Transfer Times and Steady State Throughput of TCP Connections

Paper ID: 318
No. of Pages: 20

Abstract

Various models for transfer times of lossless TCP connections and steady state throughput of TCP connections with losses have been proposed in literature. However, most TCP connections in today's Internet transfer data of the order of a few KBytes and losses are omnipresent and high loss rates not uncommon. In this paper, we propose a single model which can estimate the transfer times of TCP flows of arbitrary size, any packet loss probability and accounts for the effects of window limitation. In [2], the authors extend the steady state analysis of Padhye et al. [10] by accounting for the connection setup time and an approximate analysis of the initial slow start phase. In contrast, our model not only accounts for the connection setup phase but also gives the *exact* expressions for the transfer times when the flow experiences a single or no loss. Also, unlike [2], our model captures the slow start phases which occur subsequently in the transfer as a result of timeouts. The proposed model can also characterize the steady state throughput of TCP connections. The model is verified using both simulations and web based measurements of TCP connections. In addition, we also introduce an empirical model which allows a better "feel" of TCP latency and the nature of its dependence on loss probabilities and window limitation. Finally, we use our models to analyze the effect on window limitation and packet size on TCP latency and also show that non-persistent connections have lower overall transfer times than persistent connections.

1 Introduction

The dominance of TCP as the transport protocol of a majority of today,s networking applications has led to efforts for analytic models of TCP behavior. A number of models have been proposed by researchers to model TCP performance. Most of these analysis techniques assume very large transfers and propose models for the steady-state throughput of long TCP connections [9, 4, 8, 6, 5, 10]. In [4] the authors propose a model for the performance analysis of HTTP over TCP assuming no packet losses. In contrast to the assumptions of the steady state approach, most TCP transfers are short and may have significant packet losses. The combined effect of these two factors does not allow most connections to reach a steady state and the connections spend a significant amount of their time in slow-start phases. However, long transfers, as shown in [8] and [10] are well modeled as spending their time in congestion avoidance and are thus unsuitable for modeling shorter flows.

Recent studies have shown that most TCP connections today carry HTTP traffic [12]. Web browsers with HTTP 1.0 open a new TCP connection for each object embedded in a web page. Studies show that this leads to TCP connections with a median size of 2-3 KBytes, an average transfer size of 8-12 KBytes with the vast majority of the connections transferring files less than 10 KBytes [3, 7, 12]. With persistent TCP connections (HTTP 1.1) which allow a single TCP connection to transfer several objects, the average size of the file transfer becomes larger and as indicated in [7], the average size of all the data on a web page is around 26-32 KBytes. In general, most of these TCP transfers are quite small from a steady-state analysis point of view. For example, with a typical MSS of 1460 bytes, a 10 KByte transfer is only 7 segments long.

This motivates the development of analytical models which take the short size of most TCP transfers into account. In [2], the authors propose a model which extends the results from [10] to finite flows by accounting for the connection establishment phase and an approximate analysis of the initial slow start. In this paper, we propose a single model which can model the latency of a TCP connection of any size and loss probability as well as give the steady state throughput for infinite flows. The main contributions of the paper can be summarized as

- A single model which accurately models both short as well as steady state TCP transfers. The model is valid for any loss probability and account for the effects of window limitation.
- *Exact* expressions for the transfer times of flows experiencing a single or no loss.

- A empirical model for TCP transfer times which is particularly useful in giving a “feel” of TCP latency and developing design guidelines.
- A sensitivity analysis on the effect of packet sizes and window limitation on TCP latency.
- Study of the performance of persistent versus non-persistent TCP connections.

Also, unlike [2], our model captures the slow start effects subsequently in the transfer which arise due to timeouts. The sensitivity analysis shows that though the transfer time reduces as the maximum window size increases, the improvements tends to saturate and beyond a point (dependent on the loss probability) there is no significant improvement. Our analysis also argues in favor of non-persistent TCP connections and we show that they have a lower transfer times than persistent connections. Our models have been validated using both simulations and real life TCP transfers over the Internet.

The rest of the paper is organized as follows. In Section 2 we give a description of the assumptions about the network and the TCP transfer. Section 3 introduces the model for finite TCP transfers while Section 5 extends it to model the steady state throughput of infinite flows and in Section 4 we extend it to account for the effects of window limitation. Section 6 presents the validation results, Section 7 introduces the empirical model, Section 8 deals with the sensitivity analysis and the comparison of persistent and non-persistent connections while Section 9 presents a discussion of the results and concluding remarks.

2 Assumptions

We follow a network and TCP transfer scenario similar to those in [10] and [2]. We assume that the hosts use a congestion control algorithm from the TCP Reno family. Also, our model considers the latency arising only from TCP's performance and thus we 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. As in [2], we do not account for the effects of Nagle's algorithm and silly window avoidance.

We assume that the receiver sends an acknowledgment for every packet it receives, i.e. we do not model the effect of delayed acknowledgments. Some operating systems, for example Linux 2.2, follow an adaptive implementation of delayed acknowledgments which are better modeled by this scheme. However, as our results show, the effect of delayed acknowledgments becomes significant only for large loads and at extremely high loss probabilities and our model is fairly accurate for even in these regions.

As in [10] and [2] we model TCP latency by considering "rounds". A round begins with the transmission of a window of packets from the receiver and ends when the sender receives an acknowledgment for one or more of these packets. We assume that the losses in a round are independent of losses in other rounds. Unlike [10] and [2], in this paper we assume that losses in a round are independent. This assumption is better suited to model losses occurring in networks with RED queues while the correlated error model of [10] and [2] are better suited for FIFO drop tail queueing [2]. Also, for short transfers, the *cwnd* values are expected to be small and the flows usually do not use a significant portion of the path's bandwidth. A binomial model is thus well suited for modeling the total number of losses suffered by a flow [1].

Finally, we assume that the time to transmit all the packets in a round is smaller than the duration of the round and that the duration of a round is independent of the window size. The amount of data transferred is assumed to be arbitrary allowing on one hand for extremely short transfers which suffer extremely few or no losses and on the other hand, for large transfers whose latencies might be modeled using their steady state throughput.

3 Modeling TCP Connections

In this section we propose our model to estimate the transfer time of TCP connections as a function of the RTT and the loss probability on the path. We propose a single model which can be used for arbitrary data transfer sizes and can also give the steady state throughput of infinite TCP connections. Our approach is based on estimating the transfer time given that a flow suffers a specified number of losses. This, combined with the probability that a flow suffers a given number of losses, then gives us the expected transfer time of a TCP flow. The model, in addition to accounting for the slow start at the beginning of each connection, also accounts for such events later in the flow which may arise due to timeouts. We first derive the transfer time of a TCP connection when it does not suffer any losses and then derive the expressions for TCP flows with a single loss. The model is then extended for multiple losses. In this section, we neglect the effect of window limitations and relax the assumption in Section 4. Also, we derive the transfer times as a function of N, the total number of unique packets to be transferred, where $N = \lceil \frac{data\_size}{mss} \rceil$.

3.1 Connection Establishment

A TCP connection begins with a three-way handshake, beginning with the initiating host sending a SYN segment. The receiver responds with an ACK for the initiating host's SYN and also sends a SYN packet of its own. When the initiating host receives this SYN/ACK packet it assumes that the connection has been established and confirms this by sending an ACK of its own. 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. Then, following the arguments and the derivation of [2], it can be shown that, for loss rates p low enough that most connections are successfully established before TCP gives up, the expected duration of the connection setup phase, t_{setup}, can be approximated as

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

3.2 Flows Without Losses

In the case when there are no losses, the flow begins in slow start and the window expands exponentially till the transfer is over. Thus the time to transfer N packets, t_{no_loss} is given by

$$t_{no\_loss}(N) = \lfloor \log_2 N \rfloor + 1 \tag{2}$$

3.3 Flows with a Single Loss

We now consider the case when the flow experiences a single loss. Consider a flow of N packets where the i^{th} packet is lost. The window increases exponentially till the first i packets are transmitted. The time to transmit the first i packets is then given by $\lfloor \log_2 i \rfloor + 1$. Now, if $cwnd$ is less than 4 in the round when the packet is lost, the loss will lead to a timeout. The $cwnd$ now reduces to 1 and has an effectively linear increase (since the exponential phase after the timeout lasts till $cwnd = 2$) till the remaining packets are transmitted. The average duration of the timeout period is given by βRTT as calculated in [10] and is given by

$$\beta = \frac{2(1 + p + 2p^2 + 4p^3 + 8p^4 + 18p^5 + 32p^6)}{1-p} \tag{3}$$

After the timeout, $N - 2i + 2$ packets remain to be transmitted. The time to transmit the $N - 2i + 2$ packets is then determined by solving for n in $\sum_{j=1}^{n} j = N - 2i + 2$. The time required to transfer N packets is then given by

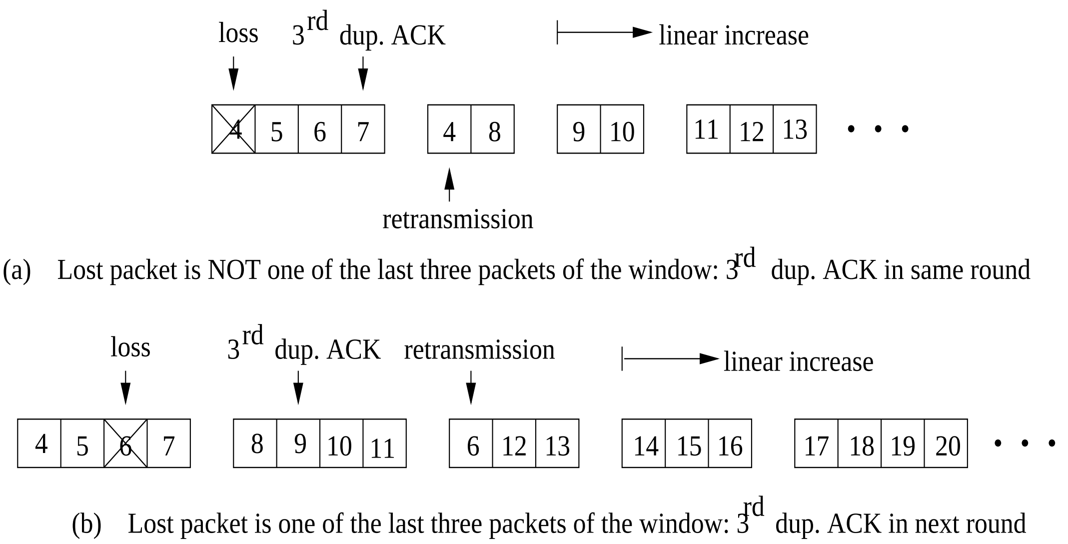


Figure 1: Window increase patterns for TCP flows with different positions of the lost packet in a window.

For $i = 1, 2, 3$

$$t_{1\_loss}(N) = \left[\lceil \log_2 i \rceil + \beta + \left\lceil \frac{-1 + \sqrt{1 + 8(N - 2i + 2)}}{2} \right\rceil \right] RTT \tag{4}$$

Observe that the first term in Equation (4) corresponds to the time to transmit the first i packets, β gives the duration of the timeout period and the last term is the solution of $\sum_{j=1}^{n} j = N - 2i + 2$.

For rounds in which *cwnd* is greater than or equal to 4, a single error in any packet will lead to congestion avoidance, not timeout. Once three duplicate acknowledgments are received the sender infers a loss and retransmits. To find the total transfer time, we first find the number of packets k that are transmitted *before* the window cuts down to $cwnd/2$. This is calculated in Equation (6). Note that we can have two cases depending on whether or not the lost packet was amongst the last three packets of the round. 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 acknowledgments. Figure 1 illustrates this concept. The time to transmit the remaining $N - k$ packets is then determined by solving for n in $\sum_{j=cwnd/2}^{n} j = N - k$ which yields the third term in Equation (5). The time required to transfer N packets is then given by

For $i = 4, \cdots, N$

$$t_{1\_loss}(N) = \left[\lfloor \log_2 i \rfloor + 3 \right] RTT + \begin{cases} \left[\left\lceil \frac{-1+\sqrt{1+4(2(N-k)+n(n-1))}}{2} \right\rceil - n + 1 \right] RTT, & \text{if } 2^{\lfloor \log_2 i \rfloor + 1} - i < 4 \\ \left[\left\lceil \frac{-1+\sqrt{1+4(2(N-k)+n(n-1))}}{2} \right\rceil - n \right] RTT, & \text{otherwise} \end{cases} \tag{5}$$

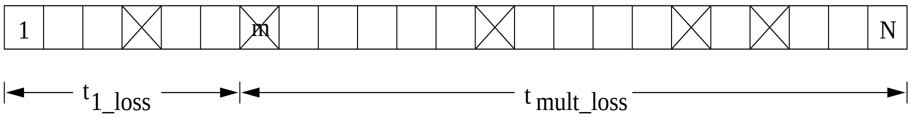


Figure 2: Modeling a flow by breaking it up into the section which has the first loss and the remaining section with the rest of the losses.

where $n = \lceil \frac{i}{2} \rceil$ and

$$k = \begin{cases} \lceil \frac{i}{2} \rceil + 2i - 2, & \text{if } 2^{\lfloor \log_2 i \rfloor + 1} - i < 4 \\ \lceil \frac{i}{2} \rceil + 2^{\lfloor \log_2 i \rfloor + 1} - 2, & \text{otherwise} \end{cases} \tag{6}$$

Note that the third term of Equation (5) simplifies to the form of the third term in Equation (4) when $n = 1$. Also, the term β does not appear in Equation (5) as there is no timeout.

3.4 Flows with Multiple Losses

We now consider the case when there is more than one loss. For a flow of N packets which experiences M losses, we assume that the second loss occurs at the m^{th} packet and find the time taken to transmit the first $m-1$ packets (where there is one loss) using Equations (4) and (5) of Section 3.3. Please refer to Figure 2 for an illustration. After the second loss, $N-m+1$ packets remain to be transmitted with $M-2$ losses within these packets. Assuming uniform distribution of losses, the average distance (in number of packets) between two consecutive losses, D_{ave}, is given by

$$D_{ave} = \frac{N-m+1}{M-1} \tag{7}$$

We now take D_{ave} and look at its substructure. We compute the average time to transmit D_{ave} packets which multiplied by $M-2$ gives the time required to transfer the remainder of the flow after the first $m-1$ packets. After the first loss, we approximate the possible range of values of *cwnd* when the subsequent losses occur by $\{1, \cdots, \lceil c\frac{-1+\sqrt{1+16D_{ave}}}{2} \rceil\}$. The factor $c = \frac{(3+10RTT)(1-p)^2}{4(1+p+p^2)^3}$ was empirically derived using simulations and curve fitting. To keep the analysis tractable, we assume that each of these possible values of *cwnd* and the position of the lost packet within a *cwnd* are equally likely when the loss occurs.

When a TCP transfer is in the congestion avoidance mode, as it will be after the first loss, a subsequent loss leads to a timeout only if the *cwnd* value during that round is less than 4. (For tractability, we neglect the cases when the subsequent losses occur while the connection is in the slow start mode after a timeout). The duration of the timeout is again given by βRTT. Also, as explained in the previous section, the time to transmit the remaining packets after the timeout is obtained by solving

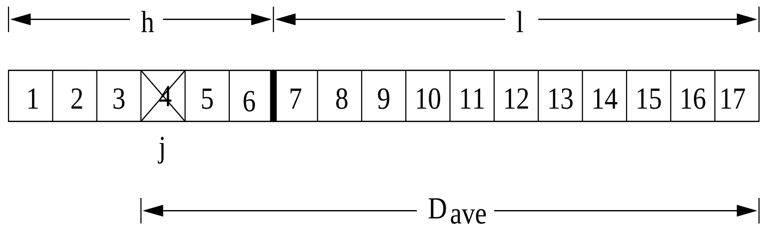


Figure 3: The parameters h, j and l and their relation to D_{ave}.

an equation of the form $\sum_{j=1}^{n} j = x$ for n and yields the last terms of Equation (8). For such a flow with $cwnd = h$ when the j^{th} packet in the round is lost (see Figure 3), the time taken for the connection to transmit l subsequent packets is given by

$$t_{timeout}(l) = \begin{cases} \left[\beta + \left\lceil \frac{-1+\sqrt{1+8(l-h+1)}}{2} \right\rceil\right] RTT, & \text{if } j = 1 \\ \left[1 + \beta + \left\lceil \frac{-1+\sqrt{1+8(l-h+1)}}{2} \right\rceil\right] RTT, & \text{otherwise} \end{cases} \tag{8}$$

On the other hand, losses which occur when $cwnd \geq 4$ lead to fast retransmissions. The time taken to transmit l packets after such a loss, given that the j^{th} packet of a window of size h is lost, is given by

$$t_{fast\_retrans}(l) = \begin{cases} \left[4 - n + \left\lceil \frac{-1+\sqrt{1+4[2(l-k)+n(n-1)]}}{2} \right\rceil\right] RTT, & \text{if } h - j < 3 \\ \left[3 - n + \left\lceil \frac{-1+\sqrt{1+4[2(l-k)+n(n-1)]}}{2} \right\rceil\right] RTT, & \text{otherwise} \end{cases} \tag{9}$$

where $n = \lceil \frac{h}{2} \rceil$ and

$$k = \begin{cases} h + n - 2 & \text{if } h - j < 3 \\ h - j + n - 1 & \text{otherwise} \end{cases} \tag{10}$$

Note that Equations (8) and (9) have a form similar to that of Equations (4) and (5).

We can now use Equations (8) and (9) to calculate the average time between two losses and thus the transfer time of the remaining $N - m + 1$ packets. We first find the expected time to transmit D_{ave} packets and multiply it by $M - 2$ to get the time required to transmit the packets after the second loss. This, combined with the time to transmit the first $m - 1$ packets using Equations (4) and (5) then gives the expected transfer time for the flow. The expected transfer time for a flow of N packets with multiple losses, $t_{mult\_loss}(N)$, is thus given by

$$t_{mult\_loss}(N) = E\{t_{1\_loss}(m-1)\} + E\{(M-2)t_{timeout}(D_{ave})\} + E\{(M-2)t_{fast\_retrans}(D_{ave})\} \tag{11}$$

where m is the sequence number of the second packet which is lost and the expectation operation is carried over all possible values of m and the number of losses M.

3.5 The Expected Transfer Time

Combining the results of the previous sections, the expected transfer time for a flow of N packets is given by

$$T_{transfer}(N) = t_{setup} + (1-p)^N t_{no\_loss}(N) + p(1-p)^{N-1} E\{t_{1\_loss}(N)\} + t_{mult\_loss}(N) \tag{12}$$

A program to calculate the expected transfer time is available online. However we do not disclose the URL in this paper to comply with the double blind review process.

4 Effect of Window Limitation

In the beginning of a TCP connection, the receiver advertises a maximum buffer size which determines the maximum window size that the sender may have. We denote this maximum limit by W_{max} and during a period when there are no losses *cwnd* grows to reach this value. In the previous sections we assumed that the connection does not experience any such limitation in the course of the transfer and in this section we find the expressions for the transfer time when this assumption is relaxed.

4.1 Flow Without Losses

When a flow does not experience any losses, the congestion window increases exponentially till it reaches W_{max} and then stays there till the transfer is complete. The number of packets transmitted when *cwnd* reaches W_{max}, $N_{exponential}$, is given by

$$N_{exponential} = 2^{\lceil log_2 W_{max} \rceil} + W_{max} - 1 \tag{13}$$

The transfer time for N packets is then

$$t_{no\_loss}(N) = \begin{cases} \lfloor \log_2 N \rfloor + 1, & \text{if } N \leq N_{exponential} \\ \lfloor \log_2 W_{max} \rfloor + \lceil \frac{N - N_{exponential}}{W_{max}} \rceil + 2, & \text{otherwise} \end{cases} \tag{14}$$

4.2 Flows with a Single Loss

For flows with a single loss, we model the flow as two parts: the slow start phase till *cwnd* reaches W_{max} or the loss occurs followed by a timeout or a congestion avoidance phase. Consider 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_{max}, can be expressed as

$$n_{max} = 2^{m+1} + \left\lceil \frac{\max\{i - 2^{m+1} + 1, 0\}}{W_{max}} \right\rceil W_{max} - 1 \tag{15}$$

where $m = \min\{\lfloor \log_2 i \rfloor, \lfloor \log_2 W_{max} \rfloor\}$. The time to successfully transmit the first $i-1$ packets is given by $t_{no\_loss}(i)$ from Equation (14). As argued in Section 3.3, the flow will lead to a timeout only if *cwnd* is less than 4. After the *cwnd* is set to 1 and the remaining packets are retransmitted while *cwnd* increases linearly (the exponential phase lasts till $cwnd = 2$). Following the arguments of Section 3.3 the total time required to transfer N packets is then given by

For $i = 1, 2, 3$

$$t_{1\_loss} = [t_{no\_loss}(i-1) + \beta]\, RTT + \begin{cases} \left\lceil \frac{-1+\sqrt{1+8(N-k)}}{2} \right\rceil RTT & \text{if } N-k \leq N_{ca} \\ \left[W_{max} + \left\lceil \frac{2(N-k) - W_{max}(W_{max}+1)}{2W_{max}} \right\rceil \right] RTT & \text{otherwise} \end{cases} \tag{16}$$

with $t_{no\_loss}(0) = 0$, $k = \min\{i, W_{max}\} + i - 2$ and $N_{ca} = W_{max}(W_{max}+1)/2$. Note that the first part of the previous expression has the same form as that of Equations (4) and (5). For rounds in which $cwnd \geq 4$, a single error in any packet will lead to congestion avoidance. As in Section 3.3, to find the total transfer time we first find the number of packets k (given in Equation (19)) that are transmitted before the window cuts down to $cwnd/2$. The time to transmit the remaining $N-k$ is then determined by solving for n in $\sum_{j=cwnd/2}^{n} j = N - k$. The time required to transfer N packets is then given by

For $i = 4, \cdots, N$

$$t_{1\_loss} = [t_{no\_loss}(i) + 3]\, RTT + \begin{cases} [t_{linear}(N-k, n) - n + 1]\, RTT, & \text{if } n_{max} - i < 3 \\ [t_{linear}(N-k, n) - n]\, RTT, & \text{otherwise} \end{cases} \tag{17}$$

where $n = \lceil \frac{\min\{i, W_{max}\}}{2} \rceil$ and

$$t_{linear}(a, b) = \begin{cases} \left\lceil \frac{-1+\sqrt{1+4(2a+b(b-1))}}{2} \right\rceil & \text{if } 2a \leq W_{max}(W_{max}+1) - b(b-1) \\ w_{max} + \left\lceil \frac{2a - W_{max}(W_{max}+1) + b(b-1)}{2W_{max}} \right\rceil & \text{otherwise} \end{cases} \tag{18}$$

$$k = \begin{cases} i + \min\left\{ W_{max}, \left\lceil \frac{\min\{i, W_{max}\}}{2} \right\rceil + \min\{i, W_{max}\} - 1 \right\} - 1, & \text{if } n_{max} - i < 3 \\ i + \min\left\{ \lceil \frac{i}{2} \rceil + n_{max} - i, W_{max} \right\} - 1, & \text{otherwise} \end{cases} \tag{19}$$

Note that the first part of Equation (18 is the case when *cwnd* does not reach W_{max} and the second part for the other case. Also, a and b in the equation are dummy variables which assume variables from Equation (17).

4.3 Flows with Multiple Losses

For the multiple loss case, we follow the same approach as in Section 3.4. To account for the effect of window limitation on the possible values of $cwnd$, we limit its range when subsequent losses occur to $\min\{\lceil c\frac{-1+\sqrt{1+16D_{ave}}}{2}\rceil, W_{max}\}$ where D_{ave} is given by Equation (7) and $c = \frac{(3+10RTT)(1-p)^2}{4(1+p+p^2)^3}$. Now, following the arguments of Section 3.4, for a flow with $cwnd = h$ when the j^{th} packet in the round is lost, the time to transmit l subsequent packets is given by

$$t_{timeout}(l) = [\beta + I(j > 1)]RTT + \begin{cases} \left\lceil \frac{-1+\sqrt{1+8(l-h+1)}}{2} \right\rceil RTT & \text{if } l-h+1 \le N_{ca} \\ \left[W_{max} + \left\lceil \frac{2(l-h+1)-W_{max}(W_{max}+1)}{2W_{max}} \right\rceil \right] RTT & \text{otherwise} \end{cases} \tag{20}$$

where $N_{ca} = W_{max}(W_{max}+1)/2$ and $I(x)$ is the indicator function. On the other hand, losses which occur when $cwnd \geq 4$ lead to fast retransmissions. The time to transmit l subsequent packets (see Figure 3) is then given by

$$t_{fast\_retrans}(l) = \begin{cases} [4 - n + t_{linear}(l-k, n)]\, RTT, & \text{if } h - j < 3 \\ [3 - n + t_{linear}(l-k, n)]\, RTT, & \text{otherwise} \end{cases} \tag{21}$$

where $t_{linear}(a, b)$ is defined in Equation (18), $n = \lceil \frac{h}{2} \rceil$ and

$$k = \begin{cases} \min\{h + n - 1, W_{max}\} - 1 & \text{if } h - j < 3 \\ \min\{h - j + n, W_{max}\} - 1 & \text{otherwise} \end{cases} \tag{22}$$

Equations (20) and (21) can now be used to calculate the transfer time of the remaining $N - m + 1$ packets. We now use the procedure outlined in Section 3.4 to calculate the transfer time. As Equations (16) and (17) can be used to obtain the time to transmit the first $m - 1$ packets, the expected transfer time for a flow of N packets with multiple losses, $t_{mult\_loss}(N)$, is thus given by

$$t_{mult\_loss}(N) = E\{t_{1\_loss}(m-1)\} + E\{(M-2)t_{timeout}(D_{ave})\} + E\{(M-2)t_{fast\_retrans}(D_{ave})\} \tag{23}$$

where m is the sequence number of the second packet which is lost and the expectation operation is carried over all possible values of m and the number of losses M.

4.4 The Expected Transfer Time

Combining the results of the previous sections, the expected transfer time for a flow of N packets is given by

$$T_{transfer}(N) = t_{setup} + (1-p)^N t_{no\_loss}(N) + p(1-p)^{N-1} E\{t_{1\_loss}(N)\} + t_{mult\_loss}(N) \tag{24}$$

where t_{setup}, $t_{no\_loss}(N)$ and $t_{mult\_loss}(N)$ are defined in Equations (1), (14) and (23) respectively and $t_{1\_loss}(N)$ is defined in Equations (16) and (17). A program to calculate the expected transfer time is available online. However we do not disclose the URL in this paper to comply with the double blind review process.

5 Modeling Infinite Flows

In this section we extend the analysis of the previous section to model infinite TCP flows and their steady state throughput. As in [10], we assume that the source has an unlimited amount of data to send. The loss model assumed for this section is the independent loss model introduced in Section 2 in contrast with the burst loss model of [10].

To extend the model of Sections 3 and 4 to an infinite flow, we first note that for an infinite flow, the probability that the flow experiences a single or no loss is 0 for $p > 0$. Since these flows will have multiple losses, we can use Equations (20) and (9) to find the average transmission time between successive errors. Also, for a packet loss probability of p, the average number of packets transmitted between two successive losses, d, is given by $1/p$. Using this and the average transmission time between successive losses, we can calculate the steady state throughput of the flow. The possible values of the $cwnd$ in this case can thus be approximated to vary uniformly between 1 and cw_{max}, where cw_{max} is given by

$$cw_{max} = min\left\{\left\lceil c\frac{-1+\sqrt{1+16/p}}{2}\right\rceil, W_{max}\right\} \tag{25}$$

where $c = \frac{(3+10RTT)(1-p)^2}{4(1+p+p^2)^3}$. The expected time to transfer d packets can then be written as

$$t_{ss} = \frac{2}{cw_{max}(cw_{max}+1)}\left[\sum_{i=1}^{3}\sum_{j=1}^{i} t_{timeout}(d) + \sum_{i=4}^{cw_{max}}\sum_{j=1}^{i} t_{fast\_retransmit}(d)\right] \tag{26}$$

where $t_{timeout}(d)$ and $t_{fast\_retransmit}(d)$ are defined in Equations (20) and (21). The steady state throughput in bytes per second of a TCP connection is thus

$$R = \frac{dMSS}{t_{ss}} = \frac{MSS}{t_{ss}p} \tag{27}$$

We note that though the expression for the steady state throughput does not have the same closed form as that derived by Padhye et al. in [10], their numerical values are almost the same. The numerical results and their comparison with measurements from TCP connections are given in the next section.

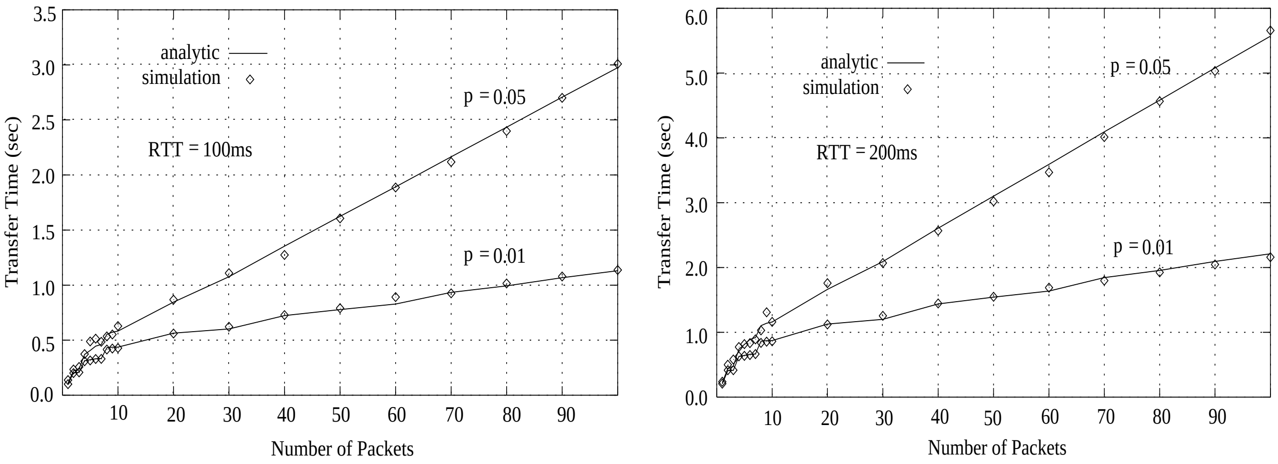


Figure 4: Transfer times from analytic and simulation results for various file sizes and loss probabilities of 0.01 and 0.05. The figures show the results for RTTs of 100 and 200ms.

6 Model Verification Results

The models proposed in the previous sections are verified in this section using both simulations and real world TCP connections over the Internet. We also verify the extension of our model to calculate the steady state throughput of TCP connections by comparing it with the results of [10].

The simulations were conducted using the network simulator *ns* [13] using a simple topology with a sender connected to a receiver through a router. The links speeds were assumed to be 1Mbps and the router had a loss probability p in the forward direction. We also assumed that there was no cross traffic and the `FullTCP` model of *ns* was used as the transport protocol. The transfer time for a given file size was determined by averaging over 1000 trials. The results are shown in Figure 4 for RTTs of 100 and 200 milliseconds loss probabilities of 0.01 and 0.05. The model fits the experimental results very well and the worst case error over all the cases we tried was less than 5%.

To verify the accuracy of the proposed model in real world TCP connections, we conducted a number of TCP connections from a local machine to machines in various domains both in the United States and abroad. We show the results for TCP file transfers to the midwest USA, west coast USA and Europe. The local machine was running Solaris 5.6 while the others were running HP-UX, FreeBSD CAIRN-2.5 and FreeBSD 3.3 respectively. In Figures 5, 6 and 7 we show the transfer times for various files sizes to these destinations and compare them to our model. The measurement results were generated by conducting FTP transfers of randomly generated file sizes (using a uniform distribution) with the sender at the local machine and the receiver at the remote domain. The two sets of results correspond

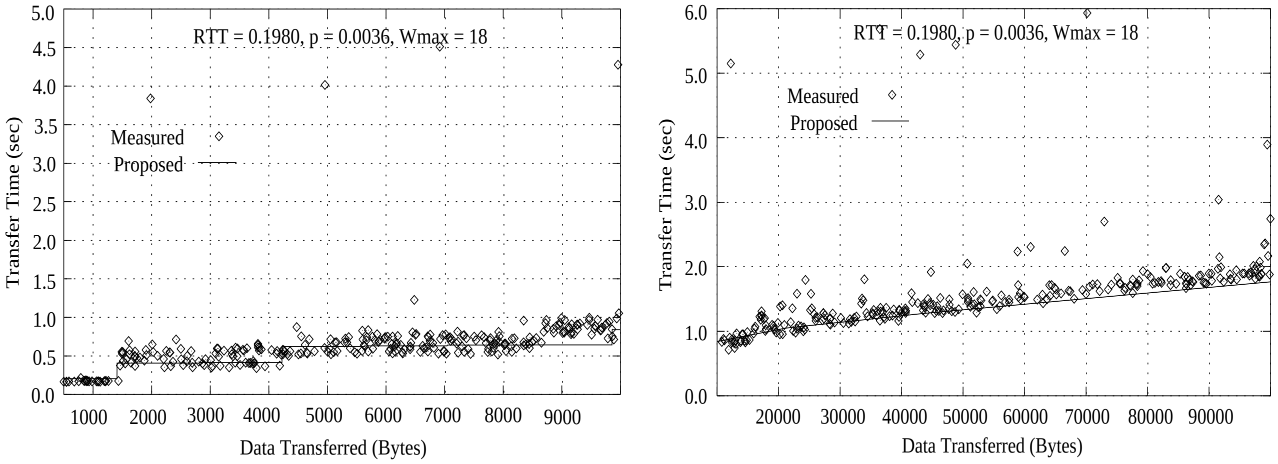


Figure 5: Transfer times of measured and modeled TCP transfer from the local machine to University of Pisa, Italy.

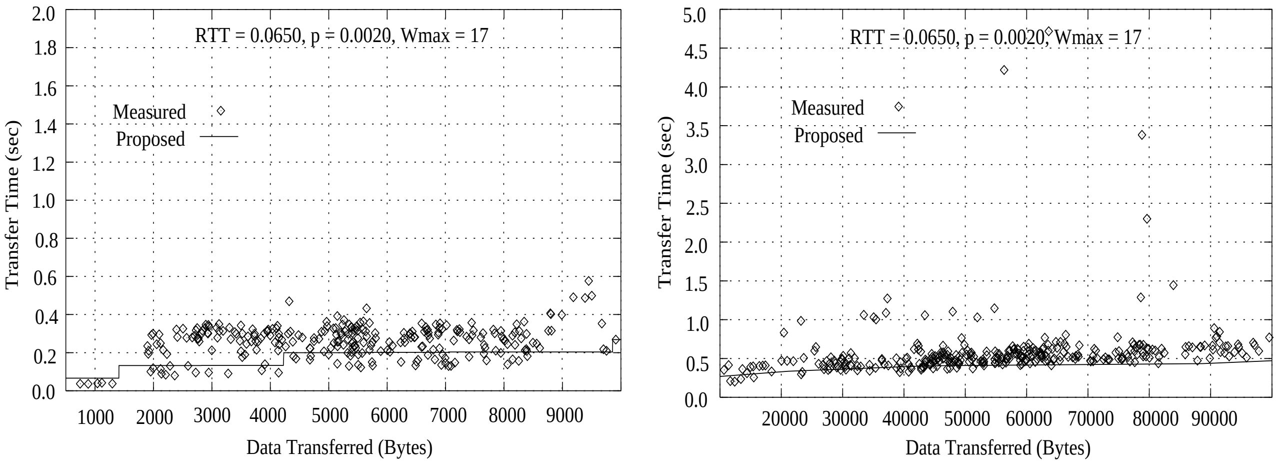


Figure 6: Transfer times of measured and modeled TCP transfer from the local machine to Ohio State University.

to the cases when the file sizes generated were between 100-10000 bytes and 10000-100000 bytes. The results from our model match very closely with those of the measurements.

Figures 8 and 9 show the results of our model extended to calculate the steady state throughput of TCP connections. We compare our results with those of Padhye et al. [10] and the traces reported therein. We plot the graphs for three of the 100 sec transfers of [10] (Fig. 13, 16 and 17 of [10]) and we see that our results match closely with both the measurements and the model proposed in [10]. We also note that there is a little deviation in the two models as the loss probability increases which might be due to the difference in the loss models. However, this does not seem to affect the accuracy and validity of the results as our results are very close to the measured values.

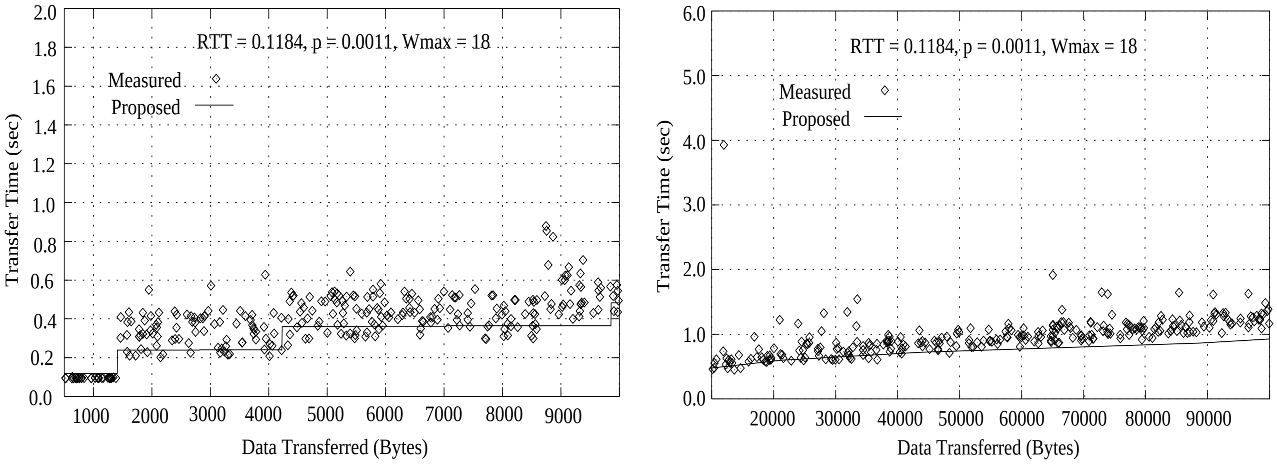


Figure 7: Transfer times of measured and modeled TCP transfer from the local machine to University of California at Los Angeles.

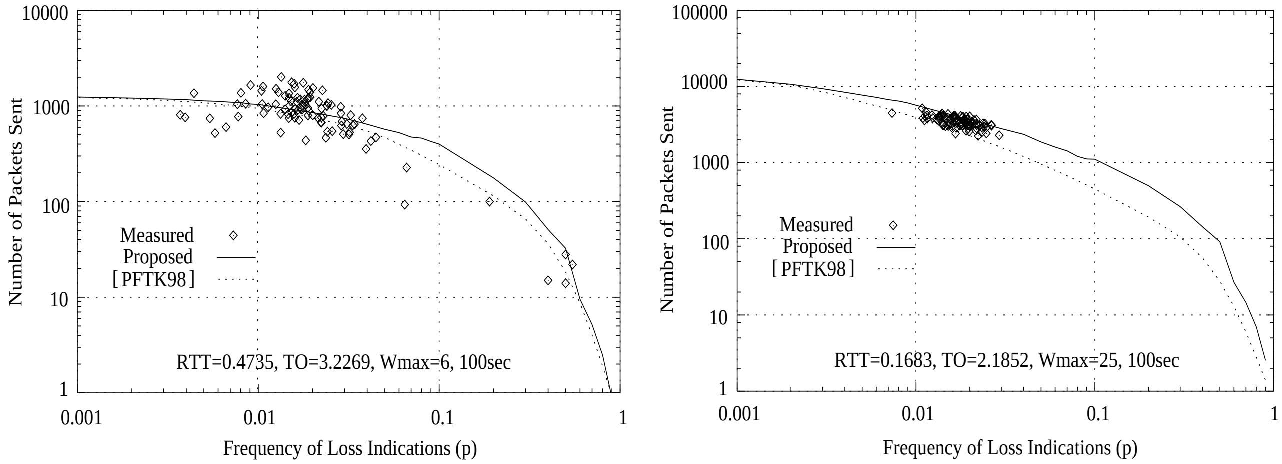


Figure 8: Steady state throughput of TCP connections from the proposed model compared with the model and measurements from [10] (Fig. 13 and 16 of [10]).

7 Empirical Model

To get a "feel" of TCP's behavior which might lead to design guidelines, we now propose an empirical model for TCP latency. the model gives a rough idea of the dependence of TCP latency on the loss probability and window limitation. Empirically, the time to transfer N packets using TCP can be approximated as

$$T_{transfer}(N) = \left[\log_2 N + \frac{4.5}{1 + RTT^2} pN + \frac{1+p}{W_{max}\sqrt{W_{max}}}\right] RTT \tag{28}$$

The transfer times predicted by the empirical model are compared with those obtained from Equation (24) in Figure 10 and as can be seen, the empirical model is a very close approximation over a wide

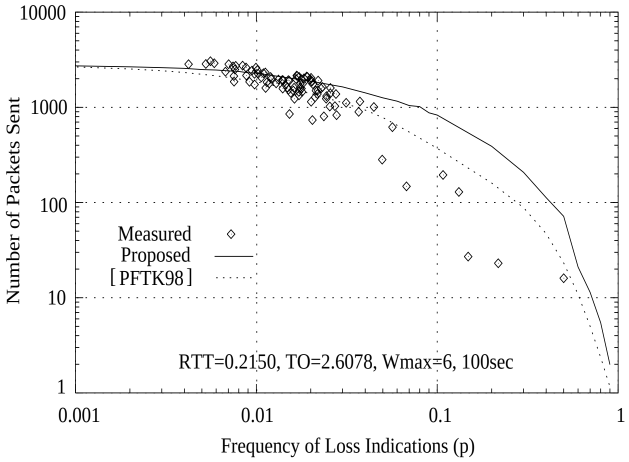


Figure 9: Steady state throughput of TCP connections from the proposed model compared with the model and measurements from [10] (Fig. 17 of [10]).

range of RTTs, loss rates, W_{max} and file sizes. The empirical model can thus be used to get a rough indication of the dependence of TCP's latency on various parameters.

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 neglecting any window limitation effects. The third term accounts for the effects of window limitation. The model indicates that the transfer time increases linearly 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} and thus there is no significant improvement in the latency if W_{max} is increased beyond a threshold. At present, we have not explored the extension of this model to steady-state flows.

8 Implications

We now investigate the sensitivity of some of the parameters on TCP latency as well as look into effects of using parallel connections for transferring large files. For the sensitivity analysis, we look into the effects of varying packet sizes and the effect of window limitation on the transfer time.

8.1 Sensitivity Analysis

In Figure 11 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

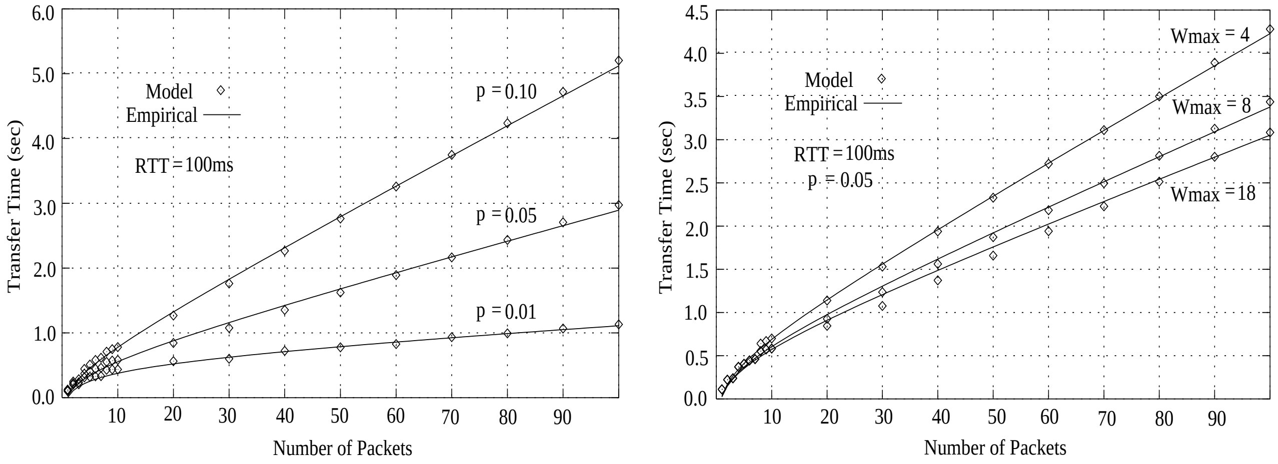
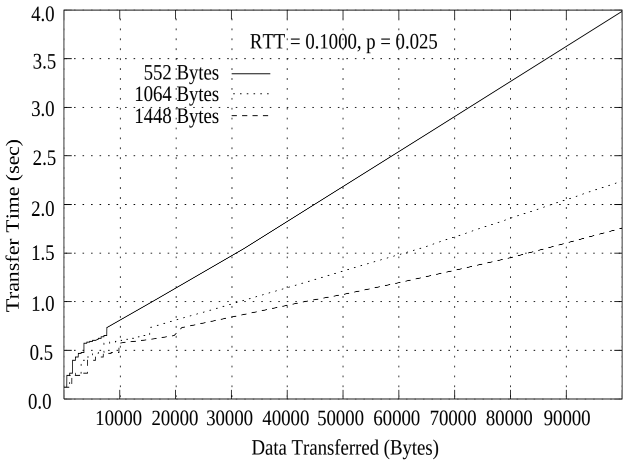


Figure 10: Transfer times from the empirical model and the analytic model of Equation (24). In the first figure there is no effect of window limitation and the transfer times are plotted for loss probabilities of 0.01, 0.05 and 0.10 while the second figure compares the transfer times for W_{max} of 4, 8 and 18.

as packet sizes get smaller and W_{max} decreases. However, we note the reduction in the transfer time with increase in W_{max} is not linear and tends to saturate. Also, 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}. However the decrease in the transfer time with large values of W_{max} is too less to justify increasing it beyond a limit determined the loss probability. To explain this, we note that for a packet loss probability p, on an 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 due to window limitation. Any further increase in W_{max} beyond this point further reduces the effect of window limitation and we see that the transfer time saturates to the value predicted by Equation (12) where we neglect the effect of window limitation. 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 with the increase in the packet size.

8.2 Persistent versus Non-Persistent Connections

As an illustration of the applications of the models developed in this paper, we now compare the performance of persistent versus non-persistent connections. Most of today's web browsers open a new TCP connection for each object embedded in a web page. However, browsers with HTTP 1.1 and some implementations of HTTP 1.0 support persistent connections allowing a single connection to download several objects. To compare their performance we consider the case when the contents of a single file are transferred using a single flow, two flows are used for the transfer with each flow



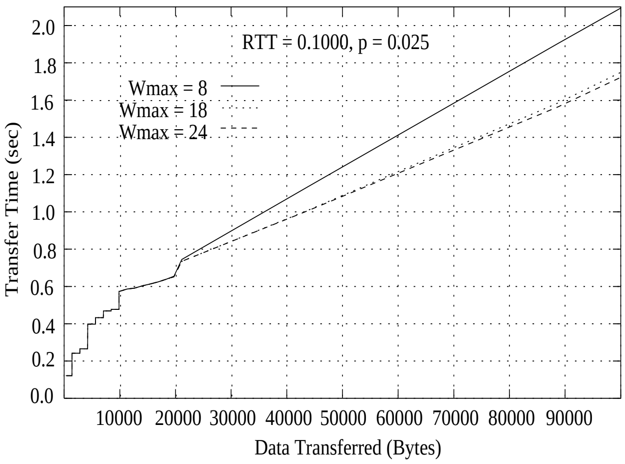


Figure 11: Effect of packet sizes and W_{max} on the transfer time of TCP connections.

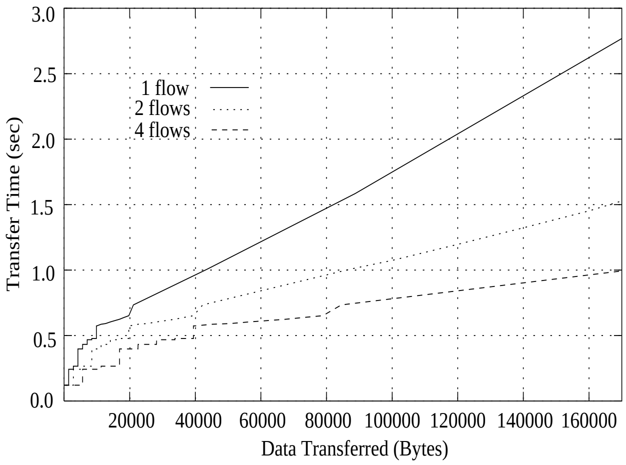


Figure 12: Effect of persistent and non-persistent connections on the transfer time of TCP connections.

transferring half of the data and finally four flows with each flow transferring one fourth of the data. We assume that the individual flows are independent which is reasonable when the data to be transferred is low. Figure 12 compares these the transfer times and it is evident that the transfer times decrease when separate connections are used to transfer different parts of the object and the transfer time is inversely proportional to the number of flows.

9 Conclusion and Discussions

In this paper we presented a model for TCP transfer times which is capable of modeling TCP latencies for arbitrary file sizes. The model can also be applied to obtain the steady state throughput of TCP connections assuming infinite flows. In contrast to earlier models which either assume no loss [4, 11], infinite flows [10] or model the slow start of TCP connections in the beginning for the transfer [2], our model accounts for arbitrary file sizes, slow start phases at all possible points in the flow and arbitrary loss probabilities. We also account for the possibility of window limitation. The model is verified using simulations as well as real life web based measurements. We also validate our steady state throughput model with the traces from [10] and show the the results are in good agreement.

The paper also presented an empirical model for estimating the latency of TCP transfers. This model 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. We show that though the transfer time decreases with increasing W_{max}, the improvement tends to saturate as W_{max} increases. This is due to the fact that for a given loss probability p, we expect to see a loss every $1/p$ packets and thus the window value of the flow will generally not exceed $1/p$. Consequently, for $W_{max} > 1/p$, the effect of window limitation is very low and further increases in W_{max} do not affect the transfer time. We also compared the performance of non-persistent and persistent connections for transferring data and show that non-persistent connections, in which a separate TCP connection is opened for each object have lower overall transfer times.

Acknowledgments

We would like to thank a long list of people who helped us including folks around the world who gave us accounts on their machines. We hide their names reluctantly for the double blind review process.

References

[1] N. Cardwell, S. Savage and T. Anderson, "Modeling the performance of short TCP Connections," University of Washington, Seattle, Oct 1998.

[2] N. Cardwell, S. Savage and T. Anderson, "Modeling TCP latency," *To appear in Proceedings of IEEE INFOCOM,* Tel Aviv, Israel, March 2000.

[3] C. R. Cunha, A. Bestavros and M. E. Crovella, "Characteristics of WWW client based traces," Technical Report BU-CS-95-010, Boston University, Jul 1995.

[4] 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, Oct 1997.

[5] 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, Aug 1997.

[6] T. V. Lakshman and U. Madhow, "The performance of TCP/IP for networks with high bandwidth-delay products and random loss," *IEEE/ACM Transactions on Networking,* Vol. 5, No. 3, pp 336-350, Jun 1997.

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

[8] 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.

[9] T. Ott, J.H.B. Kemperman and M. Mathis, "The stationary behavior of ideal TCP congestion avoidance," `ftp://ftp.bellcore.com/pub/tjo/TCPwindow.ps`, Unpublished Manuscript, Aug 1996.

[10] J. Padhye, V. Firoiu, D. Towsley and J. Kurose, "A simple model and its empirical validation," *Proceedings of ACM SIGCOMM'98,* Vancouver, BC, Canada, pp 303-314, Sep 1998.

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

[12] 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.

[13] UCB/LBLN/VINT Network Simulator - *ns* (version 2), `http://www-mash.cs.berkeley.edu/ns`, 1997.