Modeling TCP Reno Performance: A Simple Model and Its Empirical Validation

Jitendra Padhye, Victor Firoiu, Donald F. Towsley, *Fellow, IEEE*, and James F. Kurose, *Fellow, IEEE*

***Abstract*—The steady-state performance of a bulk transfer TCP flow (i.e., a flow with a large amount of data to send, such as FTP transfers) may be characterized by the *send rate*, which is the amount of data sent by the sender in unit time. In this paper we develop a simple analytic characterization of the steady-state send rate as a function of loss rate and round trip time (RTT) for a bulk transfer TCP flow. Unlike the models in [7]–[9], and [12], our model captures not only the behavior of the fast retransmit mechanism but also the effect of the time-out mechanism. Our measurements suggest that this latter behavior is important from a modeling perspective, as almost all of our TCP traces contained more time-out events than fast retransmit events. Our measurements demonstrate that our model is able to more accurately predict TCP send rate and is accurate over a wider range of loss rates. We also present a simple extension of our model to compute the *throughput* of a bulk transfer TCP flow, which is defined as the amount of data received by the receiver in unit time.**

***Index Terms*—Empirical validation, modeling, retransmission timeouts, TCP.**

I. Introduction

A SIGNIFICANT amount of today's Internet traffic, including WWW (HTTP), file transfer (FTP), e-mail (SMTP), and remote access (Telnet) traffic, is carried by the TCP transport protocol [20]. 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 [7]–[9], [12], have been directed at analytically characterizing the send rate of a bulk transfer TCP flow as a function of packet loss and round trip delay. One reason for this recent interest is that a simple quantitative characterization of TCP send rate under given operating conditions offers the possibility of defining a "fair share" or "TCP-friendly" [8] send rate 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 [21], [22].

In this paper we develop a simple analytic characterization of the steady-state send rate 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 (RTT). Unlike the recent work of [7]–[9], and [12], our model captures not only the behavior of the fast retransmit mechanism but also the effect of the time-out mechanism on send rate. The measurements we present in Section III indicate that this latter behavior is important from a modeling perspective, as we observe more time-out 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 send rate over a significantly wider range of loss rates than before; measurements presented in [9] 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 send rate and is able to do so over a wider range of loss rates.

The remainder of the paper is organized as follows. In Section II we describe our model of TCP congestion control in detail and derive a new analytic characterization of TCP send rate as a function of loss rate and average RTT. In Section III we compare the predictions of our model with a set of measured TCP flows over the Internet, having as their endpoints sites in both U.S. and Europe. Section IV discusses the assumptions underlying the model and a number of related issues in more detail. In Section V we present a simple extention of the model to calculate the throughput of a bulk transfer TCP flow. Section VI concludes the paper.

II. Model for TCP Congestion Control

In this section we develop a stochastic model of TCP congestion control and avoidance that yields a relatively simple analytic expression for the send rate 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 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 [16]. In this paper, we focus our attention on the congestion avoidance behavior of TCP and its impact on send rate, 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 time-out), limited receiver window size, and average RTT. Our model is based on the Reno flavor of TCP, as it is one of the more popular implementations in the Internet today

Manuscript received June 10, 1999; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor K. Ramakrishnan. This material is based upon work supported by the National Science Foundation under Grant NCR-95-08274, Grant NCR-95-23807, and Grant CDA-95-02639.

J. Padhye, D. F. Towsley, and J. F. Kurose are with the Department of Computer Science, University of Massachusetts, Amherst, MA 01003 USA (e-mail: jitu@cs.umass.edu; towsley@cs.umass.edu; kurose@cs.umass.edu).

V. Firoiu is with the Technology Center, Nortel Networks, Billerica, MA 01821 USA (e-mail: vfiroiu@nortelnetworks.com).

Publisher Item Identifier S 1063-6692(00)03318-5.

1063–6692/00$10.00 © 2000 IEEE

[14], [15]. We assume that the reader is familiar with TCP Reno congestion control (see for example [6], [18], and [19]) and we adopt most of our terminology from [6], [18], and [19].

Our model focuses on the congestion avoidance mechanism, where the 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 ACK's or by time-out, as discussed shortly.

We *model* the congestion avoidance behavior of TCP in terms of "rounds." A round starts with 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, 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 RTT and is assumed to be independent of the window size, an assumption also adopted (either implicitly or explicitly) in [7]–[9], and [12]. Our concept of rounds is similar to the concept of "mini-cycles" proposed in [7]. Note that we have also assumed here that the time needed to send all the packets in a window is smaller than the RTT; this behavior can be seen in observations reported in [3] and [14].

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, [19]), 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 RTT.

In the following subsections, we model the behavior of TCP 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 ACK's 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. 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 model is a simple and crude approximation to capture the loss behavior observed in studies such as [23]. We discuss this assumption further in Section IV.

We develop a stochastic model of TCP congestion control in several steps, corresponding to its operating regimes: when loss indications are exclusively TD (Section II-A), when loss indications are both TD and TO (Section II-B), and when the congestion window size is limited by the receiver's advertised window (Section II-C). Note that we do not model certain aspects of the behavior of TCP (e.g., fast recovery). However, we believe that 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 III. A more detailed discussion of model assumptions and related issues is presented in Section IV. Also note that in the following, we measure send rate in terms of packets per unit of time, instead of bytes per unit of time.

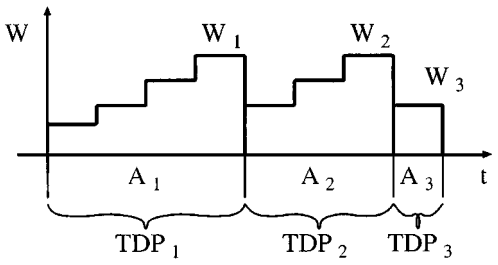

Fig. 1. Evolution of window size over time when loss indications are triple-duplicate ACK's.

A. Loss Indications are Exclusively Triple-Duplicate ACK's

In this section, we assume that loss indications are exclusively of type triple-duplicate ACK, 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 $t > 0$, define N_t to be the number of packets transmitted in the interval $[0, t]$, and $B_t = N_t/t$ to be the send rate in 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, we define the long-term steady-state send rate of a TCP connection 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 send rate of the TCP connection and p, the loss probability defined above.

A sample path of the evolution of congestion window size is given in Fig. 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 Fig. 1). For the ith TDP 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 $\{Y_i\}_i$ it can be shown that

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

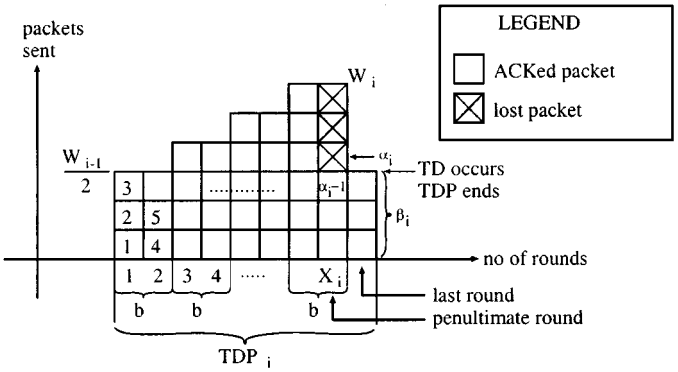

Fig. 2. Packets sent during a TDP.

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

Consider a TDP as in Fig. 2. A TDP starts immediately after a TD loss indication. Thus, the initial congestion window size is equal to $W_{i-1}/2$, half the size of the 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 Fig. 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 TDP ends), as discussed in more detail in Section II-B. 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 TDP 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 is

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

The mean of α is thus

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

From (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 (RTT) of the jth 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 i.i.d. 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

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

where

$$E[r] \sim E[r_{ij}].$$

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

Finally, to derive an expression for $E[X]$, we consider the evolution of W_i as a function of the number of rounds, as shown in Fig. 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 ith TDP, the window size increases between $W_{i-1}/2$ and W_i. Since the increase is linear with slope $1/b$, we have

$$W_i = \frac{W_{i-1}}{2} + \frac{X_i}{b}, \qquad i = 1, 2, \cdots. \tag{7}$$

The fact that Y_i packets are transmitted in TDP$_i$ is expressed by

$$Y_i = \sum_{k=0}^{X_i/b-1} \left(\frac{W_{i-1}}{2} + k\right) b + \beta_i \tag{8}$$

$$= \frac{X_i W_{i-1}}{2} + \frac{X_i}{2}\left(\frac{X_i}{b} - 1\right) + \beta_i \tag{9}$$

$$= \frac{X_i}{2}\left(\frac{W_{i-1}}{2} + W_i - 1\right) + \beta_i \quad \text{using (7)} \tag{10}$$

where β_i is the number of packets sent in the last round (see Fig. 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 is 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 (5), (7), and (10) that

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

and

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

For simplicity, we assume β_i, the number of packets in the last round, to be uniformly distributed between 1 and $W_i - 1$. 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., $E[W] \approx \sqrt{8/3bp}$ for small values of p. From (6), (11), and (13), it follows that

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

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

Observe that

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

From (1) and (5) we have

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

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

which can be expressed as

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

Thus, for small values of p, (20) reduces to the formula in [8] for $b = 1$.

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

B. Loss Indications are Triple-Duplicate ACK's and Time-Outs

So far, we have considered TCP flows where all loss indications are due to triple-duplicate ACK's. Our measurements show (see Table II) 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 ACK's) are lost, and less than three duplicate ACK's are received. The sender waits for a period of time denoted by T_0, and then retransmits nonacknowledged 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 a time-out period of $64T_0$ 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 Fig. 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

$$S_i = Z_i^{\mathrm{TD}} + Z_i^{\mathrm{TO}}.$$

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

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

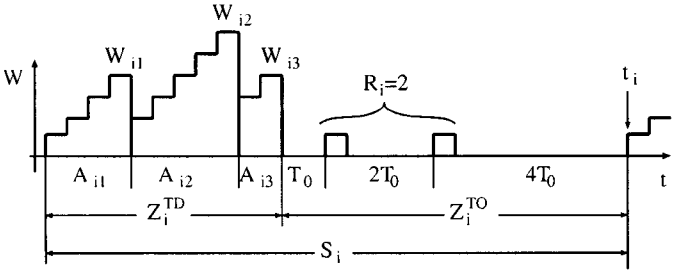

Fig. 3. Evolution of window size when loss indications are triple-duplicate ACK's and time-outs.

We extend our definition of TDP's given in Section II-A to include periods starting after, or ending in, a TO loss indication (in addition to periods between two TD loss indications). Let n_i be the number of TDP's in interval Z_i^{TD}. For the jth TDP 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 II-A, we are interested in the send rate of a TCP flow. We have

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

and, thus

$$E[M] = E\left[\sum_{j=1}^{n_i} Y_{ij}\right] + E[R]$$

$$E[S] = E\left[\sum_{j=1}^{n_i} A_{ij}\right] + E\left[Z^{\mathrm{TO}}\right].$$

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

$$E\left[\left(\sum_{j=1}^{n_i} Y_{ij}\right)_i\right] = E[n]E[Y]$$

$$E\left[\left(\sum_{j=1}^{n_i} A_{ij}\right)_i\right] = E[n]E[A].$$

To derive $E[n]$ observe that, during Z_i^{TD}, the time between two consecutive time-out sequences, there are n_i TDP's, 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

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

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

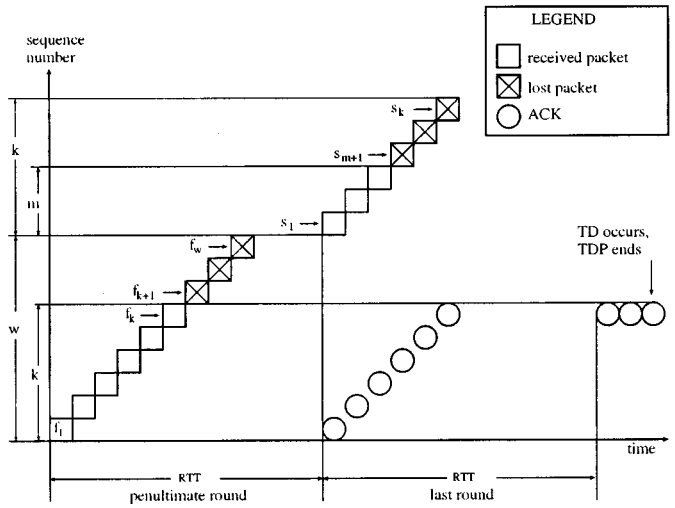

Fig. 4. Packet and ACK transmissions preceding a loss indication.

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 Fig. 4).[1] Let w be the current congestion window size. Thus packets $f_1, \cdots, f_w$ are sent in the penultimate round. Packets $f_1, \cdots, f_k$ 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 the following packets, till the end of the round. Thus, all packets following f_{k+1} in the penultimate round are also lost. However, since packets $f_1, \cdots, f_k$ are ACKed, another k packets, $s_1, \cdots, s_k$ 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 $s_{m+2} \cdots s_k$ are also lost in the last round. The m packets successfully sent in the last round are responded to by ACK's for packet f_k, which are counted as duplicate ACK's. These ACK's are not delayed [19], so the number of duplicate ACK's is equal to the number of successfully received packets in the last round. If the number of such ACK's 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

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

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

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

where $h(k)$ is given by

$$h(k) = \sum_{m=0}^{2} C(k, m). \tag{23}$$

This follows by noting that 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 $A(w, k) * C(k, m)$. After algebraic manipulations, we get the following for $\hat{Q}(w)$:

$$\min\left(1, \frac{\left(1-(1-p)^3\right)\left(1+(1-p)^3\left(1-(1-p)^{w-3}\right)\right)}{1-(1-p)^w}\right). \tag{24}$$

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

$$\hat{Q}(w) \approx \min\left(1, \frac{3}{w}\right). \tag{25}$$

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

$$Q \approx \hat{Q}(E[W]) \tag{26}$$

where $E[W]$ is given by (13).

We consider next the derivation of $E[R]$ and $E[Z^{\mathrm{TO}}]$. For this, we need the probability distribution of the number of time-outs 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 TO's occurs when there are $k-1$ consecutive losses (the first loss is given) followed by a successfully transmitted packet. Consequently, the number of TO's in a TO sequence has a geometric distribution, and thus

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

Then we can compute the mean of R:

$$E[R] = \sum_{k=1}^{\infty} kP[R = k] = \frac{1}{1-p}. \tag{27}$$

Next, we focus on $E[Z^{\mathrm{TO}}]$, 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 $2^{i-1}T_0$, $i = 1 \cdots 6$, with all immediately following time-outs having length $64T_0$. Then, the duration of a sequence with k time-outs is

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

and the mean of Z^{TO} is

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

Substituting expressions for Q, $E[S]$, $E[R]$, and $E[Z^{\mathrm{TO}}]$ in (21) we obtain the following for $B(p)$:

$$B(p) = \frac{\dfrac{1-p}{p} + E[W] + \hat{Q}(E[W]) \dfrac{1}{1-p}}{\mathrm{RTT}(E[X]+1) + \hat{Q}(E[W]) T_0 \dfrac{f(p)}{1-p}} \tag{28}$$

where

$$f(p) = 1 + p + 2p^2 + 4p^3 + 8p^4 + 16p^5 + 32p^6. \tag{29}$$

$\hat{Q}$ is given in (24), $E[W]$ in (13), and $E[X]$ in (16). Using (14), (17), and (25), we have that (28) can be approximated by

$$B(p) \approx \frac{1}{\mathrm{RTT}\sqrt{\dfrac{2bp}{3}} + T_0 \min\left(1, 3\sqrt{\dfrac{3bp}{8}}\right) p(1+32p^2)}. \tag{30}$$

C. 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_m. As a consequence, during a period without loss indications, the window size can grow up to W_m, but will not grow further beyond this value. An example of the evolution of window size is depicted in Fig. 5.

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

$$E[W_u] = \frac{2+b}{3b} + \sqrt{\frac{8(1-p)}{3bp} + \left(\frac{2+b}{3b}\right)^2}. \tag{31}$$

We assume that if $E[W_u] < W_m$, we have the approximation $E[W] \approx E[W_u]$. In other words, if $E[W_u] < W_m$, the receiver-window limitation has negligible effect on the long term average of the TCP send rate, and thus the send rate is given by (28).

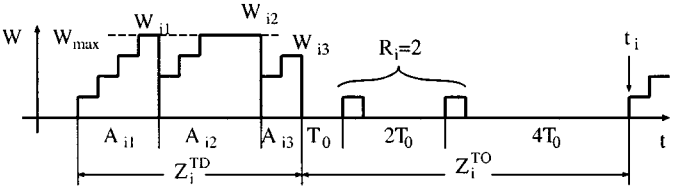

Fig. 5. Evolution of window size when limited by W_m.

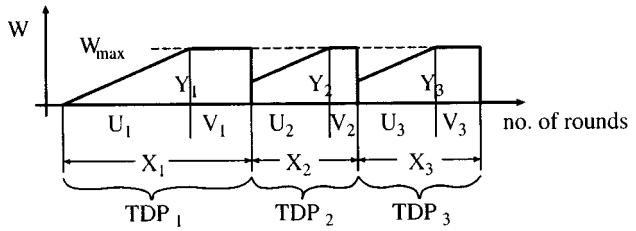

Fig. 6. Fast retransmit with window limitation.

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

$$W_m = \frac{W_m}{2} + \frac{U_i}{b} \qquad \forall i \ge 2$$

which implies $E[U] = (b/2)W_m$. Also, considering the number of packets sent in the ith TDP, we have

$$Y_i = \frac{U_i}{2}\left(\frac{W_m}{2} + W_m\right) + V_i W_m$$

and then

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

Since Y_i, the number of packets in the ith TDP, does not depend on window limitation, $E[Y]$ is given by (5), $E[Y] = (1-p)/p + W_m$, and thus

$$E[V] = \frac{1-p}{pW_m} + 1 - \frac{3b}{8} W_m.$$

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

$$E[X] = E[U] + E[V] = \frac{b}{8} W_m + \frac{1-p}{pW_m} + 1.$$

By substituting this result in (28), we obtain the TCP send rate, $B(p)$, when the window is limited:

$$B(p) = \frac{\dfrac{1-p}{p} + W_m + \hat{Q}(W_m) \dfrac{1}{1-p}}{\mathrm{RTT}\left(\dfrac{b}{8} W_m + \dfrac{1-p}{pW_m} + 2\right) + \hat{Q}(W_m) T_0 \dfrac{f(p)}{1-p}}.$$

In conclusion, the complete characterization of TCP send rate, $B(p)$, is

$$B(p) = \begin{cases} \dfrac{\dfrac{1-p}{p} + E[W] + \hat{Q}(E[W])\dfrac{1}{1-p}}{\mathrm{RTT}\left(\dfrac{b}{2}E[W_u]+1\right) + \hat{Q}(E[W])T_0\dfrac{f(p)}{1-p}}, & E[W_u] < W_m \\ \dfrac{\dfrac{1-p}{p} + W_m + \hat{Q}(W_m)\dfrac{1}{1-p}}{\mathrm{RTT}\left(\dfrac{b}{8}W_m + \dfrac{1-p}{pW_m} + 2\right) + \hat{Q}(W_m)T_0\dfrac{f(p)}{1-p}}, & \text{otherwise} \end{cases} \tag{32}$$

where $f(p)$ is given in (29), $\hat{Q}$ is given in (24), and $E[W_u]$ in (13). In the following sections, we will refer to (32) as the "full model." The following approximation of $B(p)$ follows from (30) and (32) as shown in

$$\min\left(\frac{W_m}{\mathrm{RTT}}, \frac{1}{\mathrm{RTT}\sqrt{\dfrac{2bp}{3}} + T_0 \min\left(1, 3\sqrt{\dfrac{3bp}{8}}\right) p(1+32p^2)}\right). \tag{33}$$

In Section III, we verify that (33) is indeed a very good approximation of (32). Henceforth we will refer to (33) as the "approximate model."

III. Measurements and Trace Analysis

Equations (32) and (33) provide an analytic characterization of TCP send rate 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 several TCP connections established between hosts scattered across U.S. and Europe.

Table I lists the domains and operating systems of the hosts used for the measurements.[2] 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 [14] and [15]. For example, when we analyze traces from a Linux sender, we account for the fact that TD events occur after getting only two duplicate ACK's instead of three. Our trace analysis programs were further verified by checking them against `tcptrace` [11] and `ns` [10].

We carried out two different sets of measurement experiments. Table II summarizes data from the first set. Each row in the table corresponds to a 1-h long TCP connection in which the sender behaves as an "infinite source"—it always has data to send and thus TCP send rate is only limited by the TCP congestion control. The experiments were performed at randomly selected times during 1997 and the beginning of 1998. The third and fourth column of Table II indicate the number of packets sent and the number of loss indications, respectively (triple-duplicate ACK or time-out). 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 [9]. The next six columns show a breakdown of the loss indications by type: the number of TD events, the number of "single" time-outs, having duration T_0, the number of "double" time-outs, $T_1 = 2T_0$, 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 RTT, and average duration of a single time-out T_0. These values have been averaged over the entire trace. When calculating RTT values, we follow Karn's algorithm, in an attempt to minimize the impact of time-outs and retransmissions on the RTT estimates. An important observation to be drawn from the data in these tables is that in all traces, time-outs constitute the majority or a significant fraction of the total number of loss indications. This underscores the importance of including the effects of time-outs in the model of TCP congestion control. In addition to single time-out events (column T_0), it can be seen that exponential backoff (multiple time-outs) occurs with significant frequency.

For the second set of experiments, we established 100 serially-initiated TCP connections between a given sender-receiver pair. Each connection lasted for 100 s, and was followed by a 50-s gap before the next connection was initiated. These experiments were performed at randomly selected times during 1998. These connections showed loss patterns similar to those observed for the 1-h long connections.

The graphs in Fig. 7 compare the predictions of the proposed model, and the predictions of the model proposed in [9] with measurement data for 1 h-long traces. The title of each graph indicates the average RTT, the average single time-out duration T_0, and the maximum window size W_m advertised by the receiver (in number of packets). To plot the graph, each 1 h trace was divided into 36 consecutive 100 s intervals, and each plotted point on a graph represents the number of packets sent versus the frequency of loss indications during a 100 s interval. While

TABLE I
Domains and Operating Systems of Hosts

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	cs.umass.edu	Linux 2.0.30
att	att.com	Linux

[2]The hostname for machine located in the `att.com` domain has been altered due to security concerns.

TABLE II
SUMMARY DATA FROM 1 h TRACES

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

dividing a continuous trace into fixed sized intervals can lead to some inaccuracies in measuring p, (e.g., the interval boundaries may occur within time-out 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 100 s, which are longer than most time-outs, we have minimized the impact of such inaccuracies. Each 100 s interval is classified into one of four categories: intervals of type TD did not suffer any time-out (only triple duplicate ACK's), intervals of type "$T0$" suffered at least one single time-out but no exponential backoff, "$T1$" represents intervals that suffered a single exponential backoff at least once (i.e., a double time-out), etc. The line labeled "TD only" (stands for triple-duplicate ACK's only) plots the predictions made by the model described in [9], which is essentially the same model as described in [8], while accounting for delayed ACK's. The line labeled "proposed (full)" represents the model described by (32). It has been pointed out in [8] 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 send rate that are much higher than measured. Also, in several traces [see, for example, Fig. 7(a)] we observe that TCP send rate 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 send rate at low p values.

The graphs in Fig. 8 compare the measured send rate with the predictions of the proposed model and the model in [9]. The title of each graph indicates the sender-receiver pair between which the measurements were carried out. As described earlier, each experiment consisted of 100 traces, each of which was 100 s in duration. For each trace, we measure the send rate, the loss rate, the round-trip time and T_0. We plot three points for each trace: one representing the measured send rate, a second representing the send rate predicted by the proposed model and the third representing the TD-only model in [9]. The points in each category are joined *only for better visual representation*. The x axis indicates the trace number and the y axis indicates the send rate, measured in terms of number of packets sent by the sender.

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

- *Hour-long traces*: We divide each trace into 100 s 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 RTT and time-out for the entire trace (these values are available in Table II). Then, for each 100 s interval we calculate the number of packets predicted by our proposed model, $N_{\text{predicted}} = B(p_{\text{observed}}) * 100$ s, where B is from (32). The average error is given by:

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

  The average error of our approximate model [using B from (33)] and of "TD only" are calculated in a similar manner. A smaller average error indicates better model accuracy. In Fig. 9 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*.

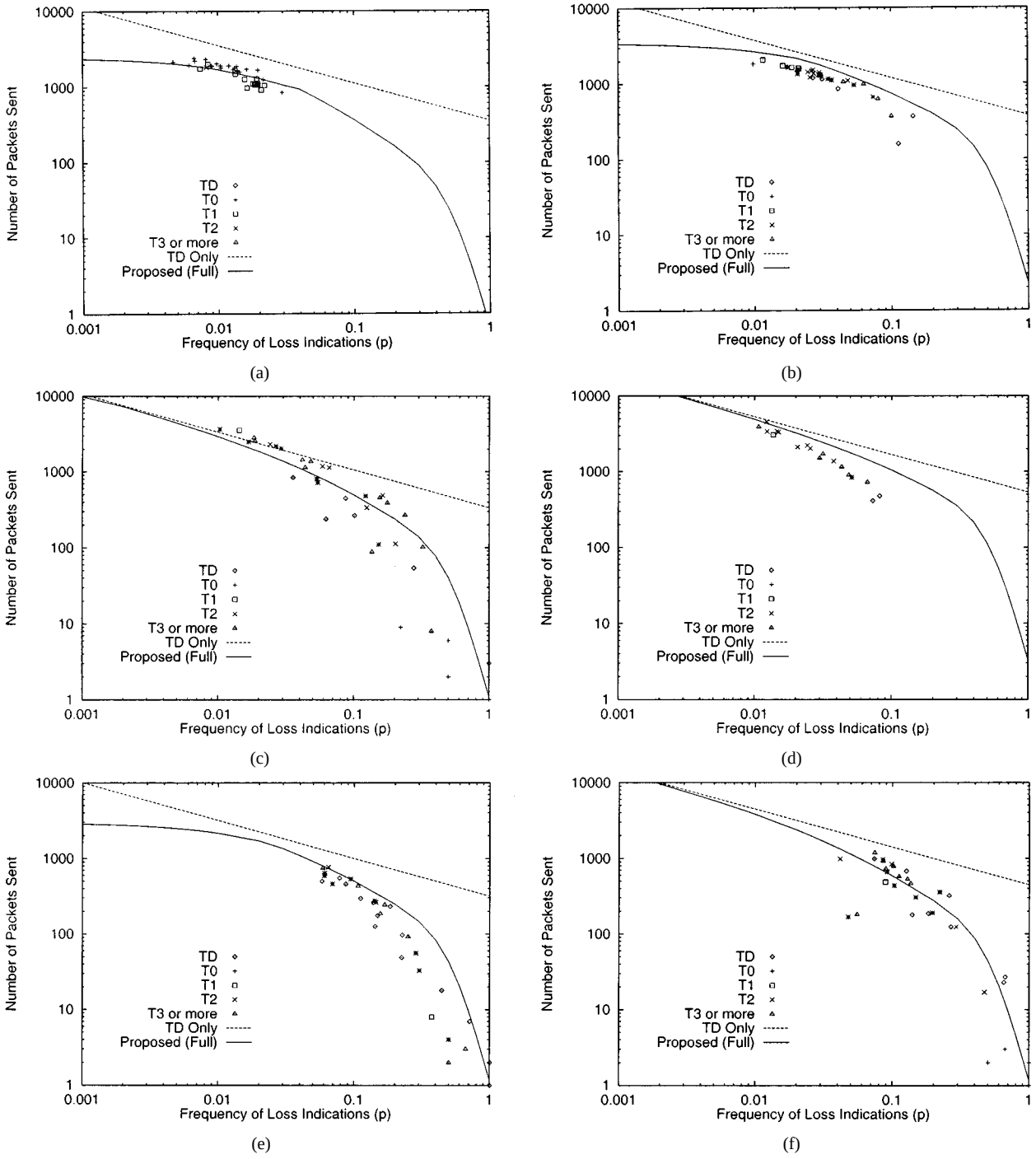

Fig. 7. One-hour traces. (a) Manic to baskerville, RTT $= 0.243$, $T_0 = 2.495$, $W_m = 6$, 1×1 h. (b) Pif to imagine, RTT $= 0.229$, $T_0 = 0.700$, $W_m = 8$, 1×1 h. (c) Pif to manic, RTT $= 0.257$, $T_0 = 1.454$, $W_m = 33$, 1×1 h. (d) Void to alps, RTT $= 0.162$, $T_0 = 0.489$, $W_m = 48$, 1×1 h. (e) Void to tove, RTT $= 0.272$, $T_0 = 1.356$, $W_m = 8$, 1×1 h. (f) Babel to alps, RTT $= 0.194$, $T_0 = 1.359$, $W_m = 48$, 1×1 h.

- *100 s traces*: We use the value of round-trip time and time-out calculated for each 100 s trace. The error values are shown in Fig. 10.

It can be seen from Figs. 9 and 10 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. Independent empirical and simulation studies of the model proposed in this paper have also been presented in [1], [2], [5], and [17]. These studies have found that the model provides a good fit to the observed send rate of TCP connections under a wide variety of network conditions.

IV. Discussion of the Model and the Experimental Results

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

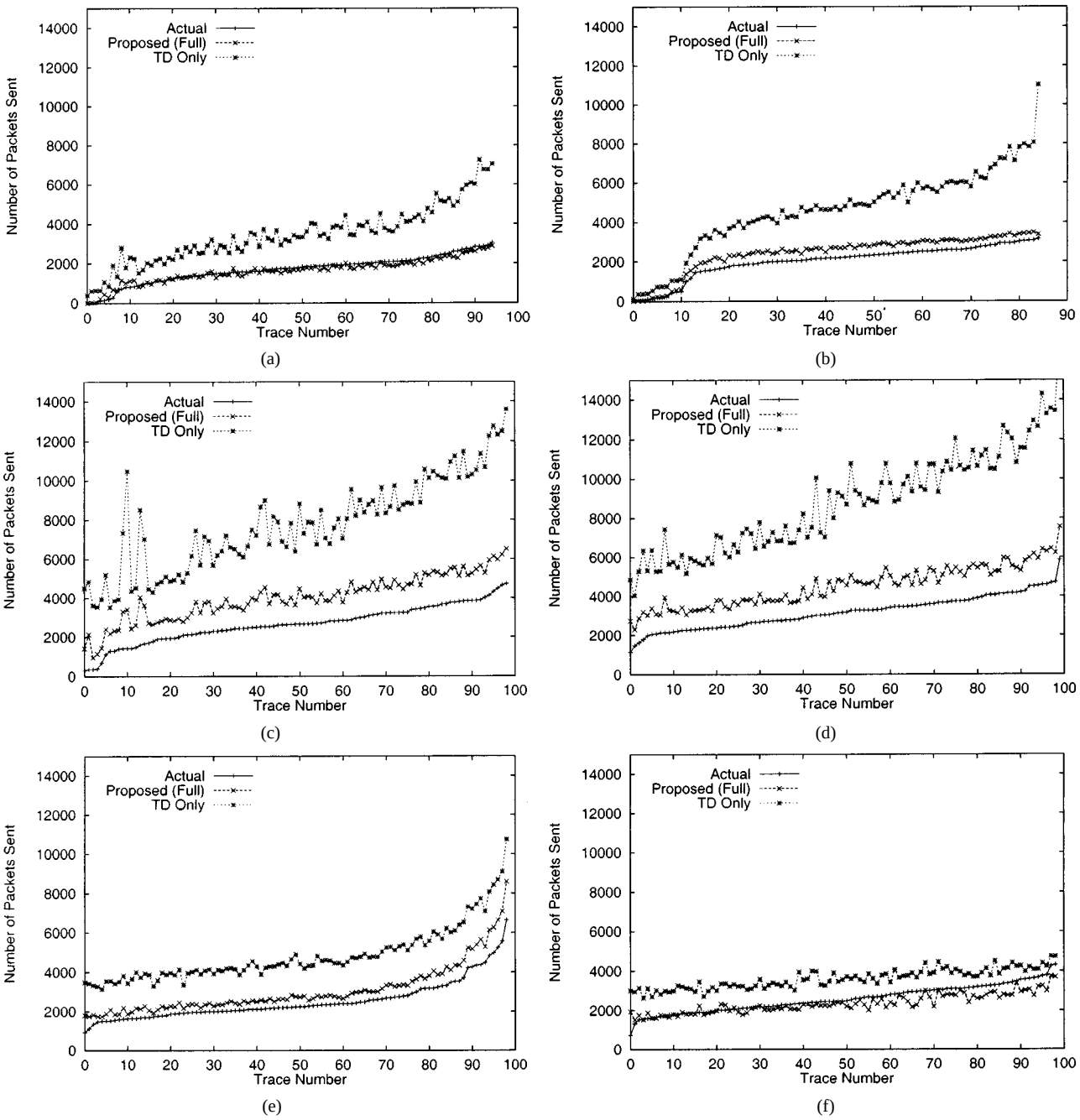

Fig. 8. The 100-s traces. (a) Manic to ganef. (b) Manic to mafalda. (c) Manic to tove. (d) Manic to maria. (e) Att to sutton. (f) Manic to afer.

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 III validate this assumption indirectly. We have also assumed that the time spent in *slow start* is negligible compared to the length of our traces. These assumptions have also been made in [8], [9], and [12].

We have assumed that packet losses within a round are *correlated* and losses in one round are *independent* of losses in other rounds. Recent studies [23] have shown the packet loss process observed on the Internet is bursty. The models provided, however, are too complicated to allow derivation of closed-formed results. Thus a simple loss model was assumed. In our simulation studies the model was able to predict the throughput of TCP connections quite well, even with Bernoulli losses. Investigation of performance of TCP under various packet loss models is an area for future work.

Another assumption we made, that is also implicit in [8], [9], and [12], 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 and the number of packets in transit during each sample. For most traces summarized in Table II, the coefficient of correlation is in the range $[-0.1, 0.1]$, 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

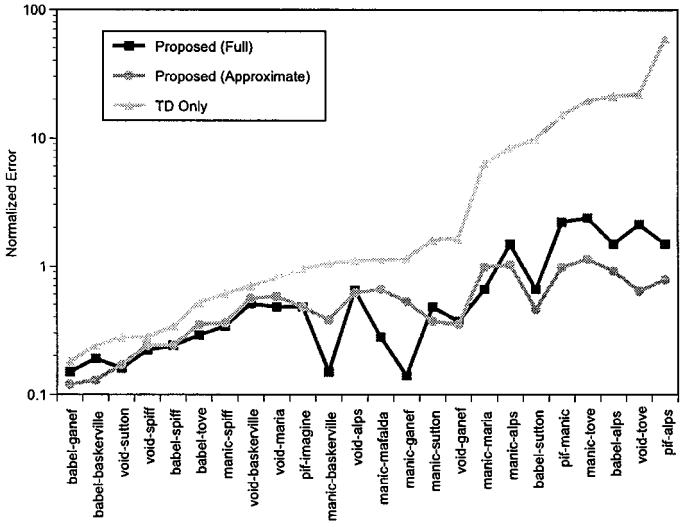

Fig. 9. Comparison of the models for 1 h traces.

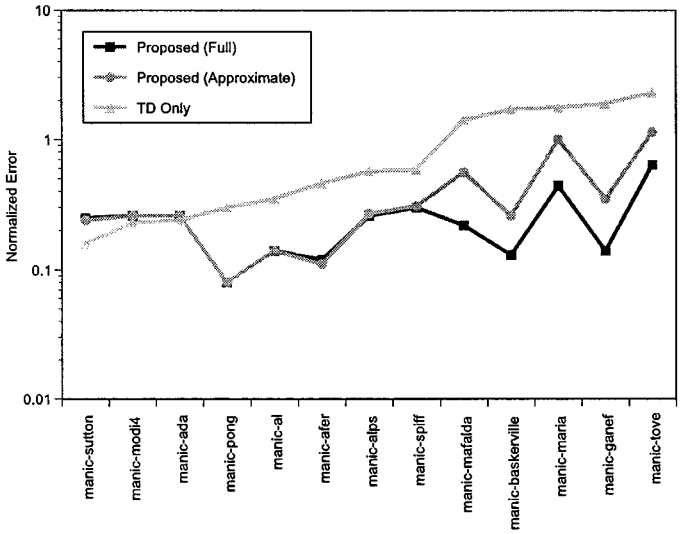

Fig. 10. Comparison of the models for 100 s traces.

connection (probably at the ISP, just before the modem). As a result, our model, as well as the models described in [8], [9], and [12] fail to match the observed data in the case of a receiver at the end of a modem. In Fig. 11, 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 28.8-kbyte/s modem. The results are for a 1-h connection divided into 100-s intervals.

We have also assumed that all of our senders implement TCP Reno as described in [6], [18], and [19]. In [14] and [15], 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 ACK's), 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 [6], [18], and [19]. Our observations also seem to indicate that in the Irix implementation, the exponential backoff

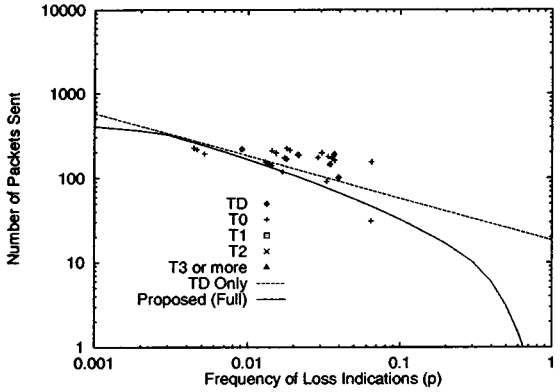

Fig. 11. Manic to $p5$, RTT $= 4.726$, $T_0 = 18.407$, $W_m = 22$, 1×1 h.

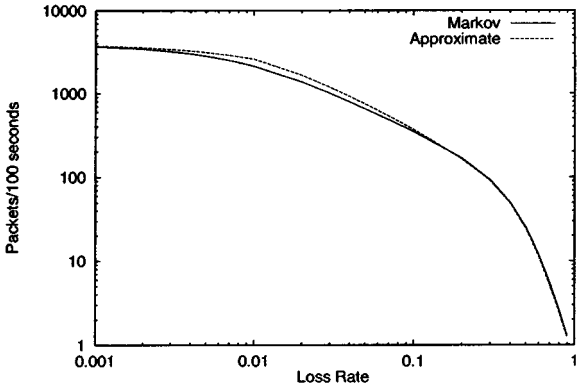

Fig. 12. Comparison with the Markov model: RTT $= 0.47$ s, $T_0 = 3.2$ s, $W_m = 12$.

is limited to 2^5, instead of 2^6. We are also aware of the observation made in [15] that the SunOS TCP implementation is derived from Tahoe and not Reno. We have not customized our model for these cases.

During the course of the analysis presented in Section II, we made several simplifying assumptions to obtain a closed-form solution. We have carried out a more detailed stochastic analysis, leading to a Markov model of TCP Reno [13]. This Markov model does not appear to have a simple closed-form solution. However, when solved numerically, the predictions of the Markov model closely match the predictions of the model proposed in this paper. In Fig. 12, we compare the Markov model with the model presented in this paper. The closeness of the match between the two models is evident.

V. Throughput of a Bulk Transfer TCP Flow

In the previous sections, we have focused our attention on investigating the send rate of a bulk transfer TCP flow. The steady-state performance of such a flow may also be characterized by *throughput*, which is the amount of data received by the receiver in unit time. The formula derived in Section II calculates the send rate. The same analysis can be easily modified to calculate throughput. Consider (21). It should be clear that to calculate throughput, instead of send rate, we only need to modify the numerator. We need to calculate the number of packets that make it to the receiver in a TDP, (counterpart of $E[Y]$) and in the time-out sequence (counterpart of $E[R]$). Let

us define these to be $E[Y']$ and $E[R']$, respectively. We can then calculate the throughput, denoted by $T(p)$, as

$$T(p) = \frac{E[Y'] + Q * E[R']}{E[A] + Q * E[Z^{\mathrm{TO}}]}. \tag{34}$$

Since only one packet makes it to the receiver in a time-out sequence (i.e., the packet that ends the time-out sequence), it is evident that

$$E[R'] = 1. \tag{35}$$

To calculate the number of packets that reach the receiver in a TDP, consider Fig. 2. The TD event is induced by the loss of packet α. Let the window size be W, when the loss occurs. Then, the number of packets received by the receiver is

$$E[Y'] = E[\alpha] + E[W] - E[\beta] - 1. \tag{36}$$

In Section II, we have shown that: $E[\alpha] = 1/p$ and $E[\beta] = E[W]/2$. From (35) and (36), along with the analysis for $E[W]$ and Q from Section II, we get

$$T(p) = \begin{cases} \dfrac{\dfrac{1-p}{p} + \dfrac{W(p)}{2} + Q(p, W(p))}{\mathrm{RTT}(W(p)+1) + \dfrac{Q(p, W(p))G(p)T_0}{1-p}}, \\ \qquad W(p) < W_m \\ \dfrac{\dfrac{1-p}{p} + \dfrac{W_m}{2} + Q(p, W_m)}{\mathrm{RTT}\left(\dfrac{W_m}{4} + \dfrac{1-p}{pW_m} + 2\right) + \dfrac{Q(p, W_m)G(p)T_0}{1-p}}, \\ \qquad \text{otherwise} \end{cases} \tag{37}$$

where $W(p)$, $Q(p, w)$, and $G(p)$ are defined as shown in (38), shown at the bottom of the page. In Fig. 13, we plot the send rate and throughput of a bulk transfer TCP flow with the following parameters: $W_m = 12$, RTT $= 470$ ms, and $T_0 = 3.2$ s.

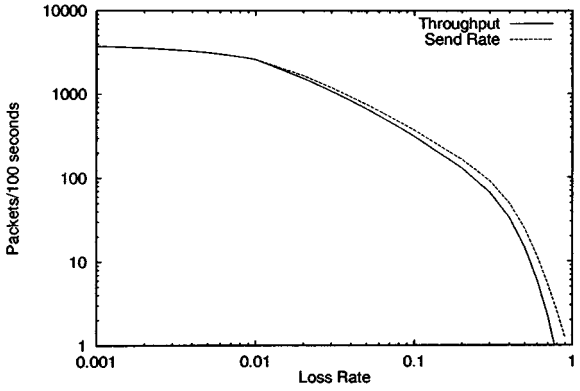

Fig. 13. Comparison of throughput and send rate.

VI. 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 send rate as a function of loss rate. The model takes into account the behavior of the protocol in the presence of time-outs, 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 time-outs. We found that our model provides a very good match to the observed behavior in most cases, while models proposed in [8], [9], and [12] significantly overestimate send rate. Thus, we conclude that time-outs 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 derived a simple expression for calculating the *throughput* of a bulk transfer TCP flow.

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, 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. Third, 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.

Acknowledgment

The authors wish to thank P. Krishnan of Hofstra University, Hempstead, NY, Z.-L. Zhang of the University of Min-

$$\begin{aligned} W(p) &= \frac{2}{3} + \sqrt{\frac{4(1-p)}{3p} + \frac{4}{9}} \\ Q(p, w) &= \min\left(1, \frac{\left(1-(1-p)^3\right)\left(1+(1-p)^3\left(1-(1-p)^{w-3}\right)\right)}{1-(1-p)^w}\right) \\ G(p) &= 1 + p + 2p^2 + 4p^3 + 8p^4 + 16p^5 + 32p^6 \end{aligned} \tag{38}$$

nesota, Minneapolis, A. Stathopoulos of the College of William and Mary, Williamsburg, VA, P. Wan of the Georgia Institute of Technology, Atlanta, L. Peterson of the University of Arizona, Tucson, L. Zhang of the University of California, Los Angeles, J. Bolot and P. Nain of INRIA, Sophia Antipolis, France, C. Cranor of Washington University, St. Louis, MO, G. Gheorhiu of the University of Southern California, Los Angeles, S. Pink of the Swedish Institute of Science, H. Schulzerinne of Columbia University, New York, NY, S. Tripathi of the University of Maryland, College Park, and S. Kasera of the University of Massachusetts, Amherst, for providing them with computer accounts that allowed them to gather the data presented in this paper. They would also like to thank D. Rubenstein and the anonymous referees for their helpful comments on earlier drafts of this paper.

REFERENCES

[1] J. Bolliger, T. Gross, and U. Hengartner, "Bandwidth modeling for network-aware applications," in *Proc. INFOCOMM'99*, 1999.
[2] N. Cardwell, "Modeling the performance of short TCP connections," unpublished.
[3] K. Fall and S. Floyd, "Simulation-based comparisons of Tahoe, Reno, and SACK TCP," *Comput. Commun. Rev.*, vol. 26, no. 3, July 1996.
[4] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Networking*, vol. 1, no. 4, Aug. 1997.
[5] M. Handley and S. Floyd, "TCP-friendly simulations," unpublished.
[6] V. Jacobson, "Modified TCP congestion avoidance algorithm," note sent to end2end-interest mailing list, 1990.
[7] T. Lakshman and U. Madhow, "The performance of TCP/IP for networks with high bandwidth-delay products," *IEEE/ACM Trans. Networking*, vol. 5, pp. 336–350, June 1997.
[8] J. Mahdavi and S. Floyd, "TCP-friendly unicast rate-based flow control," note sent to end2end-interest mailing list, Jan. 1997.
[9] M. Mathis, J. Semke, J. Mahdavi, and T. Ott, "The macroscopic behavior of the TCP congestion avoidance algorithm," *Comput. Commun. Rev.*, vol. 27, no. 3, July 1997.
[10] S. McCanne and S. Floyd. (1997) ns-LBL network simulator. [Online]. Available: http://www-nrg.ee.lbnl.gov/ns/
[11] S. McCanne and S. Ostermann. (1996) tcptrace: tcpdump file analysis tool. http://jarok.cs.ohiou.edu/software/tcptrace/. [Online]
[12] T. Ott, J. Kemperman, and M. Mathis. The stationary behavior of ideal TCP congestion avoidance. [Online]. Available: ftp://ftp.bellcore.com/pub/tjo/TCPwindow.ps
[13] J. Padhye, V. Firoiu, and D. Towsley, "A stochastic model of TCP Reno congestion avoidance and control," Tech. Rep. UMASS-CS-TR-1999-02.
[14] V. Paxson, "Automated packet trace analysis of TCP implementations," in *Proc. SIGCOMM'97*, 1997.
[15] ——, "End-to-End Internet packet dynamics," in *Proc. SIGCOMM'97*, 1997.
[16] V. Paxson and S. Floyd, "Why we don't know how to simulate the Internet," in *Proc. 1997 Winter Simulation Conf.*, 1997.
[17] L. Qiu, Y. Zhang, and S. Keshav, "On individual and aggregate TCP performance," Cornell CS Tech. Rep. TR99-1744, May 1999.
[18] W. Stevens, "TCP slow start, congestion avoidance, fast retransmit, and fast recovery algorithms," RFC2001, Jan. 1997.
[19] ——, *TCP/IP Illustrated, Vol. 1 The Protocols*, 10th ed. Reading, MA: Addison-Wesley, 1997.
[20] K. Thompson, G. Miller, and M. Wilder, "Wide-area Internet traffic patterns and characteristics," *IEEE Network*, vol. 11, Nov./Dec. 1997.
[21] T. Turletti, S. Parisis, and J. Bolot, "Experiments with a layered transmission scheme over the Internet," INRIA, Sophia Antipolis, France, Tech. Rep. RR-3296, Nov. 1997.
[22] L. Vicisano, L. Rizzo, and J. Crowcroft, "TCP-like congestion control for layered multicast data transfer," in *Proc. INFOCOMM'98*, 1998.
[23] M. Yajnik, S. Moon, J. Kurose, and D. Towsley, "Measurement and modeling of the temporal dependence in packet loss," in *Proc. INFOCOMM'99*, 1999.

Jitendra Padhye received the B.E. degree in computer engineering from the Victoria Jubilee Technical Institute, Mumbai, India, in 1992, and the M.S. degree in computer science from Vanderbilt University, Nashville, TN, in 1995. He is currently pursuing the Ph.D. degree in computer science at the University of Massachusetts, Amherst.

His research interests are in the area of Internet technologies and applications, specifically, Internet congestion control.

Victor Firoiu received the Dipl. Eng. degree in computer science from the Polytechnic Institute of Bucharest, Romania, in 1987, and the M.S. and Ph.D. degrees in computer science from the University of Massachusetts, Amherst, in 1995 and 1998, respectively.

He joined Nortel Networks in 1998, where he is now managing the QoS and Performance Engineering Group. He also cochairs the "Building Differentiated Services" IRTF Group.

Donald F. Towsley (M'78–SM'93–F'95) received the B.A. degree in physics and the Ph.D. degree in computer science from the University of Texas, Austin, in 1971 and 1975, respectively.

He was a member of the faculty of the Department of Electrical and Computer Engineering at the University of Massachusetts, Amherst, from 1976 to 1985. He is currently a Distinguished Professor in the Department of Computer Science, University of Massachusetts. He held visiting positions at IBM T. J. Watson Research Center, Yorktown Heights, NY, from 1982 to 1983, the Laboratoire MASI, Paris, France, from 1989 to 1990, INRIA, Sophia Antipolis, France, in 1996, and AT&T Labs–Research, Florham Park, NJ, in 1997. He currently serves on the editorial board of Performance Evaluation, and has previously served on several editorial boards including those of the IEEE TRANSACTIONS ON COMMUNICATIONS and IEEE/ACM TRANSACTIONS ON NETWORKING. He was Program Co-chair of the joint ACM SIGMETRICS and PERFORMANCE '92 conference. His research interests include networks, multimedia systems, and performance evaluation.

Dr. Towsley has received the 1998 IEEE Communications Society William Bennett Paper Award and two best conference paper awards from ACM SIGMETRICS in 1987 and 1996. He is a member of ORSA and the IFIP Working Groups 6.3 and 7.3 and is a fellow of the ACM.

James F. Kurose (S'81–M'84–SM'91–F'97) received the B.A. degree in physics from Wesleyan University, Middletown, CT, and the Ph.D. degree in computer science from Columbia University, New York, NY.

He is currently a Professor and Chair of the Department of Computer Science, University of Massachusetts, Amherst. He was a Visiting Scientist at IBM Research during the 1990–1991 academic year, and at INRIA and at EURECOM, both in Sophia Antipolis, France, during the 1997–1998 academic year. He has been active in theprogram committees for IEEE Infocom, ACM SIGCOMM, and ACM SIGMETRICS conferences for a number of years. His research interests include real-time and multimedia communication, network and operating system support for servers, and modeling and performance evaluation.

Dr. Kurose is a past Editor-in-Chief of the IEEE TRANSACTIONS ON COMMUNICATIONS and of the IEEE/ACM TRANSACTIONS ON NETWORKING. He has won a number of awards for on-campus and off-campus teaching. He is a member of ACM, Phi Beta Kappa, Eta Kappa Nu, and Sigma Xi.