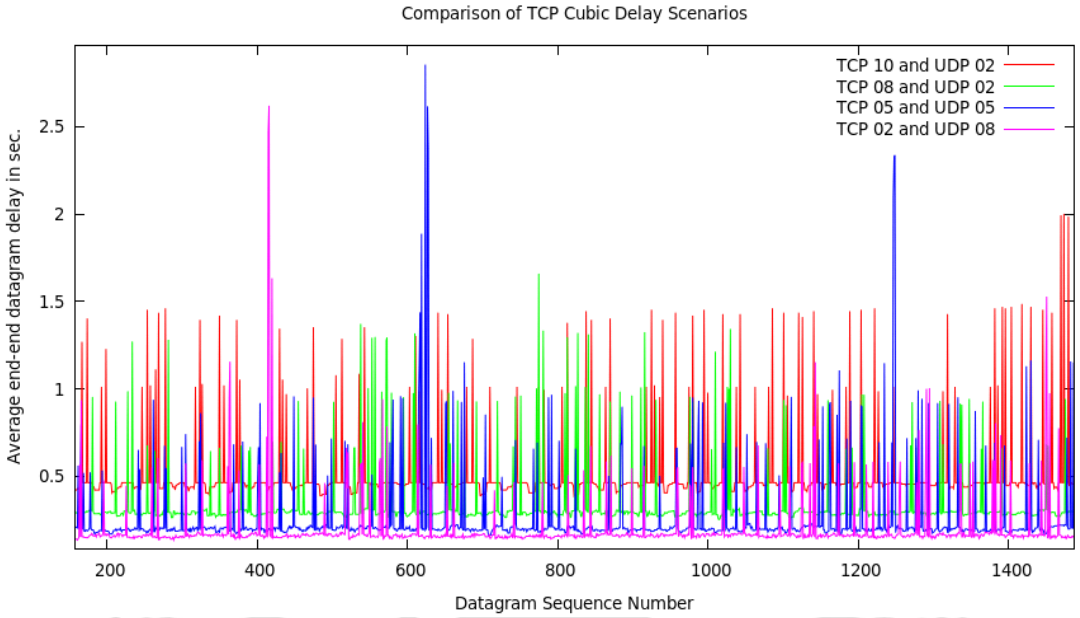


FIGURE 4.8: Instantaneous delay of a tagged TCP flow for various fractions of TCP in background traffic for homogeneous scenario

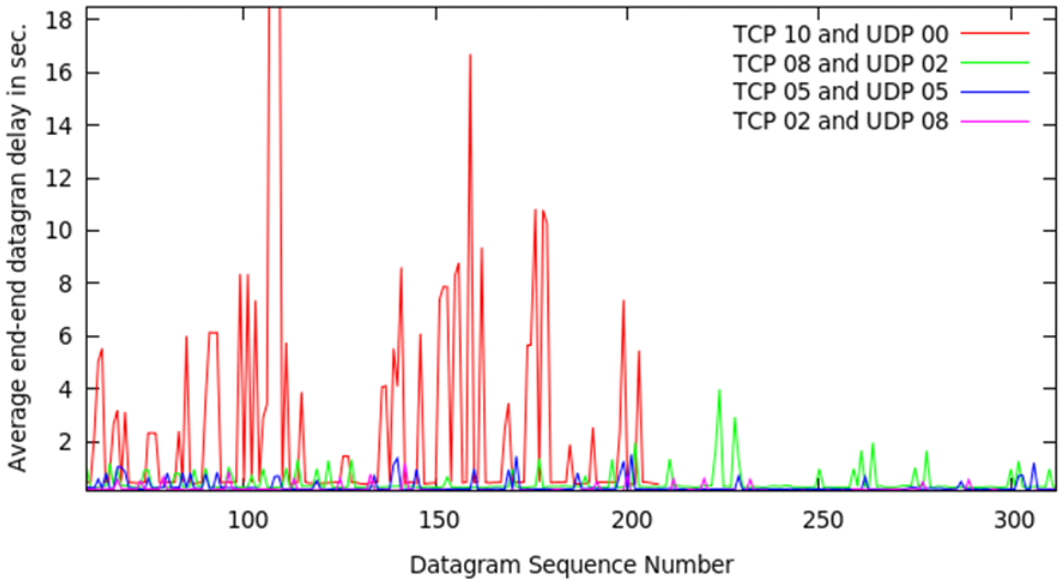


FIGURE 4.9: Instantaneous delay of a tagged TCP flow for various fractions of TCP in background traffic for heterogeneous scenario

1) In all the four cases, average end-to-end instantaneous delay for tagged TCP traffic is significantly higher compared to the UDP tagged traffic. The maximum value of UDP average instantaneous end-to-end delay is 360ms as compared to the maximum value of TCP which is 18000ms (worst case).

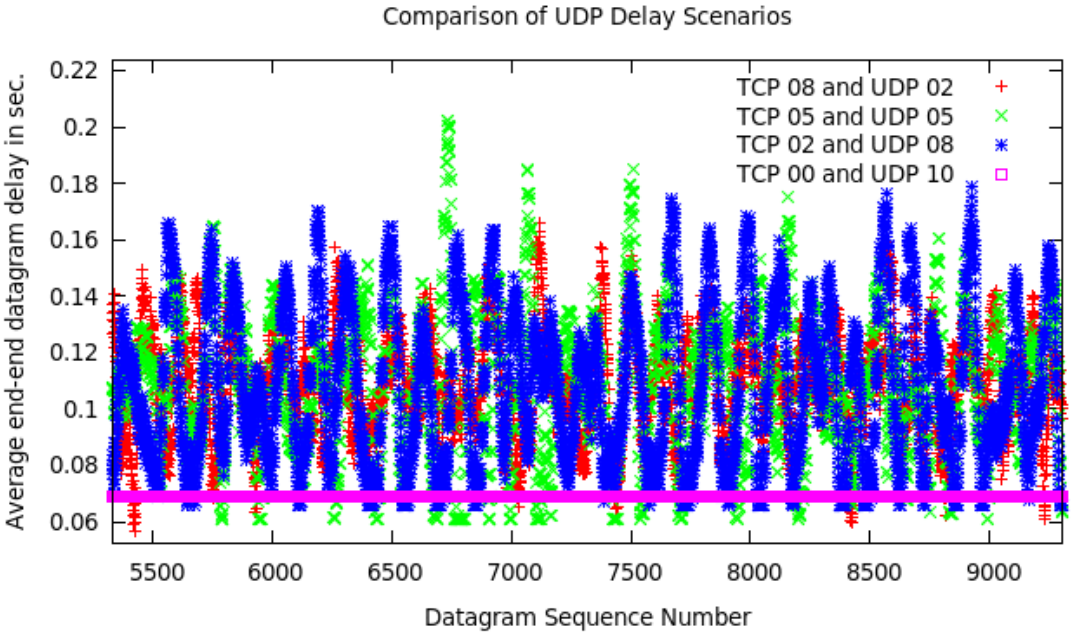


FIGURE 4.10: Instantaneous delay of a tagged UDP flow for various fractions of TCP in background traffic for homogeneous scenario

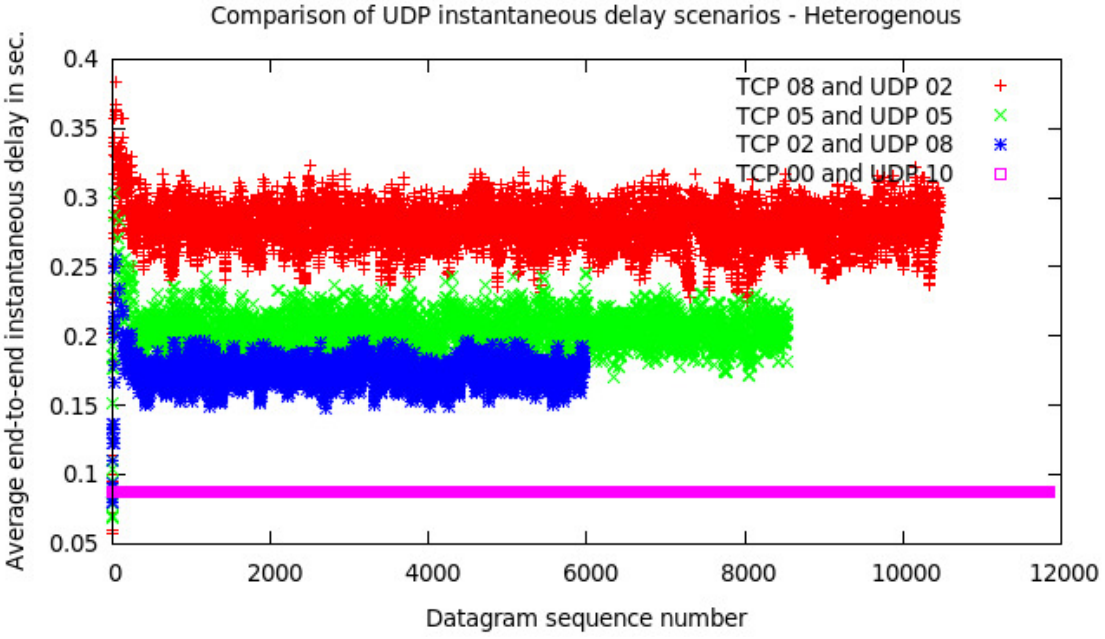


FIGURE 4.11: Instantaneous delay of a tagged UDP flow for various fractions of TCP in background traffic for heterogeneous scenario

2) As the proportion of TCP in the background traffic increases, average instantaneous end-to-end delay for both TCP and UDP tagged traffic increases. Comparing Figure 4.8 and Figure 4.9, it is observed that for TCP tagged traffic, the delay is

more adverse for heterogeneous scenarios. This is not true with UDP tagged traffic. Figure 4.10 shows that the average instantaneous end-to-end delay performance of UDP tagged traffic exhibits a lot of variation. The variation increases with the increase of TCP proportion in the background traffic. Figure 4.11, for the heterogeneous scenario, indicates clearly that the value of average instantaneous end-to-end delay increases with an increase in TCP proportion in the background traffic. The reason for this may be, the greedy nature of the TCP congestion control algorithm, which tries to grab all available outgoing capacity for TCP traffic. Interestingly, in the heterogeneous case, not all the datagrams of tagged UDP flow reach the destination. As the background TCP fraction increases, more datagrams reach the destination, but with the increased average instantaneous end-to-end delay. Variation in end-to-end instantaneous delay is more in the homogeneous case compared to the heterogeneous case, but in the homogeneous case, most datagrams reach the destination. On the Internet (heterogeneous case), most UDP based applications like DNS, DHCP, signaling protocols, etc. are all low RTT value applications.

3) In the homogeneous case for TCP tagged traffic and for 100% TCP in the background traffic, global synchronization effect in TCP is pronounced, see Figure 4.8 for α =1.

4.4.3 Jitter simulations

Average instantaneous end-to-end jitter experienced by datagrams of TCP/UDP tagged traffic has been recorded and plotted for some chosen values of α i.e. percentage of TCP in the background traffic, with respect to the datagram sequence number. Heterogeneous case of TCP Cubic and TCP Reno jitter values are compared and shown in Figure 4.12 and Figure 4.13 respectively. For paucity of space, UDP plots are not shown, but Table 4.3 shows peak jitter values for homogeneous and heterogeneous cases for both TCP and UDP datagrams.

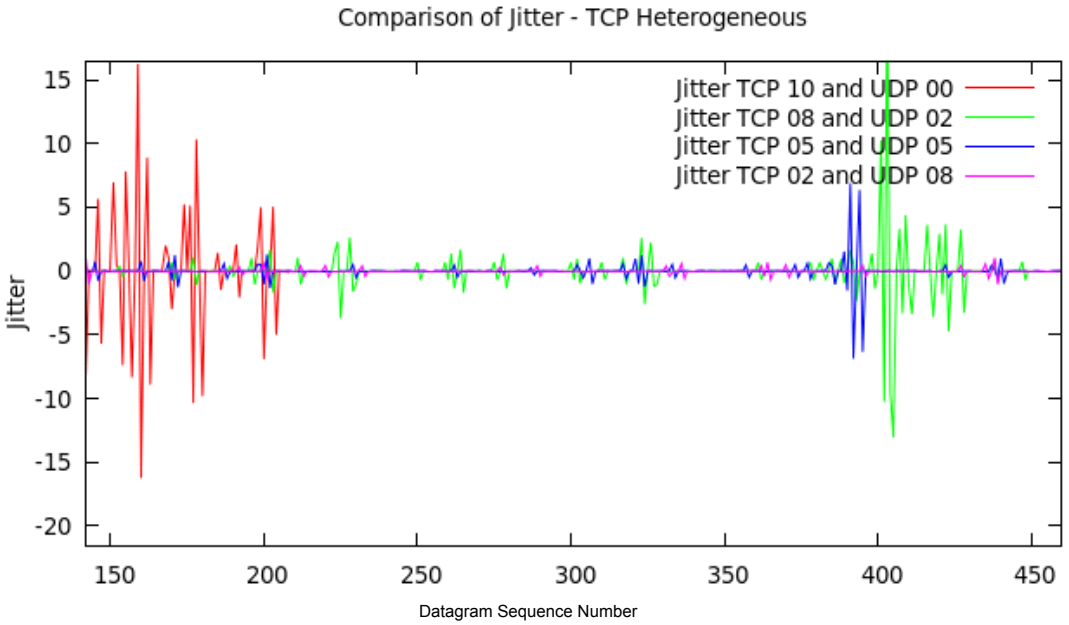


FIGURE 4.12: The Jitter of a Cubic TCP tagged flow for various fractions of TCP background traffic

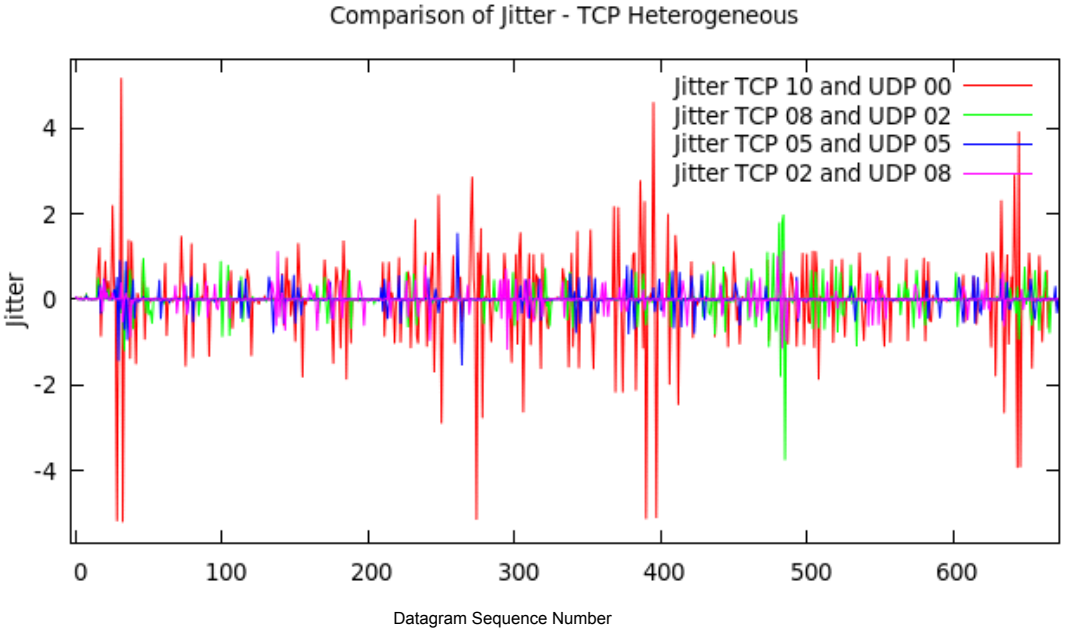


FIGURE 4.13: The Jitter of a Reno TCP tagged flow for various fractions of TCP background traffic

The following observations can be made based on these plots.

1)TCP Cubic jitter values are worse (-4.69) sec and Reno is (2.48) sec., But in either case, the jitter values increase with increased TCP fraction in background traffic

TABLE 4.3: Peak-to-peak Jitter values

	Jitter TCP flow(sec)		**Jitter UDP(sec)**	
α	**Homo**	**Hetero**	**Homo**	**Hetero**
1.0	1.513	-4.69	-	-
0.8	1.447	1.64	0.18	0.200
0.5	0.380	1.260	0.15	0.118
0.2	0.509	0.565	0.12	0.105
0	-	-	0	0

as shown in Figure 4.12 and Figure 4.13 respectively. These values are critical for real-time applications

2) When peak-to-peak jitter for tagged TCP flow is compared to UDP tagged flow, the jitter for tagged TCP flow is significantly higher compared to the UDP tagged traffic Table 4.3. The reason possibly is buffer dynamics at the router queue.

4.4.4 Comparison of different flavors of TCP using average end-to-end delay

Just to compare different flavors of TCP, the average end-to-end delays for a typical TCP tagged flow, with α= 0.8 of background traffic based on TCP have been recorded. Table 4.4 shows records for the average end-to-end delay measurements on congested links with and without RED for α using different TCP flavors.

It can be seen that when α =0.8 performances of TCP Reno and TCP Compound are comparable; whereas, the TCP Cubic gives the worst delay performance. However, the performance improves with the use of RED for the TCP Cubic scheduling algorithm by 42.3 % in the case of homogeneous network scenario and by 38.27 % for the heterogeneous network scenario.

TABLE 4.4: Average end-to-end delay measurements of congested link with and without RED for various TCP types

Average End-to-End delay in in sec. for α =0.8 (80 % nodes are TCP)		
TCP Type	**W/O RED**	**RED**
TCP Cubic (Hetero)	0.5622	0.347
TCP Cubic (Homo)	0.3434	0.1981
TCP Compound (Hetero)	0.3409	0.2132
TCP Compound (Homo)	0.3174	0.1816
TCP Reno (Hetero)	0.3894	0.1952
TCP Reno (Homo)	0.3380	0.1838

4.5 Conclusions

This chapter explores the interaction of datagrams between background TCP flows with tagged TCP, and also the interaction between datagrams of tagged UDP with TCP. The datagrams at the router are from number of flows, which constitute a stochastic process. Delay parameters for datagrams of a typical tagged TCP flow are substantially larger than that of UDP, but interestingly the interaction gives surprising results. The mean delay of datagrams worsens 400 % for TCP and around 30 % for UDP when background traffic has a maximum fraction of TCP. Secondly, average instantaneous delay and jitter for datagrams of TCP tagged flow are substantially larger when compared to UDP tagged flow when background traffic has a maximum fraction of TCP. Delay performance values are dependent directly on the fraction of TCP traffic in the background. This is true for all TCP flavors like Reno, Cubic, and Compound even after the use of scheduling mechanism (RED). Therefore, the inference is clear that the interaction of TCP with fellow TCP and/or UDP is adverse for delay performance. This affects all applications, especially the QoS for real-time applications. The reason can be attributed to queuing delay and the number of retransmissions, which are random in nature. As fraction of TCP in the background traffic increases beyond 80 %, this degradation would become quite severe and would compromise the ITU recommendations. The major reasons are buffer dynamics at the congested router, TCP congestion

control mechanism and retransmissions in TCP. Another important factor is the variable segment size of TCP. Modern-day Internet traffic has crossed 90 % mark of TCP fraction in background traffic. Therefore, the average end-to-end delay would be very much on the higher side. However, degradation in end-to-end delay being significantly higher for TCP, compared to UDP based traffic (approximately 4 times), there is no point in exploring any new TCP based solutions as they would always worsen the delay performance. Possibly one could look at exploring UDP based congestion control which may minimize the queuing delays, particularly for real-time applications. However, this may necessitate major architectural changes in the Internet which would be difficult to implement, if not impossible. Alternatively, one can explore the congestion avoidance mechanism using link-by-link congestion control at the network layer itself as suggested in the literature survey by [46], thereby trying to avoid the congestion before it happens. This, however, may also necessitate the optimum sizing of buffers in routers and might help in minimizing the effect of queuing delay on the Internet.

Chapter 5

Model of UDP Throughput in Presence of TCP

This chapter presents the proposed algorithmic model for UDP throughput in presence of TCP. The model is implemented with the values obtained in the measurement dataset and validated using NS2 simulations.

The user session translates into a single flow or multiple flows. A flow/ flows is bursty in nature and network traffic is aggregations of bursts of single or many flows and is called bursty traffic. The burstiness of arrivals is the evident characteristics of self-similarity given in a seminal work by Leland *et al.* [100]. Crovella *et al.* [101] conclude that the relation between file size distribution and self-similarity is not affected by the distribution of packet interarrival times, topology, cross-traffic and changes in network resources. Reliability and flow control measures at the transport layer preserve self-similarity. In fact, they argue that TCP moderates the performance of the network. Mushtaq *et al.* [102] statistically evaluated packet inter-arrival time and packet size and concluded that packet inter-arrival time and packet size follow a power law and can be modeled by heavy tail distribution. The Statistical characterization of measurement data given in Table 3.1 agree with the characteristics of self-similarity.

5.1 Modelling and analysis of UDP throughput

The model of arrival process and effective service process in this chapter are considered from Eq. (3.4) and Eq. (3.5). Also referred is the the measurement data from statistical characterization of traffic given in Table 3.1.

While servicing the datagrams the router will find the different fractions of TCP and UDP. The service times for TCP and UDP datagrams are fixed say, (θ) datagrams/sec. The service decision of routing is made by the router by considering the header only and it is independent of payload size of the datagram in consideration. The datagrams when served at the IP layer create space in the router buffer according to their respective sizes and in turn, is the decisive factor whether the newer arriving datagram is accommodated or discarded. This can result in loss of datagrams. The effect of the size of datagrams and continuously changing buffer occupancy which mandates a change in waiting time of queue is incorporated in the model Eq. (3.5). The effective service times will vary according to the size of the datagram being served.

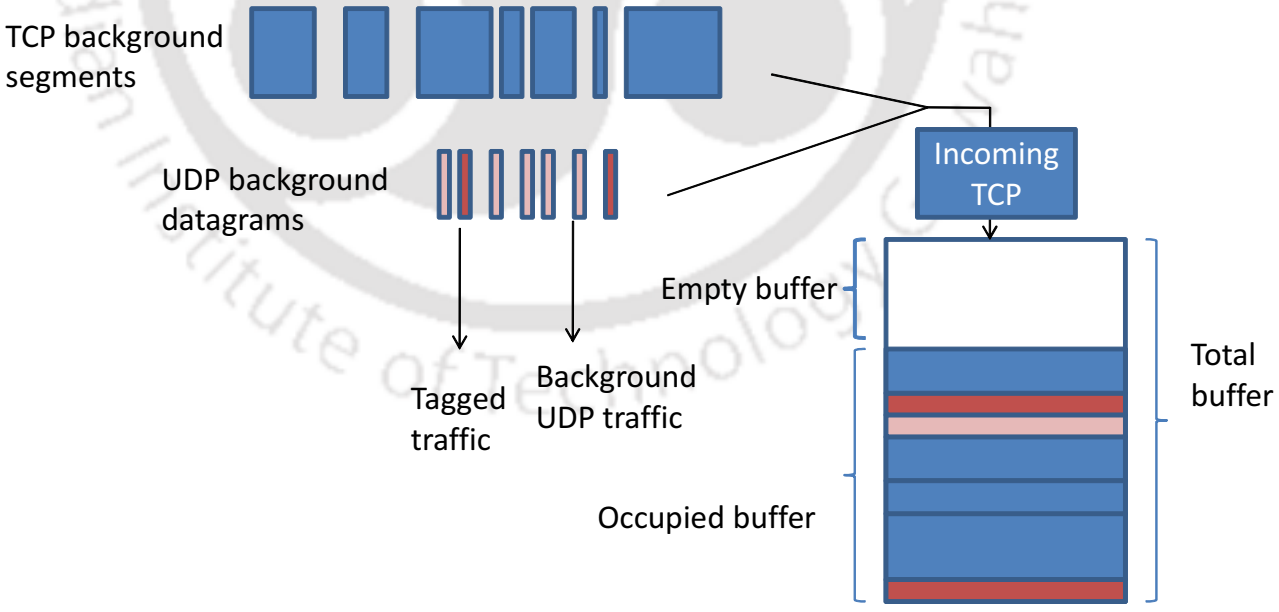


FIGURE 5.1: Buffer occupancy mechanism

When a datagram of interest from tagged traffic flow arrives at the first buffer, it would either be accommodated or discarded in the buffer depending on the empty space available in the buffer at that instant. If the empty space is larger than the size of incoming datagram, then that datagram gets accommodated. A tagged flow is identified by a source-destination IP address pair and the source-destination port-pair used for that session. Every single datagram of the tagged flow if accommodated in the router buffer gets served. This is true for all datagrams and for all buffers en-route destination, although maximum loss occurs at bottleneck nodes. The interesting concept here is that service times of router are constant, but datagram sizes not being the same, the empty space created due to the exit of datagrams depends on the size of datagrams. Therefore, the buffer occupancy constantly changes and this dynamic, changing buffer occupancy decides whether a UDP datagram will be accommodated or discarded at that buffer as depicted in Figure 5.1. The successfully stored datagrams at first buffer, then are processed in the buffer, and are routed to next router buffer en-route destination. Again at the next router hop, if the newly arriving finds empty space then the datagram will be accommodated and served else discarded and this continues till datagram approaches last hop. The mechanism is considered by the algorithmic model. The algorithmic UDP throughput model will collect all successful datagrams at the destination which pass through every single buffer en-route destination for calculating throughput. The algorithmic UDP throughput in presence of TCP is modeled as in Algorithm 1.

5.1.1 Implementation of the algorithmic model using measurement data in Matlab

The algorithmic UDP throughput model has been implemented using values from the statistical characterization of traffic from Chapter 3 Table 3.1 for the topology as shown in Figure 4.5; only the number of intermediate routers are varied as per the requirements. Secondly, the flow of interest is UDP and the performance parameter is UDP throughput observed for various fractions of TCP in the background traffic. Figure 5.2 shows the results of UDP throughput model implementation where 'a' indicates the fraction of TCP traffic in the background. Observation is that as the fraction of TCP traffic in the background and the number of hops increases, the UDP throughput drops.

Require: {Input Data}

Tagged Source IP = *sIP*

Tagged Source Port = *sPort*

Tagged Destination IP = *dIP*

Tagged Destination Port = *dPort*

packetSize: size of incoming UDP packet

emptyBuffer: Empty Buffer available at router

1: H ← Total number of routers en-route
2: h = 1
3: **while** (h < H) **do**
4: *h.StoredPackets* = 0
5: ∀ UDP packets received with (*sIP, sPort, dIP, dPort*)
6: *h.emptyBuffer* ← Get buffer space available at router h
7: **if** (*packetSize* ≤ *h.emptyBuffer*) **then**
8: Store packet at router h
h.StoredPackets++
9: **else**
10: discard the packet
11: **end if**
12: **print** *h.StoredPackets*
13: h = h + 1
14: **end while**
15: end

Algorithm 1: *UDP throughput for individual flow at each router in the presence of other background traffic*

The reason attributed is that the probability of a datagram being served at the router depends on the buffer dynamics at the router queue and this is true at each router en-route the destination. Therefore, taking into consideration the cumulative effect of the loss of the datagrams, the throughput of the UDP flow of interest decreases. If the sending rate of UDP is reduced, it hardly matters for UDP throughput, as TCP follows the congestion algorithm which will grab any bandwidth available till the congestion occurs. TCP has a greedy congestion control mechanism. There is a variation of UDP throughput for different TCP fractions in background traffic. Maximum UDP is throughput is when there is no TCP and beyond 20 hops UDP throughput falls drastically. The loss of datagrams is due to buffer dynamics only. The links considered are lossless and packets sent by source were 1000 in number refer Figure 5.2 and Figure 5.3.

To specifically analyze the reason for the fall of UDP throughput when background traffic is the entire TCP (a=1), the implementation was carried out to observe UDP throughput for the same topology as in Chapter 4 Figure 4.5. The specific case of source sending packets of 600 packets of 210 bytes is shown in refer Figure 5.3, although the trend of results is the same for the number packets sent by the source be 500, 1000, 2000 and 5000. The Figure 5.3 indicates the situation when a = 1, the total number of UDP datagrams sent is 600 and the plot is throughput (Packets) vs. Time. The observation is that the throughput falls intermittently to a value as low as 400. Secondly, to verify how UDP datagram size affects UDP throughput, the experiment was carried out for two cases 'a = 0' and 'a = 1' with varying UDP datagrams sizes for constant values as shown in Table 5.1.

Some pertinent observations from Figure 5.2, Figure 5.3, Table 5.1 and Table 5.2 are

1 Figure 5.2 indicates that the throughput of tagged traffic decreases with a fraction of TCP in background traffic and the number of hops (RTT). Figure 5.3 indicates the intermittent variability of UDP throughput when there is maximum TCP in the background traffic (a = 1).

2 From Table 5.1, when a=0, TCP background traffic is 0%. As we decrease the packet size below 210 bytes, the throughput is maximized and remains almost constant but at the cost of increasing overheads. Secondly, as we increase the

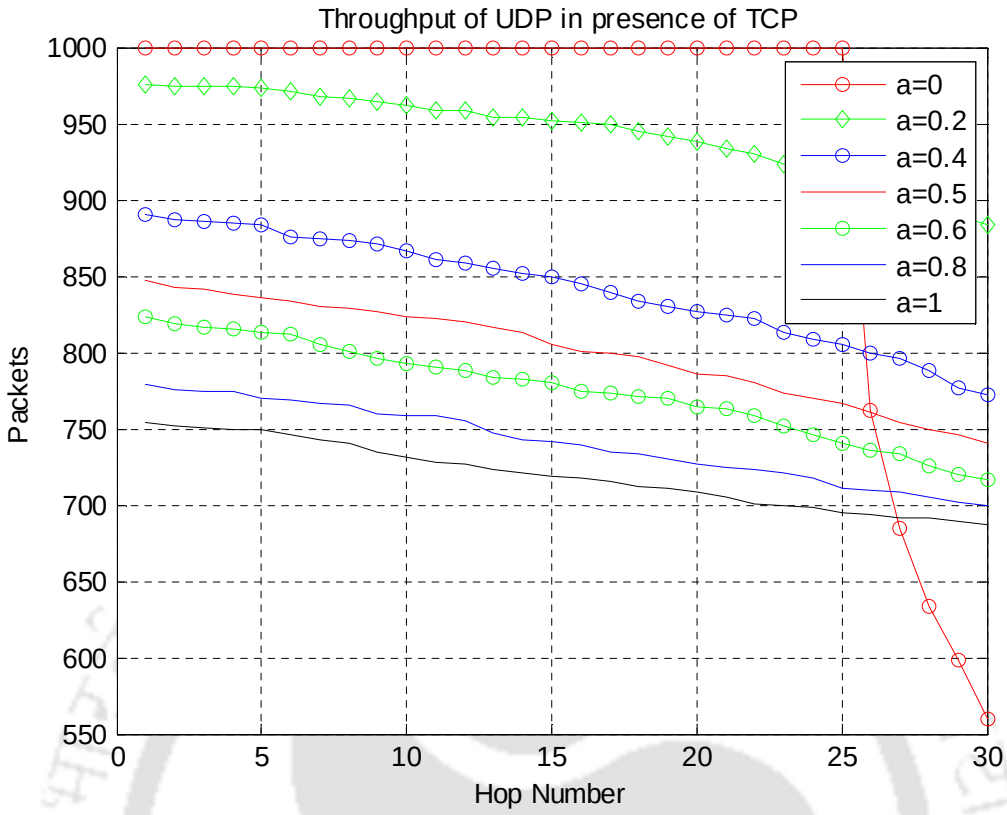


FIGURE 5.2: UDP throughput with respect to the TCP fraction

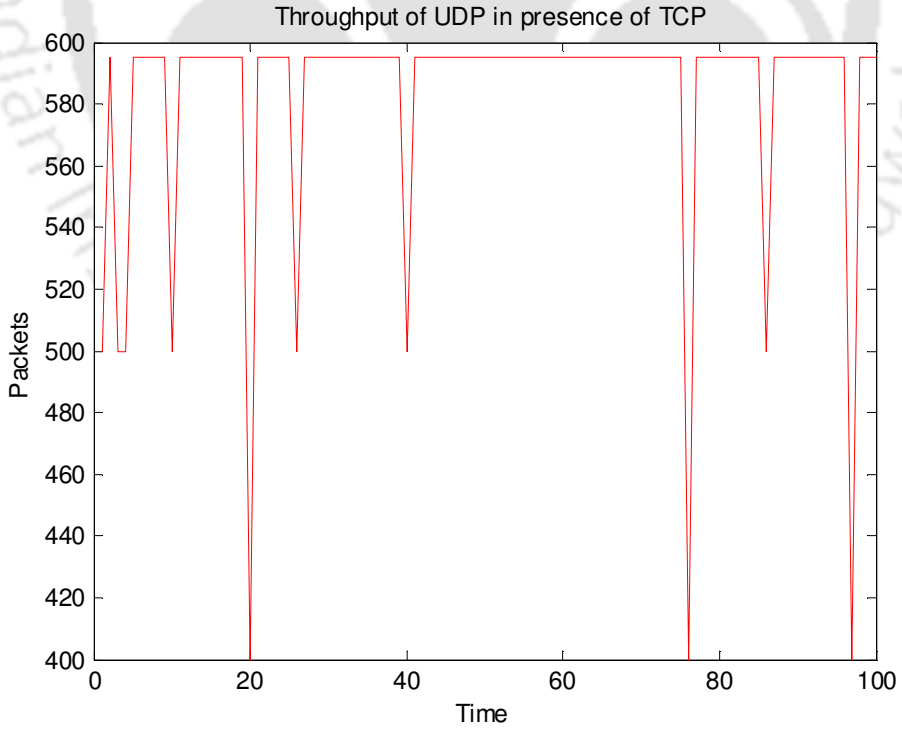


FIGURE 5.3: UDP throughput when TCP fraction is 1

TABLE 5.1: Effect of datagram size of UDP on the UDP throughput

a	**UDP datagram size** (Tagged traffic)	**Datagrams received when 1010 datagrams are sent** (Tagged traffic)
0	100	1010
	210	1000
	230	995
	240	300
	500	200
1	210	750
	500	100
	1500	50

TABLE 5.2: Effect of mean sending rate of UDP on its throughput

Mean sending rate of tagged traffic	**a**	**Total datagrams sent**	**Datagrams Received**	**Hop at which loss occurs**
10	0	1000	1000	25
	1	1000	750	30
15	0	1200	1200	20
	1	1200	850	20
20	0	1500	1500	15
	1	1500	900	15
50	0	10000	10000	10
	1	10000	5500	10

datagram size above 210 bytes, throughput decreases and nearly becomes around 25 % at datagram size equal to 240 bytes. If UDP packet size is increased the UDP throughput keeps decreasing. Hence the optimum packet size can be considered to be 210 at a=0 i.e. in the absence of TCP traffic.

3 From Table 5.1, when a=1, TCP background traffic is 100%, it is evident that the UDP throughput decreases. The trend is clear when background traffic has maximum TCP the UDP throughput is degraded.

4 From Table 5.2, it is seen that even if the mean sending rate of tagged UDP flow is increased throughput of tagged UDP flow reduces. It indicates that the throughput of tagged traffic decreases with the increased fraction of TCP in background traffic increases, and number of hops (RTT). TCP congestion control is a greedy in the sense it tries to send datagrams in the network till it hits congestion and affecting UDP. Therefore, rate at which UDP is sent hardly matters. The scenario is worse for larger fractions of TCP.

Therefore, it is evident that TCP traffic affects the UDP throughput. In the next section simulation results are presented to corroborate the trend of UDP throughput degradation in presence of TCP.

5.2 Simulation results

In an Internet environment, both TCP segments and UDP datagrams are multiplexed at routers and contend for bandwidth allocation. Typically, on the Internet, loss of datagrams takes place at the bottleneck node where datagrams move from LAN to WAN. We have considered a scenario where TCP and UDP contend for bandwidth, as shown in Figure 5.4. Figure 5.4 shows 10 sources and 10 corresponding destinations. Number of routers are varied to increase the number of hops. Initially, traffic on TCP is taken as 100% and that in UDP as 0%. Gradually the TCP traffic is reduced to 0% by simultaneously increasing the UDP traffic to 100%. All results from Figure 5.5 are for a short duration from 45 s to 50 seconds simulation time because the behaviors during the

other parts are very similar. Two cases, namely homogeneous network, and heterogeneous network have been considered. In both the two cases homogeneous network, and heterogeneous network, it is assumed that the links are error-free, the link bandwidths are as shown in Figure 5.4 and all the nodes have infinite buffer space.

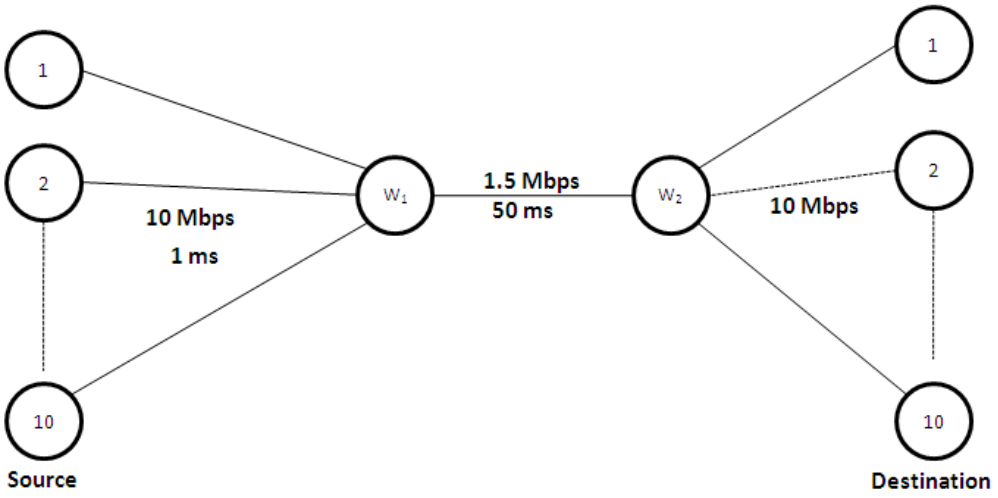


FIGURE 5.4: Network layout

TABLE 5.3: NS2 Simulation parameters

LAN bandwidth	10 Mbps
WAN bandwidth	1.5 Mbps
Access-link delay	1 ms at both sides (source and destination) in homogeneous scenarios, and in case of heterogeneous scenarios 1 ms at source side and pattern of (1,3,..19) ms for the last destination side
Bottleneck-link delay	50 ms
Queue limits	Default
TCP type	Std. Cubic TCP (with built in AIMD mechanism)
TCP's maximum window size	Default (1000 packets)
Packet size	Default (1460 in bytes for TCP and 210 for UDP)
Queue management	RED queue management was used in the bottleneck link and DropTail in the access link

In the homogeneous network, delays on source side LAN and delays on the destination side WAN are equal, whereas, in the heterogeneous network, delays on source side LAN are fixed and delays on destination side WAN follow a pattern such as 1 ms, 3 ms, 5 ms and so on. This network layout is shown in Figure Figure 5.4. Number of routers (W1, W2, W3...) are added as the number of hops varied for simulation. There are infinite FTP source(s) that are attached to generate TCP traffic while the CBR traffic source(s) is used to generate UDP traffic. For both homogeneous and heterogeneous cases, values of the mean delay for different proportions of TCP and UDP results have been presented in Figure 5.4. Care has been taken to ensure that the WAN link is never underutilized during traffic generation. For both homogeneous and heterogeneous cases, values of UDP throughput for different proportions of TCP and UDP results have been presented in Figure 5.4. For a fair comparison, care has been taken to ensure that the WAN link is never underutilized during traffic generation. Average throughput is calculated by using following equation,

$$\text{UDP Throughput} = \frac{\text{Number of bytes received*8}}{\text{Simulation time*1000}} \text{ kbps} \tag{5.1}$$

5.2.1 Observations of simulation results

The simulation results are shown in Figure 5.5. On the X-axis the ratio of TCP to UDP is plotted. Secondly, different colours indicate the number of hops for that respective case. Legend to the right indicate the colour and number of router hops for that respective case. Y-axis indicate the throughput of UDP flow in heterogeneous scenario. First observation is throughput for UDP flow degrades for two parameters namely, fraction of TCP in the background traffic and number of hops. The simulation results clearly corroborate algorithmic observations. UDP throughput is a function of TCP and decreases as the fraction of TCP in background traffic increases. Not all datagrams of tagged traffic find the place in buffer therefore get discarded. Taking into consideration the cumulative effect of the loss of the datagrams which could not be accommodated and served by routers en-route destination, the throughput decreases. If the sending rate of UDP is reduced, it hardly matters for UDP throughput, as TCP follows the congestion algorithm which

will grab any bandwidth available till the congestion occurs. Maximum UDP throughput when there is no TCP. Exactly similar trend is obtained if links with error rate of 0.01 and 0.05 were used. Results of error free links given in Figure 5.5. UDP Throughput falls with increasing number of hops and also by fraction of TCP in the background traffic. Either of cases homogeneous and heterogeneous the trend is similar in fact more worse in heterogeneous case.

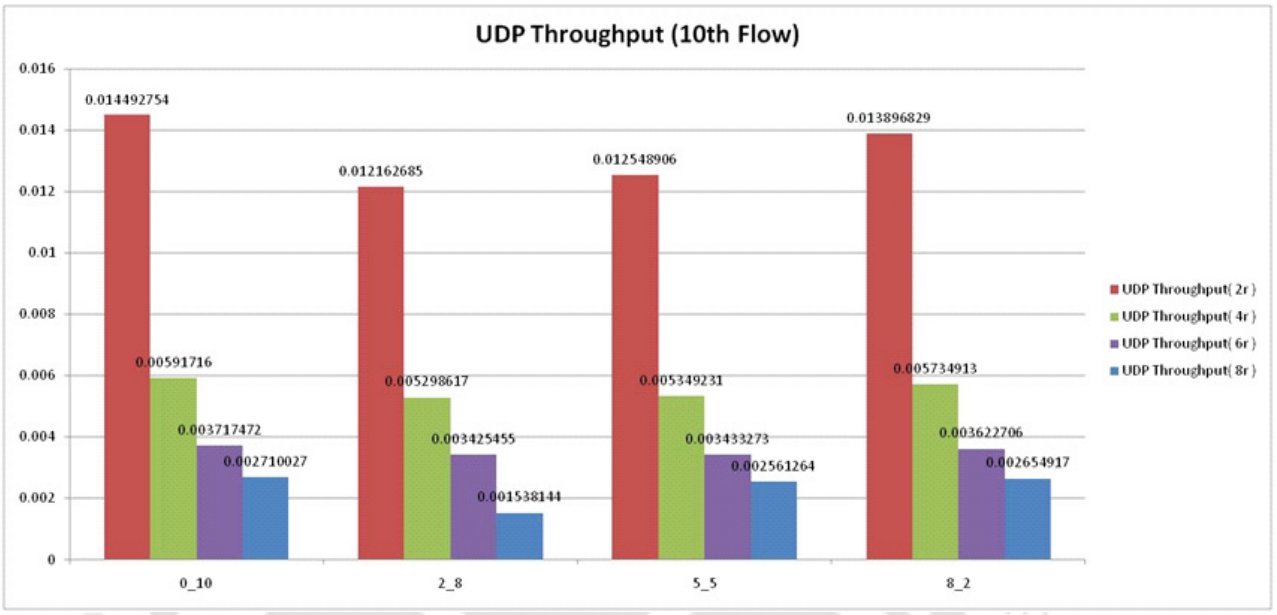


FIGURE 5.5: Simulation results for UDP throughput where X-axis indicates ratio of TCP:UDP and colors indicates number of routers and Y-axis indicates UDP throughput in heterogeneous case.

5.3 Conclusions

In this chapter, UDP throughput is algorithmically modeled probably for the first time in the presence of TCP. The algorithm is implemented based upon the data traces using measurement data from chapter 3 and validated using NS2 simulations. The results show clearly that UDP throughput degrades as TCP fraction of background traffic is increased. Even if the number of UDP flows is minimized to enhance the UDP throughput, the TCP congestion mechanism prevents UDP from achieving increased throughput due to its greedy approach. UDP throughput falls drastically as the number of hops increases. 50% degradation happens in throughput per two hops. If newer protocols like QUIC, SPDY, HTTP/2, RTMFP, MPRTP, DTLS, UTP, BitTorrent, UDT, SST

prefer UDP for transport evolution, then the inference of this chapter is that TCP and UDP are not friendly and in fact, increased fraction of TCP in the background traffic degrades UDP throughput is a valid concern. These are probably the reasons why UDP is not implemented on the Internet more in real-time applications, whereas TCP remains the most dominant protocol. This makes UDP a non-preferred protocol on the Internet in spite of its merits to better support delay-sensitive traffic. On Internet mostly UDP implementations are noticed only for a limited class of applications with low RTT requirements such as DNS, DHCP, and some signaling protocols in VOIP.

Chapter 6

Design and Performance Analysis of Link-by-Link Congestion Avoidance Algorithm (LbLCA)

In this chapter a new Link-by-link congestion avoidance algorithm (LbLCA) is proposed. The design philosophy is based on sizing considerations of router buffers. The delay performance of any congestion control protocol is directly related to the size of the router buffer. Therefore, buffer sizes are determined using the LbLCA algorithm and followed by the implementation of the proposed algorithm in NS2. The performance is compared with normal TCP (with AIMD congestion control) and the proposed LbLCA Algorithm. Next for commonly deployed linear and cross-network topologies, LbLCA and TCP are compared for performance (delay and packet delivery ratio). It is observed that LbLCA provides improved End-to-End Delay and Packet Delivery Ratio (PDR) performance, compared to the normal TCP based traffic flows, which suffer from excessive end-to-end delays, compromising on QoS parameters.

The chapter is organized as follows, Section 6.1 presents the design philosophy for the proposed LbLCA and design equations. In Section 6.2, buffer sizes are determined using LbLCA for different router buffers for some typical linear and cross configurations. In

Section 6.3 Performance evaluation of LbLCA is carried out using buffer sizes determined in Section 6.3. The NS2 simulations for the same linear and cross configurations are carried out. Results are compared of the TCP and LbLCA. In Section 6.4 Buffer Size Prediction using Multiple Linear Regression for Link-by-Link Congestion Avoidance Algorithm is carried out followed by conclusions.

6.1 Design philosophy for LbLCA algorithm

The Internet is a collection of communication links shared by many resources. A typical hierarchical network model has three parts, as shown in Figure 6.1. The first part, namely, access which grants end devices access to the network. The next part which is distributed or aggregation part combines the data received from access part devices before transmitting forward to the core part for routing to the final destination. The third part, i.e., the core part provides a path or route for datagram traversal. This part is typically controlled by service providers. If used in an enterprise private network, it is commonly called the backbone. Core part devices are routers and switches. The end-to-end argument principle of the Internet is followed and as a result, the intelligence and decision making is confined to the access part and the core part normally ensures fast and efficient routing.

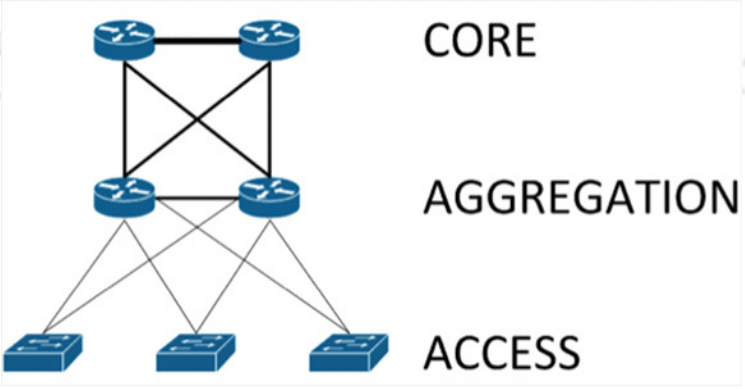


FIGURE 6.1: Hierarchical block diagram

As discussed in chapter 2, the existing AIMD based TCP protocol has some limitations. The TCP integrates the two functions i.e., flow and congestion control. The flow control is transport-related point-to-point phenomena and should be controlled by the receiver.

Congestion control, however, is a global phenomenon and can be controlled at the network layer itself. Further, it is desirable to delink the flow and congestion control and deal with congestion issues separately at the network layer, where congestion occurs. In this chapter, an attempt has been made to propose a new approach referred to as the Link-by-Link Congestion Avoidance LbLCA algorithm. LbLCA is the simple feedback-based protocol that tries to prevent congestion before it occurs. The algorithm is schematically shown in Figure 6.2 and has the following features:

1. The LbLCA algorithm is simple proactive in nature, and every router participates in the congestion avoidance process. From the feedback obtained, the router computes a sending rate that will not overwhelm its downstream neighbors and will thus ensure that congestion does not occur in the first place.

2. This algorithm has been proposed to be implemented in the core routers and at edge routers of the hierarchical model of Figure 6.1. The datagrams from different flows are aggregated at the port of the router. The output queue of the router is subjected to the feedback mechanism, before forwarding to the next hop. This implies that control is exercised at each and every router along the path of the traffic stream.

3. The edge router will decide and maintain the sending rates from its outgoing ports to the core network. The sending rate is decided by the router's buffer size and its outgoing link capacity.

4. No routing algorithms need to be modified for the implementation of LbLCA. Only outgoing sending rates are adjusted according to the feedback received from the downstream router. No new protocols are added/modified.

5. Link utilization is considered while deciding buffer size. Buffers have been optimally sized such that they never overflow and buffers always have some packets to send.

6. The router buffer feedback comes from a single hop i.e. from the immediate neighboring router(s). Therefore, LbLCA ensures a fairly quick response.

7. The LbLCA algorithm is a closed-loop control through explicit feedback about the state of the following router (i.e. the router buffer occupancy and the outgoing link(s)' sending rates).

8. Flow control and congestion control are segregated and are suggested to be implemented at transport/application and network layers respectively. LbLCA fully complies with the end-to-end argument design principle of the Internet. Explicit flow control depending on the need (partial, full, none) can be achieved either by UDP or application layer itself. The real-time control protocol (RTCP) can also be a possibly good option for achieving flow control.

As a result of LbLCA, the edge routers experience the backpressure of core routers. Further, the edge routers pass on this pressure to the routers in the access part. It is the access part network devices, which decide the mean sending rate values on a per-port basis. The buffer size of the edge router is determined by the value of the mean sending rate of the access network. Therefore, calculations of per-flow complexity get eliminated, which results in improved scalability. Estimations of buffer sizes for edge and core routers, for different link capacities, have been done in the following subsection. The buffer size directly impacts the delay related performance parameters. Larger the buffer sizes, higher will be the queuing delays; whereas smaller buffer size increases the packet loss probability.

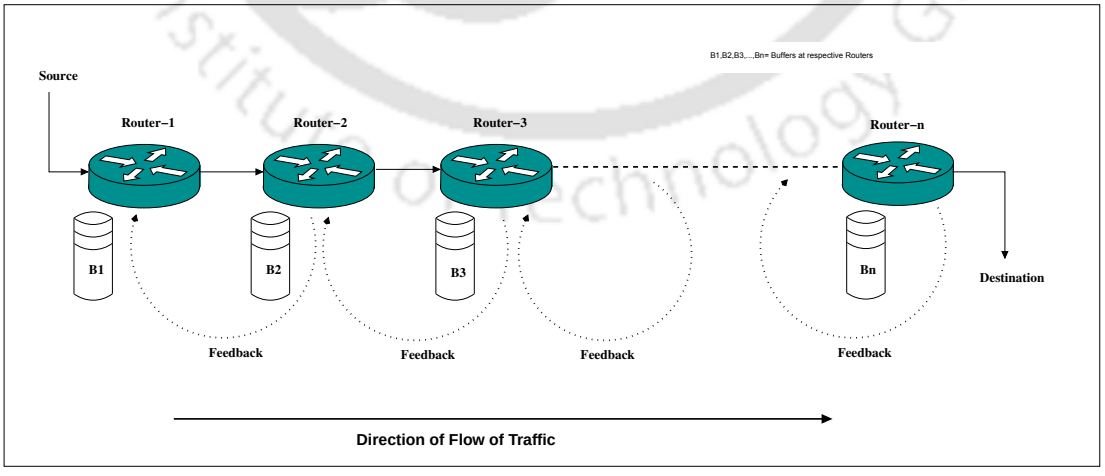


FIGURE 6.2: Link-by-Link configuration

6.1.1 Design of LbLCA algorithm

Figure 6.3 shows the typical simple network topology considered for a given source-destination pair. Let $R_i(1 \leq i \leq n)$ be the router of interest. Each router has multiple ports say $j(1 \leq j \leq m)$ with $c_1, c_2, c_3, ..., c_m$ link capacities, attached to the respective ports. Router ports can have different weights attached to them. In this thesis work, it is assumed that the link capacities considered are weights of ports. Each router has a buffer associated with it, shared by all ports. Let $\Theta(i,k)$ be the occupancy of the i^{th} router R_i at the k^{th} discrete instant. Let $R_{(i-1)}$ and $R_{(i+1)}$ be the upstream and downstream routers of the R_i^{th} router. Let $a(i,k,j)$, $s(i,k,j)$ and $ts(i,k,j)$ be the arrival rate, actual sending rate and target sending rate of the R_i^{th} router buffer, at the k^{th} instant and for port j of the router.

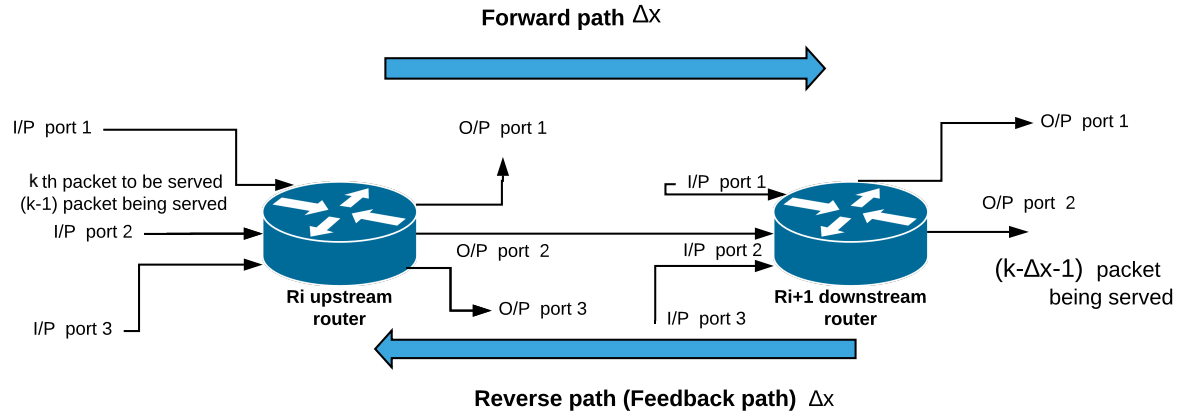


FIGURE 6.3: Typical network diagram for upstream and downstream routers

The design of the LbLCA is illustrated below with three steps:

Step.1) Calculating the occupancy of a buffer:

Let R_i be the router whose occupancy is to be estimated. Let $\Theta(i,k)$ be the occupancy of the R_i^{th} router at the k^{th} instant and Δx be the one way delay between R_i and R_{i+1} router. The assumption is that delay between feedback path is also Δx. Here d represents successive packet time difference equal to one unit of

time. Occupancy of the R_i^{th} router buffer at the k^{th} discrete instant is expressed as:

$$\Theta\,(i,k) = \min\,(Z, B_i) \tag{6.1}$$

$$Z = \max\left[\Theta\,(i,k-1) + d\left(\sum_{j=1}^{m} c_j \cdot a(i,k-1,j) - \sum_{j=1}^{m} c_j \cdot s(i,k-1,j)\right), 0\right] \tag{6.2}$$

The above equation implies that the occupancy of a router buffer at any instant is given by its occupancy at the previous instant plus what came in from its upstream neighbor minus what went out from the outgoing ports of the buffer. B_i is the maximum buffer size available.

Step.2) Calculating the target sending rate:

Every downstream router sends two components as feedback to its respective upstream router. The first component $Fb_{(a)}$ is its buffer occupancy. $Fb_{(a)}$ is the feedback component indicating the buffer occupancy of the downstream router in question as shown in Eq. (6.3) and the second component is $Fb_{(b)}$ as given in Eq. (6.4). It is the appropriately weighted sending rate at the outgoing port of the downstream router.

$$Fb_{(a)} = [\Theta\,(i+1, k-\Delta x-1)] \tag{6.3}$$

$$Fb_{(b)} = \left[\frac{\sum_{j=1}^{m} s\,(i+1, k-\Delta x-1)\,.c_j}{\sum_{j=1}^{m} c_j}\right] \tag{6.4}$$

$a(i,k,j)$, $s(i,k,j)$ and $ts(i,k,j)$ are the arrival rate, actual sending rate and target sending rate of the R_i^{th} router buffer, at the k^{th} instant and for port j of the router. This feedback is used to modify the sending rate of the corresponding output router port of R_i^{th} (upstream) router. To calculate the target sending rate, let Θ_i be the

buffer size of the i^{th} router. Therefore, the target sending rate of the R_i^{th} router at the k^{th} instant for the j^{th} port is shown in Eq. (6.5).

$$ts(i,k,j) = \begin{cases} \delta, & if\ \Theta(i,k-1) \le bl[Bi] \quad or \quad \Theta(i+1,k-\Delta x-1) \ge bu[Bi+1] \\ S(i,k-1,j) + \left\{ \frac{\left[bu^*[Bi+1] - Fb_{(a)}\right]}{d} - Fb_{(b)} \right\} & \\ & else\ if \quad \Theta(i+1,k-\Delta x-1) \ge 0.5[Bi+1] \\ S(i,k-1,j) + \left\{ \frac{\left[bl^*[Bi+1] - Fb_{(a)}\right]}{d} + Fb_{(b)} \right\} & \\ & else \quad \Theta(i+1,k-\Delta x-1) < 0.5[Bi+1] \end{cases} \tag{6.5}$$

Each router aims to make the occupancy of its downstream neighbor to around half of the maximum buffer size. This is done to provide an adequate range for input variations. If the occupancy exceeds a upper set-point [bu] of the downstream router, then there will be a finite probability of buffer overflow; therefore the sending rate of upstream router's typical port is fixed to preset value δ as per Eq. (6.5) above. At the same time, to avoid under-utilization of the buffer, a lower set-point [bl] for the upstream router is proposed. Therefore, the sending rate is again δ if the occupancy of the upstream router (here Ri) buffer falls below the lower set-point. The value of minimum sending rate [δ] is fixed at 5 % (0.05^*c_j) of the incoming arrival rate. In the rest cases, the correction factor is added to or subtracted from the sending rate of R_I^{th} router, depending on the downstream router feedback as indicated in Eq. (6.5).

Step.3) Selection of the actual sending rate:

The actual sending rate may differ from the target sending rate. The actual sending rate $s(i,k,j)$ of the R_i^{th} router at the k^{th} instant for the j^{th} port is given by,

$$s(i,k,j) = \min\left(ts(i,k,j),\ \delta, c_j\right) \tag{6.6}$$

For any edge router, the mean arrival rate of the access part of the network is decided locally in the access network as per requirement. Using the above set of LbLCA design equations, the optimum buffer sizes for different (core and edge) routers for required mean arrival rates are determined. The determination of buffer sizes is carried out using MATLAB environment simulation in the next section.

6.2 Determination of buffer sizes using LbLCA algorithm

LbLCA system has been implemented using the above three design equations for different configurations namely simple linear configuration and cross configurations involving different input and outputs. LbLCA is implemented at the network layer. The implementation of LbLCA is carried out in two steps:

1) Determination of buffer sizes of commonly used configurations using Matlab simulations.

2) Using buffer sizes incorporation of LbLCA in NS2 and simulations for performance evaluation.

When LbLCAis implemented, TCP congestion control is considered redundant. The assumption is made that datagram arrivals are Poisson distributed. Stochastic studies [103] have shown that the aggregation of traffic from a large number of flows (as can be expected at an optical core link) converges to be Poisson distributed for UDP. It is further assumed that UDP datagrams are on an average smaller in size than TCP packets and UDP traffic is much lesser (around 6 % to 8 %) of total traffic. The study showed that almost all UDP packets were around 200-byte packets, at the network layer and traffic follows Poisson distribution [17]. Therefore, traffic has been considered to be Poisson distributed at the inputs of each router in the network for buffer size estimations. Further, simulations were also carried out using the Hyperexpotential and Weibull distributions representing network traffic accounting for the self-similarity behavior observed in TCP flows. However, the results were very much similar to the situation using the Poisson distribution, except for a slightly larger variance of instantaneous buffer occupancies.

Secondly, the topologies selected for determining buffer sizes are the building blocks of the core network. A complex core network essentially can be constructed using either of the configurations used in the simulations.

6.2.1 Determination of the size of router buffers

Following are the steps carried out for every router to estimate buffer size.

1. Initialize buffers to half the capacity as mean size of buffer [c = 0.5 Bi].
2. Initialize the sending rate according to the mean arrival rate at the first buffer (λ_i).
3. Calculate the target sending rate using Eq. 6.5) after receiving feedback. Feedback includes current buffer occupancy of the appropriate weighted sending rates at the outgoing ports of the downstream router.
4. Data is transmitted to the downstream router and then occupancy of the R_i^{th} router changes according to equation Eq. (6.1).
5. Every router sends feedback to its upstream router as shown in Eq. (6.3) and Eq. (6.4). Then the process continues (step 3, step 4 and step 5).

In the following subsection sizing for router buffer is done. Then the minimum buffer size for various routers with respect to the link capacities and specified mean arrival rates at the input ports of the network are plotted. The values never exceed upper set-point and are never below lower set-point. Set-points (upper and lower) are selected as 80 % and 20% of the buffer size B of the router. Again, linear and cross configuration scenarios are chosen, which form the building blocks of the core and aggregation part of the hierarchical model.

6.2.2 Sizing of router buffer for linear configuration

Simple linear configuration, one with three routers only as shown in Figure 6.4, is simulated. The arrival rates are presumed to be Poisson distributed. Figure 6.5(a) and Figure 6.5(b) show minimum buffer size required for various values of link capacities for

the mean Poisson arrival rate as $\lambda = 170$ Mbps and $\lambda = 400$ Mbps respectively, as two cases are considered. The sending rate has been selected such that the buffer occupancy of downstream routers gets maintained between a lower limit (considered here 20%) and upper limit (considered here 80%) of the total buffer size. For linear configuration, all router buffers are of the same size and exhibit the same behavior as shown in Figure 6.5(b) and Figure 6.5(c). Table 6.1 shows the buffer sizes obtained from the Matlab simulations using LbLCA equations. These sizes are subsequently used in validation using NS2 simulations.

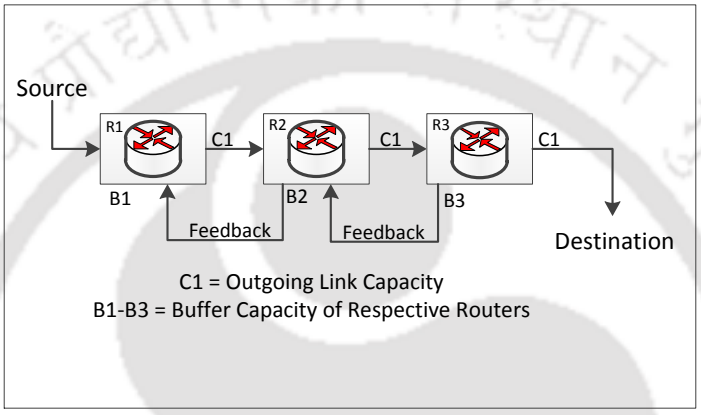
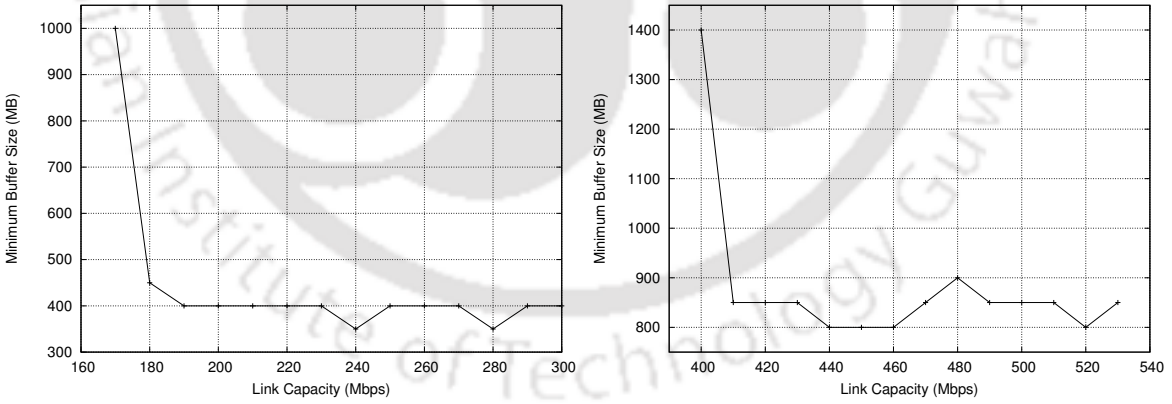


FIGURE 6.4: Router Linear configuration topology



(a) Behavior with $\lambda = 170$ (b) Behavior with $\lambda = 400$

FIGURE 6.5: Three router linear configuration and buffer behavior.

Observations for linear configurations

In a linear configuration, every router can be viewed as an M/M/1 queue (arrival and service processes verified at router input and output ports). Therefore, it is intuitive that mean arrival and mean departure rates should be maintained equal, so as to avoid

buffer overflow condition. When the mean arrival rate is more than the link capacity, packets keep accumulating in the buffer and an overflow occurs. Therefore, the outgoing link capacity should always be greater than the mean arrival rate. Buffer size stabilizes to a value for selected values of outgoing link capacity and the mean arrival rate (λ), beyond which the increase in buffer size only leads to increasing the queuing delay. It is interesting to note that for a definite capacity value of the links and the selected the mean arrival rate of the router input port, a typical minimum buffer size exists. The actual sending rate is determined by the occupancy of the router. So the n^{th} router will let out all the packets it has and they will be taken in by $(n+1)^{st}$ router, which itself will let out all the packets it has to the next downstream router. The same behavior will repeat at the next instant. The instantaneous buffer occupancies of buffers B1, B2, and B3 in linear three router configuration are shown in Figure 6.6, for $\lambda = 170$ Mbps with respect to discrete-time. Similar behavior holds true for six router and nine router configurations too, though not included here for space constraints. Similar simulations are done for all routers and results are shown in Table 6.1. Row 2 and Row 3 show the results for 2-input and 3-input cross configurations.

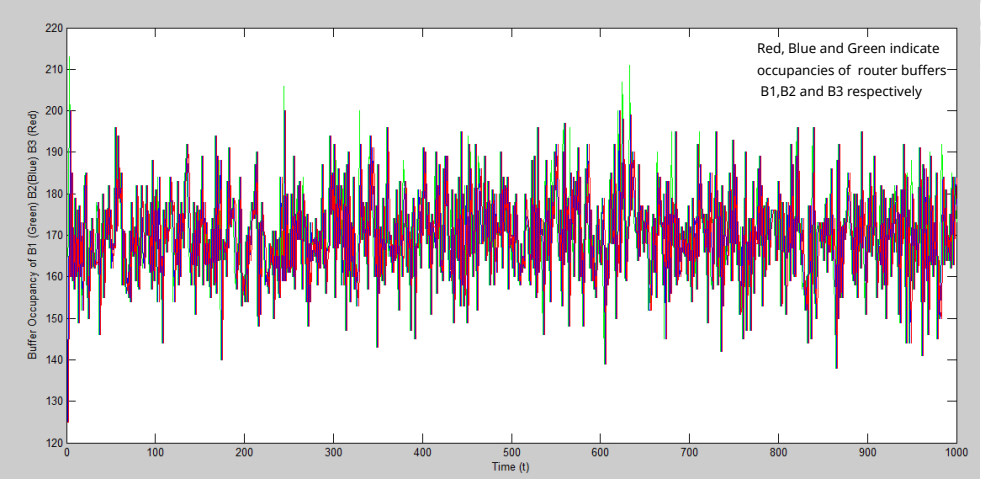


FIGURE 6.6: Instantaneous buffer occupancies for 3 routers linear configuration. Colors red, blue and green indicate instantaneous buffer occupancies of router buffers B1, B2, and B3 respectively, for $\lambda = 170$ Mbps

TABLE 6.1: Buffer sizes obtained from simulations

Sr.no.	Network topology	Mean arrival rates in Mbps	Buffer sizes in Mb
1.	Simple linear	a) $\lambda = 170$	B1=B2=B3=B4=400
		b) $\lambda = 400$	B1=B2=B3=B4=900
2.	2-input cross	a) $\lambda1 = 180$ and $\lambda2 = 70$	B1=425, B2=600 B3=170, B2=600
		b) $\lambda1$=150 and $\lambda2 = 200$	B1=390, B2=900 B3=525, B2=520
3.	3-input cross	a) $\lambda1 = 180$, $\lambda2 = 70$ and $\lambda3 = 140$	B1=810, B2=760, B3=535, B4=170, B5=1250, B6=560
		b) $\lambda1 = 600$ $\lambda2$=310 and $\lambda3 = 620$	B1=3590, B2=3000, B3=2455, B4=3180, B5=5470, B6=2480

6.2.3 Sizing of router buffers for 2-input and 3-input cross configuration

The cross-router configuration with two network topologies as shown in Figure 6.7 and Figure 6.9 (a) have been considered for simulations. For the 2-input cross configuration shown in Figure 6.7, the router buffer sizing has been done for router buffer B2, assuming the capacity of link C3. Figure 6.8(a) Figure and 6.8(b) show the variation of estimated buffer size with respect to different values of link capacities C3. The first variation Figure 6.8 (a) for specified mean arrival rates of $\lambda1 = 180$ Mbps at the input port of router buffer B1 and arrival rate of $\lambda2 = 70$ Mbps at the input port of router buffer B3. Other variation shown in Figure 6.8 (b) with different mean arrival rates of $\lambda1 = 150$ Mbps at the input port of router buffer B1 and arrival rate of $\lambda2 = 200$ Mbps at the input port of router buffer B3 is also carried out and recorded.

For the 3-input cross configuration shown in Figure 6.9(a), the router buffer sizing has been determined for router buffer B5, assuming the capacity of link C5. Figure 6.9(b) shows the variation of buffer size with respect to different values of link capacities C5, for specified mean arrival rates of $\lambda1 = 120$ Mbps at the input port of router buffer B1,

$\lambda 2 = 70$ Mbps at the input port of router buffer B3 and $\lambda 3 = 140$ Mbps at the input port of router buffer B6. Router port weights are considered while receiving feedback from downstream and providing feedback to upstream routers. To demonstrate the scalability of the proposed method and the assumption that the core routers have higher bandwidths say typically more than 500 Mb, one more scenario is added. The 3-input cross configuration with higher router bandwidth shown in Table 6.1 in the last row. Scaling up/down is the ability to scale by changing allocated resources. This is very much possible in LbLCA as the buffer is explicitly designed and LbLCA is non-partial to different flows as it works at layer 3.

Observations for cross configurations

As expected, the buffer size required for cross traffic routers is much higher than the buffer size required for routers in a linear configuration. For a given value of λ, if the outgoing link capacity increases, the minimum buffer size required increases to a certain maximum value after which it remains constant. It can be noted that if the outgoing link capacity is not considered, a larger number of packets can be sent out and higher buffer size is required to accommodate these packets. It can also be inferred that the size of the buffer required at each router depends only on outgoing link capacities.

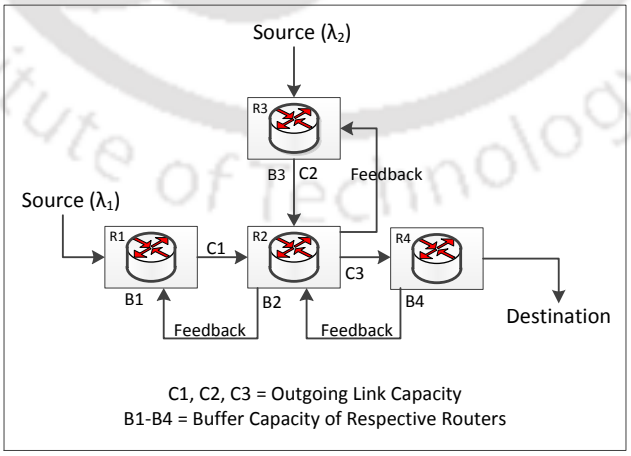
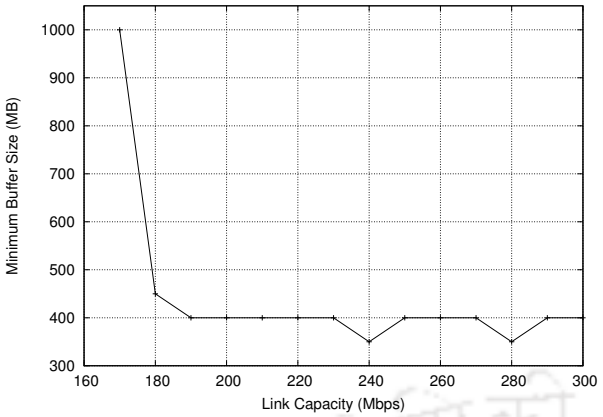
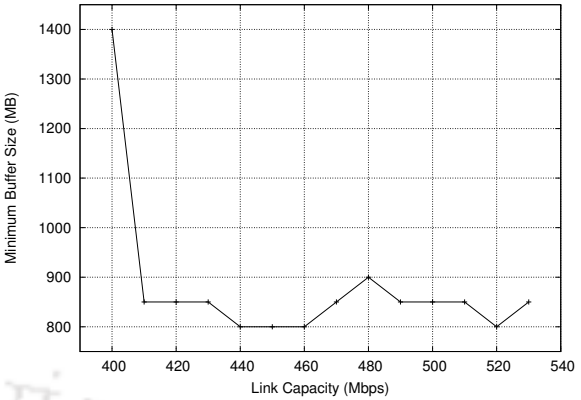


FIGURE 6.7: 2-input cross configuration topology

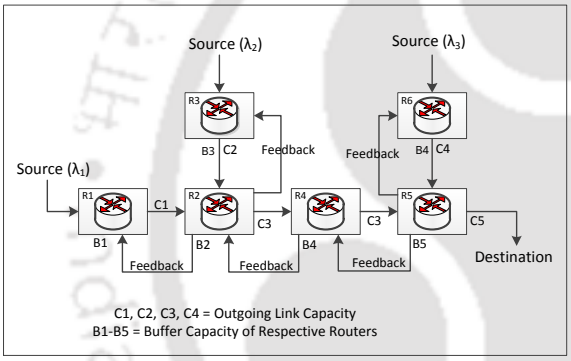


(a) Buffer (B2) behavior with $\lambda 1 = 180$ and $\lambda 2 = 70$

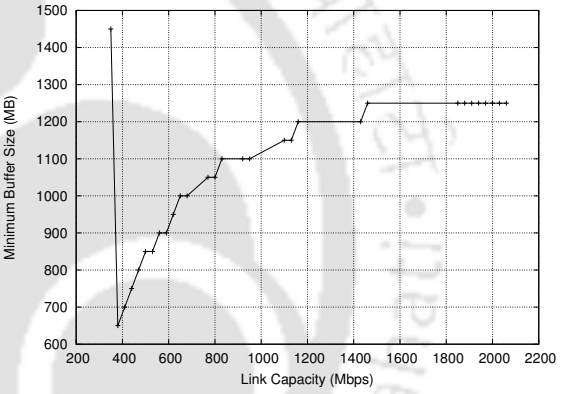


(b) Buffer (B2) behavior with $\lambda 1 = 150$ and $\lambda 2 = 200$

FIGURE 6.8: 2-input cross configuration buffer behavior graph



(a) 3-input cross configuration topology



(b) Buffer (B5) behavior with $\lambda 1 = 120$, $\lambda 2 = 70$ and $\lambda 3 = 140$

FIGURE 6.9: 3-input router cross configuration topology with buffer behavior graph

6.3 Performance evaluation of LbLCA

The primary purpose of NS2 simulations is to validate the router buffer sizes designed using LbLCA algorithm using design equations (Eq. (6.1), Eq. (6.2), Eq. (6.3), Eq. (6.4), Eq. (6.5) and Eq. (6.6)). Further, the simulations have been done using the buffer sizes of different routers, which are shown in Table. 6.1. Furthermore, LbLCA and TCP have been compared for performance parameters like Packet Delivery Ratio (PDR), End-to-end Network Delay and Efficacy, which has been defined as the ratio of PDR and end-to-end network delay. NS2 stack has been modified to implement the LbLCA equations. CBR, RTP, UDP and IP protocols are used. Feedback value is

dependent on the RTT and in simulation d = RTT. The value of d is to be carefully selected as too frequent and too late feedback messages impact the stability of the system. The number of frames and the processing time at the downstream router are accessed and are considered as the feedback and target sending rate are calculated. LbLCA is implemented at frame level in the MAC layer. Standard procedures have been used to modify the stack of NS2 by adding LbLCA.cc, LbLCA.h, and LbLCA_packet.h files. Modified stack has an existing routing algorithm and sending rates are modified at the output ports.

For traffic applications, CBR is subjected to the LbLCA congestion avoidance algorithm. The comparison of LbLCA with TCP traffic is carried out, and performance is recorded and the results are plotted. Readings from 15 iterations per configuration type are averaged out. The topologies considered are simple dumbell (linear)and cross networked with 2 and 3 inputs, as they are primary building blocks used to construct complex topologies representing core, distribution and access networks. The parameters used in the NS2 configuration are as given in Table 6.2. The LbLCA algorithm is compared with standard, widely deployed TCP type, i.e, Reno TCP (with the RED scheduling algorithm) has been used for simulations, the newer TCP types such as Cubic can also be considered.

TABLE 6.2: NS2 Simulation parameters

LAN Access-link delay	5 msec max
WAN Core-link delay	10 msec max
Bottleneck-link delay	50 ms max
Buffer size	Values obtained from section 6.2.2 (Table 6.1) are used with appropriate configuration, in Mbps
Queue limits	1000
Link Capacities	As per section 6.2 for appropriate configurations
Packet size	1000 bytes
Traffic type	CBR
Protocols	RTP,UDP,IP and LbLCA implemented in MAC layer (cross layer)
Simulation time	100 sec (to verify stability also)
Simulator version	NS-2 ver 2.35

Result: Implementation of *LbLCA* in NS2

Input : Plot all nodes and duplex links according to the scenario and apply agent and application traffic on scenario

Output: Delay and Packet Delivery Ratio in LbLCA scenario

Initialization

$R_i \leftarrow i^th$ upstream router

$R_{i+1} \leftarrow i+1^{st}$ downstream router

Bi $\leftarrow$ buffer size of i^th router in number of packets

$\theta \leftarrow$ Occupancy of router buffer in number of packets packets(0.5*Bi)

bl $\leftarrow$ 01*Bi (lower set – point)

bu $\leftarrow$ 0.9*Bi (upper set – point)

k $\leftarrow$ discete instant

d $\leftarrow$ successive packet time difference equal to one unit of time

$\delta \leftarrow$ minimum sending rate(5% of link capacity)

$S(i,j,k–1) \leftarrow$ sending rate of R_i^{th} router at j^{th} port at (k–1)st instant (packets/sec)

$Fb_a \leftarrow$ feedback occupancy of R_{i+1} at $(k-\Delta x-1)$

$Fb_b \leftarrow$ approriate weighted sumof sending rates at all the outgoing ports of the downstream router

$\Delta x \leftarrow$ delay difference between forward and reverse path, both are same for upstream and downstream

foreach *router node* **do**

  while *buffer designed using LbLCA* **do**

    read $\Theta(i)$ and $\Theta(i+1)$ after every Δx

    if $R_i(\theta) >$ bl *AND* $R_{i+1}(\theta) <$ bu **then**

      if $R_{i+1}(\theta) \geq 0.5$*B$(i+1)$ **then**

        Subtract Fb_b from $\left[S(i,j,k-1) + \frac{[bu*B(i+1)–Fba]}{d}\right]$

      else

        Add Fb_b to $\left[S(i,j,k-1) + \frac{[bl*B(i+1)–Fba]}{d}\right]$

    Send with δ rate

  end

end

Algorithm 2: Algorithm for implementation of *LbLCA* in NS2

6.3.1 Performance metrics

The following parameters have been determined from the measurement traces.

1. End-to-end Network Delay (msec): End-to-end delay or one-way delay (OWD) refers to the time taken for a packet to be transmitted across a network from source to destination. It includes transmission, processing, queuing and propagation delays.

2. Packet Delivery Ratio (PDR): The ratio of the number of packets, those are successfully delivered to a destination compared to the number of packets that have been sent out by the sender.

3. Throughput is the amount of data moved successfully from one place to another in a given time period (Mbps).

4. Jitter is defined as the variation in the network delay of consecutive received packets at the receiver due to network congestion, improper queuing, or configuration errors. The units of jitter measurement are milliseconds peak-to-peak (p-p).

To determine the effectiveness of the LbLCA algorithm, a new term Efficacy has been coined, using end-to-end network delay and PDR. Efficacy can be expressed as follows,

$$\text{Efficacy}(\eta) = \frac{\text{PDR (Packet Delivery Ratio)}}{\text{E}_\text{n} \text{ (Normalized End-to-end Network Delay)}} \tag{6.7}$$

$$\text{E}_\text{n} = \frac{\text{End-to-End delay in ms}}{\text{d}_\text{opt} \text{ in ms}} \tag{6.8}$$

Here d_{opt} represents End-to-End delay acceptable as per specification=100ms. where, $E_n \leq 1.0$ is acceptable. Maximum permissible $E_n = 1.5$ whereas $E_n > 5.0$ not acceptable as it violates ITU recommendations G.114 for one way delay [104]. Efficacy can be calculated in dB as per Eq. (6.9) below. For an ideal case, i.e, PDR of 100 % and $E_n \leq 1.0$ (close to ITU standard applicable one-way delay) the efficacy in dB is given by Eq. (6.9). Therefore $\eta > 0$ dB is a good performance. As the end-to-end delay=100 msec

is considered acceptable, the above value ($\eta = 0$ DB) becomes minimum acceptable or threshold value, below which the performance cannot be considered as acceptable.

$$\eta = 10\log_{10}(\eta) = 10\log_{10}(1.0) - 10\log_{10}(1.0) = 0 \text{ dB} \tag{6.9}$$

6.3.2 Simulations for linear configuration

This subsection deals with linear configuration. Figure 6.10 (a) and Figure 6.10 (b) depict the topology of simple 3-router linear configuration and its end-to-end network delay performance respectively. Figure 6.11 (a) and Figure 6.11 (b) represent the variation in PDR and Efficacy respectively. The mean arrival rate is λ = 170Mbps and buffer size used is B = 400Mb. The X-axis represents the outgoing link capacities of the router. Since the linear configuration is used, all the routers exhibit the same behavior. Similarly, Figure 6.12 depicts the results of a 3-router linear configuration with mean arrival rate λ = 400Mbps, maximum buffer size B = 900Mb.

Following are some important observations for linear network topology:

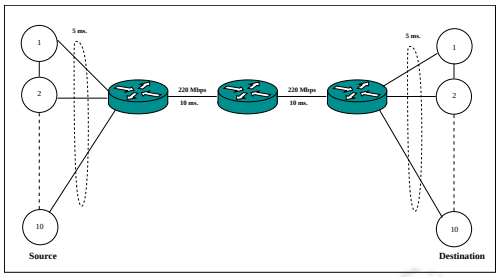
1. PDR is maximum for LbLCA (close to 100) configuration compared to TCP (about 80) as noticed from Figure 6.11(a)

2. Network delay is very minimum for LbLCA compared to TCP (350 to 550 ms) as seen from Figure 6.10(b).

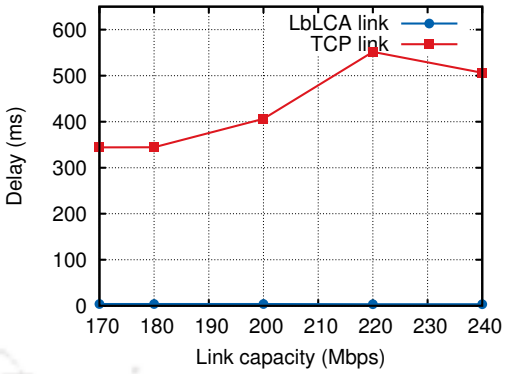
3. Efficacy is best (close to ideal) for LbLCA with CBR and low efficacy is recorded for TCP only configuration as observed from Figure 6.11(b).

4. Comparison clearly verifies that LbLCA gives the improved performance over normal TCP, for all the three performance parameters. When we compare efficacy, it can be seen that for mean arrival rates λ = 170 Mbps and λ = 400 Mbps as seen in Figure 6.11(b) and Figure 6.12(c). LbLCA gives improved performance over the conventional TCP protocol (with AIMD based congestion control) by almost 18 dB (approximately 63 times improved).

5. Similar trend is seen for throughput and network jitter refer Table 6.3

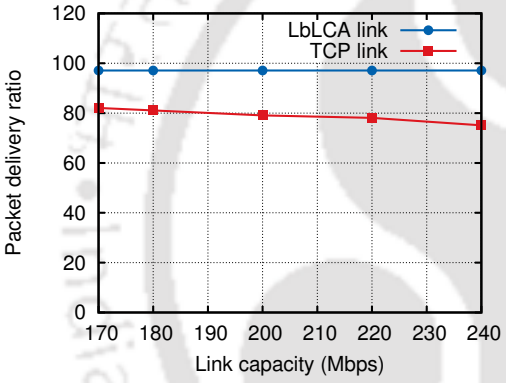


(a) Three router linear topology

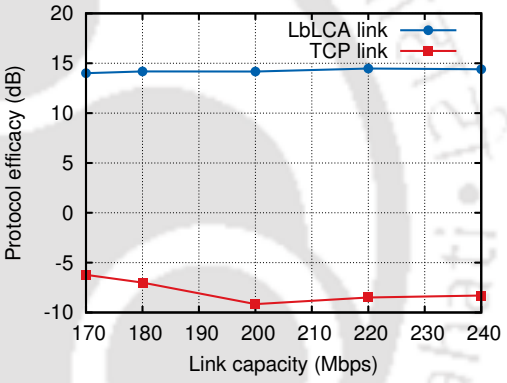


(b) Network delay for three router linear topology

FIGURE 6.10: Topology and Network delay for three router linear configuration with $\lambda = 170$

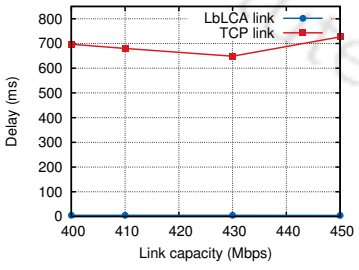


(a) Packet Delivery Ratio for three router linear topology

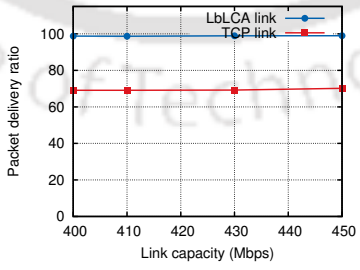


(b) Efficacy for three router linear topology

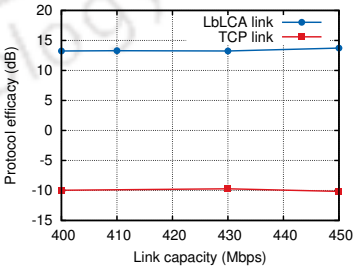
FIGURE 6.11: PDR and Efficacy for three router linear topology with $\lambda = 170$



(a) Network delay



(b) Packet Delivery Ratio



(c) Efficacy

FIGURE 6.12: Performance parameters for three router linear configuration with $\lambda = 400$

TABLE 6.3: Throughput and jitter performance for simple linear configuration when $\lambda 1 = 170$ and $\lambda 1 = 400$

Simple Linear configuration for λ =170 and λ =400 **Cap=Capacity(Mbps), Th=Throughput(Mbps) and Jitter (p-p) (msec)**									
Cap	**Th LbLCA**	**Th TCP**	**Jitter LbLCA**	**Jitter TCP**	**Cap**	**Th LbLCA**	**Th TCP**	**Jitter LbLCA**	**Jitter TCP**
170	90.69	80.69	16	960	**400**	91.59	79.59	7	1007
180	90.67	79.67	15	1003	**410**	91.57	80.62	6	993
200	90.66	78.66	17	983	**430**	91.56	81.56	8	997
220	90.69	75.69	18	998	**440**	95.58	81.58	7	988
240	90.68	73.68	18	978	**450**	91.59	81.65	8	1008

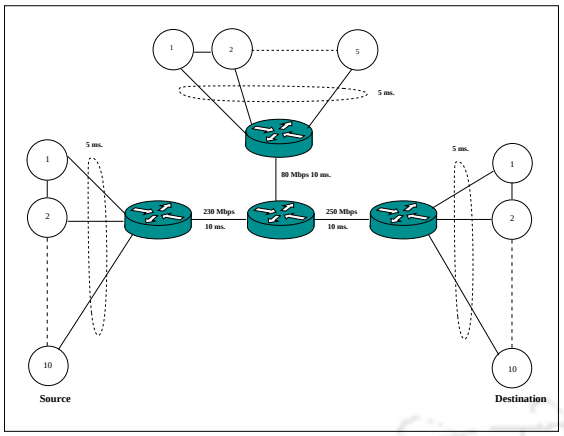
Similar performance measures for PDR, network delay, throughput, network jitter, and efficacy were observed for a six-router linear configuration.

6.3.3 Simulation for a 2-input Cross Configuration

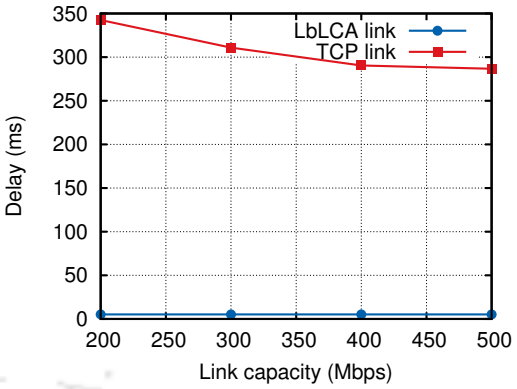
The 2-input and 3-input cross configurations have also been considered for validation of LbLCA algorithm using buffer sizes estimated in Chapter 6.2.2 (Table. 6.1).

TABLE 6.4: Throughput and jitter performance of 2-input cross configuration with $\lambda 1 = 180$ and $\lambda 1 = 70$

2-input Cross configuration				
Capacity (Mbps)	**Throughput LbLCA (Mbps)**	**Throughput TCP (Mbps)**	**Jitter (p-p) LbLCA (ms)**	**Jitter(p-p) TCP (ms)**
200	90.55	79.81	23	1298
300	90.58	79.91	19	1421
400	90.511	78.97	19	1256
500	90.52	79.78	22	1302

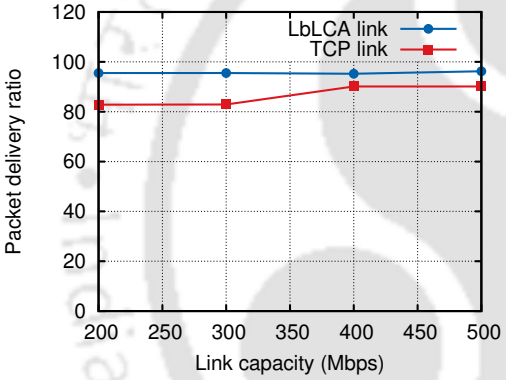


(a) 2-input cross topology

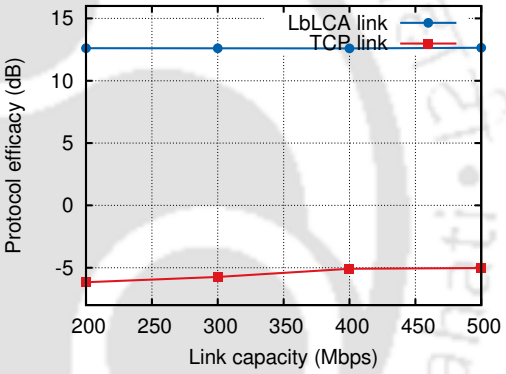


(b) End-to-End network delay for 2-input cross topology

FIGURE 6.13: Topology and End-to-End network delay for 2-input cross configuration with $\lambda 1 = 180$ and $\lambda 2 = 70$.



(a) Packet Delivery Ratio for 2-input cross configuration topology



(b) Protocol Efficacy for 2 input cross topology

FIGURE 6.14: PDR and Efficacy for 2-input cross topology when $\lambda 1 = 180$ and $\lambda 2 = 70$.

Significant observations for 2-input cross topology as shown in Figure 6.13 and Figure 6.14 are as follows:

1. Efficacy is maximum for LbLCA compared to TCP configuration refer Figure 6.14 (b). PDR is maximum for LbLCA compared to TCP. End-to-end network delay is the worst for TCP configuration refer Figure 6.13 (a).

2. Best values are obtained for LbLCA configurations. When the comparison is done for efficacy performance, as shown in Figure 6.14(b), LbLCA provides improved

TABLE 6.5

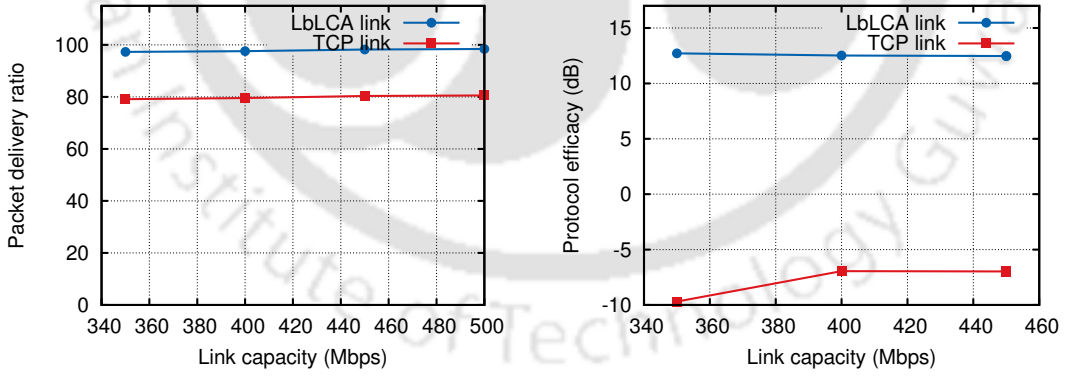
	LbLCA			TCP		
Capacity	**PDR**	**Delay in ms**	**Efficacy**	**PDR**	**Delay in ms**	**Efficacy**
340	0.973	5.21	12.712	0.791	382.5	-6.844
360	0.975	5.46	12.518	0.795	393.5	-6.942
380	0.982	5.58	12.454	0.803	400.1	-6.971
400	0.984	5.58	12.464	0.805	405.6	-7.019
420	0.985	5.6	12.460	0.806	406.3	-7.027
440	.0985	5.62	12.462	0.806	408.6	-7.029

compared to TCP; almost 17-18 dB (approximately 50 times) improved. This indicates that the LbLCA algorithm as proposed yields a higher level of performance compared to normal TCP.

3. A similar trend is seen for throughput and network jitter. Low network jitter values clearly suggest that delay-sensitive applications will have improved QoS in LbLCA.

6.3.4 3-input cross configuration

Figure 6.3.4(a) shows a 3-input cross router configuration. The performance trend is similar to the performance of 2-input cross configuration .



(a) Packet Delivery Ratio for a 3-input cross configuration topology

(b) Protocol Efficacy for a 3-input cross configuration topology

FIGURE 6.15: PDR and Efficacy for a 3-input cross configuration with $\lambda 1 = 180$, $\lambda 1 = 70$ and $\lambda 2 = 140$.

The significant observations for 3-input cross router configuration are:

1. Efficacy is maximum for LbLCA around 13dB compared to refer (around TCP -10dB) Figure 6.15(b).

TABLE 6.6: PDR and delay performance of 3-input cross configuration with $\lambda 1 = 600$, $\lambda 1 = 310$ and $\lambda 2 = 620$

3-input cross configuration				
Capacity (Mbps)	**LbLCA PDR**	**LbLCA Delay (msec)**	**TCP PDR**	**TCP Delay (msec)**
1000	97.23	6.83	78.92	520
1500	98.06	6.72	78.56	583
2000	97.63	6.64	77.03	567
2500	97.56	6.02	77.63	517

TABLE 6.7: PDR and delay performance of 3-input cross configuration with (a) $\lambda 1 = 180$, $\lambda 1 = 70$ and $\lambda 2 = 140$ (b)$\lambda 1 = 600$, $\lambda 1 = 310$ and $\lambda 2 = 620$

3-input Cross configuration [Cap= Capacity (Mbps),Th=throughput (Mbps) and Jitter(p-p) msec]									
Cap	**Th LbLCA**	**Th TCP**	**Jitter LbLCA**	**Jitter TCP**	**Cap**	**Th LbLCA**	**Th TCP**	**Jitter LbLCA**	**Jitter TCP**
350	92.11	75.39	22	1300	**1000**	91.59	78.59	19	2200
400	92.56	75.96	18	1198	**1500**	91.23	79.20	16	2183
450	93.23	76.61	20	1247	**2000**	90.56	80.01	17	1997
500	93.56	76.63	19	1088	**2500**	92.23	78.63	18	2198

2. For PDR, best values are obtained for LbLCA compared to PDR values obtained for TCP configuration. Regarding end-to-end network delay, minimum values are obtained for LbLCA.

3. Regarding end-to-end network delay, minimum values are obtained for LbLCA. Maximum delay values are observed for TCP configuration refer Figure 6.3.4(b).

4. It is clearly observed that improved performance is obtained for LbLCA. In particular, from Figure 6.15(b), it is evident that LbLCA provides efficacy of 13 dB compared to the performance of almost -7 dB given by TCP . This means the LbLCA algorithm provides 100 times improved performance compared to the TCP protocol. The point to be noted is that TCP has two random components while calculating end-to-end delay; first is queuing delay and second is number of retransmissions in TCP. In LbLCA, these issues are taken care of; firstly, minimum queuing delay due to proper buffer sizes and secondly hardly any retranmissions are required. This suggests that LbLCA with proper buffer size provides optimum performance with maximum PDR and minimum possible end-to-end network delay. This is also due to the fact that in LbLCA we are trying to avoid congestion at the network layer itself where congestion occurs.

5) Table 6.6 and Table 6.7 indicate throughput, end-to-end delay, PDR and jitter for both the protocols. Performance trend remains the same.

Chapter 7

Conclusions and Suggestions for Future Work

THE revolutionary Internet is huge success based on best effort delivery services. The connection-less services of the Internet provide no guarantees regarding delivery. The flexibility, scalability and end-to-end argument argument were the primary design requirements of the Internet and are addressed in the architecture of the Internet. Today's conflicting goals of security, portability and QoS of delay sensitive applications pose challenges to Internet. TCP is the dominant and widely used protocol on Internet and Internet performance is closely linked TCP. Performance of TCP is associated with the congestion control mechanism of TCP which also integrates flow control. TCP protocol is assumed to be fair for all flows. The datagrams at the router generated by the transport level flows contend for resources at the router. The datagrams at the router are from a number of flows, which constitute a stochastic process. These flow impact and get impacted by fellow flows. This thesis work quantitatively analyses the effect of multiplexing of TCP flows on the performance of the Internet communications. For analysing the delay sensitivity the focus is the queuing delay, as the rest components of network delay like transmission delay, propagation delay, switching delay etc. are deterministic. The uncertainty comes from queuing delay. The first contribution of this thesis is the

model proposed for the arrival and service processes for the datagrams at the router which are constituted by the multiplexed TCP flows. The proposed model accounts for different fractions of TCP and UDP along with the arrival rates of TCP and UDP. The proposed model also considers datagram sizes of TCP and UDP flows. Further, statistical characterization of real time traffic is carried out. Armed with arrival and service process models and statistical characterization of real time traffic mean queuing delay, average instantaneous delay and jitter for datagrams of TCP tagged flow is carried out. The analytical results are intriguing. The mean delay of datagrams worsens by 400 % for TCP and around 30 % for UDP when background traffic has a maximum fraction of TCP. Secondly, average instantaneous delay and jitter for datagrams of TCP tagged flow are substantially larger as compared to those of UDP tagged flow when background traffic has a maximum fraction of TCP. Therefore, clear inference is that the delay performance values are dependent directly on the fraction of TCP traffic in the background. This is true for all TCP flavors like Reno, Cubic, and Compound even after the use of scheduling mechanism (RED). The major conclusion is that the interaction of TCP with fellow TCP and/or UDP impact is adverse for delay performance. This affects all applications, especially the QoS for real-time applications. The reason can be attributed to queuing delay at the routers and the number of retransmissions due to TCP congestion control mechanism. The number of transmissions are random in nature. As fraction of TCP in the background traffic increases beyond 80 %, this degradation would become quite severe and would compromise the ITU recommendations. The major reasons are buffer dynamics at the congested router, TCP congestion control mechanism and retransmissions in TCP. Another important factor is the variable segment size of TCP. The inference was validated by extensive NS2 simulations. Solution could be possibly, one could look at exploring UDP based congestion control which may minimize the queuing delays, particularly for real-time applications. Some newer protocols like Google's Quick UDP Internet connections (QUIC) protocol, SPDY,Adobe's Real Time Media Flow Protocol, Multipath Real Time Transport Protocol MPRTP, μTorrent Transport Protocol, Bit-Torrent's UDT, SST, NEAT are some UDP newer protocols used in the Internet. The RFC 8085 mandates the use of some congestion mechanism for use of UDP.

TCP flows are not friendly with each other in-fact hurt each others for the delay related

performance. Therefore, next issue to be analysed in this thesis work is whether TCP flows are friendly with UDP. UDP throughput is algorithmically modeled for the first time in the presence of TCP in this thesis work. The algorithm is implemented based on the data traces using measurement data and validated using NS2 simulations. The results show clearly that UDP throughput degrades as TCP fraction of background traffic is increased. Even if the number of UDP flows is minimized to enhance the UDP throughput, the TCP congestion mechanism prevents UDP from achieving increased throughput due to its greedy approach. UDP throughput falls drastically as the number of hops increases. 50% degradation happens in throughput per two hops. The next inference of this thesis work is that the TCP and The UDP flows are not friendly and in fact, increased fraction of TCP in the background traffic degrades UDP throughput of the flow in consideration is a valid concern. These are probably the reasons why UDP is not implemented on the Internet more in real-time applications, whereas TCP remains the most dominant protocol. This makes UDP a non-preferred protocol on the Internet in spite of its merits to better support delay-sensitive traffic. On the Internet, mostly UDP implementations are seen only for a limited class of applications with low RTT requirements such as DNS, DHCP, and some signaling protocols.

After concluding on the delay friendliness of TCP flows amongst themselves and with UDP next issue was to contribute to the solution. Coming with a new TCP variant was not the solution, as any TCP would incorporate same TCP Slow Start and Additive Increase Multiplicative Decreases congestion control mechanism and the behavior would yield similar results of TCP delay friendliness. Alternatively, one can explore the congestion avoidance mechanism at the network layer. The queuing delay is more for end-to-end error checking compared to hop-by hop error checking mechanism Shailendra *et al.* [46]. Therefore, a mechanism to avoid the congestion before it happens is proposed. This, however, may also necessitate the optimum sizing of buffers in routers and helps in minimizing the effect of queuing delay on the Internet. The idea of implementing congestion avoidance at the network layer and delinking congestion avoidance with flow control is intuitive as congestion happens in the network whereas flow control is point-to-point phenomena at the transport layer.

The next contribution of this thesis work is the proposal of new congestion avoidance

phenomena at the network layer as an alternative to existing TCP. The LbLCA algorithm is implemented in the core routers and edge routers in a hierarchical network with incremental change on the Internet. It is very interesting to note that there exists a minimum required buffer size at a router, which prevents the overflow of packets thus minimizing the packet losses and optimizing the network delay. For a given mean arrival rate, the buffer size of the router is independent of the number of flows passing through it and depends only upon the outgoing link capacities. With proper buffer size, the LbLCA ensures the minimum queuing delay and therefore, minimizes end-to-end delay for the applications with best-effort delivery.

Buffer sizing in the Internet routers is a significant issue as it directly impacts the performance of the Internet. Today, buffer sizes are designed considering the TCP congestion control mechanism. Although none of the buffer rules is preferred and no TCP variant is widely accepted, LbLCA provides a better alternative as a congestion avoidance mechanism with improved performance as compared to TCP. Linear multiple regression provides a closed form of the equation for the buffer size design. Linear multiple regression for a typical cross traffic example proves it clearly. The closed form of regression equation can be used with constraints on outgoing link capacities of outer ports to optimize the dependent variable, the buffer size. An interesting contribution of this thesis work is the prediction of buffer sizes for LbLCA routers using linear multiple regression. The LbLCA is implemented using predicted buffer sizes by linear multiple regression in chapter 6. The performance of LbLCA and TCP are compared. QoS is improved in LbLCA compared to TCP and issue of retransmissions is minimized. A new parameter namely, efficacy is defined as a ratio of packet delivery ratio and network delay. Efficacy of LbLCA is 50 times better than TCP as seen in the implemented results.

LbLCA works at the network layer and can facilitate multicasting. Explicit flow control depending on the need (partial, full, none) can be achieved by the application layer itself. The real-time control protocol (RTCP) can also be a possibly good option for achieving flow control. LbLCA can be implemented at the network layer with minor incremental modifications. LbLCA algorithm provides better performance for all applications and protocols above the network layer. LbLCA implemented at the edge routers induces pressure on sending rate for access networks. Access networks, then have to regulate

their outgoing rates as the size of the buffer of the edge router is determined following the mean value of the access network sending rate and edge routers outgoing link capacities. In this thesis work, it is assumed that access, edge and core networks act as a team to maximize total network utility. With an exception, then, the study of access network's selfish goals coinciding with system-wide goals can be pursued as future work. Further, the use of experimental Internet testbeds for verifying LbLCA on a larger scale can also be followed. Therefore, the results of the Internet testbed for LbLCA can be considered an extension in future work.

Appendix A

Predicted Values with Predicted Limits of Individuals in the linear multiple regression results

A.1 Predicted Values with Predicted Limits of Individuals

The following tables shows the actual and predicted values of buffer size B5, the dependent variable from multiple linear regression results. Standard error of predicted buffer size for each row is also presented for 95% confidence interval.

TABLE A.1: Predicted Values with Predicted Limits of Individuals

Row	Actual B5 (Mb)	Predicted B5 (Mb)	Standard Error	95% Lower Pred. Limit of Individual	95% Upper Pred. Limit of Individual
1	520	692.8054	52.08715	588.4204	797.1904
2	560	703.9409	51.77668	600.1781	807.7037
3	600	705.78	51.91505	601.7399	809.82
4	640	719.2396	51.65981	615.711	822.7681
5	680	718.7545	51.75451	615.0361	822.4728
6	680	734.5382	51.66306	631.0031	838.0733
7	720	731.7291	51.60563	628.3091	835.149
8	720	749.8369	51.7864	646.0546	853.6191
9	760	744.7036	51.46852	641.5584	847.8488

Table A.1 continued from previous page

Row	Actual B5(MB)	Predicted B5(MB)	Standard Error	95% Lower Pred. Limit of Individual	95% Upper Pred. Limit of Individual
10	760	755.8391	51.21004	653.2119	858.4663
11	800	757.6782	51.34328	654.7839	860.5724
12	800	771.1378	51.15663	668.6176	873.658
13	800	770.6527	51.22999	667.9855	873.3199
14	800	786.4364	51.22456	683.7801	889.0928
15	840	783.6273	51.12873	681.163	886.0916
16	840	801.7351	51.41338	698.7004	904.7698
17	840	796.6018	51.03957	694.3162	898.8874
18	840	807.7373	50.83506	705.8616	909.6131
19	880	809.5764	50.96258	707.4451	911.7076
20	880	827.6842	51.29934	724.8781	930.4904
21	880	822.551	50.8978	720.5494	924.5524
22	880	836.0106	50.79944	734.2062	937.8149
23	880	835.5255	50.84529	733.6293	937.4218
24	880	830.3922	52.9188	724.3406	936.4438
25	880	864.7688	52.09401	760.3701	969.1675
26	920	854.9873	50.78962	753.2026	956.772
27	920	880.0674	52.67424	774.5059	985.629
28	920	867.9619	50.76792	766.2207	969.7031
29	920	881.4215	50.73233	779.7516	983.0914
30	920	880.9365	50.75859	779.2139	982.6589
31	920	894.3961	50.74099	792.7088	996.0833

Table A.1 continued from previous page

Row	Actual B5(MB)	Predicted B5(MB)	Standard Error	95% Lower Pred. Limit of Individual	95% Upper Pred. Limit of Individual
32	920	893.911	50.76163	792.1824	995.6396
34	920	906.8856	50.77703	805.1261	1008.645
35	960	931.9657	52.80709	826.1379	1037.793
36	960	919.8601	50.80479	818.045	1021.675
37	960	930.9956	50.73307	829.3242	1032.667
38	960	932.8347	50.84488	830.9392	1034.73
39	960	948.6183	51.11225	846.1871	1051.05
40	960	945.8092	50.89728	843.8088	1047.81
41	960	963.917	51.50272	860.7032	1067.131
42	960	958.7838	50.96194	856.6537	1060.914
43	960	979.2157	52.00982	874.9857	1083.446
44	960	971.7583	51.03883	869.4742	1074.042
45	960	999.1625	53.9365	891.0714	1107.254
46	960	984.7328	51.12788	882.2703	1087.195
47	1000	1004.195	51.28412	901.419	1106.97
48	1000	1010.682	51.34222	907.7899	1113.574
49	1000	1017.169	51.40329	914.1547	1120.184
50	1000	1010.682	51.46735	907.7899	1113.574
51	1000	1017.169	51.40329	914.1547	1120.184
52	1000	1023.657	51.46735	920.5137	1126.799
53	1000	1030.144	51.53437	917.0087	1133.421
54	1000	1036.631	51.60435	933.2137	1140.048

Table A.1 continued from previous page

Row	Actual B5(MB)	Predicted B5(MB)	Standard Error	95% Lower Pred. Limit of Individual	95% Upper Pred. Limit of Individual
55	1000	1026.85	54.80962	917.0087	1136.691
56	1000	1036.631	1036.631	933.2137	1140.048
57	1000	1026.85	54.80962	962.4033	1173.024
58	1000	1067.714	51.91356	958.5431	1166.617

Bibliography

[1] D. P. Bertsekas, R. G. Gallager, and P. Humblet, *Data networks*. Prentice-Hall International New Jersey, 1992, vol. 2.

[2] R. Srikant and L. Ying, *Communication networks: an optimization, control, and stochastic networks perspective*. Cambridge University Press, 2013.

[3] A. Mondal, I. Trestian, Z. Qin, and A. Kuzmanovic, "P2p as a cdn: A new service model for file sharing," *Computer Networks*, vol. 56, no. 14, pp. 3233–3246, 2012.

[4] B. Briscoe, A. Brunstrom, A. Petlund, D. Hayes, D. Ros, J. Tsang, S. Gjessing, G. Fairhurst, C. Griwodz, and M. Welzl, "Reducing internet latency: A survey of techniques and their merits," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 3, pp. 2149–2196, 2014.

[5] A. Csoma, L. Toka, and A. Gulyás, "On lower estimating internet queuing delay," in *2015 38th International Conference on Telecommunications and Signal Processing (TSP)*. IEEE, 2015, pp. 299–303.

[6] Y. G. Gavgani, "Buffer sizing in internet routers," Ph.D. dissertation, Stanford University, 2007.

[7] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson, "On the self-similar nature of ethernet traffic (extended version)," *IEEE/ACM Transactions on networking*, vol. 2, no. 1, pp. 1–15, 1994.

[8] W. Willinger, M. S. Taqqu, R. Sherman, and D. V. Wilson, "Self-similarity through high-variability: statistical analysis of ethernet lan traffic at the source level," *IEEE/ACM Transactions on Networking (ToN)*, vol. 5, no. 1, pp. 71–86, 1997.

[9] M. Calzarossa and G. Serazzi, "Workload characterization: a survey," *Proceedings of the IEEE*, vol. 81, no. 8, pp. 1136–1150, 1993.

[10] D. Rossi, L. Muscariello, and M. Mellia, "On the properties of tcp flow arrival process," in *Communications, 2004 IEEE International Conference on*, vol. 4. IEEE, 2004, pp. 2153–2157.

[11] A. B. Downey, "Evidence for long-tailed distributions in the internet," in *Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement*. ACM, 2001, pp. 229–241.

[12] M. E. Crovella and A. Bestavros, "Self-similarity in world wide web traffic: evidence and possible causes," *IEEE/ACM Transactions on networking*, vol. 5, no. 6, pp. 835–846, 1997.

[13] M. Mirza, J. Sommers, P. Barford, and X. Zhu, "A machine learning approach to tcp throughput prediction," *IEEE/ACM Transactions on Networking (TON)*, vol. 18, no. 4, pp. 1026–1039, 2010.

[14] G. Raina, D. Towsley, and D. Wischik, "Part ii: Control theory for buffer sizing," *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 3, pp. 79–82, 2005.

[15] D. Wischik and N. McKeown, "Part i: Buffer sizes for core routers," *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 3, pp. 75–78, 2005.

[16] M. Enachescu, Y. Ganjali, A. Goel, N. McKeown, and T. Roughgarden, "Routers with very small buffers." in *INFOCOM*, 2006.

[17] G. Rouskas, A. Vishwanath, and V. Sivaraman, "Anomalous loss performance for mixed real-time and tcp traffic in routers with very small buffers," *IEEE/ACM Transactions on Networking*, vol. 19, no. 4, 2011.

[18] T.-Y. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson, "A buffer-based approach to rate adaptation: Evidence from a large video streaming service," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 4, pp. 187–198, 2015.

[19] I. Lokshina, "Study on estimating probabilities of buffer overflow in high-speed communication networks," *Telecommunication Systems*, vol. 62, no. 2, pp. 289–302, 2016.

[20] P. Imputato and S. Avallone, "An analysis of the impact of network device buffers on packet schedulers through experiments and simulations," *Simulation Modelling Practice and Theory*, vol. 80, pp. 1–18, 2018.

[21] T. Hoeiland-Joergensen, P. McKenney, D. Taht, J. Gettys, and E. Dumazet, "The flow queue codel packet scheduler and active queue management algorithm," in *RFC 8290*, 2018.

[22] A. G. Araldo and D. Rossi, "Dissecting bufferbloat: Measurement and per-application breakdown of queueing delay," in *Proceedings of the 2013 workshop on Student workhop*. ACM, 2013, pp. 53–56.

[23] A. Afanasyev, N. Tilley, P. Reiher, and L. Kleinrock, "Host-to-host congestion control for tcp," *IEEE Communications surveys & tutorials*, vol. 12, no. 3, pp. 304–342, 2010.

[24] C. Lochert, B. Scheuermann, and M. Mauve, "A survey on congestion control for mobile ad hoc networks," *Wireless Communications and Mobile Computing*, vol. 7, no. 5, pp. 655–676, 2007.

[25] S.-u. Lar and X. Liao, "An initiative for a classified bibliography on tcp/ip congestion control," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 126–133, 2013.

[26] M. Li, A. Lukyanenko, S. Tarkoma, Y. Cui, and A. Ylä-Jääski, "Tolerating path heterogeneity in multipath tcp with bounded receive buffers," *Computer Networks*, vol. 64, pp. 1–14, 2014.

[27] P. Yang, J. Shao, W. Luo, L. Xu, J. Deogun, and Y. Lu, "Tcp congestion avoidance algorithm identification," *IEEE/ACM Transactions on Networking (TON)*, vol. 22, no. 4, pp. 1311–1324, 2014.

[28] M. A. Alrshah, M. Othman, B. Ali, and Z. M. Hanapi, "Comparative study of high-speed linux tcp variants over high-bdp networks," *Journal of Network and Computer Applications*, vol. 43, pp. 66–75, 2014.

[29] J. Wang, J. Wen, C. Li, Z. Xiong, and Y. Han, "Dc-vegas: a delay-based tcp congestion control algorithm for datacenter applications," *Journal of Network and Computer Applications*, vol. 53, pp. 103–114, 2015.

[30] S. Patel and K. Rani, "Comparative performance analysis of tcp-based congestion control algorithms," *International Journal of Communication Networks and Distributed Systems*, vol. 17, no. 1, pp. 61–75, 2016.

[31] S. K. Bisoy and P. K. Pattnaik, "Fairness between tcp reno and tcp vegas in wired and wireless network," *International Journal of Computational Systems Engineering*, vol. 3, no. 1-2, pp. 14–26, 2017.

[32] V. Kushwaha and R. Gupta, "Congestion control for high-speed wired network: A systematic literature review," *Journal of Network and Computer Applications*, vol. 45, pp. 62–78, 2014.

[33] C. Xu, Z. Li, L. Zhong, H. Zhang, and G.-M. Muntean, "Cmt-nc: improving the concurrent multipath transfer performance using network coding in wireless networks," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 3, pp. 1735–1751, 2016.

[34] R. N. Yadav and K. Sharma, "On efficient data distribution in multipath tcp," *International Journal of Communication Networks and Distributed Systems*, vol. 19, no. 3, pp. 237–256, 2017.

[35] J. Wu, B. Cheng, C. Yuen, Y. Shang, and J. Chen, "Distortion-aware concurrent multipath transfer for mobile video streaming in heterogeneous wireless networks," *IEEE Transactions on Mobile Computing*, vol. 14, no. 4, pp. 688–701, 2015.

[36] L. He and H. Zhou, "Robust lyapunov–krasovskii based design for explicit control protocol against heterogeneous delays," *Telecommunication Systems*, vol. 66, no. 3, pp. 377–392, 2017.

[37] J. Wu, C. Yuen, B. Cheng, Y. Yang, M. Wang, and J. Chen, "Bandwidth-efficient multipath transport protocol for quality-guaranteed real-time video over heterogeneous wireless networks," *IEEE Transactions on Communications*, vol. 64, no. 6, pp. 2477–2493, 2016.

[38] J. Wu, C. Yuen, M. Wang, and J.-L. Chen, "Content-aware concurrent multipath transfer for high-definition video streaming over heterogeneous wireless networks," *IEEE Transactions on Parallel & Distributed Systems*, no. 1, pp. 1–1, 2016.

[39] G. Carofiglio, M. Gallo, and L. Muscariello, "Optimal multipath congestion control and request forwarding in information-centric networks: Protocol design and experimentation," *Computer Networks*, vol. 110, pp. 104–117, 2016.

[40] H. Huang, X. Sun, and M. Hu, "Steady-state throughput discrete model of multipath tcp based on linked increase algorithm," *International Journal of Communication Networks and Distributed Systems*, vol. 21, no. 4, pp. 528–546, 2018.

[41] J. A. Arokkiam, X. Wu, K. N. Brown, and C. J. Sreenan, "Experimental evaluation of tcp performance over 10gb/s passive optical networks (xg-pon)," in *2014 IEEE global communications conference*. IEEE, 2014, pp. 2223–2228.

[42] P. Casas, A. D'Alconzo, P. Fiadino, A. Bär, A. Finamore, and T. Zseby, "When youtube does not work?analysis of qoe-relevant degradation in google cdn traffic," *IEEE Transactions on Network and Service Management*, vol. 11, no. 4, pp. 441–457, 2014.

[43] F. Ciucu and F. Poloczek, "On multiplexing flows: Does it hurt or not?" in *Computer Communications (INFOCOM), 2015 IEEE Conference on*. IEEE, 2015, pp. 1122–1130.

[44] Y. Xu, C. Yu, J. Li, and Y. Liu, "Video telephony for end-consumers: measurement study of google+, ichat, and skype," in *Proceedings of the 2012 ACM conference on Internet measurement conference*. ACM, 2012, pp. 371–384.

[45] C. Callegari, S. Giordano, M. Pagano, and T. Pepe, "Behavior analysis of tcp linux variants," *Computer Networks*, vol. 56, no. 1, pp. 462–476, 2012.

[46] S. Shailendra, R. Bhattacharjee, and S. K. Bose, "Optimized flow division modeling for multi-path transport," in *Annual IEEE India Conference (INDICON)*, 2010, pp. 1–4.

[47] J. Wu, C. Yuen, B. Cheng, M. Wang, and J. Chen, "Streaming high-quality mobile video with multipath tcp in heterogeneous wireless networks," *IEEE Transactions on Mobile Computing*, vol. 15, no. 9, pp. 2345–2361, 2016.

[48] R. Rajaboina, P. C. Reddy, and R. A. Kumar, "Performance comparison of tcp, udp and tfrc in static wireless environment," in *2015 2nd International Conference on Electronics and Communication Systems (ICECS)*. IEEE, 2015, pp. 206–212.

[49] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh, and V. Jacobson, "Bbr: congestion-based congestion control," *Communications of the ACM*, vol. 60, no. 2, pp. 58–66, 2017.

[50] C. Xu, J. Zhao, and G.-M. Muntean, "Congestion control design for multipath transport protocols: A survey," *IEEE communications surveys & tutorials*, vol. 18, no. 4, pp. 2948–2969, 2016.

[51] S. Ferlin, Ö. Alay, T. Dreibholz, D. A. Hayes, and M. Welzl, "Revisiting congestion control for multipath tcp with shared bottleneck detection," in *Computer Communications, IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on*. IEEE, 2016, pp. 1–9.

[52] S. Molnár, Z. Móczár, and B. Sonkoly, "Living with congestion: Digital fountain based communication protocol," *Computer Communications*, vol. 80, pp. 82–100, 2016.

[53] C. Perkins and V. Singh, "Multimedia congestion control: Circuit breakers for unicast rtp sessions," *Internet RFC*, no. 8083, 2017.

[54] X. Deng, T. Zhang, and X. He, "Finishing the tiny flows quickly for common data centre services," *International Journal of Security and Networks*, vol. 12, no. 2, pp. 112–119, 2017.

[55] S. K. Bisoy, P. K. Pattnaik, B. Pati, and C. R. Panigrahi, "Design and analysis of a stable aqm controller for network congestion control," *International Journal of Communication Networks and Distributed Systems*, vol. 20, no. 2, pp. 143–167, 2018.

[56] H. Owens II and A. Durresi, "Video over software-defined networking (vsdn)," *Computer Networks*, vol. 92, pp. 341–356, 2015.

[57] X. Nan, Y. He, and L. Guan, "Optimal resource allocation for multimedia cloud based on queuing model," in *Multimedia signal processing (MMSP), 2011 IEEE 13th international workshop on*. IEEE, 2011, pp. 1–6.

[58] L. Zhou and H.-C. Chao, "Multimedia traffic security architecture for the internet of things," *IEEE Network*, vol. 25, no. 3, pp. 35–40, 2011.

[59] F. Liers, T. Volkert, and A. Mitschele-Thiel, "The forwarding on gates architecture: Merging intserv and diffserv," in *Proceedings of international conference on advances in future internet (AFIN)*. Citeseer, 2012.

[60] J. W. Guck and W. Kellerer, "Achieving end-to-end real-time quality of service with software defined networking," in *Cloud Networking (CloudNet), 2014 IEEE 3rd International Conference on*. IEEE, 2014, pp. 70–76.

[61] P. Flavius and P. Ferdi, "Internet service delivery models: evolution and current issues," in *Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2011 International Conference on*. IEEE, 2011, pp. 146–153.

[62] A. Malik, J. Qadir, B. Ahmad, K.-L. A. Yau, and U. Ullah, "Qos in ieee 802.11-based wireless networks: a contemporary review," *Journal of Network and Computer Applications*, vol. 55, pp. 24–46, 2015.

[63] S. Ha, Y. Kim, L. Le, I. Rhee, and L. Xu, "A step toward realistic performance evaluation of high-speed tcp variants," in *Fourth International Workshop on Protocols for Fast Long-Distance Networks (PFLDNet06)*, 2006.

[64] J. Wang, A. Liu, T. Yan, and Z. Zeng, "A resource allocation model based on double-sided combinational auctions for transparent computing," *Peer-to-Peer Networking and Applications*, pp. 1–18, 2017.

[65] S. Li, W. Sun, and C. Hua, "Optimal resource allocation for heterogeneous traffic in multipath networks," *International Journal of Communication Systems*, vol. 29, no. 1, pp. 84–98, 2016.

[66] G. Raina, S. Manjunath, S. Prasad, K. Giridhar, G. Raina, S. Manjunath, S. Prasad, and K. Giridhar, "Stability and performance analysis of compound tcp with rem and drop-tail queue management," *IEEE/ACM Transactions on Networking (TON)*, vol. 24, no. 4, pp. 1961–1974, 2016.

[67] G. Abbas, Z. Halim, and Z. H. Abbas, "Fairness-driven queue management: A survey and taxonomy," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 324–367, 2016.

[68] Y. Yang, M. Xu, D. Wang, and S. Li, "A hop-by-hop routing mechanism for green internet," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 1, pp. 2–16, 2016.

[69] H. Ballani, Y. Chawathe, S. Ratnasamy, T. Roscoe, and S. Shenker, "Off by default!" *Microsoft.com*.

[70] M. Gholipour, A. T. Haghighat, and M. R. Meybodi, "Hop-by-hop congestion avoidance in wireless sensor networks based on genetic support vector machine," *Neurocomputing*, vol. 223, pp. 63–76, 2017.

[71] Y. Ganjali and N. McKeown, "Update on buffer sizing in internet routers," *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 5, pp. 67–70, 2006.

[72] A. Vishwanath, V. Sivaraman, and M. Thottan, "Perspectives on router buffer sizing: Recent results and open problems," *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 2, pp. 34–39, 2009.

[73] A. Vishwanath, V. Sivaraman, and G. N. Rouskas, "Anomalous loss performance for mixed real-time and tcp traffic in routers with very small buffers," *IEEE/ACM Transactions on Networking (TON)*, vol. 19, no. 4, pp. 933–946, 2011.

[74] M. Allman, "Comments on bufferbloat," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 1, pp. 30–37, 2013.

[75] H. H. Gharakheili, A. Vishwanath, and V. Sivaraman, "Comparing edge and host traffic pacing in small buffer networks," *Computer Networks*, vol. 77, pp. 103–116, 2015.

[76] D. Ghosh, K. Jagannathan, and G. Raina, "Right buffer sizing matters: some dynamical and statistical studies on compound tcp," *arXiv preprint arXiv:1604.05516*, 2016.

[77] K. Edeline, M. Kühlewind, B. Trammell, E. Aben, and B. Donnet, "Using udp for internet transport evolution," *arXiv preprint arXiv:1612.07816*, 2016.

[78] G. Papastergiou, G. Fairhurst, D. Ros, A. Brunstrom, K.-J. Grinnemo, P. Hurtig, N. Khademi, M. Tüxen, M. Welzl, D. Damjanovic *et al.*, "De-ossifying the internet transport layer: A survey and future perspectives," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 1, pp. 619–639, 2017.

[79] A. Tsioliaridou, C. Zhang, C. Liaskos, and V. Tsaoussidis, "Fast-fair handling of flows," *International Journal of Communication Networks and Distributed Systems*, vol. 18, no. 1, pp. 32–57, 2017.

[80] A. Arfeen, K. Pawlikowski, D. McNickle, and A. Willig, "The role of the weibull distribution in modelling traffic in internet access and backbone core networks," *Journal of network and computer applications*, vol. 141, pp. 1–22, 2019.

[81] I. V. Strelkovskaya, T. I. Grygoryeva, and I. N. Solovskaya, "Self-similar traffic in g/m/1 queue defined by the weibull distribution," *Radioelectronics and Communications Systems*, vol. 61, no. 3, pp. 128–134, 2018.

[82] M. Wajahat, A. Yele, T. Estro, A. Gandhi, and E. Zadok, "Analyzing the distribution fit for storage workload and internet traffic traces," *Performance Evaluation*, vol. 142, p. 102121, 2020.

[83] M. Alasmar, G. Parisis, R. Clegg, and N. Zakhleniu, "On the distribution of traffic volumes in the internet and its implications," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2019, pp. 955–963.

[84] P. Jurkiewicz, G. Rzym, and P. Boryło, "Flow length and size distributions in campus internet traffic," *Computer Communications*, vol. 167, pp. 15–30, 2021.

[85] O. J. Adeyemi, S. I. Popoola, A. A. Atayero, D. G. Afolayan, M. Ariyo, and E. Adetiba, "Exploration of daily internet data traffic generated in a smart university campus," *Data in brief*, vol. 20, pp. 30–52, 2018.

[86] I. Strelkovskaya and I. Solovskaya, "Using spline-extrapolation in the research of self-similar traffic characteristics," *Journal of Electrical Engineering*, vol. 70, no. 4, pp. 310–316, 2019.

[87] V. Kartashevskiy and M. Buranova, "Analysis of packet jitter in multiservice network," in *2018 International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T)*. IEEE, 2018, pp. 797–802.

[88] L. Arshadi and A. H. Jahangir, "An empirical study on tcp flow interarrival time distribution for normal and anomalous traffic," *International Journal of Communication Systems*, vol. 30, no. 1, 2017.

[89] G. Srivastava, M. Singh, P. Kumar, and J. Singh, "Internet traffic classification: A survey," in *Recent Advances in Mathematics, Statistics and Computer Science*. World Scientific, 2016, pp. 611–620.

[90] L. Arshadi and A. H. Jahangir, "Benford's law behavior of internet traffic," *Journal of Network and Computer Applications*, vol. 40, pp. 194–205, 2014.

[91] S. K. Bose, *An introduction to queueing systems*. Springer Science & Business Media, 2013.

[92] M. Liu, Y. Xue, Y. Zhao, and H. Guo, "Research on the distribution and self-similarity characteristic of end-to-end network delay," *International Journal of Future Generation Communication and Networking*, vol. 8, no. 3, pp. 291–302, 2015.

[93] I. Ivanisenko, L. Kirichenko, and T. Radivilova, "Investigation of self-similar properties of additive data traffic," in *2015 Xth International Scientific and Technical Conference" Computer Sciences and Information Technologies"(CSIT)*. IEEE, 2015, pp. 169–171.

[94] O. Fedevych, I. Droniuk, and M. Nazarkevych, "Monitoring and analysis of measured and modeled traffic of tcp/ip networks," in *International Conference on Computer Networks*. Springer, 2016, pp. 32–41.

[95] S. M. Popa and G. M. Manea, "Using traffic self-similarity for network anomalies detection," in *2015 20th International Conference on Control Systems and Computer Science*. IEEE, 2015, pp. 639–644.

[96] S. K. Patra and S. Mishra, "Self similarity effect of rtt and rto in network congestion control mechanism," in *Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies*, 2016, pp. 1–4.

[97] L. Vassio, D. Giordano, M. Trevisan, M. Mellia, and A. P. C. da Silva, "Users' fingerprinting techniques from tcp traffic," in *Proceedings of the Workshop on Big Data Analytics and Machine Learning for Data Communication Networks*, 2017, pp. 49–54.

[98] M. C. Weigle, P. Adurthi, F. Hernández-Campos, K. Jeffay, and F. D. Smith, "Tmix: a tool for generating realistic tcp application workloads in ns-2," *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 3, pp. 65–76, 2006.

[99] P. Oppenheimer, K. Nabozny, and J. B. Wilson, *Top-down network design*. Cisco Press Indianapolis, 2011.

[100] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson, "On the self-similar nature of ethernet traffic (extended version)," *Networking, IEEE/ACM Transactions on*, vol. 2, no. 1, pp. 1–15, 1994.

[101] M. E. Crovella and A. Bestavros, "Self-similarity in world wide web traffic: evidence and possible causes," *Networking, IEEE/ACM Transactions on*, vol. 5, no. 6, pp. 835–846, 1997.

[102] S. A. Mushtaq and A. A. Rizvi, "Statistical analysis and mathematical modeling of network (segment) traffic," in *Emerging Technologies, 2005. Proceedings of the IEEE Symposium on*. IEEE, 2005, pp. 246–251.

[103] A. Vishwanath, V. Sivaraman, and G. N. Rouskas, "Anomalous loss performance for mixed real-time and tcp traffic in routers with very small buffers," *IEEE/ACM Transactions on Networking*, vol. 19, no. 4, pp. 933–946, 2011.

[104] R. ITU-T and I. Recommend, "G. 114," *One-way transmission time*, vol. 18, 2000.

List of Publications from the Thesis

International Journals

1. Thombre, Sneha K., Lalit M. Patnaik, and Anil S. Tavildar. "Effect of multiplexing TCP flows on delay sensitivity in the internet communications." CCF Transactions on Networking 1.1-4 (2019): 37-51.
   (**Chapter - 3**)

2. Thombre, Sneha, Lalit Patnaik, and Anil Tavildar. "Delay and jitter sensitivity analysis with varying TCP fraction for multiplexed internet communications." International Journal of Internet Protocol Technology 13.2 (2020): 61-77..
   (**Chapter - 4**)

3. Thombre, Sneha, Lalit M. Patnaik, and Anil S. Tavildar. "Design and performance analysis of link-by-link congestion avoidance algorithm." International Journal of Communication Networks and Distributed Systems 24.4 (2020): 381-408.
   (**Chapter - 6**)

National and International Conferences

1. Thombre, Sneha. "Network Jitter Analysis with varying TCP for Internet Communications." *2018 3rd International Conference for Convergence in Technology (I2CT)*. IEEE, 2018. (**Chapter - 4**)

2. Thombre, Sneha. "Modelling of UDP throughput." 2016 IEEE Region 10 Conference *(TENCON)*. IEEE, 2016.
   (**Chapter - 5**)

3. Thombre, Sneha. "Rethinking the design of Internet: End-to-end arguments vs. the brave new world." *Large Scale Multi-disciplinary Systems of National Significance (LAMSYS)* ISRO , 2016. (**Chapter - 6**)

4. Thombre, Sneha, Lalit M. Patnaik, and Sanjay K. Bose. "TCP delay-friendliness." *2016 International Conference on Communication and Signal Processing (ICCSP)*. IEEE, 2016. (**Chapter - 3**)

Awards/Honor/Talks

- Presented paper at Ph.D. Forum "Quantitative Analysis of Internet Traffic for TCP and Hop-by-Hop Congestion Control" at the IEEE International conference Advanced Networks and Telecommunications Systems 2014
- Selected for Zonal Level Research Project in Avishkar 2015 organised by Savitribai Phule Pune University, Board of College Level and University Development (BCUD)