TCP Reno with Random losses: Latency, Throughput and Sensitivity Analysis

B. Sikdar, S. Kalyanaraman and K. S. Vastola
Department of Electrical, Computer and Systems Engineering
Rensselaer Polytechnic Institute, Troy, NY, 12180
email:{bsikdar,shivkuma,vastola}@networks.ecse.rpi.edu

Abstract

Current models for TCP performance focus mainly on the steady-state throughput of infinite flows, making them inappropriate for the short TCP flows which dominate the transfers over the Internet. In this paper, we present a model for the latency of finite TCP Reno flows with independent losses for which currently no models exist. The proposed model can also estimate the steady-state throughput of long flows. We also present a sensitivity analysis and empirical models which investigate the effects of various factors like loss rates, window limitation, delayed acknowledgments and packet size on TCP latency and quantify and isolate their individual contributions. Our model captures the effect of timeouts and slow start phases which occur anywhere during the transfer and uses a more accurate model for the slow start phase, leading to a more accurate estimation of TCP latencies.

1 Introduction

Most of the existing models for TCP performance [4, 5, 6, 7, 9, 10, 11] confine their attention to the steady state throughput of long TCP connections. In contrast, recent studies show that most TCP connections today are short with average connection sizes of 8-12 KB with the vast majority transferring files less than 10 KB [8, 13]. Short flows spend most of their time in the slow start phase and recover most losses using timeouts. The underlying assumptions of steady state models thus cease to hold, making them unsuitable for modeling short flows. In [3], the steady-state model of [11] is extended for finite flows by accounting for the connection establishment phase and an approximate analysis of the initial slow start, assuming correlated losses.

While TCP modeling has been widely investigated in literature, this paper's contribution is more than just marginal. In addition to concentrating mainly on the steady-state throughput, the expressions derived in existing literature are complex and do not shed much light on the individual contribution of various network parameters on TCP's performance. In this paper, we explore and quantify the effects of factors like window limitation, maximum segment sizes, effect of delayed versus undelayed ACKs etc. on TCP performance. Also, we focus on the performance of TCP Reno in the presence of independent losses, both in terms of its latency as well as the steady-state throughput. While the throughput of TCP Reno under independent losses has been considered in [6, 7, 9], no models exist for estimating its latency under independent losses. The independent loss model becomes important while considering scenarios in the Internet [2], losses and time variations due to wireless links [7] and UBR and ABR service classes in ATM networks [1].

Our model for TCP Reno latency shows a better match to latencies measured over the Internet as compared to the only other latency model existing in literature [3] (which assumes correlated losses). Unlike [3], our model captures the slow start effects subsequently in the transfer which arise due to timeouts. We also use a more accurate expression for modeling the *cwnd* increase pattern in TCP flows during slow start and both these factors lead to more a more accurate model.

The rest of the paper is organized as follows. In Section 2 we describe the assumptions used in our models. Latency and steady-state throughput models for TCP Reno with delayed ACKs are presented in Sections 3 and 4 respectively. We then model TCP Reno without delayed ACKs in Section 5. Section 6 presents the validation results and Section 7 introduces the empirical models. Section 8 deals with the sensitivity analysis and Section 9 presents the concluding remarks.

2 Assumptions

Our assumptions on the network and transfer scenario are similar to those in [11] and [3]. We consider the latency arising only from TCP's performance and do not account for delays arising at the end points from factors such as buffer limitations. We assume that the sender sends full-sized segments as fast as its congestion window allows and the receiver advertises a consistent flow control window. We assume that the receiver uses the delayed ACK scheme specified in RFC 2581. As in [3], we do not account for the effects of Nagle's algorithm and silly window avoidance. As in [11] and [3] we model TCP latency by considering "rounds". A round begins with the transmission of a window of packets from the sender and ends when the sender receives an ACK for one or more of these packets. The time to transmit all the packets in a round is assumed to be smaller than the duration of the round and that the duration of a round is independent of the window size. We assume that the losses in a round are independent of losses in other rounds and unlike [11] and [3] we assume that losses in a single round are also independent of each other, leading to a binomial distribution for the number of losses suffered by a flow.

3 Latency of TCP Reno Connections

Our approach to modeling the latency is based on estimating the transfer time, given that a flow suffers a given number of losses. This, combined with the probability that a flow suffers a given number of losses, then gives us the expected transfer time. We derive the transfer times as a function of N, the total number of unique packets to be transferred, the RTT and the loss probability p on the path.

3.1 Connection Establishment

A TCP connection begins with a three-way handshake, beginning with the initiating host sending a SYN segment, the receiver responding with a SYN/ACK packet to which the the initiating host replies with an ACK, indicating that the connection has been established. During this process, if either host does not receive the ACK it is expecting within a timeout period T_s, it retransmits its SYN and then waits twice as long for an ACK. Following the arguments and the derivation of [3], the expected duration of the connection setup phase, t_{setup}, is given by

$$t_{setup} = RTT + 2T_s\left((1-p)/(1-2p) - 1\right) \qquad (1)$$

3.2 Window Increase Pattern

TCP starts its transmission in the slow start phase and increases *cwnd* by one for each ACK it receives. With delayed ACKs, the receiver sends one ACK for every two packets that it gets or if the delayed ACK timer expires. Examples of the *cwnd* increase pattern are shown in Fig. 1. In the figure, consider the round with $cwnd = 3$ (where packets 4, 5 and 6 are sent). The receiver sends an ACK for the first two packets and delays sending the ACK for packet 6. If a new

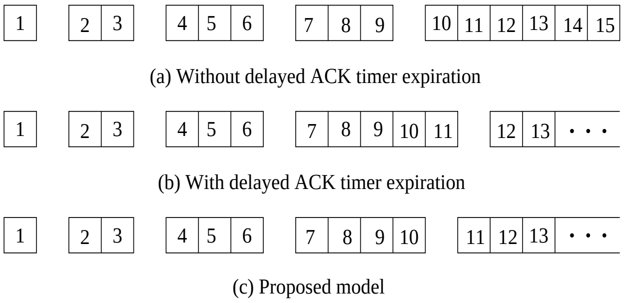

Figure 1: *cwnd* increase patterns during slow start.

packet arrives before the ACK timer expires, packet 6 is acknowledged along with the new packet and the *cwnd* increases as shown in Fig. 1(a). However, if the ACK timer expires before a new packet arrives, an ACK is sent which results in a *cwnd* of 5 as shown in Fig. 1(b).

To account for such complex behavior of the *cwnd*, we develop a model which is more accurate than that in [3] which models the number of packets transmitted in the n^{th} round as 1.5^n. The number of packets transmitted in the n^{th} round according to our model is given by

$$pkts(n) = \left\lfloor 2^{\frac{n-1}{2}} + 2^{\frac{n-2}{2}} \right\rfloor \qquad (2)$$

Note that our model predicts the number of packets transmitted in the fourth round as 4 (Fig. 1(c)) which is the average of the packets transmitted in examples (a) and (b). The number of packets transmitted in the first k rounds of the slow start phase is then given by

$$\sum_{n=1}^{k} pkts(n) = \left\lfloor 2^{\frac{n+1}{2}} + 3(2)^{\frac{4n-3}{8}} - (4+3\sqrt{2})/2 \right\rfloor \quad (3)$$

Note that after the first packet of a flow is sent, the receiver has to wait until the delayed ACK timer expires and an ACK is sent, increasing the *cwnd* to 2. This delay due to the delayed ACK, t_{dack}, is 100ms for UNIX and 150ms for Windows platforms.

3.3 Flows Without Losses

When a flow does not experience any losses, the congestion window increases exponentially until it reaches W_{max} and then stays there until the transfer is complete. From Equation (2), the number of rounds required to reach a *cwnd* of W_{max} is given by

$$n_{wm} = \left\lfloor 2\log_2\left(2W_{max}/(1+\sqrt{2})\right) \right\rfloor \qquad (4)$$

and the number of packets transmitted when *cwnd* reaches W_{max}, N_{exp}, is given by

$$N_{exp} = \left\lfloor 2^{\frac{n_{wm}+1}{2}} + 3(2)^{\frac{4n_{wm}-3}{8}} - (4+3\sqrt{2})/2 \right\rfloor + W_{max} \qquad (5)$$

The time to transfer N packets, $N < N_{exp}$, can be obtained by solving $N = \sum_{n=1}^{k} pkts(n)$ for k and is

$$t_{nl}(N) = \begin{cases} \left\lceil 2\log_2\left(\frac{2N+4+3\sqrt{2}}{2\sqrt{2}+3(2)^{\frac{5}{8}}}\right)\right\rceil RTT, & \text{if } N \le N_{exp} \\ \left[n_{wm} + \left\lceil \frac{N-N_{exp}}{W_{max}} \right\rceil\right] RTT, & \text{otherwise} \end{cases} \tag{6}$$

3.4 Flows with a Single Loss

Consider a flow of N packets where the i^{th} packet is lost. The time to transmit the first i packets is given by $t_{nl}(i)$ from Eqn. (6). Now, if $cwnd < 4$ when the packet is lost, the loss will lead to a timeout. The average duration of the timeout period, $E[TO]RTT$, as calculated in [11] is given by

$$E[TO] = T_o \frac{2(1 + p + 2p^2 + 4p^3 + 8p^4 + 18p^5 + 32p^6)}{1-p}$$

where T_o is the time the sender waits before retransmitting the first lost packet. During congestion avoidance, $cwnd$ increases linearly, increasing by 1 every two RTTs. The number of rounds required to transmit a packets in the congestion avoidance mode with the initial value of $cwnd = b$ is obtained by solving $a = \sum_{i=1}^{k}(b + \lfloor\frac{i-1}{2}\rfloor)$ for k whose solution is given by

$$t_{lin}(a,b) = \begin{cases} \left\lceil \frac{a-x(x+1)+b(b-1)}{x+1} \right\rceil + 2(x-b+1) & \text{if } a \le N_{lin} \\ \left\lceil \frac{a-N_{lin}}{W_{max}} \right\rceil + 2(W_{max} - b + 1) & \text{otherwise} \end{cases} \tag{7}$$

where $x = \lfloor(-1+\sqrt{1+4(a+b(b-1))})/2\rfloor$ and $N_{lin} = W_{max}(W_{max}+1) - b(b-1)$ and denotes the number of packets that can be sent before $cwnd$ reaches W_{max}. Also, we denote the $cwnd$ of the round when the i^{th} packet was transmitted by $cwnd(i)$, the sequence number of the last packet of that round by $n_m(i)$ and the the number of rounds it takes to transmit i packets without any losses in the absence of window limitation by $r(i)$. From Eqn. (6), $r(i)$ is given by

$$r(i) = \lceil 2\log_2((2i + 8.243)/7.455)\rceil \tag{8}$$

If $r(i) \ge n_{wm}$, we know that $cwnd = W_{max}$. For all other cases, $cwnd < W_{max}$ and is given by Eqn. (2). Then, $cwnd(i)$ is given by

$$cwnd(i) = \begin{cases} W_{max} & \text{if } r(i) > n_{wm} \\ \left\lfloor 2^{\frac{r(i)-1}{2}} + 2^{\frac{r(i)-2}{2}} \right\rfloor & \text{otherwise} \end{cases} \tag{9}$$

and $n_m(i)$, obtained using Eqn. (3), is given by

$$n_m(i) = \begin{cases} \left\lceil \frac{\max(0, i-N_{exp})}{W_{max}} \right\rceil W_{max} + N_{exp} & \text{if } r(i) > n_{wm} \\ \left\lfloor 2^{\frac{r(i)+1}{2}} + 3(2)^{\frac{4r(i)-3}{8}} - \frac{4+3\sqrt{2}}{2} \right\rfloor & \text{otherwise} \end{cases} \tag{10}$$

In the round the i^{th} packet is transmitted, $i - n_m(i) + cwnd(i) - 1$ packets are sent before the i^{th} packet and we can expect $\lceil(i - n_m(i) + cwnd(i) - 1)/2\rceil$ ACKs for these packets. Thus the $cwnd$ when the i^{th} packet is lost $cwnd_{i+1} = \min\{W_{max}, \lceil\frac{i-n_m(i)+cwnd(i)-1}{2}\rceil + cwnd(i)\}$. The number of packets sent before the flow experiences the timeout is then $k = cwnd_{i+1} + i - 2$ with one more packet getting sent in the slow-start phase following the timeout. The total transfer time (for $i = 1, \cdots, 6$) is then given by

$$t_{sl}(N) = [t_{nl}(i) + E[TO] + t_{lin}(N-k-1, 2) + 1]\, RTT \tag{11}$$

If $cwnd \ge 4$, a single loss is recovered with a fast-retransmit. As in the case of timeouts, $cwnd_{i+1} = \min\{W_{max}, \lceil(i - n_m(i) + cwnd(i) - 1)/2\rceil + cwnd(i)\}$. On receiving the third duplicate ACK, the lost packet is retransmitted. Once the lost packet is acknowledged, the flow goes into congestion avoidance. Now, we can have two cases depending on the position of the lost packet. If the lost packet is among the last three packets in the round, a new round of packets is transmitted before TCP gets three duplicate ACKs. Otherwise, a third duplicate ACK is received at the end of the same round. Then, $N - k$ packets remain to be transmitted in the congestion avoidance mode with

$$k = \begin{cases} \min\{W_{max}, \lceil cwnd_{i+1}/2\rceil + cwnd_{i+1} - 1\} + i - 1 & \text{if } n_m(i) - i < 3 \\ \min\{W_{max}, \lceil cwnd_{i+1}/2\rceil + n_m(i) - i\} + i - 1 & \text{otherwise} \end{cases}$$

At the beginning of the congestion avoidance mode, $cwnd$ halves its value to $n = \lceil\frac{cwnd_{i+1}}{2}\rceil$. The total transfer time (for $i = 7, \cdots, N$) is then given by

$$t_{sl}(N) = \begin{cases} [t_{nl}(i) + t_{lin}(N-k, n) + 2]\, RTT & \text{if } n_m(i) - i < 3 \\ [t_{nl}(i) + t_{lin}(N-k, n) + 1]\, RTT & \text{otherwise} \end{cases} \tag{12}$$

3.5 Flows with Multiple Losses

Consider a flow of N packets with M losses, with the second loss at the m^{th} packet. The transmission time for the first $m-1$ packets (where there is one loss) is obtained using Eqns. (11) and (12). The average number of packets between two consecutive losses in the remaining $N - m + 1$ packets, D_a, is given by

$$D_a = (N - m + 1)/(M - 1) \tag{13}$$

We compute the average time to transmit D_a packets, which, multiplied by $M - 2$ gives the time required to transfer the remainder of the flow. After the first

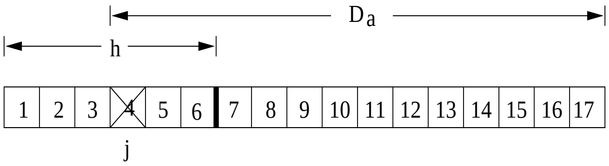

Figure 2: The relation between h, j and D_a.

loss, we approximate the possible values of *cwnd* when the subsequent losses occur to vary uniformly in the range $\{1, \cdots, \min\{W_{max}, \lceil c\frac{-1+\sqrt{1+16D_a}}{2}\rceil\}\}$. The factor $c = \frac{(3+10RTT)(1-p)^2}{4(1+p+p^2)^3}$ was empirically derived using simulations and curve fitting.

Now, consider such a flow with $cwnd = h$ when the j^{th} packet in the round is lost (Figure 2). If $cwnd < 4$, the loss leads to a timeout. The number of packets successfully transmitted following the j^{th} packet before the timeout can be shown to be $h-1$. The subsequent slow start phase lasts for only one round since congestion avoidance takes over when *cwnd* reaches 2 and transmits the remaining $D_a - h$ packets. The total time to transmit the D_a packets is then given by

$$t_{TO}(D_a) = [E[TO] + I(j > 1) + t_{lin}(D_a - h, 2)]RTT \tag{14}$$

where $I(x)$ is the indicator function.

On the other hand, losses which occur when $cwnd \geq 4$ lead to fast retransmissions. Following the arguments of the previous section, $D_a - k$ packets remain to be transmitted in the congestion avoidance phase, which begins with a *cwnd* of $\lceil\frac{h}{2}\rceil$ and

$$k = \begin{cases} \min\{h + \lceil\frac{h}{2}\rceil - 1, W_{max}\} - 1 & \text{if } h - j < 3 \\ \min\{h - j + \lceil\frac{h}{2}\rceil, W_{max}\} - 1 & \text{otherwise} \end{cases} \tag{15}$$

The time taken to transmit the D_a packets is then

$$t_{FR}(D_a) = \begin{cases} \left[t_{lin}(D_a - k, \lceil\frac{h}{2}\rceil) + 2\right] RTT & \text{if } h - j < 3 \\ \left[t_{lin}(D_a - k, \lceil\frac{h}{2}\rceil) + 1\right] RTT & \text{otherwise} \end{cases} \tag{16}$$

Eqns. (14) and (16), combined with the expected time to transmit the first $m-1$ packets using Eqns. (11) and (12) then gives the expected transfer time for the flow. The expected transfer time for a flow of N packets with multiple losses, $t_{ml}(N)$, is thus given by

$$\begin{aligned} t_{ml}(N) &= E\{t_{sl}(m-1)\} + E\{(M-2)t_{TO}(D_a)\} \\ &\quad + E\{(M-2)t_{FR}(D_a)\} \end{aligned} \tag{17}$$

where the expectation operation is carried over all possible values of m and M.

3.6 The Expected Transfer Time

Combining the results of the previous sections, the expected time to transfer N packets is given by

$$\begin{aligned} T_{transfer}(N) &= t_{setup} + t_{dack} + (1-p)^N t_{nl}(N) + \\ &\quad p(1-p)^{N-1} E\{t_{sl}(N)\} + t_{ml}(N) \end{aligned} \tag{18}$$

where t_{setup}, $t_{nl}(N)$ and $t_{ml}(N)$ are defined in Eqns. (1), (6) and (17) respectively and $t_{sl}(N)$ is defined in Eqns. (11) and (12).

4 Steady-State Throughput

To extend the model of Sec. 3 to infinite flows, we first note that for $p > 0$, the average number of packets between two successive losses, $d = 1/p$. We can now use Eqns. (14) and (16) to find the average time to transmit d packets. Dividing d by this expected transfer time gives us the steady state throughput. The possible values of the *cwnd* in this case is approximated to vary uniformly between 1 and c_w, where

$$c_w = min\left\{\left\lceil c(-1 + \sqrt{1 + 16/p})/2\right\rceil, W_{max}\right\} \tag{19}$$

and $c = \frac{(3+10RTT)(1-p)^2}{4(1+p+p^2)^3}$. Again, note that any loss with $cwnd < 4$ leads to a timeout and in these cases, the transfer time is obtained using $t_{TO}(d)$ of Eqn. (14). On the other hand, any loss with $cwnd \geq 4$ is recovered using a fast retransmit and we can use $t_{FR}(d)$ of Eqn. (16) to calculate the transfer time. Averaging the transfer time over the c_w possible values of *cwnd* and all possible positions of the lost packet within the window, the expected transfer time is given by

$$t_{ss}(p) = \frac{2}{c_w(c_w+1)}\left[\sum_{i=1}^{3}\sum_{j=1}^{i} t_{TO}(d) + \sum_{i=4}^{c_w}\sum_{j=1}^{i} t_{FR}(d)\right] \tag{20}$$

The steady state throughput in bytes per second of a TCP connection is thus

$$R(p) = \frac{dMSS}{t_{ss}(p)} = \frac{MSS}{t_{ss}(p)p} \tag{21}$$

5 TCP Reno Without Delayed ACKs

In the absence of delayed ACKs, the receiver sends an ACK for each new packet it receives. The model follows the same structure and arguments as in the previous sections and due to space restrictions, we do not elaborate them in detail again in this section. Also, the absence of delayed ACKs does not affect the connection setup time and we again use Eqn. (1) to evaluate the setup time t_{setup}. In the slow-start phase, the *cwnd* in the n^{th} round is given by 2^{n-1} and the time to transmit a packets in the congestion avoidance phase, with an initial window of b is given by

$$t_{lin}(a,b) = \begin{cases} \left\lceil \frac{-1+\sqrt{1+4(2a+b(b-1))}}{2} \right\rceil - b + 1 & \text{if } 2a \leq N_{lin} \\ \left\lceil \frac{2a - N_{lin}}{2W_{max}} \right\rceil + W_{max} - b + 1 & \text{otherwise} \end{cases} \tag{22}$$

where $N_{lin} = W_{max}(W_{max}+1) - b(b-1)$.

5.1 Flow Without Losses

In the absence of losses, the number of packets transmitted when *cwnd* reaches W_{max}, N_{exp}, is given by $N_{exp} = 2^{\lceil log_2 W_{max} \rceil} + W_{max} - 1$. The transfer time for N packets is then

$$t_{nl}(N) = \begin{cases} \lfloor \log_2 N \rfloor + 1 & \text{if } N \le N_{exp} \\ \lfloor \log_2 W_{max} \rfloor + \lceil \frac{N - N_{exp}}{W_{max}} \rceil + 2 & \text{otherwise} \end{cases} \quad (23)$$

5.2 Flows with a Single Loss

Consider again a flow of N packets where the i^{th} packet is lost. The highest sequence number of any packet sent in the round where the i^{th} packet was transmitted, $n_m(i)$, can be expressed as

$$n_m(i) = 2^{m+1} + \left\lceil \frac{\max\{i - 2^{m+1} + 1, 0\}}{W_{max}} \right\rceil W_{max} - 1 \quad (24)$$

where $m = \min\{\lfloor \log_2 i \rfloor, \lfloor \log_2 W_{max} \rfloor\}$. Again, a loss will lead to a timeout only if *cwnd* is less than 4 and the number of packets successfully transmitted before the congestion avoidance phase is given by $k = \min\{i, W_{max}\} + i - 2$. The time required to transfer N packets (for $i = 1, 2, 3$) can be expressed as

$$t_{sl}(N) = [t_{nl}(i) + E[TO] + t_{lin}(N - k - 1, 2) + 1]\, RTT \quad (25)$$

For rounds in which $cwnd \ge 4$, a single loss in any packet is recovered using a fast-retransmit. The number of packets k that are transmitted before the window cuts down to $cwnd/2$ is given by

$$k = \begin{cases} \min\{W_{max}, \min\{i, W_{max}\} & \text{if } n_m(i) - i < 3 \\ +\lceil \min\{i, W_{max}\}/2 \rceil - 1\} + i - 1 & \\ \min\{W_{max}, \lceil \frac{i}{2} \rceil + n_m(i) & \text{otherwise} \\ -i\}\, i - 1 & \end{cases}$$

and the time to transmit the remaining $N - k$ is obtained using Eqn. (22). The time required to transfer N packets (for $i = 4, \cdots, N$ and with $n = \lceil \frac{\min\{i, W_{max}\}}{2} \rceil$) is then given by

$$t_{sl}(N) = \begin{cases} [t_{nl}(i) + t_{lin}(N - k, n) & \text{if } n_m(i) - i < 3 \\ +2]\, RTT & \\ [t_{nl}(i) + t_{lin}(N - k, n) & \text{otherwise} \\ +1]\, RTT & \end{cases} \quad (26)$$

5.3 Flows with Multiple Losses

For the multiple loss case, we follow the same approach as in Section 3.5. We limit the range of possible values of the *cwnd* to $\min\{\lceil c \frac{-1 + \sqrt{1 + 16 D_a}}{2} \rceil, W_{max}\}$ where D_a is given by Eqn. (13) and $c = \frac{(3 + 10RTT)(1-p)^2}{4(1 + p + p^2)^3}$. We first consider the cases with timeouts. Again, for a flow with $cwnd = h$ when the j^{th} packet in the round is lost, $D_a - h$ packets remain to be transmitted after the slow-start phase. The time to transmit D_a packets is then

$$t_{TO}(D_a) = [E[TO] + I(j > 1) + t_{lin}(D_a - h, 2)]RTT \quad (27)$$

As before, losses occurring when $cwnd \ge 4$ lead to fast retransmissions. Again, $D_a - k$ packets remain to be transmitted in the congestion avoidance phase with

$$k = \begin{cases} \min\{h + \lceil \frac{h}{2} \rceil - 1, W_{max}\} - 1 & \text{if } h - j < 3 \\ \min\{h - j + \lceil \frac{h}{2} \rceil, W_{max}\} - 1 & \text{otherwise} \end{cases}$$

The time to transmit D_a packets is then given by

$$t_{FR}(D_a) = \begin{cases} [t_{lin}(D_a - k, \lceil \frac{h}{2} \rceil) + 2]\, RTT & \text{if } h - j < 3 \\ [t_{lin}(D_a - k, \lceil \frac{h}{2} \rceil) + 1]\, RTT & \text{otherwise} \end{cases} \quad (28)$$

The expected time to transfer the N packets in the presence of multiple losses, $t_{ml}(N)$, is thus

$$\begin{aligned} t_{ml}(N) &= E\{t_{sl}(m-1)\} + E\{(M-2)t_{TO}(D_a)\} \\ &\quad + E\{(M-2)t_{FR}(D_a)\} \end{aligned} \quad (29)$$

where the expectation operation is carried over all possible values of m and M.

5.4 The Expected Transfer Time

The expected transfer time for a flow of N packets is given by

$$\begin{aligned} T_{transfer}(N) &= t_{setup} + (1-p)^N t_{nl}(N) + \\ &\quad p(1-p)^{N-1} E\{t_{sl}(N)\} + t_{ml}(N) \end{aligned} \quad (30)$$

where t_{setup}, $t_{nl}(N)$ and $t_{ml}(N)$ are defined in Eqns. (1), (23) and (29) respectively and $t_{sl}(N)$ is defined in Eqns. (25) and (26).

5.5 Steady-State Throughput

Following the methodology of Sec. 4, we first find the expected time to transfer $d = 1/p$ packets, which can be written as

$$t_{ss}(p) = \frac{2}{c_w(c_w + 1)} \left[\sum_{i=1}^{3} \sum_{j=1}^{i} t_{TO}(d) + \sum_{i=4}^{c_w} \sum_{j=1}^{i} t_{FR}(d) \right]$$

where $t_{TO}(d)$ and $t_{FR}(d)$ are defined in Eqns. (27) and (28) and c_w is defined in Eqn. (19). The steady state throughput in bytes per second of a TCP connection is thus $R(p) = \frac{MSS}{t_{ss}(p)p}$.

6 Model Verification Results

The models proposed in this paper are verified in this section using measurements over the Internet and simulations using the simulator *ns*. The measurements over the Internet were carried out for TCP connections from a local machine to machines in various domains in the USA and abroad. Due to space limitations, we show the results for only two sets of measurements. The sender was running Solaris 5.6 while the

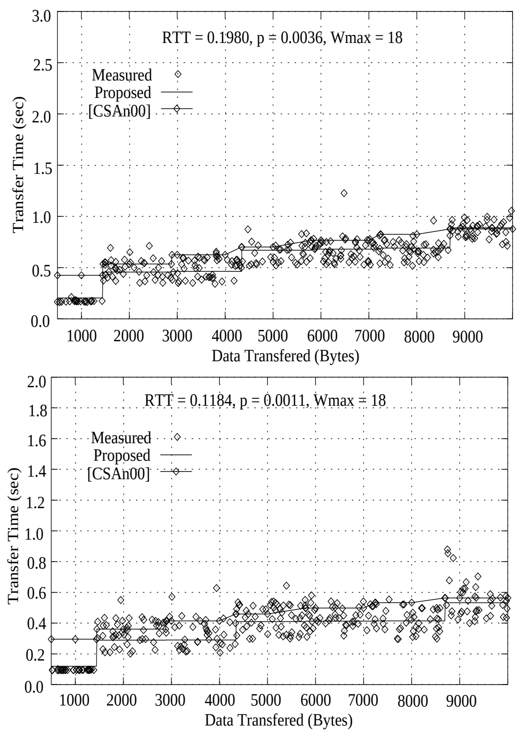

Figure 3: Comparison of latency of TCP Reno transfer between Troy, NY and Pisa, Italy (top) and Los Angeles, CA (bottom).

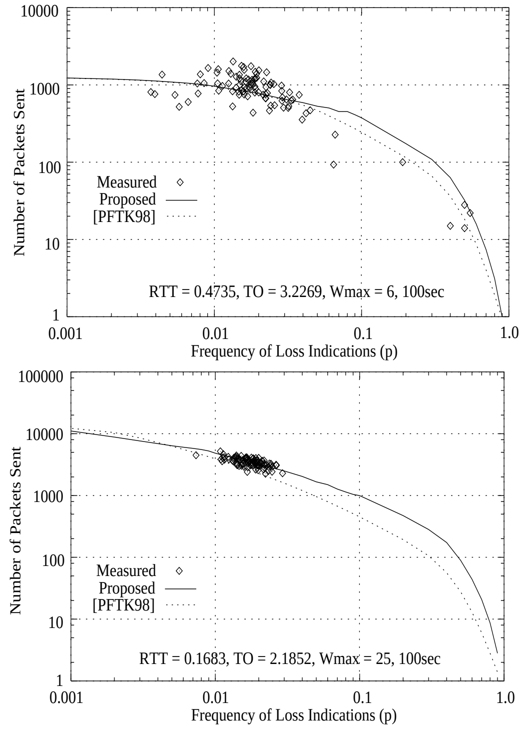

Figure 4: Steady state throughput of TCP connections from the proposed model compared with the model and measurements from [11] (Fig. 16 and 17 of [11]).

receivers used FreeBSD CAIRN-2.5 and FreeBSD 3.3. The traces were collected at the sender using `tcpdump`. Fig. 3 shows the transfer times for various files sizes and compare them to our model. Our results are a better match with latencies measured over the Internet as compared to the model proposed in [3], particularly for transfers less than 10000 bytes. The improvement is primarily due to fact that our model is able to capture the effects of timeouts and slow start more accurately.

Fig. 4 shows the results of our model for the steady state throughput of TCP connections. We compare our results with those of Padhye et al. [11] (Figs. 16 and 17 of [11]) and we see that our results match closely with both the measurements and the model from [11]. We note that there is a little deviation in the two models as the loss rate increases which is due to the difference in the loss models. However, this does not affect the accuracy and validity of the model as our results are very close to the measured values.

The model for TCP Reno without delayed ACKs is verified using the simulator *ns* (since most TCP implementations use delayed ACKs and it is difficult to access machines running TCP without delayed ACKs). For our simulations, we used a simple topology with a sender connected to a receiver through a router which randomly dropped packets with a fixed packet drop probability. The results are shown in Figure 5. The model fits the experimental results very well over a wide range of loss rates and RTTs and the worst case error over all the cases we tried was less than 5%.

7 Empirical Model

To get a better "feel" of TCP's behavior which might lead to design guidelines, we now propose an empirical model for TCP latency. Empirically, the time to transfer N packets can be approximated as

$$T_{emp}(N) = [\log_{1.57} N + \{f(p, RTT)N + 4p\log_{1.57} N + 20p\} + \frac{(10+3RTT)}{4(1-p)W_{max}\sqrt{W_{max}}}N]RTT \quad (31)$$

where $f(p, RTT) = \frac{2.32(2p+4p^2+16p^3)}{(1+RTT)^3}N + \frac{(1+p)}{RTT10^3}$. The first expression in the model corresponds to the case when the flow does not experience any losses. The second term accounts for the increase in the transfer times dues to losses while the third term accounts for the effects of window limitation. The model indicates that the transfer time increases exponentially with an increase in the loss rate. Also, for larger RTTs, the rate of increase decreases. The third term shows that the transfer time is inversely proportional to W_{max}

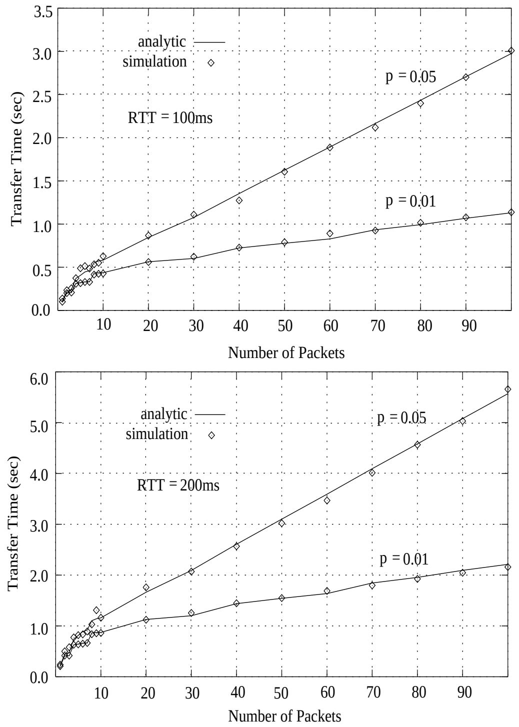

Figure 5: Transfer times from analytic and simulation results for TCP flows without delayed ACKs.

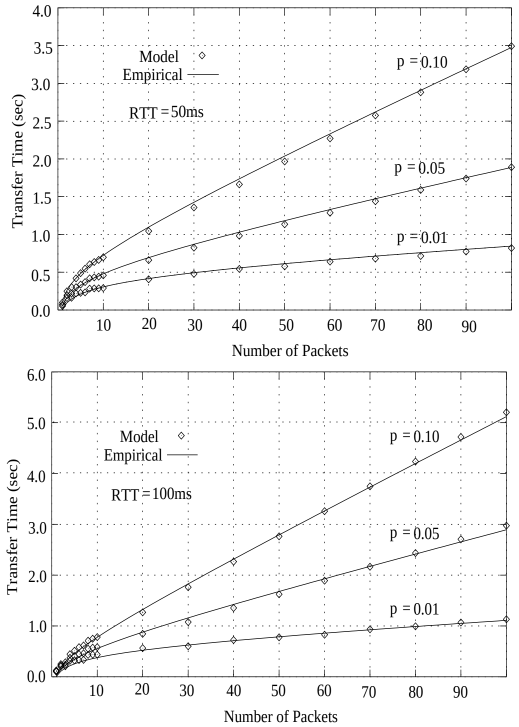

Figure 6: Latencies from the empirical and analytic models of Eqn. (18) (top) and Eqn. (30) (bottom).

and thus there is no significant improvement in the latency if W_{max} is increased beyond a threshold. The results from the empirical model are compared with those from Eqn. (18) in Figure 6 and as can be seen, the model is a very close approximation over a wide range of RTTs, loss rates, W_{max} and file sizes.

For TCP Reno without delayed ACKs, empirically, the time to transfer N packets can be modeled by

$$T_{emp}(N) = \left[\log_2 N + \frac{4.5pN}{1+RTT^2} + \frac{1+p}{(W_{max})^{1.5}}\right] RTT \tag{32}$$

The first expression in the model corresponds to the case when the flow does not experience any losses. The second term accounts for the increase in the transfer times dues to losses while the third term accounts for the effects of window limitation. Figure 6 compares the the empirical model to the analytic model of Eqn. (30) and as can be seen, the model is a very close approximation over a wide range of network parameters.

8 Sensitivity Analysis

We now investigate the sensitivity of TCP latency to variations in the packet size and the maximum window size, W_{max}. In Figure 7 we plot the transfer time of TCP connections as a function of the packet size and W_{max} with other parameters kept constant. As expected, the graphs indicate that the transfer time increases as packet sizes get smaller and as W_{max} decreases. For smaller data transfers, there is no effect of W_{max} on the transfer time as the transfer is over before *cwnd* reaches W_{max}. It is interesting to note that the reduction in the transfer time with increase in W_{max} is not linear and tends to saturate, making the decrease in latency too small to justify increasing W_{max} beyond a limit which is determined by the loss probability. To explain this, we note that for a packet loss probability p, on average we see a loss every $1/p$ packets. Thus *cwnd* for the flow is generally lower than $1/p$. If $W_{max} > 1/p$, then $cwnd < W_{max}$ for most of the flow and there is no significant increase in the transfer time if W_{max} is increased. We also note that the transfer time decreases as the packet size increases though here too the reduction in the transfer time does not scale linearly with the increase in the packet size.

9 Conclusion and Discussions

In this paper we presented models for TCP Reno transfer times and steady-state throughput. In addition to being the first model for Reno latency with independent losses, our model accounts for arbitrary file sizes, slow start phases at all possible points in the

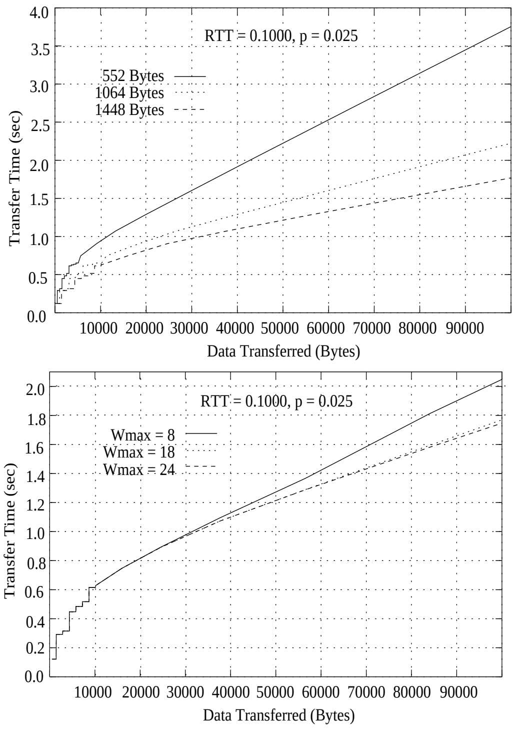

Figure 7: Effect of packet sizes and W_{max} on the transfer time of TCP connections.

flow and arbitrary loss probabilities. Our model is also more accurate for estimating latencies over the Internet than those predicted by [3] (which uses correlated losses) which can be attributed to the better modeling of the timeouts and slow-start phases of a TCP flow.

In the second part of the paper, we investigated the effects of various network and protocol parameters on TCP latency. We presented a model for the latency and throughput in the absence of delayed ACKs. The paper also presented an empirical model for estimating the latency of TCP transfers which gives an indication of the dependence of the latency on various parameters and is thus helpful in formulating design guidelines. We also carried out sensitivity studies on TCP with respect to the packet sizes and the effect of window limitations and show that the improvements in the latency with increasing packet sizes and W_{max} is not linear and saturates beyond a point which is determined by the loss rate.

References

[1] *ATM forum traffic management specification version 4.0, Draft specification ATM Forum/95-13RII,* ATM Forum, March 1996.

[2] J-C. Bolot, "End-to-end packet delay and loss behavior in the Internet," *Proc. of ACM SIGCOMM,* pp. 289-298, San Francisco, CA, September 1993.

[3] N. Cardwell, S. Savage and T. Anderson, "Modeling TCP latency," *Proceedings of IEEE INFOCOM,* pp. 1742-1751, Tel Aviv, Israel, March 2000.

[4] C. Casetti and M. Meo, "A New Approach to Model the Stationary Behavior of TCP Connections," *Proceedings of IEEE INFOCOM,* pp. 367-375, Tel Aviv, Israel, March 2000.

[5] J. Heidemann, K. Obraczka and J. Touch, "Modeling the performance of HTTP over several transfer protocols," *IEEE/ACM Transactions on Networking,* vol. 5, no. 5, pp. 616-630, Oct 1997.

[6] A. Kumar, "Comparative performance analysis of versions of TCP in a local network with a lossy link," *IEEE/ACM Transactions on Networking,* vol. 6, no. 4, pp. 485-498, Aug 1998.

[7] T. V. Lakshman and U. Madhow, "The performance of TCP/IP for networks with high bandwidth-delay products and random loss," *IEEE/ACM Trans. on Networking,* vol. 5, pp 336-350, Jun 1997.

[8] B. A. Mah, "An empirical model of HTTP network traffic," *Proceedings of IEEE INFOCOM'97,* Kobe, Japan, April 1997.

[9] M. Mathis, J. Semke, J. Mahdavi and T. Ott, "The macroscopic behavior of the TCP congestion avoidance algorithm," *ACM Computer Communications Review,* vol. 27, no. 3, pp 67-82, Jul 1997.

[10] A. Misra and T. J. Ott, "The window distribution of idealized TCP congestion avoidance with variable packet loss," *Proceedings of IEEE INFOCOM,* pp. 1564-1572, New York, NY, March 1999.

[11] J. Padhye, V. Firoiu, D. Towsley and J. Kurose, "Modeling TCP throughput: A simple model and its empirical validation," *Proc. of ACM SIGCOMM'98,* Vancouver, BC, Canada, pp 303-314, September 1998.

[12] C. Partridge and T. J. Shepard, "TCP/IP performance over satellite links," *IEEE Network,* pp 44-49, Sep/Oct 1997.

[13] K. Thompson, G. J. Miller and R. Wilder, "Wide-area Internet traffic patters and characteristics," *IEEE Network,* vol. 6, no. 11, pp 10-23, Nov 1997.