Modeling TCP Throughput: A Simple Model and its Empirical Validation *

Jitendra Padhye	Victor Firoiu †	Don Towsley	Jim Kurose
jitu@cs.umass.edu	vfiroiu@cs.umass.edu	towsley@cs.umass.edu	kurose@cs.umass.edu
1-413-545-2447	1-413-545-3179	1-413-545-0207	1-413-545-1585

Fax: 1-413-545-1249
Department of Computer Science
University of Massachusetts
LGRC, Box 34610
Amherst, MA 01003-4610 USA

May 30, 1998

Abstract

In this paper we develop a simple analytic characterization of the steady state throughput, as a function of loss rate and round trip time for a bulk transfer TCP flow, i.e., a flow with an unlimited amount of data to send. Unlike the models in [6, 7, 10], our model captures not only the behavior of TCP's fast retransmit mechanism (which is also considered in [6, 7, 10]) but also the effect of TCP's timeout mechanism on throughput. Our measurements suggest that this latter behavior is important from a modeling perspective, as almost all of our TCP traces contained more timeout events than fast retransmit events. Our measurements demonstrate that our model is able to more accurately predict TCP throughput and is accurate over a wider range of loss rates.

* This material is based upon work supported by the National Science Foundation under grants NCR-95-08274, NCR-95-23807 and CDA-95-02639. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

† http://www.cs.umass.edu/∼vfiroiu/

1 Introduction

A significant amount of today's Internet traffic, including WWW (HTTP), file transfer (FTP), email (SMTP), and remote access (Telnet) traffic, is carried by the TCP transport protocol [18]. TCP together with UDP form the very core of today's Internet transport layer. Traditionally, simulation and implementation/measurement have been the tools of choice for examining the performance of various aspects of TCP. Recently, however, several efforts have been directed at analytically characterizing the throughput of TCP's congestion control mechanism, as a function of packet loss and round trip delay [6, 10, 7]. One reason for this recent interest is that a simple quantitative characterization of TCP throughput under given operating conditions offers the possibility of defining a "fair share" or "TCP-friendly" [6] throughput for a non-TCP flow that interacts with a TCP connection. Indeed, this notion has already been adopted in the design and development of several multicast congestion control protocols [19, 20].

In this paper we develop a simple analytic characterization of the steady state throughput of a bulk transfer TCP flow (i.e., a flow with a large amount of data to send, such as FTP transfers) as a function of loss rate and round trip time. Unlike the recent work of [6, 7, 10], our model captures not only the behavior of TCP's fast retransmit mechanism (which is also considered in [6, 7, 10]) but also the effect of TCP's timeout mechanism on throughput. The measurements we present in Section 3 indicate that this latter behavior is important from a modeling perspective, as we observe more timeout events than fast retransmit events in almost all of our TCP traces. Another important difference between ours and previous work is the ability of our model to accurately predict throughput over a significantly wider range of loss rates than before; measurements presented in [7] as well the measurements presented in this paper, indicate that this too is important. We also explicitly model the effects of small receiver-side windows. By comparing our model's predictions with a number of TCP measurements made between various Internet hosts, we demonstrate that our model is able to more accurately predict TCP throughput and is able to do so over a wider range of loss rates.

The remainder of the paper is organized as follows. In Section 2 we describe our model of TCP congestion control in detail and derive a new analytic characterization of TCP throughput as a function of loss rate and average round trip time. In Section 3 we compare the predictions of our model with a set of measured TCP flows over the Internet, having as their endpoints sites in both United States and Europe. Section 4 discusses the assumptions underlying the model and a number of related issues in more detail. Section 5 concludes the paper.

2 A Model for TCP Congestion Control

In this section we develop a stochastic model of TCP congestion control that yields a relatively simple analytic expression for the throughput of a saturated TCP sender, i.e., a flow with an unlimited amount of data to send, as a function of loss rate and average round trip time (RTT).

TCP is a protocol that can exhibit complex behavior, especially when considered in the context of the current Internet, where the traffic conditions themselves can be quite complicated and subtle [14]. In this paper, we focus our attention on the congestion avoidance behavior of TCP and its impact on throughput, taking into account the dependence of congestion avoidance on ACK behavior, the manner in which packet loss is inferred (e.g., whether by duplicate ACK detection and fast retransmit, or by timeout), limited receiver window size, and average round trip time (RTT). Our model is based on the Reno flavor of TCP, as it is by far the most popular implementation in the Internet today [13, 12]. We assume that the reader is familiar with TCP Reno congestion control (see for example [4, 17, 16]) and we adopt most of our terminology from [4, 17, 16].

Our model focuses on TCP's congestion avoidance mechanism, where TCP's congestion control window size, W, is increased by $1/W$ each time an ACK is received. Conversely, the window is decreased whenever a lost packet is detected, with the amount of the decrease depending on whether packet loss is detected by duplicate ACKs or by timeout, as discussed shortly.

We *model* TCP's congestion avoidance behavior in terms of "rounds." A round starts with the back-to-back transmission of W packets, where W is the current size of the TCP congestion window. Once all packets falling within the congestion window have been sent in this back-to-back manner, no other packets are sent until the first ACK is received for one of these W packets. This ACK reception marks the end of the current round and the beginning of the next round. In this model, the duration of a round is equal to the round trip time and is assumed to be independent of the window size, an assumption also adopted (either implicitly or explicitly) in [6, 7, 10]. Note that we have also assumed here that the time needed to send all the packets in a window is smaller than the round trip time; this behavior can be seen in observations reported in [2, 12].

At the beginning of the next round, a group of W' new packets will be sent, where W' is the new size of the congestion control window. Let b be the number of packets that are acknowledged by a received ACK. Many TCP receiver implementations send one cumulative ACK for two consecutive packets received (i.e., delayed ACK, [16]), so b is typically 2. If W packets are sent in the first round and are all received and acknowledged correctly, then W/b acknowledgments will be received. Since each acknowledgment increases the window size by $1/W$, the window size at the beginning of the second round is then $W' = W + 1/b$. That is, during congestion avoidance and in the absence of loss, the window size increases linearly in time, with a slope of $1/b$ packets per round trip time.

In the following subsections, we model TCP's behavior in the presence of packet loss. Packet loss can be detected in one of two ways, either by the reception at the TCP sender of "triple-duplicate" acknowledgments, i.e., four ACKs with the same sequence number, or via time-outs. We denote the former event as a "TD" (triple-duplicate) loss indication, and the latter as a "TO" loss indication.

We assume that a packet is lost in a round independently of any packets lost in *other* rounds, a modeling assumption justified to some extent by past studies [1] that have shown that periodic UDP packets that are separated by as little as 40 msec tend to get lost only in singleton bursts. On the other hand, we assume that packet losses are correlated among the back-to-back transmissions within a round: if a packet is lost, all remaining packets transmitted until the end of that round are also lost. This bursty loss behavior, which has been shown to arise from the drop-tail queuing discipline (adopted in many Internet routers), is discussed in

[2, 3]. We discuss it further in Section 4.

We develop a stochastic model of TCP congestion control in several steps, corresponding to its operating regimes: when loss indications are exclusively TD (Section 2.1), when loss indications are both TD and TO (Section 2.2), and when the congestion window size is limited by the receiver's advertised window (Section 2.3). We note that we do not model certain aspects of TCP's behavior (e.g., fast recovery) but believe we have captured the essential elements of TCP behavior, as indicated by the generally very good fits between model predictions and measurements made on numerous commercial TCP implementations, as discussed in Section 3. A more detailed discussion of model assumptions and related issues is presented in Section 4. Also note that in the following, we measure throughput in terms of packets per unit of time, instead of bytes per unit of time.

2.1 Loss indications are exclusively "triple-duplicate" ACKs

In this section we assume that loss indications are exclusively of type "triple-duplicate" ACK (TD), and that the window size is not limited by the receiver's advertised flow control window. We consider a TCP flow starting at time $t = 0$, where the sender always has data to send. For any given time $t > 0$, we define N_t to be the number of packets transmitted in the interval $[0, t]$, and $B_t = N_t/t$, the throughput on that interval. Note that B_t is the number of packets sent per unit of time regardless of their eventual fate (i.e., whether they are received or not). Thus, B_t represents the throughput of the connection, rather than its goodput. We define the long-term steady-state TCP throughput B to be

$$B = \lim_{t\to\infty} B_t = \lim_{t\to\infty} \frac{N_t}{t}$$

We have assumed that if a packet is lost in a round, all remaining packets transmitted until the end of the round are also lost. Therefore we define p to be the probability that a packet is lost, given that either it is the first packet in its round or the preceding packet in its round is not lost. We are interested in establishing a relationship $B(p)$ between the throughput of the TCP connection and p, the loss probability defined above.

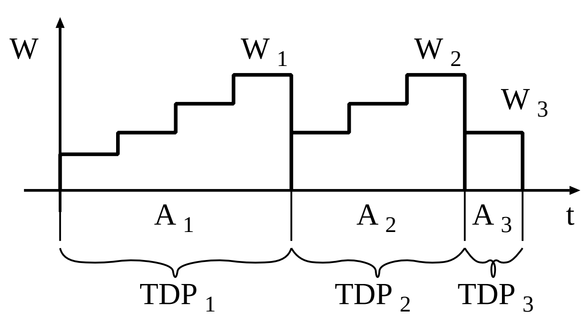

Figure 1: Evolution of window size over time when loss indications are triple duplicate ACKs

A sample path of the evolution of congestion window size is given in Figure 1. Between two TD loss indications, the sender is in congestion avoidance, and the window increases by $1/b$ packets per round, as discussed earlier. Immediately after the loss indication occurs, the window size is reduced by a factor of two.

We define a TD period (TDP) to be a period between two TD loss indications (see Figure 1). For the i-th TD period we define Y_i to be the number of packets sent in the period, A_i the duration of the period, and W_i the window size at the end of the period. Considering $\{W_i\}_i$ to be a Markov regenerative process with rewards $\{Y_i\}_i$ (see for example [15]), it can be shown that

$$B = \frac{E[Y]}{E[A]} \tag{1}$$

In order to derive an expression for B, the long-term steady-state TCP throughput, we must next derive expressions for the mean of Y and A.

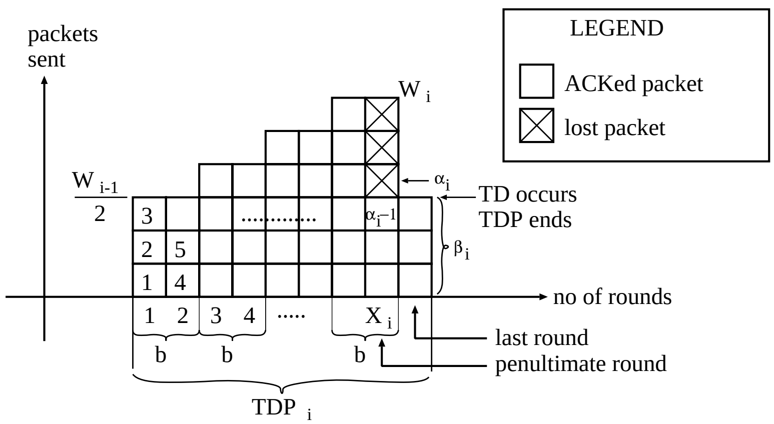

Figure 2: Packets sent during a TD period

Consider a TD period as in Figure 2. A TD period starts immediately after a TD loss indication, and thus the current congestion window size is equal to $W_{i-1}/2$, half the size of window before the TD occurred. At each round the window is incremented by $1/b$ and the number of packets sent per round is incremented by one every b rounds. We denote by α_i the first packet lost in TDP_i, and by X_i the round where this loss occurs (see Figure 2). After packet α_i, $W_i - 1$ more packets are sent in an additional round before a TD loss indication occurs (and the current TD period ends), as discussed in more detail in Section 2.2. Thus, a total of $Y_i = \alpha_i + W_i - 1$ packets are sent in $X_i + 1$ rounds. It follows that:

$$E[Y] = E[\alpha] + E[W] - 1 \tag{2}$$

To derive $E[\alpha]$, consider the random process $\{\alpha_i\}_i$, where α_i is the number of packets sent in a TD period up to and including the first packet that is lost. Based on our assumption that packets are lost in a round independently of any packets lost in *other* rounds, $\{\alpha_i\}_i$ is a sequence of independent and identically distributed (i.i.d.) random variables. Given our loss model, the probability that $\alpha_i = k$ is equal to the probability that exactly $k-1$ packets are successfully acknowledged before a loss occurs

$$P[\alpha = k] = (1-p)^{k-1}p, \quad k = 1, 2, \ldots \tag{3}$$

The mean of α is thus

$$E[\alpha] = \sum_{k=1}^{\infty} (1-p)^{k-1}pk = \frac{1}{p} \tag{4}$$

Form (2) and (4) it follows that

$$E[Y] = \frac{1-p}{p} + E[W] \tag{5}$$

To derive $E[W]$ and $E[A]$, consider again TDP_i. We define r_{ij} to be the duration (round trip time) of the j-th round of TDP_i. Then, the duration of TDP_i is $A_i = \sum_{j=1}^{X_i+1} r_{ij}$. We consider the round trip times r_{ij} to be random variables, that are assumed to be independent of the size of congestion window, and thus independent of the round number, j. It follows that

$$E[A] = (E[X]+1)E[r] \tag{6}$$

Henceforth, we denote by $RTT = E[r]$ the average value of round trip time.

Finally, to derive an expression for $E[X]$, we consider the evolution of W_i as a function of the number of rounds, as in Figure 2. To simplify our exposition, in this derivation we assume that $W_{i-1}/2$ and X_i/b are integers. First we observe that during the i-th TD period, the window size increases between $W_{i-1}/2$ and W_i. Since the increase is linear with slope $1/b$, we have:

$$W_i = \frac{W_{i-1}}{2} + \frac{X_i}{b}, \quad i = 1, 2, \ldots \tag{7}$$

The fact that Y_i packets are transmitted in TDP_i is expressed by

$$\begin{aligned} Y_i &= \sum_{k=0}^{X_i/b-1} \left(\frac{W_{i-1}}{2} + k\right)b + \beta_i & (8)\\ &= \frac{X_i W_{i-1}}{2} + \frac{X_i}{2}\left(\frac{X_i}{b} - 1\right) + \beta_i & (9)\\ &= \frac{X_i}{2}\left(\frac{W_{i-1}}{2} + W_i - 1\right) + \beta_i \quad \text{using (7)} & (10) \end{aligned}$$

where β_i is the number of packets sent in the last round (see Figure 2). $\{W_i\}_i$ is a Markov process for which a stationary distribution can be obtained numerically, based on (7) and (10) and on the probability density function of $\{\alpha_i\}$ given in (3). We can also compute the probability distribution of $\{X_i\}$. However, a simpler approximate solution can be obtained by assuming that $\{X_i\}$ and $\{W_i\}$ are mutually independent sequences of i.i.d. random variables. With this assumption, it follows from (7), (10) and (5) that

$$E[W] = \frac{2}{b} E[X] \tag{11}$$

and,

$$\frac{1-p}{p} + E[W] = \frac{E[X]}{2}\left(\frac{E[W]}{2} + E[W] - 1\right) + E[\beta] \tag{12}$$

We consider that β_i, the number of packets in the last round, is uniformly distributed between 1 and W_i, and thus $E[\beta] = E[W]/2$. From (11) and (12), we have

$$E[W] = \frac{2+b}{3b} + \sqrt{\frac{8(1-p)}{3bp} + \left(\frac{2+b}{3b}\right)^2} \tag{13}$$

Observe that,

$$E[W] = \sqrt{\frac{8}{3bp}} + o(1/\sqrt{p}) \tag{14}$$

i.e., $E[W] \approx \sqrt{\frac{8}{3bp}}$ for small values of p. From (11), (6) and (13), it follows

$$E[X] = \frac{2+b}{6} + \sqrt{\frac{2b(1-p)}{3p} + \left(\frac{2+b}{6}\right)^2} \tag{15}$$

$$E[A] = RTT\left(\frac{2+b}{6} + \sqrt{\frac{2b(1-p)}{3p} + \left(\frac{2+b}{6}\right)^2} + 1\right) \tag{16}$$

Observe that,

$$E[X] = \sqrt{\frac{2b}{3p}} + o(1/\sqrt{p}) \tag{17}$$

From (1) and (5) we have

$$B(p) = \frac{\frac{1-p}{p} + E[W]}{E[A]} \tag{18}$$

$$= \frac{\frac{1-p}{p} + \frac{2+b}{3b} + \sqrt{\frac{8(1-p)}{3bp} + \left(\frac{2+b}{3b}\right)^2}}{RTT\left(\frac{2+b}{6} + \sqrt{\frac{2b(1-p)}{3p} + (\frac{2+b}{6})^2} + 1\right)} \tag{19}$$

Which can be expressed as:

$$B(p) = \frac{1}{RTT}\sqrt{\frac{3}{2bp}} + o(1/\sqrt{p}) \tag{20}$$

Thus, for small values of p, (20) reduces to the throughput formula in [6] for $b = 1$.

We next extend our model to include TCP behaviors (such as timeouts and receiver-limited windows) not considered in previous analytic studies of TCP congestion control.

2.2 Loss indications are triple-duplicate ACKs and time-outs

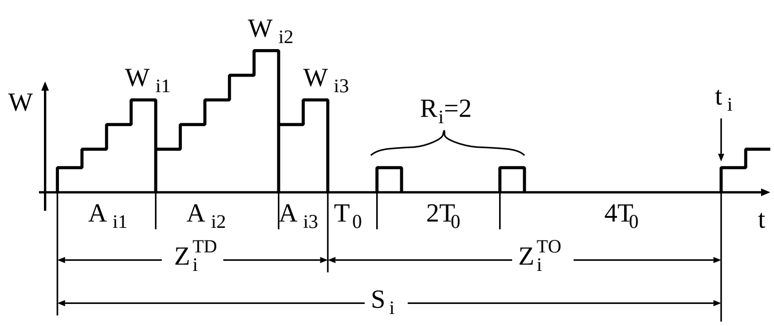

Figure 3: Evolution of window size when loss indications are triple-duplicate ACKs and time-outs

So far, we have considered TCP flows where all loss indications are due to "triple-duplicate" ACKs. Our measurements show (see Table 2) that in many cases the majority of window decreases are due to time-outs, rather than fast retransmits. Therefore, a good model should capture time-out loss indications.

In this section we extend our model to include the case where the TCP sender times-out. This occurs when packets (or ACKs) are lost, and less than three duplicate ACKs are received. The sender waits for a period of time denoted by T_0, and then retransmits non-acknowledged packets. Following a time-out, the congestion window is reduced to one, and one packet is thus resent in the first round after a time out. In the case that another time-out occurs before successfully retransmitting the packets lost during the first time out, the period of time out doubles to $2T_0$; this doubling is repeated for each unsuccessful retransmission until $64T_0$ is reached, after which the time out period remains constant at $64T_0$.

An example of the evolution of congestion window size is given in Figure 3. Let Z_i^{TO} denote the duration of a sequence of time-outs and Z_i^{TD} the time interval between two consecutive time-out sequences. Define S_i to be

$$S_i = Z_i^{TD} + Z_i^{TO}$$

Also, define M_i to be the number of packets sent during S_i. Then, $\{(S_i, M_i)\}_i$ is an i.i.d. sequence of random variables, and we have

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

We extend our definition of TD periods given in Section 2.1 to include periods starting after, or ending in, a TO loss indication (besides periods between two TD loss indications). Let n_i be the number of TD periods in interval Z_i^{TD}. For the j-th TD period of interval Z_i^{TD} we define Y_{ij} to be the number of packets sent in the period, A_{ij} to be the duration of the period, X_{ij} to be the number of rounds in the period, and W_{ij} to be the window size at the end of the period. Also, R_i denotes the number of packets sent during time-out sequence Z_i^{TO}. Observe here that R_i counts the total number of packet transmissions in Z_i^{TO}, and not just the number of different packets sent. This is because, as discussed in Section 2.1, we are interested in the throughput of a TCP flow, rather than its goodput. We have

$$M_i = \sum_{j=1}^{n_i} Y_{ij} + R_i, \quad S_i = \sum_{j=1}^{n_i} A_{ij} + Z_i^{TO}$$

and, thus,

$$E[M] = E[\sum_{j=1}^{n_i} Y_{ij}] + E[R], \quad E[S] = E[\sum_{j=1}^{n_i} A_{ij}] + E[Z^{TO}]$$

If we assume $\{n_i\}_i$ to be an i.i.d. sequence of random variables, independent of $\{Y_{ij}\}$ and $\{A_{ij}\}$, then we have

$$E[(\sum_{j=1}^{n_i} Y_{ij})_i] = E[n]E[Y], \quad E[(\sum_{j=1}^{n_i} A_{ij})_i] = E[n]E[A]$$

To derive $E[n]$ observe that, during Z_i^{TD}, the time between two consecutive time-out sequences, there are n_i TDPs, where each of the first $n_i - 1$ end in a TD, and the last TDP ends in a TO. It follows that in Z_i^{TD} there is one TO out of n_i loss indications. Therefore, if we denote by Q the probability that a loss indication ending a TDP is a TO, we have $Q = 1/E[n]$. Consequently,

$$B = \frac{E[Y] + Q * E[R]}{E[A] + Q * E[Z^{TO}]} \tag{21}$$

Since Y_{ij} and A_{ij} do not depend on time-outs, their means are those derived in (4) and (16). To compute TCP throughput using (21) we must still determine $Q, E[R]$ and $E[Z^{TO}]$.

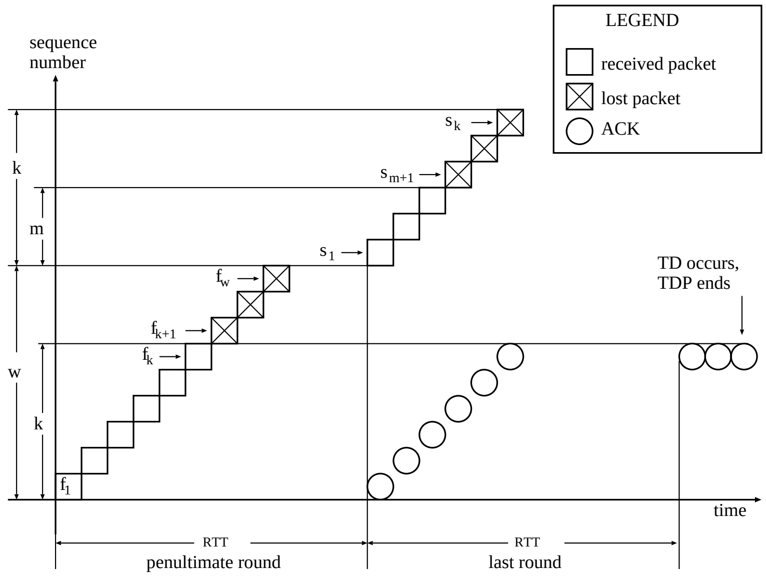

Figure 4: Packet and ACK transmissions preceding a loss indication

We begin by deriving an expression for Q. Consider the round of packets where a loss indication occurs; it will be referred to as the "penultimate" round (see Figure 4) [1]. Let w be the current congestion window size. Thus packets $f_1..f_w$ are sent in the penultimate round. Packets $f_1..f_k$ are acknowledged, and packet f_{k+1} is the first one to be lost (or not ACKed). We again assume that packet losses are correlated within a round: if a packet is lost, so are all packets that follow, till the end of the round. Thus, all packets following f_{k+1} in the penultimate round are also lost. However, since packets $f_1..f_k$ are ACKed, another k packets, $s_1..s_k$ are sent in the next round, which we will refer to as the "last" round. This round of packets may have another loss, say packet s_{m+1}. Again, our assumptions on packet loss correlation mandates that packets $s_{m+2}..s_k$ are also lost in the last round. The m packets successfully sent in the last round are responded to by ACKs for packet f_k, which are counted as duplicate ACKs. These ACKs are not delayed ([16], p. 312), so the number of duplicate ACKs is equal to the number of successfully received packets in the last round. If the number of such ACKs is higher than three, then a TD indication occurs, otherwise, a TO occurs. In both cases the current period between losses, TDP, ends. We denote by $A(w,k)$ the probability that the first k packets are ACKed in a round of w packets, given there is a sequence of one or more losses in the round. Then

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

Also, we define $C(n,m)$ to be the probability that m packets are ACKed in sequence in the last round (where n packets were sent) and the rest of the packets in the round, if any, are lost. Then,

$$C(n,m) = \begin{cases} (1-p)^m p, & m \leq n-1 \\ (1-p)^n, & m = n \end{cases}$$

[1] In Figure 4 each ACK acknowledges individual packets (i.e., ACKs are not delayed). We have chosen this for simplicity of illustration. We will see that the analysis does not depend on whether ACKs are delayed or not.

Then, $\hat{Q}(w)$, the probability that a loss in a window of size w is a TO, is given by

$$\hat{Q}(w) = \begin{cases} 1 & \text{if } w \leq 3 \\ \\ \sum_{k=0}^{2} A(w,k) + \sum_{k=3}^{w} A(w,k) \sum_{m=0}^{2} C(k,m) & \text{otherwise} \end{cases} \tag{22}$$

since a TO occurs if the number of packets successfully transmitted in the penultimate round, k, is less than three, or otherwise if the number of packets successfully transmitted in the last round, m is less than three. Also, due to the assumption that packet s_{m+1} is lost independently of packet f_{k+1} (since they occur in different rounds), the probability that there is a loss at f_{k+1} in the penultimate round and a loss at s_{m+1} in the last round equals $A(w,k) * C(k,m)$, and (22) follows.

After algebraic manipulations, we have

$$\hat{Q}(w) = \min\left(1, \frac{(1-(1-p)^3)(1+(1-p)^3(1-(1-p)^{w-3}))}{1-(1-p)^w}\right) \tag{23}$$

Observe (for example, using L'Hopital's rule) that

$$\lim_{p \to 0} \hat{Q}(w) = \frac{3}{w}.$$

Numerically we find that a very good approximation of $\hat{Q}$ is

$$\hat{Q}(w) \approx \min(1, \frac{3}{w}) \tag{24}$$

Q, the probability that a loss indication is a TO, is

$$Q = \sum_{w=1}^{\infty} \hat{Q}(w) P[W=w] = E[\hat{Q}]$$

We approximate

$$Q \approx \hat{Q}(E[W]) \tag{25}$$

where $E[W]$ is from (13).

We consider next the derivation of $E[R]$ and $E[Z^{TO}]$. For this, we need the probability distribution of the number of timeouts in a TO sequence, given that there is a TO. We have observed in our TCP traces that in most cases, one packet is transmitted between two time-outs in sequence. Thus, a sequence of k TOs occurs when there are $k-1$ consecutive losses (the first loss is given) followed by a successfully transmitted packet. Consequently, the number of TOs in a TO sequence has a geometric distribution, and thus

$$P[R=k] = p^{k-1}(1-p)$$

Then we can compute R's mean

$$E[R] = \sum_{k=1}^{\infty} k P[R=k] = \frac{1}{1-p} \tag{26}$$

Next, we focus on $E[Z^{TO}]$, the average duration of a time-out sequence excluding retransmissions, which can be computed in a similar way. We know that the first six time-outs in one sequence have length $2^{i-1}T_0$, $i = 1 \ldots 6$, with all immediately following timeouts having length $64T_0$. Then, the duration of a sequence with k time-outs is

$$L_k = \begin{cases} (2^k - 1)T_0 & \text{for } k \leq 6 \\ (63 + 64(k-6))T_0 & \text{for } k \geq 7 \end{cases}$$

and the mean of Z^{TO} is

$$\begin{aligned} E[Z^{TO}] &= \sum_{k=1}^{\infty} L_k P[R = k] \\ &= T_0 \frac{1 + p + 2p^2 + 4p^3 + 8p^4 + 16p^5 + 32p^6}{1 - p} \end{aligned}$$

Armed now with expressions for $Q, E[S], E[R]$ and $E[Z^{TO}]$ we can now substitute these expressions into equation (21) to obtain the following for $B(p)$:

$$B(p) = \frac{\frac{1-p}{p} + E[W] + \hat{Q}(E[W])\frac{1}{1-p}}{RTT(E[X]+1) + \hat{Q}(E[W])T_0\frac{f(p)}{1-p}} \tag{27}$$

where:

$$f(p) = 1 + p + 2p^2 + 4p^3 + 8p^4 + 16p^5 + 32p^6 \tag{28}$$

$\hat{Q}$ is given in (23), $E[W]$ in (13) and $E[X]$ in (16). Using (24), (14) and (17), we have that (27) can be approximated by

$$B(p) \approx \frac{1}{RTT\sqrt{\frac{2bp}{3}} + T_0 \min\left(1, 3\sqrt{\frac{3bp}{8}}\right)p(1 + 32p^2)} \tag{29}$$

2.3 The impact of window limitation

So far, we have not considered any limitation on the congestion window size. At the beginning of TCP flow establishment, however, the receiver advertises a maximum buffer size which determines a maximum congestion window size, W_{max}. As a consequence, during a period without loss indications, the window size can grow up to W_{max}, but will not grow further beyond this value. An example of the evolution of window size is depicted in Figure 5.

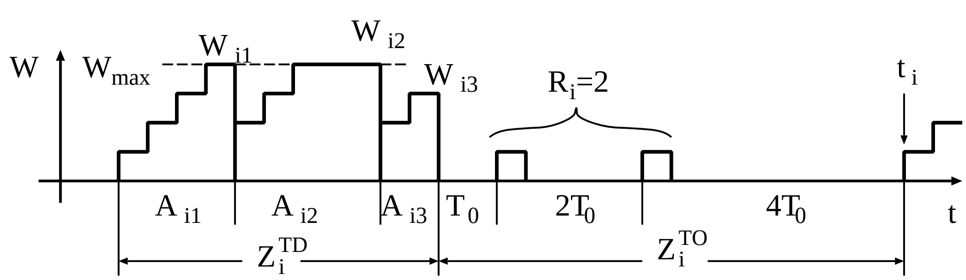

Figure 5: Evolution of window size when limited by W_{max}

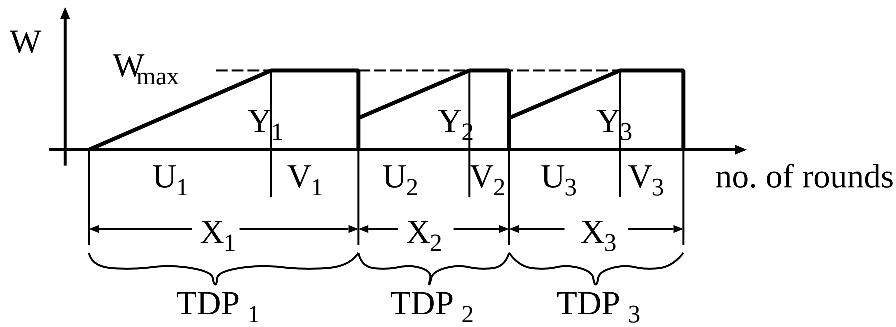

Figure 6: Fast retransmit with window limitation

To simplify the analysis of the model, we make the following assumption. Let us denote by W_u the unconstrained window size, the mean of which is given in (13)

$$E[W_u] = \frac{2+b}{3b} + \sqrt{\frac{8(1-p)}{3bp} + \left(\frac{2+b}{3b}\right)^2} \tag{30}$$

We assume that if $E[W_u] < W_{max}$, we have the approximation $E[W] \approx E[W_u]$. In other words, if $E[W_u] < W_{max}$, the receiver-window limitation has negligible effect on the long term average of the TCP throughput, and thus the TCP throughput is given by (27).

On the other hand, if $W_{max} \leq E[W_u]$, we approximate $E[W] \approx W_{max}$. In this case, consider an interval Z^{TD} between two time-out sequences consisting of a series of TD periods as in Figure 6. During the first TDP, the window grows linearly up to W_{max} for U_1 rounds, then remains constant for V_1 rounds, and then a TD indication occurs. The window then drops to $W_{max}/2$, and the process repeats. Thus,

$$W_{max} = \frac{W_{max}}{2} + \frac{U_i}{b}, \quad \forall i \geq 2$$

which implies $E[U] = (b/2)W_{max}$. Also, considering the number of packets sent in the i-th TD period, we have

$$Y_i = \frac{U_i}{2}(\frac{W_{max}}{2} + W_{max}) + V_i W_{max}$$

and then

$$E[Y] = \frac{3}{4} W_{max} E[U] + W_{max} E[V] = \frac{3b}{8} W_{max}^2 + W_{max} E[V]$$

Since Y_i, the number of packets in the i-th TD period, does not depend on window limitation, $E[Y]$ is given by (5), $E[Y] = (1-p)/p + W_{max}$, and thus

$$E[V] = \frac{1-p}{pW_{max}} + 1 - \frac{3b}{8} W_{max}$$

Finally, since $X_i = U_i + V_i$, we have

$$E[X] = E[U] + E[V] = \frac{b}{8} W_{max} + \frac{1-p}{pW_{max}} + 1$$

By substituting this result in (27), we obtain the TCP throughput, $B(p)$, when the window is limited

$$B(p) = \frac{\frac{1-p}{p} + W_{max} + \hat{Q}(W_{max})\frac{1}{1-p}}{RTT(\frac{b}{8}W_{max} + \frac{1-p}{pW_{max}} + 2) + \hat{Q}(W_{max})T_0\frac{f(p)}{1-p}}$$

In conclusion, the complete characterization of TCP throughput, $B(p)$, is:

$$B(p) = \begin{cases} \frac{\frac{1-p}{p}+E[W]+\hat{Q}(E[W])\frac{1}{1-p}}{RTT(\frac{b}{2}E[W_u]+1)+\hat{Q}(E[W])T_0\frac{f(p)}{1-p}} & \text{if } E[W_u] < W_{max} \\ \\ \frac{\frac{1-p}{p}+W_{max}+\hat{Q}(W_{max})\frac{1}{1-p}}{RTT(\frac{b}{8}W_{max}+\frac{1-p}{pW_{max}}+2)+\hat{Q}(W_{max})T_0\frac{f(p)}{1-p}} & \text{otherwise} \end{cases} \tag{31}$$

where $f(p)$ is given in (28), $\hat{Q}$ is given in (23) and $E[W_u]$ in (13). In the following sections we will refer to (31) as the "full model". The following approximation of $B(p)$ follows from (29) and (31):

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

In Section 3 we verify that equation (32) is indeed a very good approximation of equation 31. Henceforth we will refer to (32) as the "approximate model".

3 Measurements and Trace Analysis

Equations (31) and (32) provide an analytic characterization of TCP as a function of packet loss indication rate, RTT, and maximum window size. In this section we empirically validate these formulae, using measurement data from 37 TCP connections established between 18 hosts scattered across United States and Europe.

Table 1 lists the domains and operating systems of the 18 hosts. All data sets are for unidirectional bulk data transfers. We gathered the measurement data by running `tcpdump` at the sender, and analyzing its output with a set of analysis programs developed by us. These programs account for various measurement and implementation related problems discussed in [13, 12]. For example, when we analyze traces from a Linux sender, we account for the fact that TD events occur after getting only two duplicate acks instead of three. Our trace analysis programs were further verified by checking them against `tcptrace`[9] and `ns` [8].

Table 2 summarizes data from 24 data sets, each of which corresponds to a 1 hour long TCP connection in which the sender behaves as an "infinite source" – it always has data to send and thus TCP throughput is only limited by the TCP congestion control. The experiments were performed at randomly selected times during 1997 and beginning of 1998. The third and forth column of Table 2 indicate the number of packets sent and the number of loss indications respectively (triple duplicate ack or timeout). Dividing the total number of loss indications by the total number of packets sent gives us an approximate value of p. This approximation is similar to the one used in [7]. The next six columns show a breakdown of the

Receiver	Domain	Operating System
ada	hofstra.edu	Irix 6.2
afer	cs.umn.edu	Linux
al	cs.wm.edu	Linux 2.0.31
alps	cc.gatech.edu	SunOS 4.1.3
babel	cs.umass.edu	SunOS 5.5.1
baskerville	cs.arizona.edu	SunOS 5.5.1
ganef	cs.ucla.edu	SunOS 5.5.1
imagine	cs.umass.edu	win95
manic	cs.umass.edu	Irix 6.2
mafalda	inria.fr	SunOS 5.5.1
maria	wustl.edu	SunOS 4.1.3
modi4	ncsa.uiuc.edu	Irix 6.2
pif	inria.fr	Solaris 2.5
pong	usc.edu	HP-UX
spiff	sics.se	SunOS 4.1.4
sutton	cs.columbia.edu	SunOS 5.5.1
tove	cs.umd.edu	SunOS 4.1.3
void	US site	Linux 2.0.30

Table 1: Domains and Operating Systems of Hosts

loss indications by type: the number of TD events, the number of "single" timeouts, having duration T_0, the number of "double" timeouts, $T_1 = 2T_0$, etc. Note that p depends only on the $total$ number of loss indications, and not on their type. The last two columns report the average value of round trip time, and average duration of a "single" timeout T_0. These values have been averaged over the entire trace. When calculating round trip time values, we follow Karn's algorithm, in an attempt to minimize the impact of timeouts and retransmissions on the RTT estimates.

Table 3 reports summary results form additional 13 data sets. In these cases, each data set represents 100 serially-initiated TCP connections between a given sender-receiver pair. Each connection lasted for 100 seconds, and was followed by a 50 second gap before the next connection was initiated. These experiments were performed at randomly selected times during 1998. The data in columns 3-10 of Table 3 are cumulative over the set of 100 traces for the given source-destination pair. The last two columns report the average value of round trip time and "single" timeout. These values have been averaged over all hundred traces for the given source-destination pair.

An important observation to be drawn from the data in these tables is that in all traces, timeouts constitute the majority or a significant fraction of the total number of loss indications. This underscores the importance of including the effects of timeouts in the model of TCP congestion control. In addition to "single" timeout events (column T_0), it can be seen that exponential backoff (multiple timeouts) occurs with significant frequency.

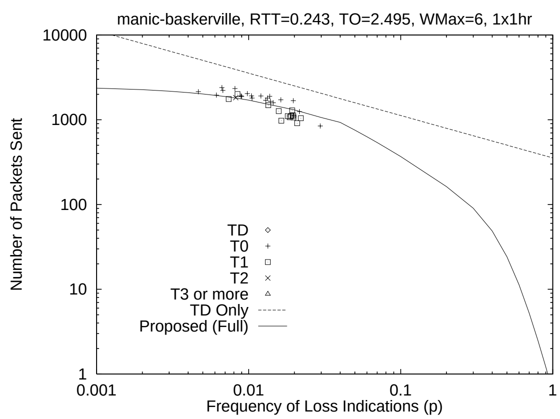

Figure 7: manic to baskerville

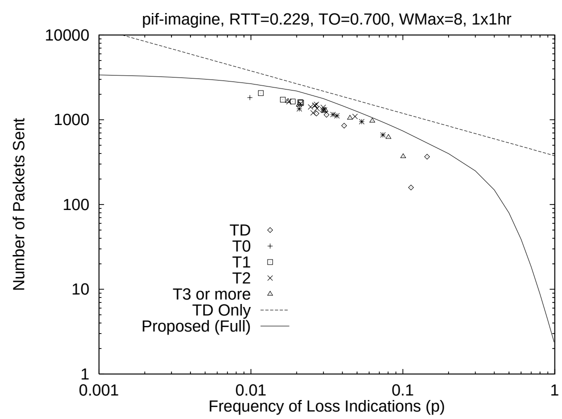

Figure 8: pif to imagine

Next, we use the measurement data described above to validate our model proposed in Section 2. Figures 7-12 plot the measured throughput in our trace data, the model of [7], as well as the predicted throughput from our proposed model given in (31) as described below. The title of the trace indicates the average round trip time, the average "single" timeout duration T_0, and the maximum window size W_{max} advertised by the receiver (in number of packets). The x-axis represents the frequency of loss indications, p, while y-axis represents the number of packets sent.

Each one-hour trace was divided into 36 consecutive 100 second intervals, and each plotted point on a graph represents the number of packets sent versus the number of loss indications during a 100s interval. While dividing a continuous trace into fixed sized intervals can lead to some inaccuracies in measuring p,

Sender	Receiver	Packets Sent	Loss Indic.	TD	T_0	T_1	T_2	T_3	T_4	T_5 or more	RTT	Time Out
manic	alps	54402	722	19	611	67	15	6	2	2	0.207	2.505
manic	baskerville	58120	735	306	411	17	1	0	0	0	0.243	2.495
manic	ganef	58924	743	272	444	22	4	1	0	0	0.226	2.405
manic	mafalda	56283	494	2	474	17	1	0	0	0	0.233	2.146
manic	maria	68752	649	1	604	35	8	1	0	0	0.180	2.416
manic	spiff	117992	784	47	702	34	1	0	0	0	0.211	2.274
manic	sutton	81123	1638	988	597	41	7	3	1	1	0.204	2.459
manic	tove	7938	264	1	190	37	18	8	3	7	0.275	3.597
void	alps	37137	838	7	588	164	56	17	4	2	0.162	0.489
void	baskerville	32042	853	339	430	67	12	5	0	0	0.482	1.094
void	ganef	60770	1112	414	582	79	20	9	4	2	0.254	0.637
void	maria	93005	1651	33	1344	197	54	15	5	3	0.152	0.417
void	spiff	65536	671	72	539	56	4	0	0	0	0.415	0.749
void	sutton	78246	1928	840	863	152	45	18	9	1	0.211	0.601
void	tove	8265	856	5	444	209	100	51	27	12	0.272	1.356
babel	alps	13460	1466	0	1068	247	87	33	18	8	0.194	1.359
babel	baskerville	62237	1753	197	1467	76	10	3	0	0	0.253	0.429
babel	ganef	86675	2125	398	1686	38	2	1	0	0	0.201	0.306
babel	spiff	57687	1120	0	939	137	36	7	1	0	0.331	0.953
babel	sutton	83486	2320	685	1448	142	31	9	4	1	0.210	0.705
babel	tove	83944	1516	1	1364	118	17	7	5	3	0.194	0.520
pif	alps	83971	762	0	577	111	46	16	8	2	0.168	7.278
pif	imagine	44891	1346	15	1044	186	63	21	10	5	0.229	0.700
pif	manic	34251	1422	43	944	272	105	36	14	6	0.257	1.454

Table 2: Summary data from 1hr traces

Sender	Receiver	Packets Sent	Loss Indic.	TD	T_0	T_1	T_2	T_3	T_4	T_5 or larger	RTT	Time Out
manic	ada	531533	6432	4320	2010	93	7	2	0	0	0.1419	2.2231
manic	afer	255674	4577	2584	1898	83	10	1	1	0	0.1804	2.3009
manic	al	264002	4720	2841	1804	70	5	0	0	0	0.1885	2.3542
manic	alps	667296	3797	841	2866	85	5	0	0	0	0.1125	1.9151
manic	baskerville	89244	1638	627	955	42	11	2	1	0	0.4735	3.2269
manic	ganef	160152	2470	1048	1308	89	18	6	1	0	0.2150	2.6078
manic	mafalda	171308	1332	9	1269	48	5	1	0	0	0.2501	2.5127
manic	maria	316498	2476	5	2362	99	8	2	0	0	0.1166	1.8798
manic	modi4	282547	6072	3976	1988	99	8	1	0	0	0.1749	2.2604
manic	pong	358535	4239	2328	1830	74	7	0	0	0	0.1769	2.1371
manic	spiff	298465	2035	159	1781	75	14	4	2	0	0.2539	2.4545
manic	sutton	348926	6024	3694	2238	87	5	0	0	0	0.1683	2.1852
manic	tove	262365	2603	6	2422	135	30	8	2	0	0.1153	1.9551

Table 3: Summary data from 100 second traces

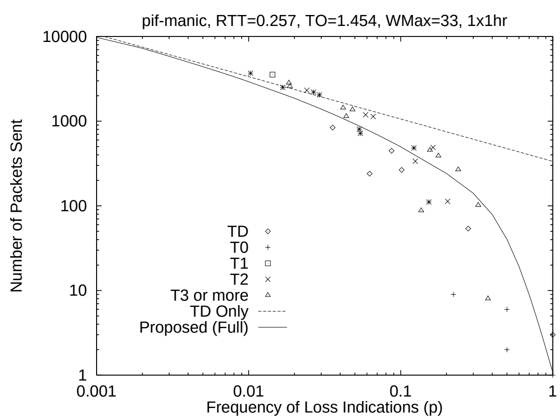

Figure 9: pif to manic

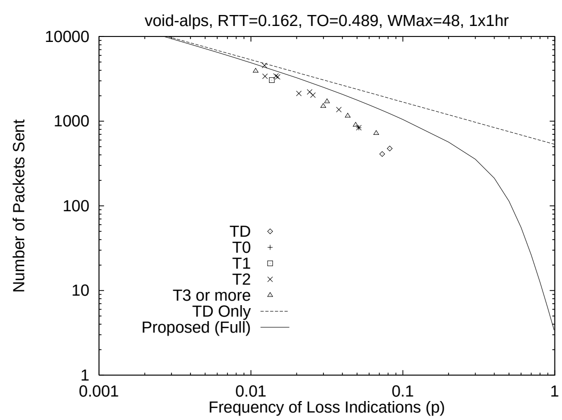

Figure 10: void to alps

(e.g., the interval boundaries may occur within timeout intervals, thus perhaps not attributing a loss event to the interval where most of its impact is felt), we believe that by using interval sizes of 100s, which are longer than most timeouts, we have minimized the impact of such inaccuracies. Each 100 second interval is classified into one of four categories: intervals of type “TD” did not suffer any timeout (only triple duplicate acks), intervals of type “$T0$” suffered at least one “single” timeout but no exponential backoff, “$T1$” represents intervals that suffered a single exponential backoff at least once (i.e a “double” timeout) etc. The line labeled “TD Only” (stands for Triple-Duplicate acks Only) plots the predictions made by the model described in [7], which is essentially the same model as described in [6], while accounting for delayed acks. The line labeled “Proposed (Full)” represents the model described by Equation (31). It has been pointed out

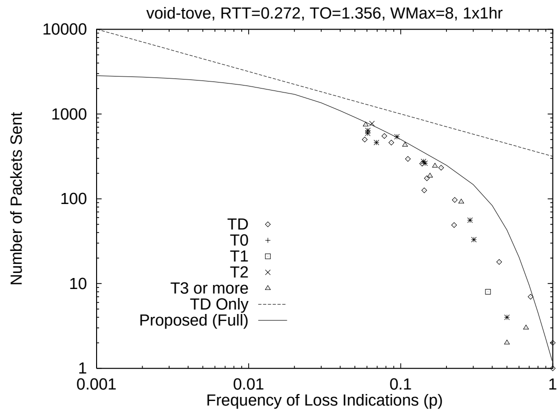

Figure 11: void to tove

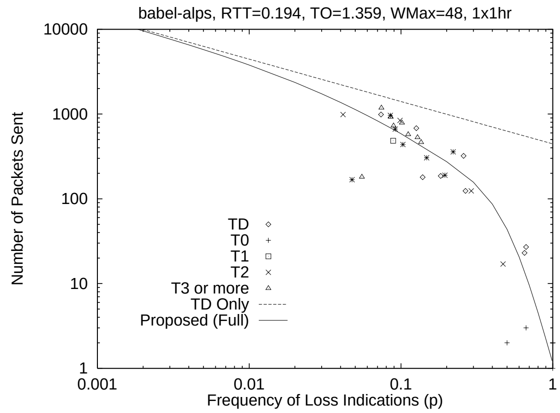

Figure 12: babel to alps

in [6] that the "TD Only" model may not be accurate when the frequency of loss indications is higher than 5%. We observe that in many traces the frequency of loss indications is higher than 5% and that indeed the "TD Only" model predicts values for TCP throughput much higher than measured. Also, in several traces (see for example, Figure 7) we observe that TCP throughput is limited by the receiver's advertised window size. This is not accounted for in the "TD Only" model, and thus "TD Only" overestimates the throughput at low p values.

Figures 13-17 show similar graphs, where each point represents an individual 100 second TCP connection. To plot the model predictions, we used round trip and timeout durations that were averaged over all 100 traces (these values also appear in Table 3). Equation (32) in Section 2 represents the simple, but approximate form (32) of the full model given in (31). In Figure 18, we plot the predictions of the approximate model along with the full model. The results for other data sets are similar.

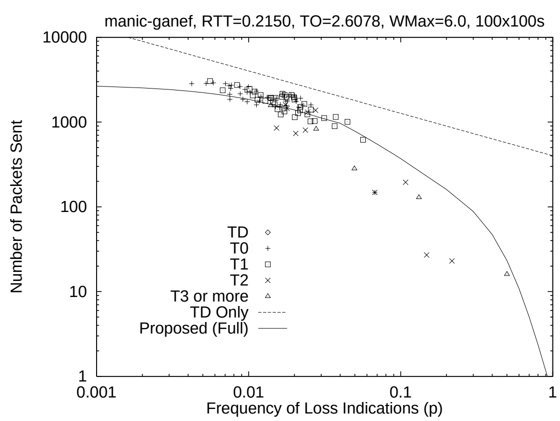

Figure 13: manic to ganef

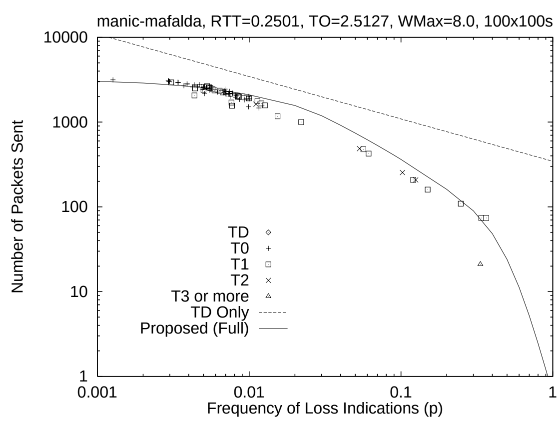

Figure 14: manic to mafalda

In order to accurately evaluate the models, we compute the average error as follows:

- **Hour-long traces:** We divide each trace into 100 second intervals, and compute the number of packets sent during that interval (here denoted as $N_{observed}$) as well as the value of loss frequency (here $p_{observed}$). We also calculate the average value of round trip time and timeout for the entire trace

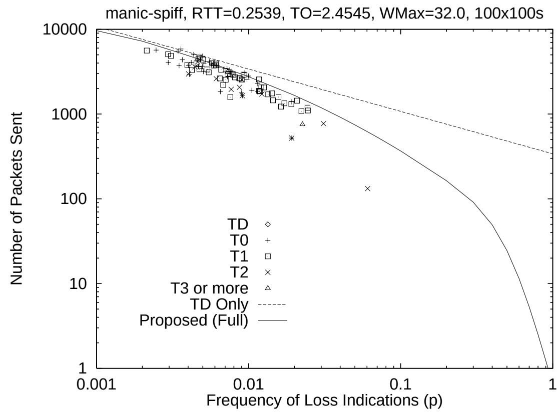

Figure 15: manic to spiff

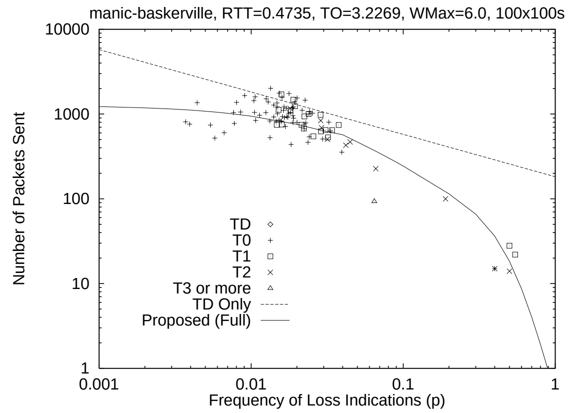

Figure 16: manic to baskerville

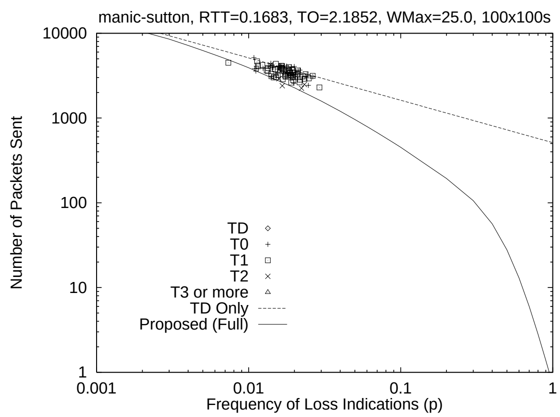

Figure 17: manic to sutton

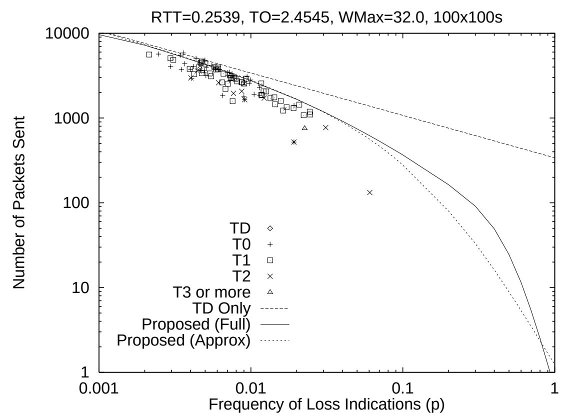

Figure 18: manic to spiff, with approximate model

(these values are available in Table 2). Then, for each 100 second interval we calculate the number of packets predicted by our proposed model, $N_{predicted} = B(p_{observed}) * 100s$, where B is from (31). The average error is given by:

$$\frac{\sum_{observations} \frac{|N_{predicted} - N_{observed}|}{N_{observed}}}{\text{number of observations}}$$

The average error of our approximate model (using B from (32)) and of "TD Only" are calculated in a similar manner. A smaller average error indicates better model accuracy. In Figure 19 we plot these error values to allow visual comparison. On the x-axis, the traces are identified by sender and receiver names. The order in which the traces appear is such that, from left to right, the average error for the "TD Only" model is increasing. The points corresponding to a given model are joined by line segments *only for better visual representation of the data*.

- **100 second traces:** We use the value of round trip time and timeout calculated for each 100-second trace. The error values are shown in Figure 20.

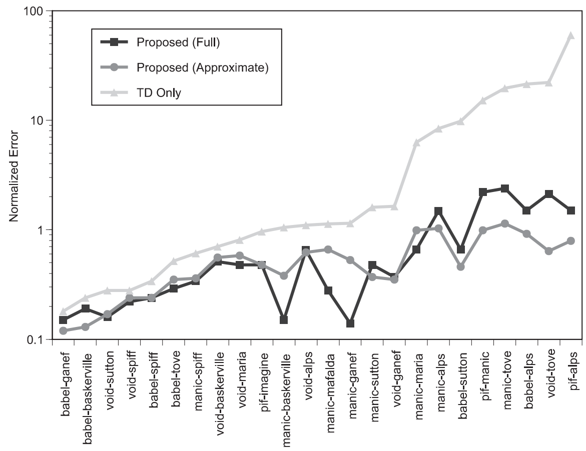

Figure 19: Comparison of the models for 1hr traces

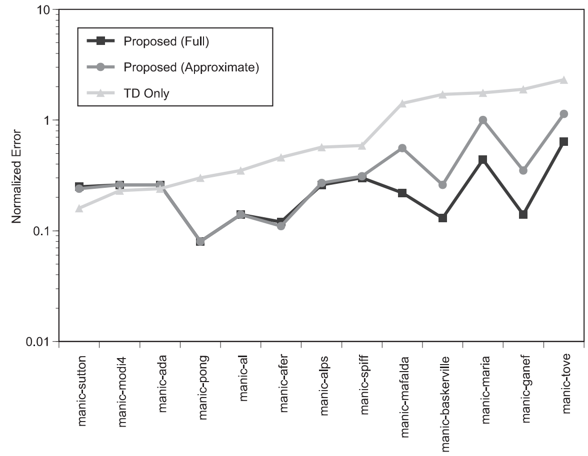

Figure 20: Comparison of the models for 100s traces

It can be seen from Figures 19 and 20 that in most cases, our proposed model is a better estimator of the observed values than the “TD Only” model. Our approximate model also generally provides more accurate predictions than the “TD Only” model, and is quite close to the predictions made by the full model. As one would expect, our model does not match all of the observations. We show an example of this in Figure 17. This is probably due to a large number of triple duplicate acks observed for this trace set.

4 A Discussion of the Model and the Experimental Results

In this section, we discuss various simplifying assumptions made while constructing the model in Section 2, and their impact on the results described in Section 3.

Our model does not capture the subtleties of the *fast recovery* algorithm. We believe that the impact of this omission is quite small, and that the results presented in Section 3 validate this assumption indirectly. We have also assumed that the time spent in *slow start* is negligible compared to the length of our traces. Both these assumptions have also been made in [6, 7, 10].

We have assumed that packet losses within a round are *correlated*. Justification for this assumption comes from the fact that the vast majority of the routers in Internet today use the drop-tail policy for packet discard. Under this policy, all packets that arrive at a full buffer are dropped. As packets in a round are sent back-to-back, if a packet arrives at a full buffer, it is likely that the same happens with the rest of the packets in the round. Packet loss correlation at drop-tail routers was also pointed out in [2, 3]. In addition, we assume that losses in one round are *independent* of losses in other rounds. This is justified by the fact that packets in different rounds are separated by one RTT or more, and thus they are likely to encounter buffer states that are independent of each other. This is also confirmed by findings in [1].

Another assumption we made, that is also implicit in [6, 7, 10], is that the round trip time is independent of the window size. We have measured the coefficient of correlation between the duration of round samples

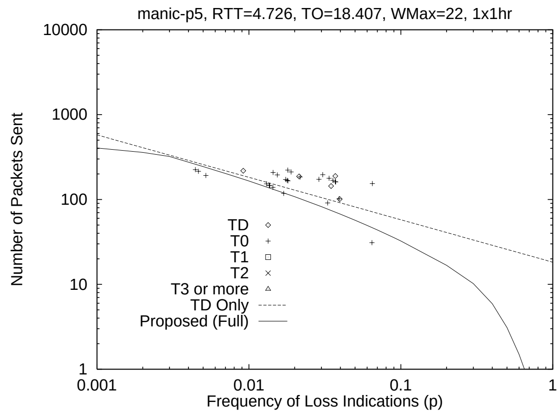

Figure 21: manic to p5

and the number of packets in transit during each sample. For most traces summarized in Table 2, the coefficient of correlation is in the range of -0.1 to +0.1, thus lending credence to the statistical independence between round trip time and window size. However, when we conducted similar experiments with receivers at the end of a modem line, we found the coefficient of correlation to be as high as 0.97. We speculate that this is a combined effect of a slow link and a buffer devoted exclusively to this connection (probably at the ISP, just before the modem). As a result, our model, as well as the models described in [6, 10, 7] fail to match the observed data in the case of a receiver at the end of a modem. In Figure 21, we plot results from one such experiment. The receiver was a Pentium PC, running Linux 2.0.27 and was connected to the Internet via a commercial service provider using a 28.8Kbps modem. The results are for a 1 hour connection divided into 100 second intervals.

We have also assumed that all of our senders implement TCP-Reno as described in [4, 17, 16]. In [13, 12], it is observed that the implementation of the protocol stack in each operating system is slightly different. While we have tried to account for the significant differences (for example in Linux the TD loss indications occur after two duplicate ACKs), we have not tried to customize our model for the nuances of each operating system. For example, we have observed that the Linux exponential backoff does not exactly follow the algorithm described in [4, 17, 16]. Our observations also seem to indicate that in the Irix implementation, the exponential backoff is limited to 2^5, instead of 2^6. We are also aware of the observation made in [13] that the SunOS TCP implementation is derived from Tahoe and not Reno. We have not customized our model for these cases.

5 Conclusion

In this paper we have presented a simple model of the TCP-Reno protocol. The model captures the essence of TCP's congestion avoidance behavior and expresses throughput as a function of loss rate. The model takes into account the behavior of the protocol in the presence of timeouts, and is valid over the entire range of loss probabilities.

We have compared our model with the behavior of several real-world TCP connections. We observed that most of these connections suffered from a significant number of timeouts. We found that our model provides a very good match to the observed behavior in most cases, while models proposed in [6, 7, 10] significantly overestimate throughput. Thus, we conclude that timeouts have a significant impact on the performance of the TCP protocol, and that our model is able to account for this impact.

We have also presented a simplified expression for TCP bandwidth in Equation (32), which is a good approximation for the proposed model in most cases. This simple approximation can be used in protocols such as those described in [19, 20] to ensure "TCP-friendliness'.

A number of avenues for future work remain. First, our model can be enhanced to account for the effects of fast recovery and fast retransmit. Second, a more precise throughput calculation can be obtained if the congestion window size is modeled as a Markov chain. Third, we have assumed that once a packet in a given round is lost, all remaining packets in that round are lost as well. This assumption can be relaxed, and the model can be modified to incorporate a loss distribution function. Estimating this distribution function for a given path in the Internet is a significant research effort in itself. Fourth, it is interesting to further investigate the behavior of TCP over slow links with dedicated buffers (such as modem lines). We are currently investigating more closely the data sets for which our model is not a good estimator. We are also working on a TCP-friendly protocol to control transmission of continuous media. This protocol will use our model to modulate its throughput to ensure TCP friendliness.

References

[1] J. Bolot and A. Vega-Garcia. Control mechanisms for packet audio in the Internet. In *Proceedings IEEE Infocom96*, 1996.

[2] K. Fall and S. Floyd. Simulation-based comparisons of Tahoe, Reno, and SACK TCP. *Computer Communication Review*, 26(3), July 1996.

[3] S. Floyd and V. Jacobson. Random Early Detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking*, 1(4), August 1997.

[4] V. Jacobson. Modified TCP congestion avoidance algorithm. Note sent to end2end-interest mailing list, 1990.

[5] P. Karn and C. Partridge. Improving Round-Trip time estimates in reliable transport protocols. *Computer Communication Review*, 17(5), August 1987.

[6] J. Mahdavi and S. Floyd. TCP-friendly unicast rate-based flow control. Note sent to end2end-interest mailing list, Jan 1997.

[7] M. Mathis, J. Semske, J. Mahdavi, and T. Ott. The macroscopic behavior of the TCP congestion avoidance algorithm. *Computer Communication Review*, 27(3), July 1997.

[8] S. MCanne and S. Flyod. ns-LBL Network Simulator, 1997. Obtain via http://www-nrg.ee.lbnl.gov/ns/.

[9] S. Ostermann. tcptrace: TCP dump file analysis tool, 1996. http://jarok.cs.ohiou.edu/software/tcptrace/.

[10] T. Ott, J. Kemperman, and M. Mathis. The stationary behavior of ideal TCP congestion avoidance. in preprint.

[11] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modeling TCP throughput: A simple model and its empirical validation. Technical report UMASS-CS-TR-1998-08.

[12] V. Paxson. Automated packet trace analysis of TCP implementations. In *Proceedings of SIGCOMM 97*, 1997.

[13] V. Paxson. End-to-End Internet packet dynamics. In *Proceedings of SIGCOMM 97*, 1997.

[14] V. Paxson and S. Floyd. Why we don't know how to simulate the Internet. In *Proccedings of the 1997 Winter Simulation Conference*, 1997.

[15] S. Ross. *Applied Probability Models with Optimization Applications*. Dover, 1970.

[16] W. Stevens. *TCP/IP Illustrated, Vol.1 The Protocols*. Addison-Wesley, 1994.

[17] W. Stevens. TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms. RFC2001, Jan 1997.

[18] K. Thompson, G. Miller, and M. Wilder. Wide-area internet traffic patterns and charateristics. *IEEE Network*, 11(6), November-December 1997.

[19] T. Turletti, S. Parisis, and J. Bolot. Experiments with a layered transmission scheme over the Internet. Technical report RR-3296, INRIA, France. Obtain via http://www.inria.fr/RRRT/RR-3296.html.

[20] L. Vicisano, L. Rizzo, and J. Crowcroft. TCP-like congestion control for layered multicast data transfer. In *Proceedings of INFOCOMM'98*, 1998.