On the Modeling of TCP Latency and Throughput

Dong Zheng

ON THE MODELING OF TCP LATENCY AND THROUGHPUT

By

Dong Zheng

A Thesis
Submitted to the Faculty of
Mississippi State University
in Partial Fulfillment of the Requirements
for the Degree of Master of Science
in Electrical Engineering
in the Department of Electrical and Computer Engineering

Mississippi State, Mississippi

August 2002

ON THE MODELING OF TCP LATENCY AND THROUGHPUT

By

Dong Zheng

Approved:

Name: Dong Zheng

Date of Degree: August 3, 2002

Institution: Mississippi State University

Major Field: Electrical Engineering

Major Professor: Dr. Georgios Y. Lazarou

Title of Study: ON THE MODELING OF TCP LATENCY AND THROUGHPUT

Pages in Study: 64

Candidate for Degree of Master of Science

In this thesis, a new model for the slow start phase based on the discrete evolutions of congestion window is developed. We then integrate this result into our improved TCP steady state model for a better prediction performance. Combining these short and steady state models, we propose an extensive stochastic model which can accurately predict the throughput and latency of the TCP connections as functions of loss rate, round-trip time (RTT), and file size. We validate our model through simulation experiments. The results show that our model's predictions match the simulation results better than the stochastic models in [17, 4], about 75% improvement in the accuracy of performance predictions compared with other steady-state models and 20% improvement compared with other short-lived TCP connection models.

DEDICATION

To my parents and Beryl.

ACKNOWLEDGMENTS

My sincere thanks goes to Dr. Georgios Lazarou, my major adviser, for his guidance and support. I appreciate his patience while I am trying to find my way out. His insight and knowledge in networks, the social grace with which he delivers his idea are a constant inspiration.

I wish to thank my thesis committee members for their valuable suggestions. I especially thank Dr. Nicholas Younan, without his help and long-lasting support, I wouldn't be able to finish my study. My thanks also goes to Mr. Bill Chapman and Mr. Michale Lane for their help on my various kinds of computer questions.

I thank my friends, especially Manimaran Selvaraj and Gomathi Anandan. Mani's knowledge in simulation always be my last resort of NS questions. He and Gomathi leaded me to this wonderful $\LaTeX$world.

I also want to thank my wife, Beryl, whose appearance has changed every aspect of my life. Her support and lovely companionship is another important source of strength for me.

This degree is dedicated to my parents, Cheng Rong and Riyong Zheng. They spend all their time and energies on me. Without their devoted love and unstingting sacrifices, none of this would have been possible.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS, ABBREVIATIONS, AND NOMENCLATURE

g	stands for golden number, 1.61804;
$cwnd$	is the size of the congestion window;
$ssthresh$	is the size of the slow start threshold;
RTT	is the round trip time;
MSS	is the maximum segment size.

CHAPTER I

INTRODUCTION

A multitude of Internet applications, such as the world wide web, usenet news, file transfer and remote login have opted for TCP as their transport medium. TCP's performance thus greatly influences Internet traffic behavior [23, 5]. Hence, many stochastic models of TCP latency and throughput have been proposed, trying to capture its characteristics [20, 8, 17, 4]. In most of these models, the TCP performance (latency and throughput) is described based on the network parameters such as TCP round trip time and packet loss rate. With the help of these models, a better understanding of TCP's sensitivity to network parameters can be obtained, and this can further aid in the design of active queue management techniques [7, 16] and TCP-friendly multicast protocols [2, 25]. In order to distribute resources in a fair manner between TCP and other transport protocol, we need to know the TCP connections' characteristics. Therefore, an accurate model of TCP performance is needed. However due to the complexity of the protocol, itself, and the fast-changing network conditions, the development of an accurate TCP stochastic model is a very challenging task.

There are at present two major approaches in the TCP modeling area. The first one makes use of simulation tools to simulate the behavior of TCP [9]. But simulation studies neither provide a full mathematical description of TCP nor estimate the performance of TCP when the network parameters change. Hence simulation based approaches are only useful to test the validating analytical models of TCP [14].

The second approach models TCP performance through the use of mathematical analysis. The work in [17, 4] follow this approach. These modeling techniques assume the underlying congestion control window size growth process to be a Markov generative process. These techniques start with mathematical expressions that relate the TCP sending rate with the loss probability and the round trip time. Then, in order to develop a mathematically tractable and solvable model, many assumptions are made to simplify the analysis. This makes it possible to derive a complete mathematical description of the expectation of TCP sending rate based on the network parameters (round trip time and loss probability).

Stochastic models of TCP can be classified into three types: models for the steady-state performance of a bulk transfer flow [3, 17], models for the short-lived flows with small packet losses [18, 8, 11], and lately, models that combine the above analyses together [20, 4].

1.1 Problem Statement and Motivation

Most existing stochastic TCP models do not capture the effect of time-outs [12, 13, 15]. As observed in [17], around 90% of packet losses cause time-outs. Hence, time-outs must always be an integral part of any TCP performance prediction model.

The work in [17] proposes a model which accounts for the time-outs, however, it always underestimates the performance of TCP. This is due on several inaccurate assumptions and approximations [1].

At the best of our knowledge, none of the steady-state models proposed so far account for the slow start phase which begins at the end of every single time-out [17, 3]. Hence, for accurate TCP performance predictions, new models are needed that can capture the effects of the slow-start phases on performance.

Since delayed acknowledgment effects has made the stochastic analysis complex in the model building, most models circumvent it by just ignoring their effects, or using a simple approximation. And these simplifications inevitably bring some inaccuracy to the model's prediction. Obviously, accurate TCP models should include the analysis of the delayed acknowledgment effects into their model.

All steady state models assume the availability of unlimited data to send. Hence, the impact of the transient phase on performance is insignificant, and is therefore ignored [17]. These models can only be used to predict the TCP send rate or throughput of bulk data transfers. Therefore, they are not applicable in predicting the performance of short-lived TCP flows.

It is noted in [6, 1] that the majority of TCP traffic in the Internet consists of short-lived flows, i.e., the transmission comes to an end during the slow start phase before switching

[1]See Chapter V for details.

to the congestion-avoidance phase. Hence, new models are needed that are capable of predicting the performance of long-lived or short-lived TCP flows.

1.2 Summary of Main Contributions of This Research

In this thesis, we constructed:

1. a better and tractable model of the congestion window growth pattern for the slow-start phase;
2. a complete steady-state model including the slow-start phase and more accurate analysis of time-outs;
3. an accurate model for the short-lived TCP flows.

1.3 Organization

The remainder of the thesis is organized as follows. In Chapter II we briefly describe the TCP's implementation, especially congestion control mechanism on which we are trying to build our model. The following Chapter III gives the summary of the work related to ours. After that, Chapter IV describes the assumptions, and then, gives a very detailed analysis of our proposed extended model step by step. The slow start congestion window growth pattern is also discussed here. In Chapter V, we will propose a short TCP connection model based on the extended steady state model developed in the previous chapter. Chapter VI describes the simulation and model validation experiments and analyze the simulation results to verify the accuracy our model. Finally, Chapter VII concludes the paper and discusses the future work.

CHAPTER II

TCP: THE MAJOR TRANSFER PROTOCOL IN THE INTERNET

Transmission Control Protocol (TCP) is the most widely used communication protocol corresponding to the transport layer of the Open System Interconnection (OSI) networking model. In this chapter, we describe the important features of TCP protocol which are relevant to our model building in the next chapter. We begin our discussion with an overview of TCP and its implementations, especially the TCP Reno release.

2.1 BSD TCP's implementation

TCP, originally defined in RFC 793[19], provides a connection-oriented and reliable byte stream service to applications. The most popular TCP implementation is released by the Computer Systems Research Group at the University of California at Berkeley, usually called the "BSD Networking Releases". This implementation of TCP has served as the starting point for many TCP/IP suites running in different UNIX and windows systems [22]. A chronology of the various BSD releases is shown in Table 2.1. Two of the most popular BSD implementations are TCP Tahoe and TCP Reno. They are widely used as a basis on the effect congestion control to the TCP performance [21]. In this thesis, we focus on the following TCP Reno features: the explicit and acknowledged connection establishment, the reliable stream exchange across the TCP connection, the end-to-end flow control, and the congestion control.

Features	RFC 793 and RFC 1122	TCP Tahoe (1988)	TCP Reno (1990)	TCP Lite (1993)
RTT Variance Estimation	√	√	√	√
Exponential RTO Backoff	√	√	√	√
Slow start	√	√	√	√
Karn's Algorithm	√	√	√	√
Congestion Avoidance	√	√	√	√
Fast Retransmit		√	√	√
Fast Recovery			√	√

Table 2.1 A history of BSD releases and important TCP features added with each release.

2.2 Basic Features of TCP

TCP is a connection-oriented communication protocol, i.e., whenever applications want to transfer data, they are required to establish a logical connection before they can do so. This explicit TCP connection is established by the three-way-handshake process [22].

When data is passed from an application to TCP for delivery, TCP breaks the data stream into smaller chunks, and adds a protocol information header to form a segment. The largest chunk of data that TCP can include in each segment is limited by the maximum segment size (MSS). During the initialization of a connection, each host announces its MSS, and TCP chooses the smallest value to avoid further fragmentation. These segments are passed to IP, and in turn IP appends its own header information to form datagrams or packets.

TCP operates on top of IP suite, which provides a best effort service. That is, IP does not guarantee that a packet would successfully be received by its destination. Packets could be dropped somewhere in the network, or get corrupted by channel noise. Both of these situations lead to the failure of the packets' delivery. TCP is designed to deal with these situations and provides the reliability to the data delivery. It achieves this by using a technique called positive acknowledgment with retransmission. This technique requires TCP sender to assign unique sequence numbers to packets, keep record of each packet sent, and wait for the receiver to send back acknowledgments (ACKs) upon successfully receiving the packets. The sender also starts a retransmission timer whenever it sends a packet. If an acknowledgment does not arrive before the expiration of the timer, TCP will assume that the packet has been lost, and therefore TCP will retransmit it.

The total amount of time a sender waits for an ACK before retransmission is called retransmission time-out (RTO). RTO is calculated based on the round trip time (RTT), the duration required for a segment to travel to the destination and an ACK to return to the sender. However, RTT changes significantly over time due to the variation of network conditions. Therefore, TCP uses an adaptive retransmission algorithm to track delay changes on each connection, and adjust the RTO accordingly. More exact details on calculating the RTO can be found in [22].

2.2.1 Flow Control

However, if TCP sends one segment at a time and waits for the acknowledgment, the send rate would be around $1/RTT$ packets per second. That is, TCP will not fully utilize the network bandwidth if the simple stop-and-wait algorithm is applied. Therefore, TCP uses the sliding window technique instead [22]. It allows the sender to transmit multiple segments before it stops and wait for an ACK, thus a higher send rate is achieved by TCP.

When TCP sends data at a rate greater than the maximum receiving rate of the receiver, the receiver's buffer will overflow. Hence, many packets are dropped by the receiver. This leads to time-outs at the sender, and therefore, retransmissions. In that situation, TCP performance is severely degraded. Thus, flow control is also about regulating the sending rate to prevent fast senders overloading slow receivers. Obviously, the sending rate depends on the receiver's processing capacity and buffer size, and therefore, it is controlled by the receiver. The receiver achieves this by advertising the current "send" window size along with each ACK, thus, limiting the size of the transmission window of the sender. This "send" window indicates how much buffer space the receiver has available for the incoming packets. A zero window tells the sender that there is no space left for further packets, and the sender should stop the transmission until it receives a non-zero window advertisement.

2.2.2 Slow Start and Congestion Avoidance

Flow control does not help reducing the possibility of congestion within the network, nor does relieve congestion when it happens. Congestion is defined as a condition of severe delay caused by an overload of packets at one or more switching points. It can occur whenever the offered load exceeds the available bandwidth at a bottleneck node.

TCP Reno has adopted two congestion control mechanism, and they are usually implemented together, known as the slow start and congestion avoidance scheme. With this scheme, a TCP connection can be in one of the two modes: the slow-start mode or the congestion-avoidance mode.

TCP changes from slow start to congestion avoidance based on a threshold. The switching is done based on two state variables: the current congestion window size ($cwnd$), and the slow-start threshold ($ssthresh$). The sender always sends the minimum of $cwnd$ and the window advertised by the receiver (snd_wnd). Thus the $cwnd$ reflects the flow control imposed by the sender according to the network condition, and snd_wnd is the flow control imposed by the receiver according to the its processing capacity and buffer size.

TCP always begins its transmission from the slow-start phase. During the slow-start phase, TCP tries to send as many packets as possible until a packet loss is detected. It does this by increasing the congestion window ($cwnd$) in an exponential fashion, i.e., whenever an acknowledgment is received, it increases the $cwnd$ by one. To see the exponential growth pattern of the $cwnd$, assume that at first $cwnd$ is one, and therefore, only one packet is sent. Once this packet is acknowledged by the receiver, $cwnd$ is increased by one, and hence, two more packets are transmitted. When these two packets are again ACKed by the receiver, $cwnd$ is set to four. This exponential growth pattern of $cwnd$ continues until $cwnd$ reaches the maximum window size or the slow start threshold. Figure 2.1 shows a typical initial slow start phase evolution without any packet losses. The number of packets that are sent back to back is printed on the arrowhead line. As shown in the figure, TCP establishes a connection through the “3-way handshake” mechanism. It first sends a SYN packet and upon receiving the corresponding ACK, TCP resumes with slow-start. Even though we include the delayed acknowledgment scheme, the evolution of cwnd is still exponential.

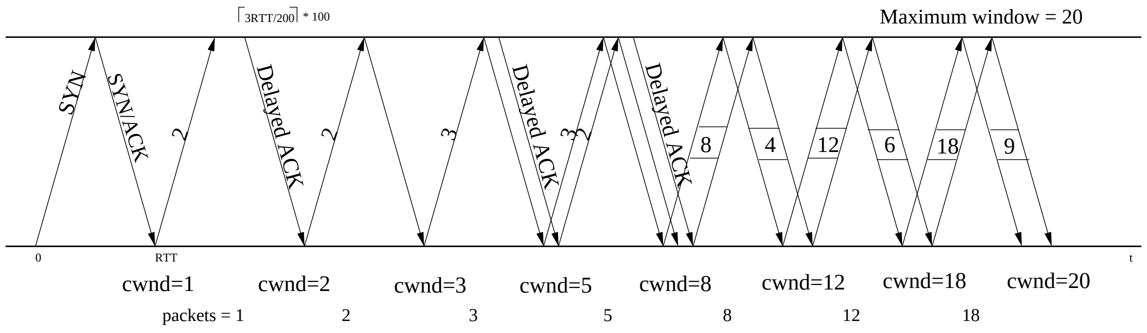

Figure 2.1 A typical TCP initial start phase evolution

TCP enters the congestion avoidance mode once the congestion window ($cwnd$) becomes greater than the slow-start threshold ($ssthresh$). During this mode, TCP increases $cwnd$ by $1/cwnd$ for every acknowledgment it receives. Thus, for every RTT period, TCP receives $cwnd$ worth of ACKs or $cwnd/2$ worth of ACKs if considering the delayed acknowledgment mechanism. That is, during the congestion avoidance mode, TCP increases $cwnd$ by one or one half segment per RTT.

Congestion is indicated by a packet loss, which could cause TCP time-outs or the reception of duplicate ACKs. TCP treats these two events with different strategies. When TCP detects a packet loss by a time-out, it sets $ssthresh$ to one half of the current $cwnd$, $cwnd$ to one packet, and then resends the lost packet. It then switches the slow-start mode. However, when three duplicate ACKs are received, TCP assumes that a packet is lost due to channel noise or mild congestion. It then sets $ssthresh$ to half of the current $cwnd$, and immediately retransmits the lost packet without waiting for the retransmission timer to expire. This is called the fast retransmit mechanism. Next, it sets $cwnd$ to $ssthresh$ plus three segments. Each time another duplicate ACK is received, TCP increases $cwnd$ by one segment and transmits a new segment if allowed by the current $cwnd$. Upon receiving a non-duplicate ACK, TCP resumes with the congestion avoidance mode. This is called the fast recovery algorithm [22].

2.2.3 Delayed Acknowledgments

Most TCP implementations use delayed acknowledgments. That is, TCP does not send ACK instantly upon receiving a segment. Instead, it delays to send an ACK in case there is data to send to the sender, so that the ACK can be carried along [1]. Usually TCP

[1]This is called piggybacking.

hold this ACK for maximum 200ms. During this time, if another packet is received, TCP sends an ACK immediately to acknowledge the reception of both packets. Another exception is: when out-of-order segment is received, an immediate ACK is sent.

2.2.4 TCP's Self-Clocking Property

TCP's flow control and congestion control are designed according to a "self-clocking" concept. As indicated in Jacobson's work [10], the flow of a TCP connection should conform to a conservation-of-packets principle. It means that for the flow to stay in equilibrium, i.e., maintaining the stability with a full window size of data in transition, the TCP sender should not release new packets into the network until it receives an indication that an old packet has exited the network. These indications are done by the acknowledgment packets. In other words, the sender uses ACKs (acknowledgments) as a clock to strobe new packets into the network. Since the TCP receiver can not generate the ACKs faster than the sender sends the packets into the network, the protocol is called "self-clocking".

This self-clocking mechanism is very important to the congestion-avoidance scheme because it helps the system adjust to the bandwidth and delay variations. When ACKs are generated at the same rate as of the incoming packets, the sender releases packets at a rate limited by the link bandwidth, so to prevent overloading the network. A more detailed description is provided in [10].

CHAPTER III

RELATED WORK

As mentioned before, stochastic models of TCP can be classified into three types: models for the steady-state performance of a bulk transfer flow [3, 17], models for the short-lived flows with small packet losses [18, 8, 11], and lately, models that combine the above analyses together [20, 4].

A review of the models given in [17, 4] follows since their models have been widely referenced, and have provided a base from which our models are developed. The stochastic steady-state model proposed in [17] for predicting TCP's throughput as a function of packet loss rate and round-trip time (RTT), is the first model that captures not only the behavior of fast retransmission mechanism, but also the effect of the time-outs. Based on this steady-state model, [4] proposed a new model for capturing the performance of short-lived TCP connections. Unlike the steady-state model of [17], this model includes the three-way-handshake and slow-start mechanisms. Further, it was developed under the assumption of no packet losses, or packets were getting lost rarely.

The following is a brief overview of these models. The notations and terminology are same as those used in [17, 4].

3.1 The Short-lived Connections Model

Cardwell, Savage and Anderson have proposed a model in [4] trying to capture the performance of short-lived TCP connections since they observed that most of the traffic of the Internet today are composed by short TCP connections. Short-lived TCP connections end their data transmission before entering the congestion-avoidance phase or soon after they enter congestion-avoidance phase for the first time.

The expected “three-way-handshake” duration is approximated in [4] as[1]:

$$E[T_{twhs}] = RTT + T_s(\frac{1-p}{1-2p} - 1) \tag{3.1}$$

where the RTT is the round trip time, T_s is the duration of SYN time-out and p is the packet loss rate.

Data transfer begins with the slow start phase. In this phase, the sender’s congestion window (cwnd) increases exponentially until either of the following two events occur: a packet gets lost or the cwnd is bounded by the receiver’s maximum congestion window size, W_m.

$E[Y_{init}]$, the expected number of packets sent until a loss occurs is given as:

$$\begin{aligned} E[Y_{init}] &= \sum_{k=0}^{d-1}(1-p)^k pk + (1-p)^d d \\ &= \frac{(1-(1-p)^d)(1-p)}{p} \\ &= \begin{cases} \frac{1-p}{p} & \text{when } d \to \infty \\ d(1-p) & \text{when } d \text{ is small and } p \to 0 \end{cases} \end{aligned} \tag{3.2}$$

[1]In this thesis, we assume that the acknowledgment-packet loss rate is zero.

where d is the total number of packets to be transmitted. The first condition of Equation (3.2) stands when we have infinite number of packets to send and the second condition holds when the TCP connection is short-lived and the loss rate is small.

Using the expected number of packets to be sent in the slow start phase, the expected window size $E[W^{ss}]$ and expected transfer time $E[Z^{ss}]$ are given as [4]:

$$E[W^{ss}] = \frac{E[Y^{ss}](\gamma - 1)}{\gamma} + \frac{w_1}{\gamma} \tag{3.3}$$

and

$$E[Z^{ss}] = \begin{cases} RTT[log_\gamma(\frac{W_m}{w_1}) + 1 + \frac{1}{W_m}(E[Y^{ss}] - \frac{\gamma W_m - w_1}{\gamma - 1})] & \text{when } E[W^{ss}] > W_m \\ RTT log_\gamma(\frac{E[Y^{ss}](\gamma-1)}{w_1} + 1) & \text{when } E[W^{ss}] \le W_m \end{cases} \tag{3.4}$$

where γ is 1.5 if the delayed acknowledgment mechanism is applied, otherwise it is 2 . w_1 is the initial window size, usually set to 1, and W_m is the limit on the receiver's congestion window size.

3.2 The Steady-state Model

As a consequence of the exponential growth of the cwnd, TCP tries to send as many packets as possible. Finally the connection reaches saturation and packets are lost. At this point the slow start phase terminates, and congestion control mode takes over the transmission. The model in [17] describes this phase and establishes the send rate as:

$$B(p, RTT) = min\left(\frac{W_m}{RTT}, \frac{1}{RTT\sqrt{\frac{2bp}{3}} + T_0 min(1, 3\sqrt{\frac{3bp}{8}}p(1 + 32p^2))}\right) \tag{3.5}$$

and throughput as:

$$T(p) = \begin{cases} \frac{\frac{1-p}{p}+\frac{W(p)}{2}+Q(p,W(p))}{RTT(W(p)+1)+\frac{Q(p,W(p))f(p)T_0}{1-p}} & \text{where } W(p) \leq W_m \\ \frac{\frac{1-p}{p}+\frac{W_m}{2}+Q(p,W_m)}{RTT(\frac{W_m}{4}+\frac{1-p}{pW_m}+2)+\frac{Q(p,W_m)f(p)T_0}{1-p}} & \text{otherwise} \end{cases} \tag{3.6}$$

where T_0 is the average duration of the first time out. $Q(p, w)$ is the probability that a sender in congestion control will detect a packet loss with re-transmission time-outs (RTOs). It is formulated as a function of the loss probability and current window size, and is given as:

$$Q(p, w) = min(1, \frac{(1-(1-p)^3)(1+(1-p)^3(1-(1-p)^{w-3}))}{1-(1-p)^w} \tag{3.7}$$

where $W(p)$ is the expected cwnd value when RTOs occur. This is also given in [17] as:

$$W(p) = \frac{2+b}{3b} + \sqrt{\frac{8(1-p)}{3bp} + (\frac{2+b}{3b})^2} \tag{3.8}$$

And finally, $f(p)$, the function of loss rate, is defined as:

$$f(p) = 1 + p + 2p^2 + 4p^3 + 8p^4 + 15p^5 + 32p^6 \tag{3.9}$$

CHAPTER IV

STEADY STATE MODEL INCORPORATING THE SLOW START PHASE

4.1 Assumptions

The model is based on the TCP Reno release from Berkeley [22]. Since we are only concerned about modeling TCP performance, we assume that the link speed is very high and the sender sends full-sized segments whenever the congestion window ($cwnd$) allows. The advertised window is always a constant, i.e., the receiver fetches the data so fast that the buffer is always empty. Thus, the congestion window evolution, alone, determines the send rate of the TCP connection which could roughly be described by $cwnd/RTT$, where RTT is the round trip time.

We model the dynamics of TCP in terms of “rounds” as done in [17]. A round starts when a window of packets is sent by the sender and ends when one or more acknowledgments are received for these packets. The delayed acknowledgment’s effect is taken into consideration, but neither the Nagle algorithm nor the silly window syndrome avoidance is considered. In addition, we assume that packet losses are in accordance with the bursty loss model. The packet losses in different rounds are independent, but they are correlated within a single round, i.e., if one packet in a round is lost, then the following back to back packets in the same round are also assumed to be lost. It is an idealization of the packet loss dynamics observed in the paths where FIFO drop-tail queues are used [4].

4.2 The Steady State Model that Incorporates the Slow-Start Phase

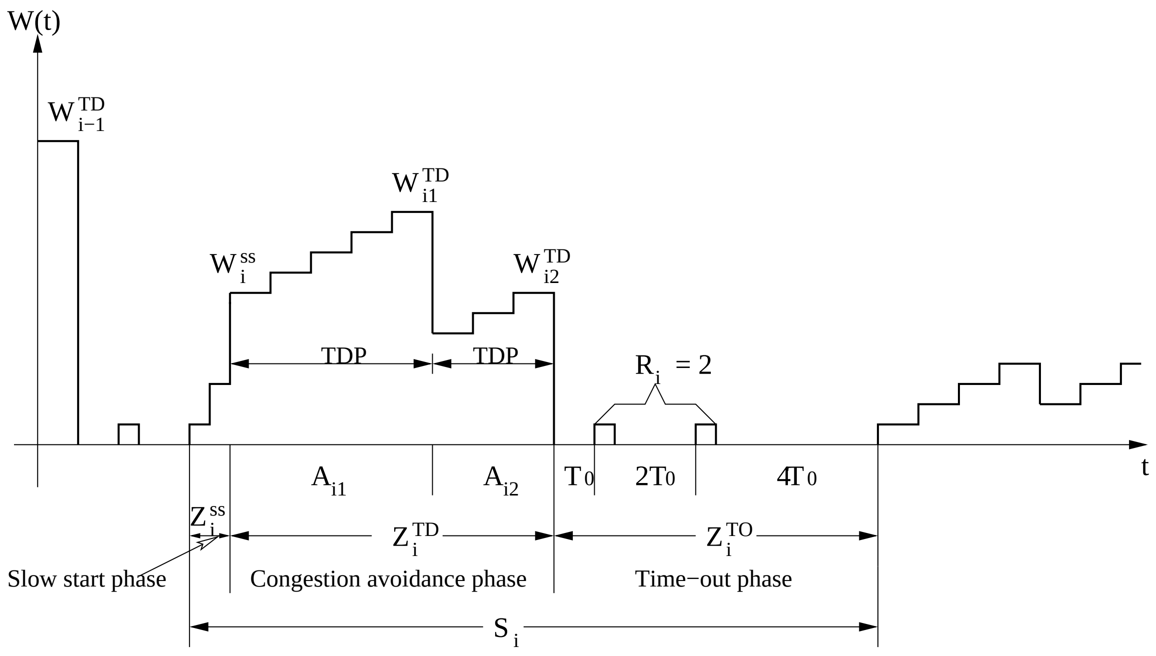

Figure 4.1 The extended steady state model - evolution of congestion window size when loss indications are triple-duplicate ACK's and time-outs.

Figure 4.1 depicts our model which includes the slow start phase. It shows a typical congestion window's ($cwnd$) evolution over time. Our model is an extension of the one used in [17]. We assume that in the steady state, the connections will go through and repeat periodically the slow-start phase, congestion-avoidance phase and time-out phase.

In the figure, the slow-start phase always begins after the first time-out following by the congestion-avoidance phase. Since the throughput in the slow-start phase is lower than the throughput in the congestion-avoidance phase, including the slow-start phase into the steady-state phase will definitely decrease the performance predictions given by the model in [17].

We define TDP to be the period between two triple-duplicate (TD) losses [17]. Let Z_i^{ss} to be the time spent in the slow-start phase, Z_i^{TD} be the duration of the congestion control phase, and Z_i^{TO} be the time interval of the time-out phase. Let M_i be the number of packets sent during the total time S_i. The above parameters are related as follows:

$$M_i = Y_i^{ss} + \sum_{j=1}^{n_i} Y_{ij} + R_i \tag{4.1}$$

$$\begin{aligned} S_i &= Z_i^{ss} + Z_i^{TD} + Z_i^{TO} \\ &= Z_i^{ss} + \sum_{j=1}^{n_i} A_{ij} + Z_i^{TO} \end{aligned} \tag{4.2}$$

where Y_i^{ss} is the number of packets sent during the slow-start phase, A_{ij} is the duration of the jth TDP, n_i is the total number of the TDPs in the interval Z_i^{TD}, Y_{ij} is the number of packets sent during the jth TDP of interval Z_i^{TD}, and R_i is the number of packets sent during the time-out phase. W_i^{ss} is the window size at the end of a slow start and finally W_{ij}^{TD} is the window size at the end of the j^{th} TDP.

Assuming (S_i, M_i) to be a sequence of independent and identically distributed (i.i.d.) random variables, we can determine the send rate as:

$$B = \frac{E[M]}{E[S]}$$

Considering n_i to be i.i.d. random variables and independent of Y_{ij} and A_{ij}, we have:

$$\begin{aligned} B &= \frac{E[Y^{ss}] + E[\sum_{j=1}^{n_i} Y_{ij}] + E[R]}{E[Z^{ss}] + E[\sum_{j=1}^{n_i} A_{ij}] + E[Z^{TO}]} \\ &= \frac{E[Y^{ss}] + E[n]E[Y] + E[R]}{E[Z^{ss}] + E[n]E[A] + E[Z^{TO}]} \end{aligned} \tag{4.3}$$

In the next subsections, we derive the closed form expressions for these expected values in the different TCP phases: the slow-start phase, the congestion-avoidance phase and the time-out phase.

4.2.1 The Slow-Start Phase

According to TCP Reno [10, 22], the current state of a TCP connection is determined based upon the values of the congestion window size ($cwnd$) and the slow-start threshold ($ssthresh$). If $cwnd$ is less than $ssthresh$, TCP is in the slow-start phase, otherwise, it is in the congestion-avoidance phase.

Since TCP has no knowledge of the network conditions during the slow-start phases, it tries to probe for the available bandwidth "greedily", i.e., increasing the $cwnd$ by one upon the receipt of a non-repeated acknowledgment. This algorithm can further be formulated using discrete mathematics as:

$$cwnd_i = \lceil \frac{cwnd_{i-1}}{2} \rceil + cwnd_{i-1}; \tag{4.4}$$

in which $cwnd_i$ is the congestion window size for the i^{th} round. Equation (4.4) is due to the fact that assuming no loss, in round $(i-1)$, there is a total of $cwnd_{i-1}$ packets sent to the destination, which, in turn, causes the TCP receiver to generate $\lceil cwnd_{i-1}/2 \rceil$

acknowledgments[1]. According to the slow-start algorithm, upon receiving these ACKs, the TCP sender increases the $cwnd$ by the number of ACKs it has obtained, which is $\lceil cwnd_{i-1}/2 \rceil$.

Noting that the congestion window is an integer, we can simplify Equation (4.4) as follows[2]:

$$cwnd_i = \lceil \frac{3}{2} cwnd_{i-1} \rceil. \tag{4.5}$$

Rearranging, we get:

$$\begin{aligned} \lceil \frac{cwnd_{i-1}}{2} \rceil &= \lceil \frac{1}{2} \lceil \frac{3}{2} cwnd_{i-2} \rceil \rceil \\ &\approx cwnd_{i-2}. \end{aligned} \tag{4.6}$$

Substituting this result in (4.4), we get the following discrete equation:

$$cwnd_i \approx cwnd_{i-2} + cwnd_{i-1}. \tag{4.7}$$

Noting that (4.7) generates the Fibonacci sequence and we have the closed form of $cwnd$ as:

$$cwnd_n = C_1 X_1^n + C_2 X_2^n, \text{ n = } 1, 2, 3... \tag{4.8}$$

in which X_1 and X_2 are[3]:

$$X_{1,2} = \frac{1 \pm \sqrt{5}}{2}. \tag{4.9}$$

[1] $\lceil x \rceil$ =the smallest integer bigger than x

[2] In deriving a model for the latency of the short-lived TCP flows, Equation (4.5) was approximated in [4] as: $cwnd_i = 3cwnd_{i-1}/2$

[3] X_1 is also called the golden number

C_1 and C_2 are determined by the initial value of $cwnd$. Assuming the initial value of $cwnd$ is one, we have:

$$C_{1,2} = \frac{5 \pm \sqrt{5}}{10}. \tag{4.10}$$

By knowing the evolution of the congestion window, we can calculate the total number of packets, Y_n^{ss}, that are sent until the n_{th} round, by summing the congestion window size during each round:

$$\begin{aligned} Y_n^{ss} &= \sum_{i=1}^{n} cwnd_i \\ &= \sum_{i=1}^{n} C_1 X_1^i + \sum_{i=1}^{n} C_2 X_2^i \\ &= \frac{C_1 X_1 (1 - X_1^n)}{1 - X_1} + \frac{C_2 X_2 (1 - X_2^n)}{1 - X_2} \\ &= cwnd_{n+2} - 2 \\ &= C_1 X_1^{n+2} + C_2 X_2^{n+2} - 2 \\ &\approx C_1 X_1^{n+2} - 2. \end{aligned} \tag{4.11}$$

The last approximation is due to the fact that:

$$C_2 X_2^{n+2} \leq |\frac{5 - \sqrt{5}}{10} \times (\frac{1 - \sqrt{5}}{2})^3| = 0.065 \tag{4.12}$$

Thus, from Equation (4.11), the number of rounds, n, can be computed as:

$$n = log_{X_1}(\frac{Y_n^{ss} + 2}{C_1}) - 2. \tag{4.13}$$

Substituting (4.13) into (4.8), we can get the approximate relationship between the congestion window size and the total count of packets that have been sent, as follows:

$$cwnd_n = \frac{Y_n^{ss} + 2}{X_1^2}. \tag{4.14}$$

Taking the expectation of both sides of Equation (4.14), we have:

$$E[W^{ss}] = \frac{E[Y^{ss}] + 2}{X_1^2} \tag{4.15}$$

in which $E[W^{ss}]$ is the expectation of $cwnd_n$.

After each time-out, the slow-start threshold is set to half of the current congestion window. Hence,

$$E[ssthresh] = \frac{E[W^{TD}]}{2}. \tag{4.16}$$

The expected data that have been sent by TCP before a loss happens can be calculated as:

$$E[Y^{ss}] = \frac{1-p}{p} \tag{4.17}$$

in which p is the loss rate.

Substituting the value of $E[Y^{ss}]$ in the Equation (4.15), we get:

$$E[W^{ss}] = \frac{1+p}{pX_1^2}. \tag{4.18}$$

This is the expected value of the congestion window when the slow-start phase ends due to a lost packet. Noting that when p is small, the expected value would be much bigger than the expected value of $ssthresh$, i.e.,

$$E[W^{ss}] \gg E[ssthresh]. \tag{4.19}$$

Thus, we can assume that most TCP connections enter the congestion-avoidance phase before a packet is lost. So, at this point, we take it for granted that a TCP connection switches from the slow-start phase to the congestion-avoidance phase when the congestion window size is equal to the $ssthresh$. As a consequence, we have the expected congestion window size at the end of the slow start constrained by the limitation of the slow-start threshold:

$$E[W^{ss}] = E[ssthresh] = \frac{E[W^{TD}]}{2}. \tag{4.20}$$

Using (4.20) in (4.15) and rearranging, we have the expected number of packets sent in the slow-start phase as:

$$E[Y^{ss}] = \frac{E[W^{TD}]g^2}{2} - 2, \tag{4.21}$$

where $g = X_1 = 1.61804$. The time spent in the slow-start phase can be obtained by multiplying the number of rounds with the round trip time (RTT) [4]:

$$E[Z^{ss}] = log_g(\frac{E[W^{TD}]}{2C_1}) \cdot RTT \tag{4.22}$$

We derive $E[W^{TD}]$ in the following subsection.

4.2.2 The Congestion-Avoidance Phase

In this subsection, the expected value of the number of packets sent in a round, $E[Y]$, and the expected value of duration of a round, $E[A]$, are derived. With reference to Figure 4.2, the following enhanced equations are given based on [17][5]:

$$Y_i = \alpha_i + W_i^{TD} - 1 \tag{4.23}$$

$$A_i = \sum_{j=1}^{X_i+1} r_{ij} \tag{4.24}$$

[4]For simplicity, we assume RTT is a constant

[5]Readers are advised to refer [17] for detailed derivation of these equations

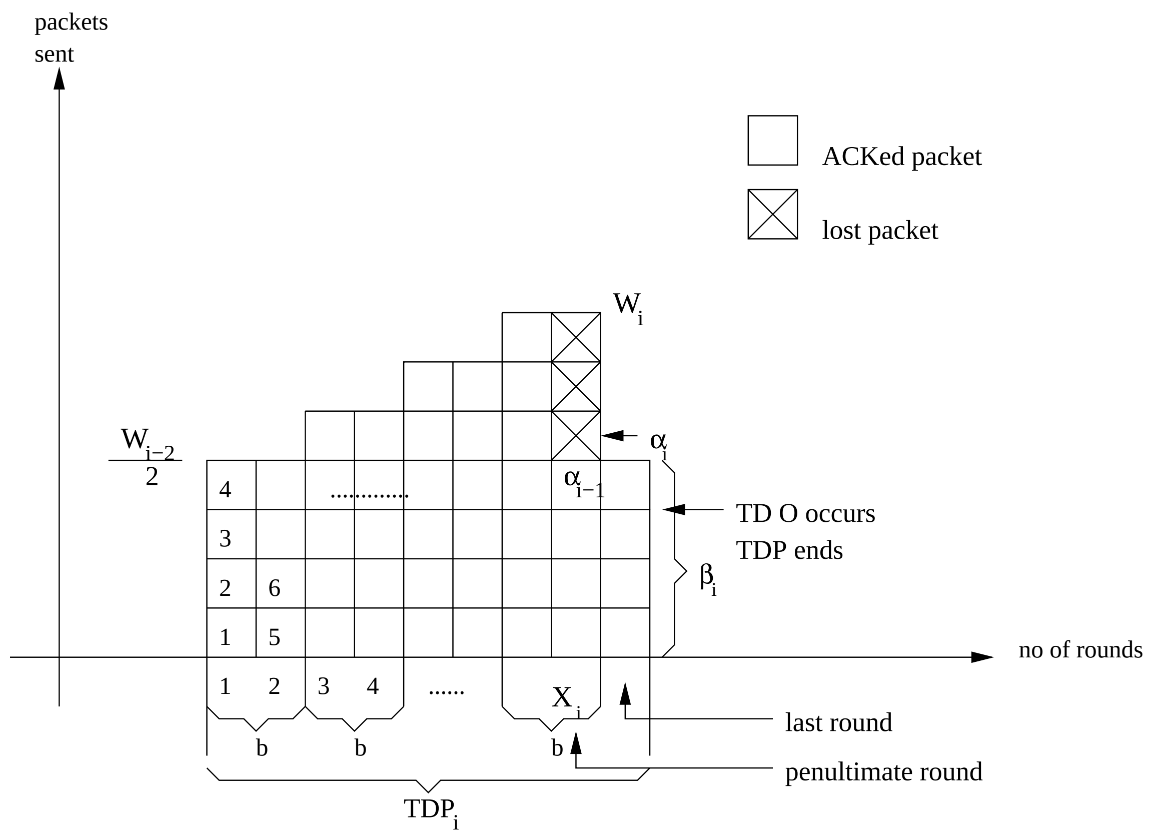

Figure 4.2 The steady state analysis.

$$W_i^{TD} = \frac{W_{i-1}^{TD}}{2} + \frac{X_i}{b} - 1 \tag{4.25}$$

and

$$\begin{aligned} Y_i &= \sum_{k=0}^{X_i/b-1} (\frac{W_{i-1}^{TD}}{2} + k)b + \beta_i \qquad (4.26) \\ &= \frac{X_i}{2}(\frac{W_{i-1}^{TD}}{2} + W_i^{TD} - 1) + \beta_i \qquad (4.27) \end{aligned}$$

where X_i is the penultimate round in the TDP which experiences packet losses, r_{ij} is the round trip time, W_i^{TD} is the window size at the end of a TDP, α_i is the number of packets sent in a TDP until the first loss happens, and β_i is the number of packets sent in the fast retransmit phase, which is the last round [17].

α_i is a geometrically-distributed random variable based on the independent and burst loss assumptions. Hence,

$$P[\alpha_i = k] = (1-p)^{k-1}p, \qquad k = 1, 2, \ldots \tag{4.28}$$

Based on Equations (4.23) and (4.28), we get:

$$\begin{aligned} E[Y] &= E[\alpha] + E[W^{TD}] - 1 \qquad (4.29) \\ &= \frac{1-p}{p} + E[W^{TD}]. \qquad (4.30) \end{aligned}$$

From Equation (4.25) and (4.27), we can have the following:

$$\begin{aligned} E[X] &= b(\frac{E[W^{TD}]}{2} + 1) \qquad (4.31) \\ E[Y] &= \frac{E[X]}{2}(\frac{E[W^{TD}]}{2} + E[W^{TD}] - 1) + E[\beta], \qquad (4.32) \end{aligned}$$

where $E[X]$ is the expected number of rounds in a single TDP. We assume X_i and W_i^{TD} to be mutually independent i.i.d. random variables. Then, from Equations (4.30), (4.32) and (4.31), we get the following expression which involves $E[W^{TD}]$:

$$\begin{aligned} \frac{1-p}{p} + E[W^{TD}] &= \frac{E[X]}{2}(\frac{E[W^{TD}]}{2} + E[W^{TD}] - 1) + E[\beta] \\ &= \frac{b(\frac{E[W^{TD}]}{2}+1)}{2}(\frac{E[W^{TD}]}{2} + E[W^{TD}] - 1) + E[\beta]. \end{aligned} \quad (4.33)$$

As we know, β_i is the number of packets sent when k packets in the penultimate round are ACKed. Thus, the value of β_i should be equal to k, whose probability can be described by:

$$A(w,k) = \frac{(1-p)^k p}{1-(1-p)^w}. \quad (4.34)$$

Now we can obtain the expected value of β, $E[\beta]$, using a conditional probability on the window size at the end of a TDP, in the following manner:

$$\begin{aligned} E[\beta] &= E[\sum_{k=0}^{w-1} k \cdot P(\beta = k)|w] \\ &= E[\sum_{k=0}^{w-1} \frac{k(1-p)^k p}{1-(1-p)^w}|w] \\ &= E[\frac{(1-p)(1-pw(1-p)^{w-1}-(1-p)^w)}{p(1-(1-p)^w)}|w]. \end{aligned} \quad (4.35)$$

When the loss rate p is very small, Equation (4.35) can be simplified as:

$$E[\beta] = (E[W^{TD}] - 1)(1-p). \quad (4.36)$$

Combining (4.36) with(4.33), we get:

$$E[W^{TD}] = -\frac{2(b-2p)}{3} + \sqrt{\frac{4(bp+2(1-p^2))}{3bp} + (\frac{2b-4p}{3b})^2} \quad (4.37)$$

The value of $E[W^{TD}]$ computed above is smaller than the value of $E[W^{TD}]$ given in [17]. This can be observed in Figure 4.3, where the majority of the difference is around 2 packets. But this difference is trivial compared to the values of the window size ($E[W^{TD}]$) when p is very small. However, if p is in the middle to high range, the difference is not small enough to be ignored when compared with the window size. Such a difference can have a significant impact on the estimation of TCP send rate or throughput.

Combining (4.31) and (4.24), we have the expected number of rounds in a single of TDP, $E[X]$ as:

$$E[X] = \frac{(2p+3)b-b^2}{3} + \sqrt{\frac{b^2p+2b(1-p^2)}{3p} + (\frac{b-2p}{3})^2}, \tag{4.38}$$

and the expected value of A, the duration of a TDP, as

$$\begin{aligned} E[A] &= (E[X]+1)E[r] \\ &\approx RTT(E[X]+1) \\ &= RTT\Bigg(-\frac{b^2-(2p+6)b}{3} + \sqrt{\frac{b^2p+2b(1-p^2)}{3p} + (\frac{b-2p}{3})^2}\Bigg), \end{aligned} \tag{4.39}$$

where we assume r_{ij} to be i.i.d. random variables and $E[r] \approx RTT$.

In the previous subsection, we stated without proof that the slow-start phase will enter the congestion-avoidance phase before a packet loss happens. This can be proved if $E[W^{ss}]^*$, the expected congestion window size at the end of the slow-start phase due to a packet loss, is bigger than the value of $E[ssthresh] = E[W^{TD}]/2$, which is the expected threshold at the beginning of the slow-start phase. In other words, we need to prove:

$$\frac{1+p}{pX_1^2} \geq \frac{E[W^{TD}]}{2},$$

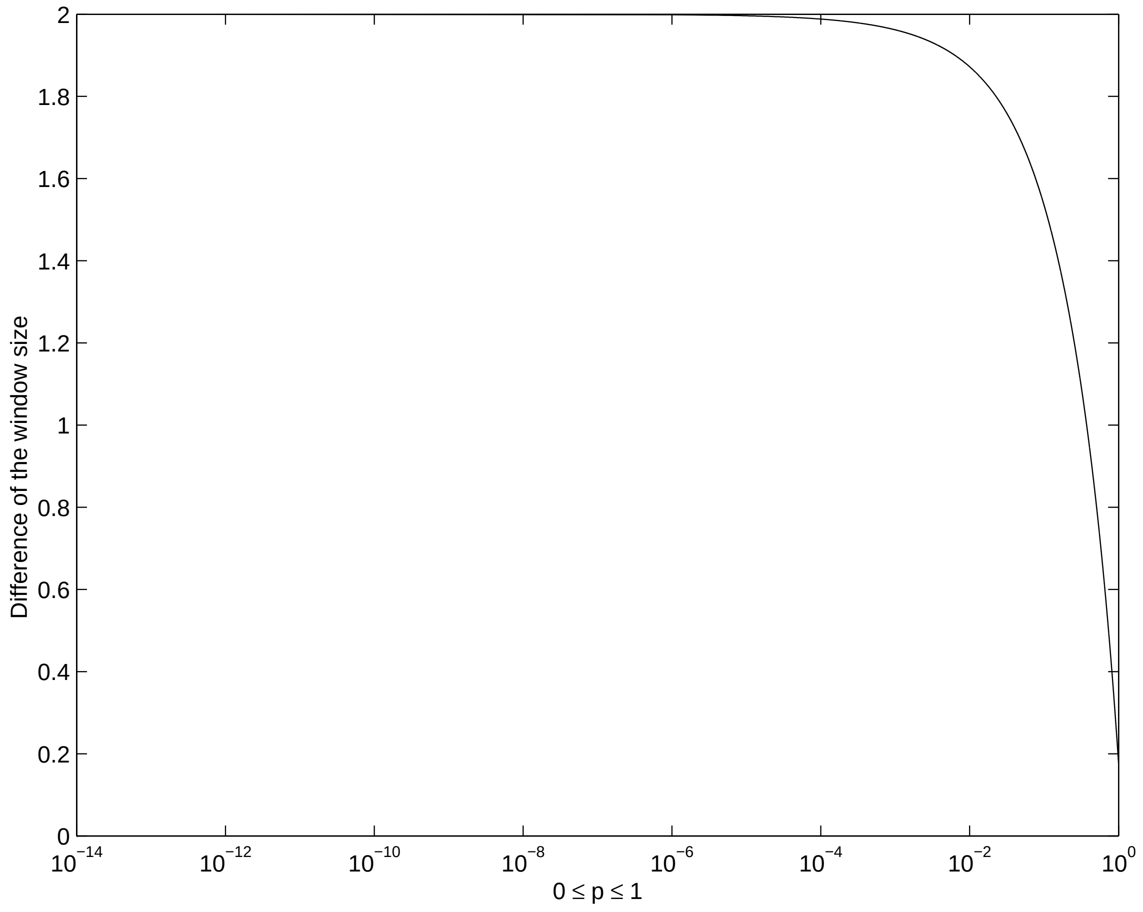

Figure 4.3 The difference of the expected value of window size.

where $E[W^{TD}]$ is given by (4.37). We prove this inequality below, under the normal condition where p is small:

$$\begin{aligned} \frac{1+p}{pX_1^2} &\geq \sqrt{\frac{2}{3bp}} \\ \Leftrightarrow 1 - 0.3p + p^2 &\geq 0 \end{aligned}$$

The last inequality stands obviously.

4.2.3 The Time-out Phase

During the congestion-avoidance phase, the probability that a loss indication is a time-out under the current congestion window size w, is given in [17] as:

$$Q^{TD}(w) = min(1, \frac{(1-(1-p)^3)(1+(1-p)^3(1-(1-p)^{w-3}))}{1-(1-p)^w}) \tag{4.40}$$

which gets simplified when the loss rate, p, is small:

$$Q^{TD}(w) = min(1, \frac{3}{w}). \tag{4.41}$$

Thus, Q^{TD}, the expected probability that a loss leads to a time-out at the end of the congestion-avoidance phase, is approximated as follows [17]:

$$\begin{aligned} Q^{TD} &= E[Q^{TD}(w)] \\ &= min(1, \frac{3}{E[W^{TD}]}). \end{aligned} \tag{4.42}$$

The traffic traces collected in [17] indicate that the effect of the time-outs must always be captured by any TCP performance prediction model. In most of the traces, time-out events out numbered the fast retransmit events, i.e., Q^{TD} is around 90% of the total loss. This value is larger than the value given by the formula of (4.42), as we further calculated that the $E[W^{TD}]$ is greater than 10, which, in turn, renders the Q^{TD} to be less than 30%. So, we believe that this approximation underestimates the real Q^{TD}. As a matter of fact, the underestimation of Q^{TD} in [17] is due to the approximation of $E[1/W] \approx 1/E[W]$ by noting that:

$$\begin{array}{rcl} E[(\frac{1}{\sqrt{W}})(\sqrt{W})]^2 & \leq & E[(\frac{1}{\sqrt{W}})^2]E[(\sqrt{W})^2] \\ \Longrightarrow \qquad \frac{1}{E[W]} & \leq & E[\frac{1}{W}] \end{array}$$

The equality holds only when $1/\sqrt{W} = \sqrt{W}$ or when $W = 1$. As we know, the window size is always bigger than 1, except at the start of a connection. In the following, we derive a more accurate expression of Q^{TD}.

Using Taylor's formula and expectation properties, we get the following [6]:

$$E[\frac{1}{W}] \approx \frac{1}{E[W]}(1 + \frac{Var(W)}{E[W]^2}). \tag{4.43}$$

The above approximation[7] holds when $E[W]^{(i+1)} \gg E[(W - E[W])^i]$. Hence, to find a better approximation of Q^{TD} we need to find the variance of W.

After rigorous analysis[8], we obtain the variance of W^{TD}, the congestion window size at the end of TDP, as follows:

$$Var[W^{TD}]_{p\to 0} \approx \frac{8(\sqrt{3}-1)}{3bp}. \tag{4.44}$$

[6]For complete derivations, please refer to Appendix A

[7]From (4.43), we can also prove $E[1/W] \geq 1/E[W]$ since $Var(W), E[W]$ are both greater than 0.

[8]Please refer to Appendix B for details

Substituting Equations (4.37) and (4.44) into Equation (4.43), we get:

$$\begin{aligned} E[\frac{1}{W^{TD}}] &= \frac{1}{E[W^{TD}]}(1+\frac{Var(W^{TD})}{E[W^{TD}]^2}) \\ &= \frac{1}{E[W^{TD}]}(1+\frac{\frac{8(\sqrt{3}-1)}{3bp}}{\frac{8}{3bp}}) \\ &= \frac{\sqrt{3}}{E[W^{TD}]}. \end{aligned} \quad (4.45)$$

Thus, Equation(4.45) gives a better, but still simple, estimation of $E[1/W^{TD}]$. Then, Q^{TD}, the probability that a loss detection is a time-out (TO), is obtained as:

$$\begin{aligned} Q^{TD} &= E[Q^{TD}(w)] \\ &= E[min(1,\frac{3}{w})] \\ &= min(1,3E[\frac{1}{W^{TD}}]) \\ &\approx min(1,\frac{3\sqrt{3}}{E[W^{TD}]}). \end{aligned} \quad (4.46)$$

The probability of n_i, the number of TDPs, is derived according to Q^{TD}:

$$p(n_i=k)=(1-Q^{TD})^{(k-1)}\cdot Q^{TD}.$$

This is due to the fact that, with probability Q^{TD}, the packets lost at the end of the congestion control phase lead to a TO, and, with probability $1-Q^{TD}$ the TCP connection stays in TDP.

By taking the expectation of n_i, we get:

$$E[n] = \frac{1}{Q^{TD}}. \quad (4.47)$$

The expressions for the number of packets sent in the time-out phase, $E[R]$ and its duration, $E[Z^{TO}]$ are given in [17] as:

$$\begin{aligned} E[R] &= \frac{1}{1-p} && (4.48) \\ E[Z^{TO}] &= T_0 \frac{f(p)}{1-p}, && (4.49) \end{aligned}$$

where $f(p)$ is defined as:

$$f(p) = 1 + p + 2p^2 + 4p^3 + 8p^4 + 15p^5 + 32p^6. \tag{4.50}$$

Utilizing these equations, we obtain the send rate in the following subsection.

4.2.4 The Steady State Send Rate and Throughput

Substituting Equations (4.21), (4.22), (4.30), (4.37), (4.39), (4.46), (4.47), (4.48) and (4.49) into (4.3), and taking into consideration the limitation of the window size[9], we finally derive the send rate as:

$$B(p, RTT) = \begin{cases} \dfrac{\frac{E[W^{TD}]g^2}{2} - 2 + \frac{1}{Q^{TD}(E[W^{TD}])}(\frac{1-p}{p} + E[W^{TD}]) + \frac{1}{1-p}}{\left(log_g(\frac{E[W^{TD}]}{2C_1}) + \frac{1}{Q^{TD}(E[W^{TD}])}(\frac{bE[W^{TD}]}{2} + b + 1)\right)RTT + \frac{f(p)T_0}{1-p}} \\ \qquad\qquad \text{when } E[W^{TD}] < W_m \\ \dfrac{\frac{W_m g^2}{2} - 2 + \frac{1}{Q^{TD}(W_m)}(\frac{1-p}{p} + W_m) + \frac{1}{1-p}}{log_g(\frac{W_m}{2C_1})RTT + \frac{1}{Q^{TD}(W_m)}((\frac{b}{8}W_m + \frac{1-p}{pW_m} + 2) + 1)RTT + \frac{f(p)T_0}{1-p}} \\ \qquad\qquad \text{when } E[W^{TD}] \geq W_m. \end{cases} \tag{4.51}$$

This can be further simplified as

$$B(p, RTT) = min(\frac{W_m}{RTT}, \frac{1}{RTT\sqrt{\frac{2bp}{3}} + min\left(1, 9\sqrt{\frac{bp}{8}}\right)p(\frac{RTT}{2}log_g(\frac{2}{3bpC_1^2}) + T_0(1+32p^2))}). \tag{4.52}$$

[9]we used the result from [17]

To derive the throughput, we only need to change $E[Y]$, the expected size of packets that have been sent in a TDP, to $E[Y']$, the expected size of packets that have been received in a TDP. $E[Y']$ can be expressed as:

$$E[Y'] = E[\alpha] + E[\beta] - 1, \tag{4.53}$$

where $E[\alpha]$ is $1/p$ and $E[\beta]$ is given by Equation (4.36). Also we substitute $E[R]$ with $E[R']$, the expected number of packets received in the time out phase, where [17]

$$E[R'] = 1 \tag{4.54}$$

Thus, the throughput can be formulated as:

$$H = \frac{E[Y^{ss}] + E[n]E[Y'] + E[R']}{E[Z^{ss}] + E[n]E[A] + E[Z^{TO}]} \tag{4.55}$$

or

$$H(p, RTT) = \begin{cases} \dfrac{\frac{E[W^{TD}]g^2}{2} - 2 + \frac{1}{Q^{TD}(E[W^{TD}])}(\frac{1-p}{p} + (E[W^{TD}]-1)(1-p)) + 1}{\left(log_g(\frac{E[W^{TD}]}{2C_1}) + \frac{1}{Q^{TD}(E[W^{TD}])}(\frac{bE[W^{TD}]}{2} + b + 1)\right)RTT + \frac{f(p)T_0}{1-p}} \\ \qquad \text{when } E[W^{TD}] < W_m \\ \dfrac{\frac{W_m g^2}{2} - 2 + \frac{1}{Q^{TD}(W_m)}(\frac{1-p}{p} + (W_m - 1)(1-p)) + 1}{log_g(\frac{W_m}{2C_1})RTT + \frac{1}{Q^{TD}(W_m)}((\frac{b}{8}W_m + \frac{1-p}{pW_m} + 1) + 1)RTT + \frac{f(p)T_0}{1-p}} \\ \qquad \text{when } E[W^{TD}] \geq W_m \end{cases} \tag{4.56}$$

which, when p is small, can be simplified as (4.52). This can be explained by noting that, if a loss seldom happens, then the send rate should just equal the throughput.

Figure 4.4 shows the relationship between our proposed steady state model and the model from [17]. It presents the differences of the throughput predictions under different packet loss rates (p) given by our model from Equation (4.56), and the full model of [17], which is described by:

$$H = \begin{cases} \dfrac{\frac{1-p}{p} + \frac{W(p)}{2} + Q(p, W(p))}{RTT(W(p)+1) + \frac{Q(p,W(p))f(p)T_0}{1-p}} & \text{where } W(p) \leq W_m \\ \dfrac{\frac{1-p}{p} + \frac{W_m}{2} + Q(p, W_m)}{RTT(\frac{W_m}{4} + \frac{1-p}{pW_m} + 2) + \frac{Q(p,W_m)f(p)T_0}{1-p}} & \text{otherwise} \end{cases} \tag{4.57}$$

Obviously when the loss rate approaches zero, both models give the same maximum performance predictions which is determined by W_m/RTT. But when p becomes larger, our model's predictions are smaller than those of the model from [17]. This is due to the introduction of the slow-start phase into the steady state model. However, when p approaches one, again both models give the same predictions because the throughput is near zero.

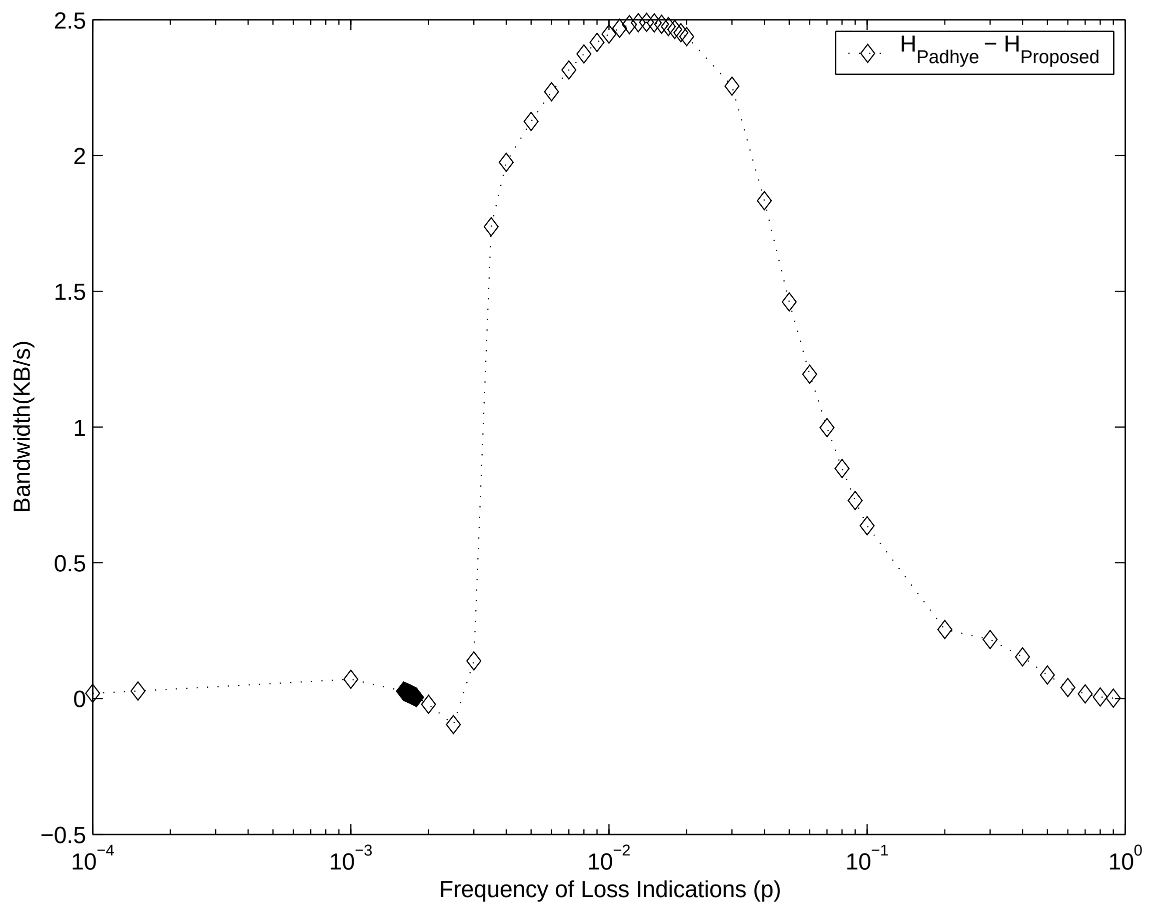

Figure 4.4 The difference of throughput predictions given by our proposed model and the model from Padhye's paper. The conditions are: $RTT = 200ms$, $MSS = 536bytes$, $w_1 = 1segment$, $T_0 = 1sec$, $W_m = 20segments$, $b = 2$.

CHAPTER V

SHORT-LIVED TCP CONNECTION MODEL

In the above sections, we derived the steady state model for predicting the performance of bulk data TCP connections. However, the steady state model is only applicable for either of the following conditions:

1. The loss rate (p) is high.
2. There are unlimited data to be sent.

Several recent studies [6, 1] have empirically observed that most TCP connections are short-lived. These short-lived connections have a common property, i.e., they usually spend their life time in the first slow-start phase without experiencing a single loss. Thus the steady-state model can not be used to predict the performance of these short-lived TCP flows. Therefore, a new model is needed.

Our proposed model is partially based on our results given in Section 4.2.1.

Every TCP connection starts with the three-way-handshake process. Assuming that no ACK packets can be lost, this process can be well modeled as follows [4]:

$$E[T_{twhs}] = RTT + T_s(\frac{1-p}{1-2p} - 1) \tag{5.1}$$

where T_s is the duration of SYN time-out.

We further assume that two or more time-outs within the three-way-handshake process is very rare. Otherwise the slow start threshold will be set to one, and therefore, the connection will be forced into the congestion-avoidance phase directly instead of into the slow-start phase.

5.1 The First Slow Start

After the three-way-handshake, the connection moves to the slow-start phase. In this phase, the sender's congestion window ($cwnd$) increases exponentially until either of the following two events occur: a packet gets lost or the $cwnd$ is bounded by the receiver's maximum congestion window size, W_m. In order to derive the latency for this phase, $E[Y_{init}]$, the expected number of packets sent until a loss occurs is given by the following enhanced equation (based on the one given in [4]):

$$E[Y_{init}] = \frac{(1-(1-p)^d)(1-p)}{p} \tag{5.2}$$

where d is the total file size measured in packets that must be transmitted. Using the same derivation as in Section 4.2.1, and according to Equation (4.15), we obtain the expected congestion window size at the end of the slow-start phase due to the packet losses as:

$$E[W_{init}] = \frac{E[Y_{init}] + 2}{g^2} \tag{5.3}$$

If $E[W_{init}]$ is bigger than the value of W_m, then the congestion window first grows to W_m and then remains there while sending the rest of the packets. Thus, the whole procedure is divided into two parts [4]. From Equation (4.15), the number of packets sent when the $cwnd$ grows to W_m is given by:

$$data_1 = g^2 \cdot W_m - 2. \tag{5.4}$$

Substituting (5.4) into (4.13), we can obtain the duration of this step measured in rounds as[1]:

$$n_1 = log_g(\frac{W_m}{C_1}). \tag{5.5}$$

In the second part,

$$n_2 = \frac{E[Y_{init}] - data_1}{W_m} \tag{5.6}$$

rounds are needed to transmit the remaining $E[Y_{init}] - data_1$ packets.

Combining the previous results and using Equation (4.13) for the $E[W_{init}] \leq W_m$ case, the expected slow-start latency is calculated as follows:

$$E[n] = \begin{cases} [\lceil log_g(\frac{W_m}{C_1})\rceil + \frac{1}{W_m}(E[Y_{init}] - g^2 W_m - 2)] & \text{when } E[W_{init}] > W_m \\ \lceil log_g(\frac{E[Y_{init}]+2}{C_1})\rceil - 2 & \text{when } E[W_{init}] \leq W_m \end{cases} \tag{5.7}$$

5.2 The First Loss

The initial slow-start phase ends when a packet loss is detected with a probability of $1 - (1-p)^d$. When a packet gets lost, it could cause re-transmission time-out (RTO), or lead to a triple duplicate ACKs in which case TCP will recover in a round or two by using the fast re-transmit and recovery. So first we have to derive the probability that a packet loss leads to a time-out (TO).

[1]We have chosen g to represent the golden number instead of X_1

Due to the exponential growing pattern of $cwnd$ in the slow-start phase, Q_{init}, the probability that a loss is a TO is different from the probability when the sender is in the congestion-avoidance phase. Hence, Q_{init} is derived as follows (See Figure 5.1):

In the round where a TD (triple-duplicate) occurs, let W^{ss} be the current size of $cwnd$, which has a value w. k packets have successfully been transmitted and ACKed among a total of w packets that are sent. Since the connection is still in the slow-start phase, the $cwnd$ increases to $w + k$ and another $2k$ packets are sent in the next round[2]. If more than three packets in these $2k$ packets have been ACKed, then a TD occurs. Otherwise a TO takes place. Let $A(w, k)$ be the probability that the first k packets have been successfully transmitted and ACKed in a round of w packets, provided that there might be one or more packets loss, which is given by Equation (4.34). Also, let $h(m)$ be the probability that no more than 2 packets have been transmitted successfully in a round of m packets, which can be given as:

$$h(m) = \sum_{i=0}^{2} (1-p)^i p \qquad \text{if } m \geq 3 \tag{5.8}$$

We thus obtain Q_{init} as:

$$\begin{aligned} Q_{init}(W^{ss}) &= \begin{cases} 1, & W^{ss} \leq 2 \\ \sum_{k=0}^{1} A(W^{ss}, k) + \sum_{k=2}^{W^{ss}-1} A(W^{ss}, k) h(2k), & \text{otherwise} \end{cases} \\ &= min\left(1, \frac{p(2-p) + (1-(1-p)^3)(1-p)^2(1-(1-p)^{W^{ss}-2})}{1-(1-p)^{W^{ss}}}\right) \end{aligned} \tag{5.9}$$

[2]The delayed acknowledgment concept is not applied here, but we prove it later that it does not affect the analysis of the Q_{init}

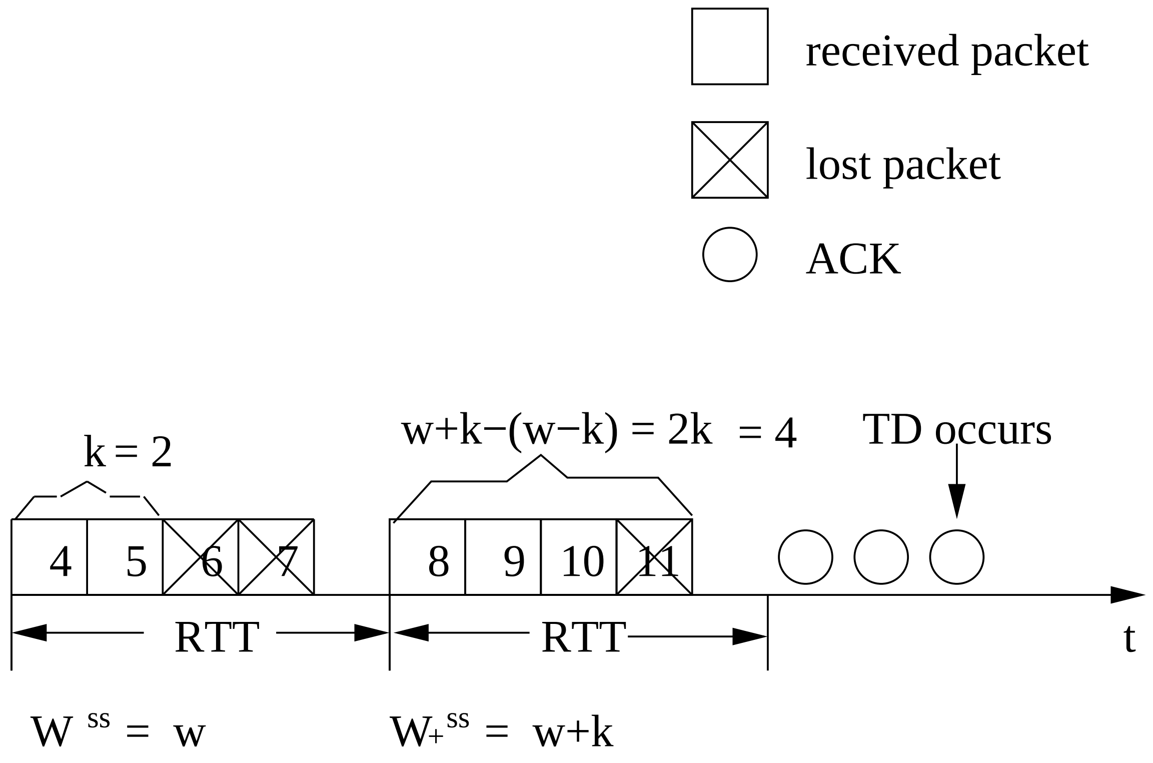

Figure 5.1 A sample situation when TD happens.

As p approaches zero, Equation (5.9) reduced to:

$$Q_{init} = \lim_{p \to 0} E[Q_{init}(W^{ss})] = min(1, \frac{2}{E[W^{ss}]}) \tag{5.10}$$

In case of delayed acknowledgment, k successfully received packets generate $\lfloor k/2 \rfloor$[3] ACKs, and thus the size of the $cwnd$ increases to $\lfloor k/2 \rfloor + w$ and $\lfloor k/2 \rfloor + k$ packets are sent. Therefore Q_{init} can be computed as:

$$Q_{init} = \begin{cases} 1, & W^{ss} \leq 2 \\ \sum_{k=0}^{1} A(W^{ss}, k) + \sum_{k=2}^{W^{ss}-1} A(W^{ss}, k) h(\lfloor \frac{k}{2} \rfloor + k), & \text{otherwise} \end{cases}$$

which is the same with Equation (5.9) since

$$h(2k) = h(\lfloor \frac{k}{2} \rfloor + k). \qquad \text{for } k \geq 2$$

The expected time which TCP spends in the RTOs is given in [17] and presented in Equation (4.49). The time which TCP spends in the fast re-transmit phase, n_t depends on where the loss occurs [20]:

$$n_t = \begin{cases} 2RTT, & \text{if the lost packet is in the last} \\ & \text{three packets of the window} \\ RTT, & \text{otherwise} \end{cases} \tag{5.11}$$

When the congestion window size W^{ss} is bigger than three, the expected time, $E[n_t]$ is computed as follows:

$$\begin{aligned} E[n_t] &= \frac{1-(1-p)^{W^{ss}-3}}{1-(1-p)^{W^{ss}}} \times 2RTT + \frac{(1-p)^{W^{ss}-3}(1-(1-p)^3)}{1-(1-p)^{W^{ss}}} \times RTT \\ &= RTT \cdot \frac{2-(1-p)^{W^{ss}-3}-(1-p)^{W^{ss}}}{1-(1-p)^{W^{ss}}} \end{aligned} \tag{5.12}$$

[3] $\lfloor k/2 \rfloor$ is the biggest integer small than $k/2$

Finally, the expected latency that this loss would incur is:

$$T_{loss} = (1 - (1-p)^d)(Q_{init}E[Z^{TO}] + (1 - Q_{init})E[n_t]) \tag{5.13}$$

where W^{ss} is

$$W^{ss} = min(W_m, \frac{E[Y_{init}] + 2}{g^2}). \tag{5.14}$$

5.3 Sending the Rest of the Packets

After the first packet loss, the transmission latency of the rest ($d - E[Y_{init}]$) packets is obtained by using our extended steady-state model as follows:

$$\begin{aligned} T_{rest} &= \frac{d - E[Y_{init}]}{H} \\ &= \frac{dp - (1 - (1-p)^d)(1-p)}{p \cdot H} \end{aligned} \tag{5.15}$$

in which Th is given by Equation (4.55).

5.4 Total Latency

Grouping (5.1), (5.7), (5.12) and (5.15) together and considering the delay (T_{delay}) caused by the delayed acknowledgment for the first packet (whose mean value is 100ms for the BSD-derived implementations), we now have the total expected latency:

$$T_{latency} = E[T_{twhs}] + E[n]RTT + T_{loss} + T_{rest} + T_{delay} - \frac{RTT}{2} \tag{5.16}$$

Note that the last term is due to the fact that only half of a round is needed to send the last window of packets. This short-lived TCP connection model is compared with our steady-state model in Figure 5.2. It shows that as the transferred file size increases, the short TCP model approaches the steady state model. This is because when the connection has a large amount of data to send, TCP would spend most of its time in the steady-state. Also, the figure illustrates that as the loss rate increases, the throughput predicted by the short-lived TCP model approaches the one predicted by the steady-state model. This is because as the connection loses its packets at higher probability, the transient slow-start phase ends quickly and the remaining packets are sent in the steady-state phase.

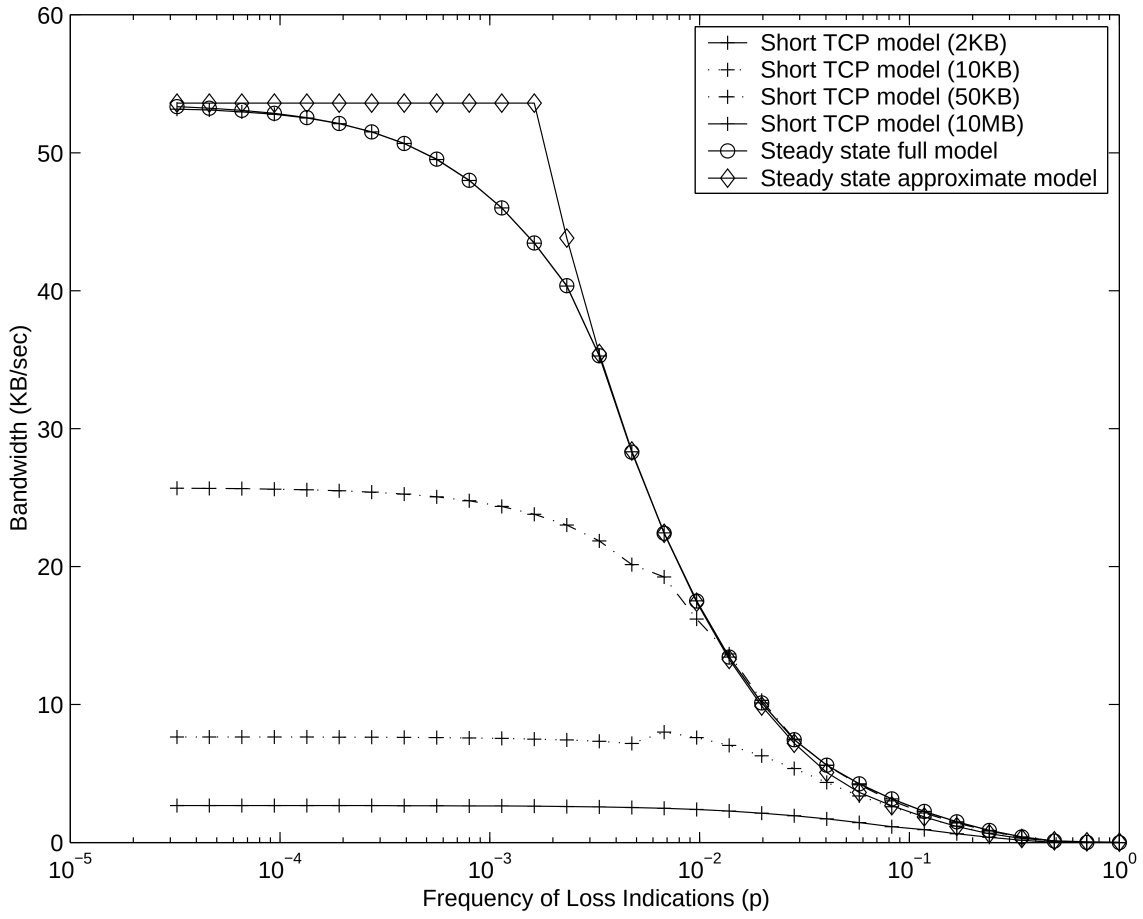

Figure 5.2 Throughput predictions given by the short TCP model and the steady state model. The conditions are: $RTT = 200ms$, $MSS = 536bytes$, $w_1 = 1segment$, $T_0 = 1sec$, $W_m = 20segments$, $b = 2$.

CHAPTER VI

MODEL VALIDATION THROUGH SIMULATION

In this chapter, we validate our proposed analytical models with simulation experiments. We performed all experiments in NS-2 [24] using the FullTCP agent. The FullTCP agent is modeled based on the 4.4BSD TCP implementation and can simulate all the important features of TCP Reno. The simulation topology used in all experiments is shown in Figure 6.1.

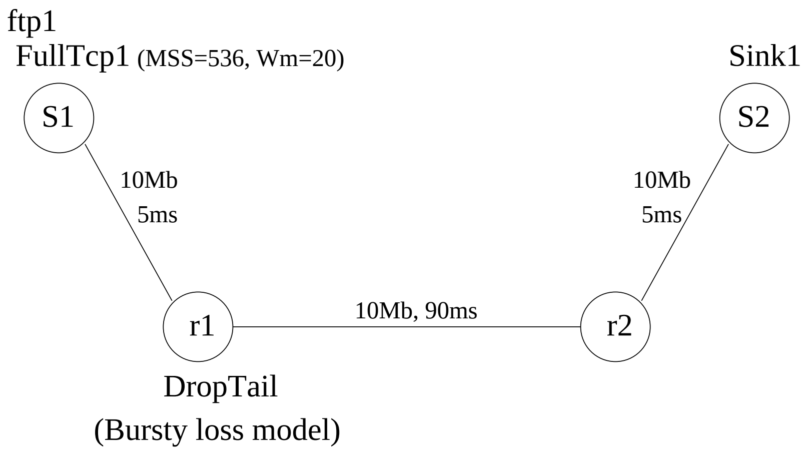

Figure 6.1 Simulation topology

Unlike in [4] where the Bernoulli loss model is used, in our experiments packets were getting lost according to the bursty loss model. Since NS-2 does not have a bursty model built-in, we added our own BurstyError Model, which was derived from the basic Error Model class. This BurstyError Model drops packets with probability p, which is a Bernoulli trial. After a packet is selected to be dropped with probability p, all the subsequent packets in transit are also dropped. This emulates the DropTail queues behavior under congestion conditions.

Since TCP does not care about what kind of data it transmits, we used FTP [1] as the application for sending a controlled number of packets over a 10Mbps link. The experiments were designed such that the minimum RTT was 200ms.

6.1 Steady-State Model

As stated previously, when the loss rate is middle-to-high, the steady-state phase has a significant impact on the performance of the short-lived TCP connections. Thus, the steady-state model is important for both bulk data transfers and short-lived TCP connections. We compared our steady-state model with the one developed in [17] [2]. Figure 6.2 shows the results for the case of $0.005 < p < 0.1$. For each value of p, we ran 1000 simulation experiments. The file size was set to 10MB. Clearly, our model's predictions are closer to the simulation results. We further compared our model with the one in [17] by using the average error criterion (as defined in [17]):

$$\frac{\sum_{observations} |Th_{predicted}(p) - Th_{observed}(p)|/Th_{observed}(p)}{Number\ of\ observations}, \tag{6.1}$$

[1]FTP is the major application used to transfer files in the network.

[2]See Figure 4.4

where $Th_{predicted}$ is the predicted value and $Th_{observed}$ is the observation value. A smaller average error implies a better model accuracy. We plotted these average errors against loss rates in Figure 6.3. It shows that in most cases the average error is 5% for our proposed full model and is 20% for the one in [17]. That is, our model is 75% more accurate than the model proposed in [17]. We can further observe from Figure 6.3 that the average error increases as p decreases. For these case, the short-lived TCP model should be used instead of the steady-state model. Nevertheless, the simulation results support our previous claim that by including the slow-start phase into the steady-state model more accurate predictions can be obtained.

6.2 Relation between the Transferred File Size and Latency

Figure 6.4 shows the relationship between the latency and the transferred file size under no loss conditions. It compares the latency predictions given by our proposed short-lived TCP model (Equation (5.16)) with the ones obtained by the short-lived TCP models of [4] and [20]. Again, our model's prediction values match the simulated values better than the values obtained by the other models. The average error is 5.83% for our model, 9.40% for the model of [4], and 14.53% for the model of [20].

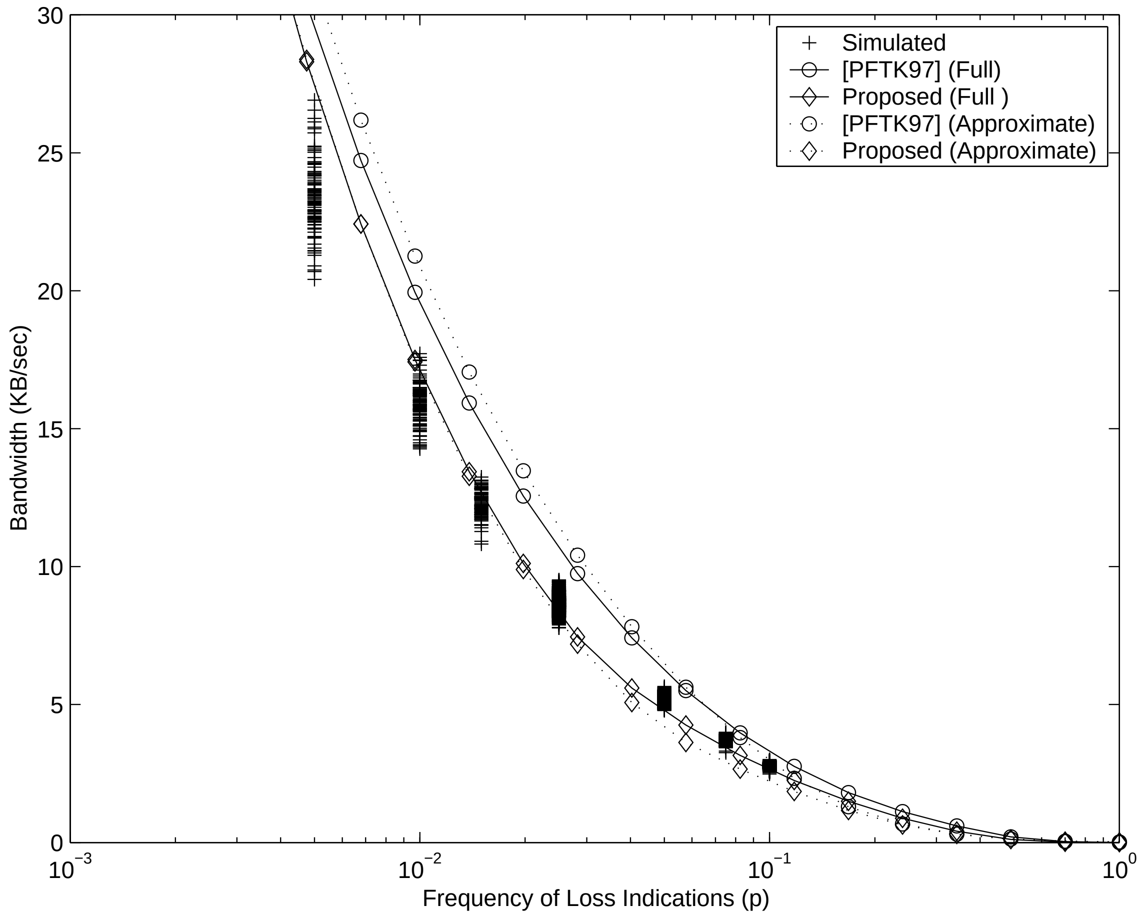

Figure 6.2 Comparing the steady state throughput predicted by the models in the middle-to-high loss rate range. The parameters are: $RTT = 200ms$, $MSS = 536bytes$, $w_1 = 1segment, T_0 = 1sec, W_m = 20segments.$

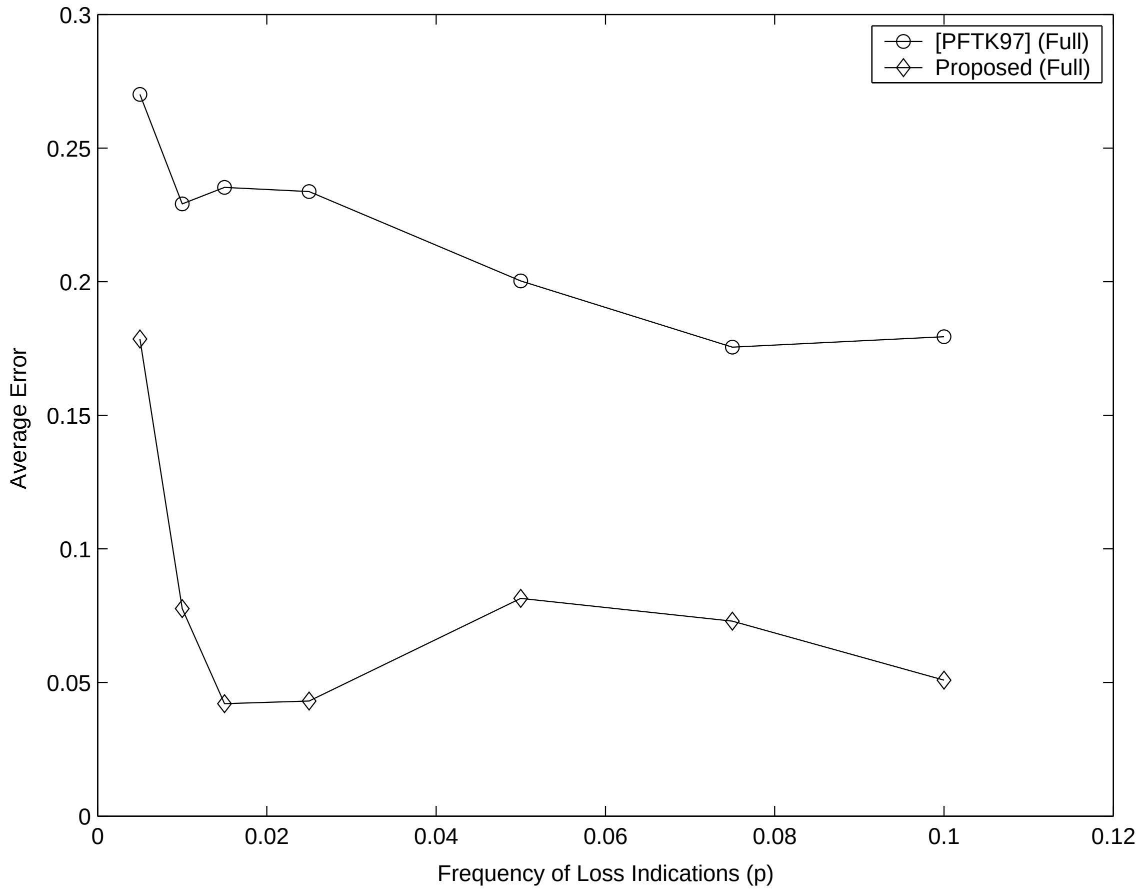

Figure 6.3 Average error comparison of the models in the middle-to-high loss rate range. The parameters are: $RTT = 200ms$, $MSS = 536bytes$, $w_1 = 1segment$, $T_0 = 1sec$, $W_m = 20segments$.

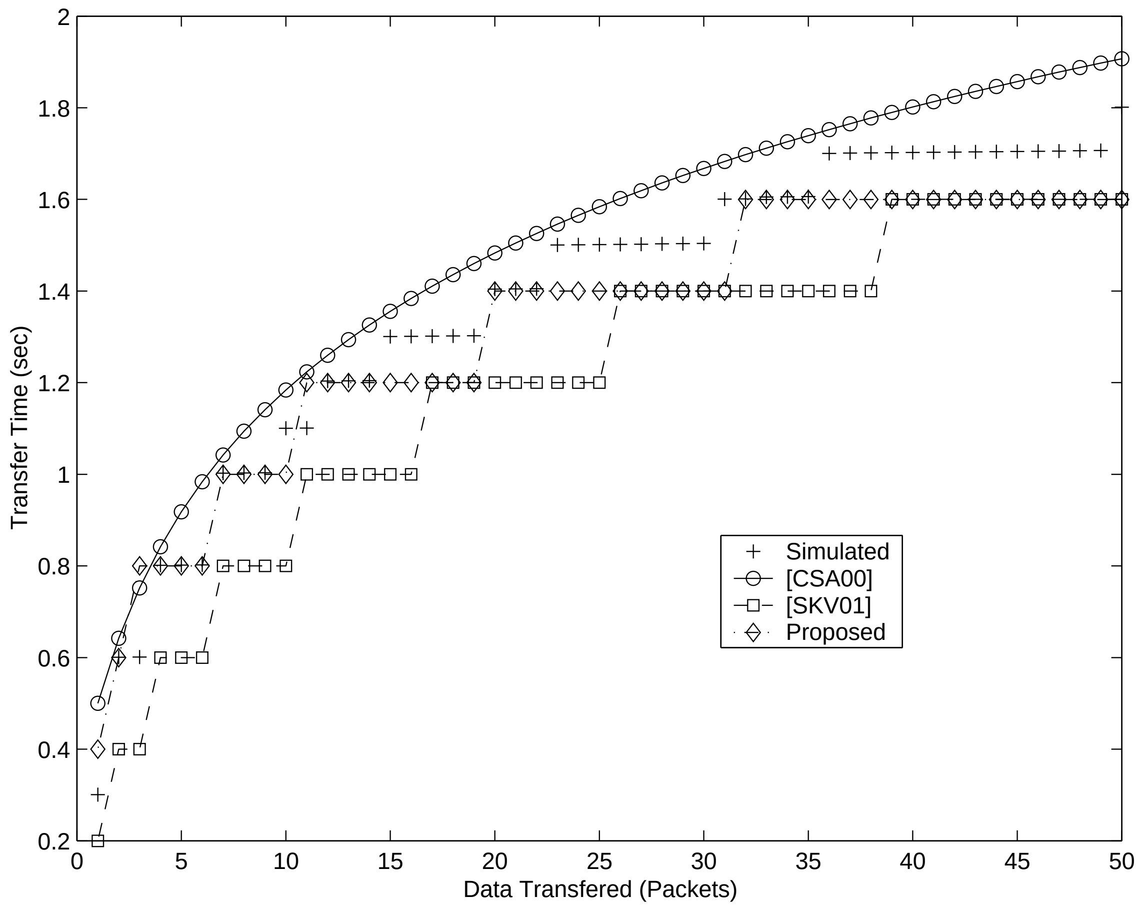

Figure 6.4 Comparing the latency predicted by the short connection models for small transferred file size. The parameters are: $p = 0$, $RTT = 100ms$, $MSS = 536bytes$, $w_1 = 1segment, T_0 = 1sec, W_m = 20segments$.

6.3 Relation between File Size, Loss Rate and Throughput

Figures 6.5, 6.6 and 6.7 compare the accuracy of our model with the one proposed in [4] in terms of throughput versus transferred file size and loss rate. Table 6.1 compares the two models in terms of the average error.

As can be observed, when the transferred file size is small and the loss rate is low, our model gives more accurate predictions than the model from [4]. This is because we include the delay acknowledgment mechanism in our model. This is also due to the fact that we use g in our prediction expression rather γ which is used in [4]. However, when the file size is big and the loss rate is high, both of the models agree closely with our steady-state model, as expected.

Table 6.1 Comparison of the predictions average error.

Loss Rate	$p = 0$	$3 \times 10^{-3} \sim 10^{-1}$		
File Size	0.5 $\sim$ 26KB	2KB	6KB	11KB
[CSA00]	9.40%	4.08%	6.43%	8.38%
Proposed	5.83%	0.59%	7.54%	7.64%

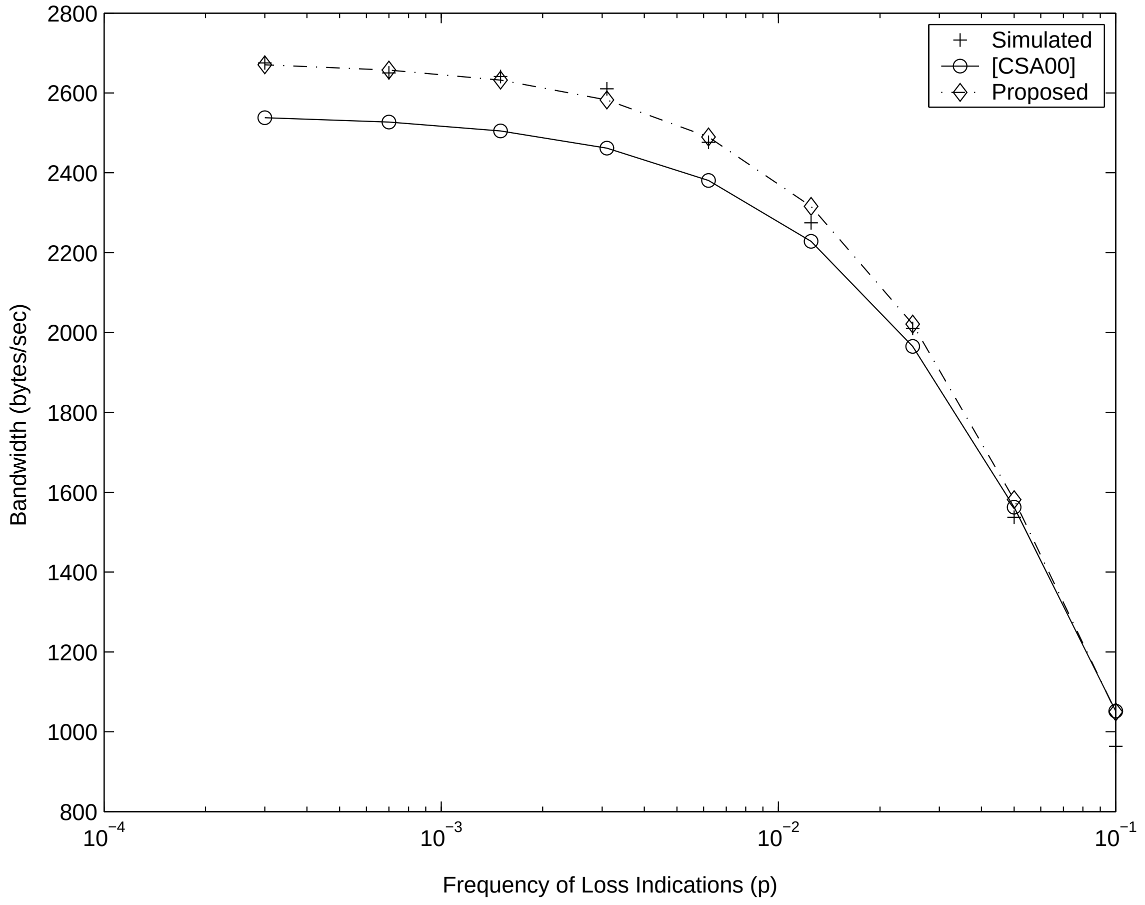

Figure 6.5 Comparing the throughput predicted by the models for varying loss rate. The transferred file size is fixed at $2KB$. The parameters are: $RTT = 100ms$, $MSS = 536bytes, w_1 = 1segment, T_0 = 1sec, W_m = 20segments$.

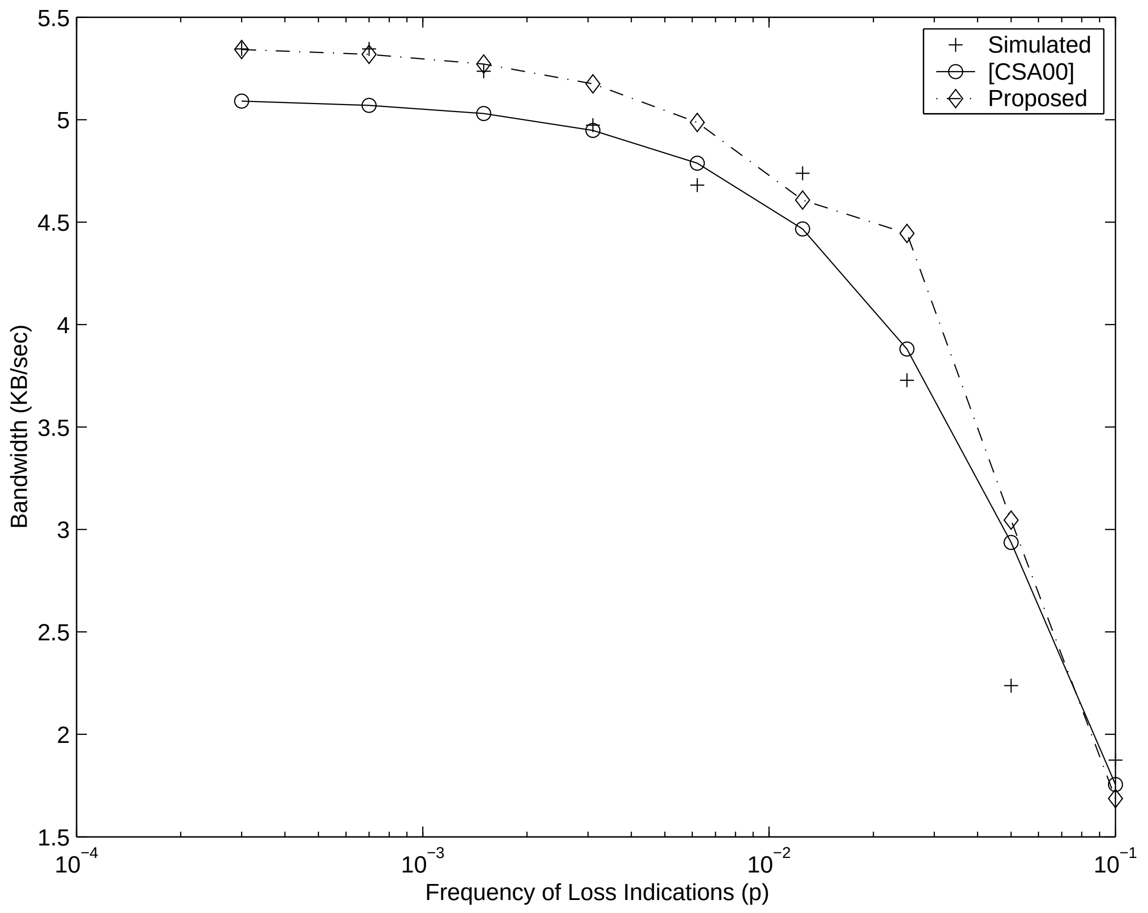

Figure 6.6 Comparing the throughput predicted by the models for varying loss rate. The transferred file size is fixed at $6kB$. The parameters are: $RTT = 100ms$, $MSS = 536bytes$, $w_1 = 1segment$, $T_0 = 1sec$, $W_m = 20segments$.

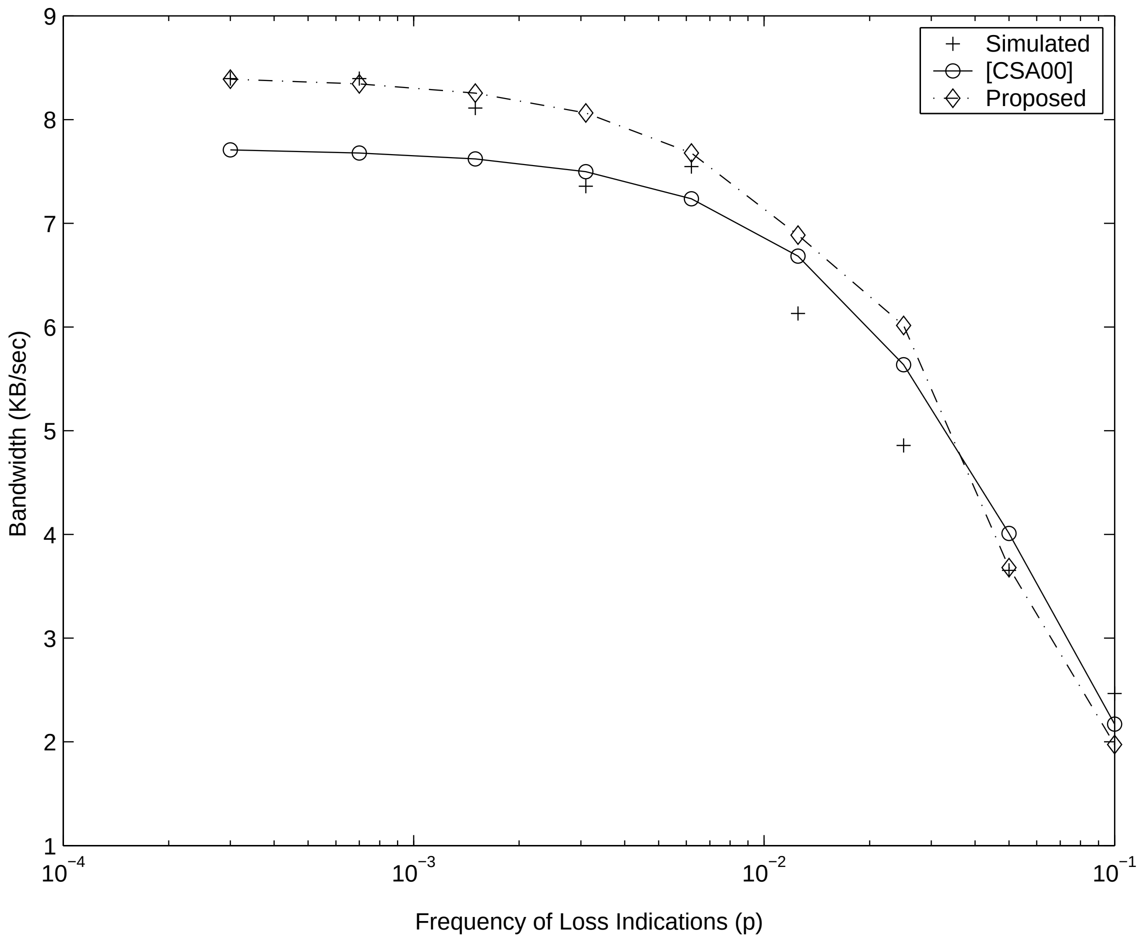

Figure 6.7 Comparing the throughput predicted by the models for varying loss rate. The transferred file size is fixed at $11kB$. The parameters are: $RTT = 100ms$, $MSS = 536bytes, w_1 = 1segment, T_0 = 1sec, W_m = 20segments.$

CHAPTER VII

CONCLUSIONS AND FUTURE WORK

7.1 Summary of the Achievements

In this thesis, we developed an extended and improved stochastic TCP model for predicting the latency and throughput. We constructs a new model for the slow-start phase based on discrete evolutions of the congestion window. This model is very important in determining the performance of short-lived TCP connections. We also integrated it into our extended steady-state model to achieve better performance predictions. The sending rate, B and the throughput, H are given by our model as follows based on the loss rate (p) and the round trip time (RTT):

$$B(p, RTT) = \begin{cases} \dfrac{\frac{E[W^{TD}]g^2}{2} - 2 + \frac{1}{Q^{TD}(E[W^{TD}])}(\frac{1-p}{p} + E[W^{TD}]) + \frac{1}{1-p}}{\Big(log_g(\frac{E[W^{TD}]}{2C_1}) + \frac{1}{Q^{TD}(E[W^{TD}])}(\frac{bE[W^{TD}]}{2} + b + 1)\Big)RTT + \frac{f(p)T_0}{1-p}} \\ \qquad\qquad \text{when } E[W^{TD}] < W_m \\ \dfrac{\frac{W_m g^2}{2} - 2 + \frac{1}{Q^{TD}(W_m)}(\frac{1-p}{p} + W_m) + \frac{1}{1-p}}{log_g(\frac{W_m}{2C_1})RTT + \frac{1}{Q^{TD}(W_m)}((\frac{b}{8}W_m + \frac{1-p}{pW_m} + 2) + 1)RTT + \frac{f(p)T_0}{1-p}} \\ \qquad\qquad \text{when } E[W^{TD}] \geq W_m, \end{cases} \tag{7.1}$$

or,

$$H(p, RTT) = \begin{cases} \dfrac{\frac{E[W^{TD}]g^2}{2} - 2 + \frac{1}{Q^{TD}(E[W^{TD}])}(\frac{1-p}{p} + (E[W^{TD}]-1)(1-p)) + 1}{\left(log_g(\frac{E[W^{TD}]}{2C_1}) + \frac{1}{Q^{TD}(E[W^{TD}])}(\frac{bE[W^{TD}]}{2} + b + 1)\right) RTT + \frac{f(p)T_0}{1-p}} \\ \qquad\qquad\qquad \text{when } E[W^{TD}] < W_m \\ \dfrac{\frac{W_m g^2}{2} - 2 + \frac{1}{Q^{TD}(W_m)}(\frac{1-p}{p} + (W_m-1)(1-p)) + 1}{log_g(\frac{W_m}{2C_1}) RTT + \frac{1}{Q^{TD}(W_m)}((\frac{b}{8} W_m + \frac{1-p}{pW_m} + 1) + 1) RTT + \frac{f(p)T_0}{1-p}} \\ \qquad\qquad\qquad \text{when } E[W^{TD}] \geq W_m, \end{cases} \tag{7.2}$$

where W_m is the maximum congestion window size, $E[W^{TD}], Q^{TD}, f(p)$ are defined in Equations (4.37), (4.46) and (4.50) respectively.

Our simulation results suggest that our model can predict the performance of bulk data transfers and short-lived TCP connections here accurately than the models proposed in [17, 4, 20].

7.2 Future Work

A number of avenues are left to be studied for future work. First in the constructing the steady-state model, we did not consider the effects of the fast recovery. Second, our model is not suitable for slow links such as telephone line using modem whose speed is usually less than $56Kb$. Third, we selected the bursty loss model to describe the packet loss behavior which is an approximation of the FIFO drop-tail queuing algorithm. Since the majority of router are now using RED as their queuing algorithm, it is important to repeat our analysis by using different loss models.

REFERENCES

[1] H. Balakrishnan, V. Padmanabhan, S. Seshan, R. H. Katz, and M. Stemm, "TCP behavior of a busy Internet server. Analysis and improvements," *Proceedings of the INFOCOM '98,* April 1998.

[2] J. Blot and T. Turletti, "Experience with rate control mechanisms for packet video in the Internet," *Computer Communications Review*, vol. 28, no. 1, January 1998.

[3] J. Bolliger, T. Gross, and U. Hengartner, "Bandwidth modeling for network-aware applications," *Proceedings of the INFOCOM'99,* March 1999.

[4] N. Cardwell, S. Savage, and T. Anderson, "Modeling TCP latency," *Proceedings of the INFOCOM '2000,* March 2000.

[5] K. Claffy, G. Miller, and K. Thompson, "The nature of the beast: Recent traffic measurements from an Internet backbone," *Proceedings of the INET '98*, July 1998.

[6] C. R. Cunha, A. Bestavros, and M. E. Crovella, *Characteristics of WWW client-based traces,* technical report BU-CS-95-010, Boston University, July 1995.

[7] S. Floyd and K. Fall, "Promoting the use of end-to-end congestion control in the Internet," *IEEE/ACM Transactions on Networking,* August 1999.

[8] J. Heidemann, K. Obraczka, and J. Touch, "Modeling the performance of HTTP over several transport protocols," *IEEE/ACM Transactions on Networking*, vol. 8, no. 2, April 2000.

[9] J. C. Hoe, *Start-up dynamics of TCP's congestion control an davoideance schemes,* master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, June 1995.

[10] V. Jacobson, "Congestion avoidance and control," *Proceedings of the SIGCOMM '88*.

[11] J. Mahdavi, "TCP performance tuning," April 1997, http://www.psc.edu/networking/tcptune/slides/.

[12] J. Mahdavi and S. Flyod, "TCP-Friendly Unicast Rate-Based Flow Control," Jan 1997, Note sent to end2end-interest mailing list.

[13] M. Mathis, J. Semske, J. Mahdavi, and T. Ott, "The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm," *Computer Communication Review*, vol. 27, no. 3, July 1997.

[14] M. Mitzenmacher and R. Rajaraman, "Towards more complete models of TCP latency and throughput," *Journal of Supercomputing*, vol. 20, no. 2, September 2001.

[15] T. Ott, J. Kemperman, and M. Mathis, "Promoting the use of end-to-end congestion control in the Internet," August 1996, ftp://ftp.bellcore.com/pub/tjo/TCPwindow.ps.

[16] T. J. Ott, T. V. Lakshman, and L. H. Wong, "SRED: Stabilized RED," *Proceedings of the INFOCOM '99*, March 1999.

[17] J. Padhye, V. Firoiu, D. F. Towsley, and J. F. Kurose, "Modeling TCP Reno Performance: A simple model and its empirical validation," *IEEE/ACM Transactions on Networking*, vol. 8, no. 2, April 2000, pp. 133–145.

[18] C. Partridge and T. J. Shepard, "TCP/IP performance over satellite links," *IEEE Network*, September 1997, pp. 44–49.

[19] J. Postel, "Transmission control protocol," September 1981, Request for Comments 793, DDN Network Information Center, SRI International.

[20] B. Sikdar, S. Kalyanaraman, and K. S. Vastola, "An integrated model for the latency and steady-state throughput of TCP connections," *Performance Evaluation*, vol. 46, October 2001.

[21] W. Stallings, *High-speed networks: TCP/IP and ATM design principles*, Prentice Hall, 1998.

[22] W. R. Stevens, *TCP/IP illustrated*, vol. 1, Addison Wesley, 1994.

[23] K. Thompson, G. J.Miller, and R. Wilder, "Wide-area Internet traffic patterns and characteristics," *IEEE Network*, vol. 11, no. 6, November 1997.

[24] UCB/LBNL/VINT, "The Network Simulator ns-2," May 2002, http://www.isi.edu/nsnam/ns/.

[25] L. Vivisano, L. Rizzo, and J. Crowcroft, "TCP-like congestion control for layered multicast data transfer," *Proceedings of the INFOCOM '98*, April 1998.

APPENDIX A

THE EXPECTATION OF 1/W

From Taylor formula, we know:

$$f(W) = \sum_{i=0}^{\infty} \frac{f^i(a)}{i!}(W-a)^i \tag{A.1}$$

Let $f(W)$ and a be $1/W$ and $E[W]$ respectively. We thus have:

$$f^n(W) = (-1)^n n! W^{-(n+1)} \tag{A.2}$$

Substituting $f^i(a)$ in Equation (A.1) and making use of Equation (A.2) and $E[W]$, we get:

$$\begin{aligned}\frac{1}{W} &= \sum_{i=0}^{\infty} \frac{(-1)^i i! E[W]^{-(i+1)}}{i!}(W - E[W])^i \\ &= \sum_{i=0}^{\infty} \frac{(-1)^i (W - E[W])^i}{E[W]^{(i+1)}} \end{aligned} \tag{A.3}$$

Taking expectation on both sides of Equation (A.3), results in:

$$\begin{aligned} E[\frac{1}{W}] &= E[\sum_{i=0}^{\infty} \frac{(-1)^i (W - E[W])^i}{E[W]^{(i+1)}}] \\ &= \sum_{i=0}^{\infty} E[\frac{(-1)^i (W - E[W])^i}{E[W]^{(i+1)}}] \\ &= \sum_{i=0}^{\infty} \frac{(-1)^i E[(W - E[W])^i]}{E[W]^{(i+1)}} \\ &= \frac{1}{E[W]} + \frac{Var(W)}{E[W]^3} + \sum_{i=3}^{\infty} \frac{(-1)^i E[(w - E[W])^i]}{E[W]^{(i+1)}} \\ &\approx \frac{1}{E[W]} + \frac{Var(W)}{E[W]^3} \\ &= \frac{1}{E[W]}(1 + \frac{Var(W)}{E[W]^2}) \end{aligned} \tag{A.4}$$

The approximation holds when $E[W]^{(i+1)} \gg E[(W - E[W])^i]$.

APPENDIX B

THE VARIANCE OF W^{TD}

Using similar assumptions as in the previous analysis, from (4.28), we know

$$Var[\alpha] = \frac{1-p}{p^2} \tag{B.1}$$

Thus from Equation (4.23):

$$Var[Y] = \frac{1-p}{p^2} + Var[W^{TD}] \tag{B.2}$$

From Equation (4.25), we get the auto-correlation at the zero point[1]:

$$\begin{aligned} R_w(0) &= \frac{R_w(0)}{4} + \frac{R_x(0)}{b^2} \\ R_x(0) &= \frac{3b^2}{4} R_w(0) \end{aligned} \tag{B.3}$$

And from Equations (4.34), (4.36) and (4.37), we can compute the variance of β as following:

$$\begin{aligned} Var[\beta] &= R_\beta(0) - E[\beta]^2 \\ &= E[\beta^2] - E[\beta]^2 \\ &= E[\sum_{k=0}^{w-1} k^2 p(\beta = k)|w] - E[\beta]^2 \\ &= E[\sum_{k=0}^{w-1} \frac{k^2(1-p)^k p}{1-(1-p)^w}|w] - E[\beta]^2 \\ &\approx \frac{2(1-p)^2}{p^2} - (1-p)[\sqrt{\frac{8}{3bp}} - 1]^2 \end{aligned} \tag{B.4}$$

From Equation (4.27), we can also get:

$$Var[Y] = Var[\frac{X_i}{2}(\frac{W_{i-1}^{TD}}{2} + W_i^{TD} - 1)] + Var[\beta]$$

[1]This is equal to $E[X^2]$

$$\begin{aligned}
&= E[(\frac{X_i}{2})^2]E[(\frac{W_{i-1}^{TD}}{2}+W_i^{TD}-1)^2] \\
&\quad -(E[\frac{X_i}{2}(\frac{W_{i-1}^{TD}}{2}+W_i^{TD}-1)])^2+Var[\beta] \\
&= \frac{R_x(0)}{4}\frac{5R_w(0)}{4}-E[\frac{X}{2}]^2E[\frac{W_{i-1}^{TD}}{2}+W_i^{TD}-1]^2+Var[\beta] \\
&= \frac{\frac{3b^2}{4}R_w(0)}{4}\frac{5R_w(0)}{4}-\frac{E[X]^2}{4}[\frac{3}{2}E[W^{TD}]-1]^2+Var[\beta] \\
&= \frac{15b^2}{64}[Var[W^{TD}]+E[W^{TD}]^2]^2-\frac{E[X]^2}{4}[\frac{3}{2}E[W^{TD}]-1]^2+Var[\beta] \\
&\approx \frac{15b^2}{64}[Var[W^{TD}]+\frac{8}{3bp}]^2-\frac{1}{4}\Big(\frac{b}{2}\sqrt{\frac{8}{3bp}}+b\Big)^2\Big(\frac{3}{2}\sqrt{\frac{8}{3bp}}-1\Big)^2 \\
&\quad +\frac{2(1-p)^2}{p^2}-(1-p)[\sqrt{\frac{8}{3bp}}-1]^2 \qquad \text{(B.5)}
\end{aligned}$$

Combining (B.5) and (B.2), we obtain the final equation:

$$\begin{aligned}
\frac{1-p}{p^2}+Var[W^{TD}] &= \frac{15b^2}{64}[Var[W^{TD}]+\frac{8}{3bp}]^2 \\
&\quad -\frac{1}{4}\Big(\frac{b}{2}\sqrt{\frac{8}{3bp}}+b\Big)^2\Big(\frac{3}{2}\sqrt{\frac{8}{3bp}}-1\Big)^2 \\
&\quad +\frac{2(1-p)^2}{p^2}-(1-p)[\sqrt{\frac{8}{3bp}}-1]^2 \qquad \text{(B.6)}
\end{aligned}$$

Solving Equation(B.6), we obtain the variance of W^{TD} as follows:

$$Var[W^{TD}]_{p\to 0}\approx\frac{8(\sqrt{3}-1)}{3bp} \qquad \text{(B.7)}$$