A Stochastic Model for Short-lived TCP Flows

Dong Zheng
Department of Electrical Engineering
Arizona State University
Dong.Zheng@asu.edu

Georgios Y. Lazarou
TITL
Mississippi State University
glaz@ece.msstate.edu

Rose Hu
TITL
Mississippi State University
hu4@ece.msstate.edu

***Abstract*— In this paper, we propose a new model for the slow-start phase based on the discrete evolutions on the congestion window, and we use this slow-start model together with our improved TCP steady-state model to develop an extensive stochastic model which can more accurately predict the throughput and latency of short-lived TCP connections as functions of loss rate, round-trip time (RTT), and file size. The results from simulation experiments show that our model's performance predictions are up to 20% more accurate than the predictions obtained from the models proposed in [1] and [2].**

I. Introduction

Internet has become an important part of our daily life in the past ten years, in which the most commonly used applications such as world wide web, usenet news, file transfer and remote login have all opted for TCP as their transport medium. Hence, many stochastic models of TCP latency and throughput have been proposed, trying to capture its steady-state characteristics [3], [4].

All steady state models assume the availability of unlimited data to send. Hence, the impact of the transient phase on performance is insignificant, and is therefore ignored [4]. These models can only be used to predict the TCP send rate or throughput of bulk data transfers. Therefore, they are not applicable in predicting the performance of short-lived TCP flows.

It is noted in [5], [6], [7] that the majority of TCP traffic in the Internet consists of short-lived flows, i.e., the transmission comes to an end during the slow start phase before switching to the congestion-avoidance phase. Hence, new models are needed that are capable of predicting the performance of long-lived or short-lived TCP flows.

In this paper, we propose a better and tractable model for the congestion window growth pattern in the slow start phase. Using this new slow-start model and the extended steady-state model proposed in [8], we construct an accurate model for the short-lived TCP flows.

The remainder of the paper is organized as follows. Section II describes the assumptions. Section III presents a detailed analysis of our proposed model. Section IV describes the simulation and model validation experiments. Finally, Section V concludes the paper.

II. Assumption

The model is based on the TCP Reno release from Berkeley [9]. Since we are only concerned about modeling the performance of TCP, we assume that the link speed is very high and the sender sends full-sized segments whenever the congestion window ($cwnd$) allows. The advertised window is always a constant, i.e., the receiver fetches the data so fast that the buffer is always empty. Thus, the congestion window evolution, alone, determines the send rate of the TCP connection which could roughly be described by $cwnd/RTT$, where RTT is the round trip time.

We model the dynamics of TCP in terms of "rounds" as done in [4]. A round starts when a window of packets is sent by the sender and ends when one or more acknowledgments are received for these packets. Unlike the model proposed in [7], the delayed acknowledgment's effect is taken into consideration, but neither the Nagle algorithm nor the silly window syndrome avoidance is considered. In addition, we assume that packet losses are in accordance with the bursty loss model. The packet losses in different rounds are independent, but they are correlated within a single round, i.e., if one packet in a round is lost, then the following back to back packets in the same round are also assumed to be lost. It is an idealization of the packet loss dynamics observed in the paths where FIFO drop-tail queues are used [1].

III. Model Building

Our proposed model is composed of four parts according to a typical short-lived flow evolution: the start of the connection (three-way-handshake), the initial slow-start phase, the first loss part, and the subsequent losses. In the following subsections we derive the latency a connection experience in each part, and then we sum them to obtain the total latency.

A. The Connection start-up phase

Every TCP connection starts with the three-way-handshake process. Assuming that no ACK packets can get lost, this process can be well modeled as follows [1]:

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

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

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

B. The Initial Slow-Start Phase

After the three-way-handshake, the slow-start phase begins. In this phase, the sender's congestion window ($cwnd$) increases exponentially, until either of the following two events occurs: a packet gets lost or the $cwnd$ reaches its maximum value, W_m.

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

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

in which $cwnd_i$ is the congestion window size for the i^{th} round. Equation (2) is due to the fact that assuming no loss, in round $(i-1)$, there is a total of $cwnd_{i-1}$ packets sent to the destination, which, in turn, causes the receiver to generate $\lceil cwnd_{i-1}/2 \rceil$ acknowledgments[1]. According to the slow-start algorithm, upon receiving these ACKs, the sender increases the $cwnd$ by the number of ACKs it has obtained, which is $\lceil cwnd_{i-1}/2 \rceil$.

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

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

Rearranging, we get:

$$\begin{aligned} \lceil \frac{cwnd_{i-1}}{2} \rceil &= \lceil \frac{1}{2} \lceil \frac{3}{2} cwnd_{i-2} \rceil \rceil \\ &\approx cwnd_{i-2}. \end{aligned} \tag{4}$$

Substituting this in (2), we get the following:

$$cwnd_i \approx cwnd_{i-2} + cwnd_{i-1}. \tag{5}$$

In order to examine the accuracy of this approximation, let's see a typical evolution of $cwnd$:

$$1, 2, 3, 5, 8, 12, 18, 27...$$

Compared with the sequence generated by (5):

$$1, 2, 3, 5, 8, 13, 21, 34...$$

and the evolution of $cwnd$ proposed by the model in [1]:

$$1, 1.5^1, 1.5^2, 1.5^3, 1.5^4, 1.5^5, 1.5^6, 1.5^7...$$

or calculated as:

$$1, 1.5, 2.25, 3.38, 5.06, 7.59, 11.39, 17.09...$$

The similarity between the two previous sequences and the discrepancy between the real evolution of $cwnd$ with the proposed model in [1] show that Equation (5) gives a better approximation of the slow start phase.

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

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

Obviously (5) generates the Fibonacci sequence, and therefore we get the following close form expression for the cwnd:

$$cwnd_n = C_1 X_1^n + C_2 X_2^n, \text{ n = } 1, 2, 3... \tag{6}$$

where[3]:

$$X_{1,2} = \frac{1 \pm \sqrt{5}}{2}. \tag{7}$$

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

$$C_{1,2} = \frac{5 \pm \sqrt{5}}{10}. \tag{8}$$

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

$$\begin{aligned} Y_n^{ss} &= \sum_{i=1}^{n} cwnd_i \\ &= C_1 X_1^{n+2} + C_2 X_2^{n+2} - 2 \\ &\approx C_1 X_1^{n+2} - 2. \end{aligned} \tag{9}$$

The last approximation is due to the fact that:

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

Thus, from Equation (9), the number of rounds, n, can be computed as:

$$n = log_{X_1}\left(\frac{Y_n^{ss}+2}{C_1}\right) - 2. \tag{10}$$

Substituting (10) into (6), we get the approximate relationship between the congestion window size and the total count of packets that have been sent, as follows:

$$cwnd_n = \frac{Y_n^{ss}+2}{X_1^2}. \tag{11}$$

Taking the expectation of both sides of Equation (11), we have:

$$E[W^{ss}] = \frac{E[Y^{ss}]+2}{X_1^2} \tag{12}$$

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

In order to derive the latency for this phase, $E[Y^{ss}]$, the expected number of packets sent until a loss occurs is given by the following enhanced equation (based on the one given in [1]):

$$E[Y^{ss}] = \frac{(1-(1-p)^d)(1-p)}{p} \tag{13}$$

where d is the total file size measured in packets that must be transmitted.

Substituting the value of $E[Y^{ss}]$ in the Equation (12), we get the expected congestion window size at the end of the slow-start phase due to the packet losses [4]:

$$E[W^{ss}] = \frac{(1-(1-p)^d)(1-p)+2p}{pg^2} \tag{14}$$

[3] X_1 is also called the golden number

[4] We will use g instead of X_1 in following equations

If $E[W^{ss}]$ is bigger than the value of W_m, the maximum congestion window size, then the congestion window first grows to W_m and then remains there while sending the rest of the packets. Thus, the whole procedure can be divided into two parts [1]. From Equation (12), the number of packets sent when the $cwnd$ grows to W_m is given by:

$$data_1 = g^2 \cdot W_m - 2. \tag{15}$$

Substituting (15) into (10), we obtain the duration of this step measured in rounds as:

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

In the second part,

$$n_2 = \frac{E[Y^{ss}] - data_1}{W_m} \tag{16}$$

rounds are needed to transmit the remaining $E[Y^{ss}] - data_1$ packets.

Combining the previous results together and using Equation (10) for the $E[W^{ss}] \leq W_m$ case, the expected slow-start latency is calculated as follows:

$$E[n] = \begin{cases} [\lceil log_g(\frac{W_m}{C_1})\rceil + \frac{1}{W_m}(E[Y^{ss}] - g^2 W_m - 2)] \\ \qquad\qquad\qquad \text{when } E[W^{ss}] > W_m \\ \lceil log_g(\frac{E[Y^{ss}]+2}{C_1})\rceil - 2 \quad \text{when } E[W^{ss}] \leq W_m \end{cases} \tag{17}$$

C. The First Loss

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

Due to the exponential growing pattern of $cwnd$ in the slow-start phase, Q^{ss}, the probability that a loss is a TO is different from the probability that when the sender is in the congestion-avoidance phase. We derive the expression of Q^{ss} as follows (See Fig. 1).

In the round where a TD (triple-duplicate) occurs, let W^{ss} be the current size of $cwnd$, which has a value w. In this round, w packets were sent. Among them, k packets are assumed to be ACKed. Since the connection is still in the slow-start phase, $cwnd$ increases to $w+k$ and another $2k$ packets are sent in the next round[5]. If more than three packets from these $2k$ packets get ACKed, then a TD would occur; otherwise, a TO would take place. Let

$$A(w,k) = \frac{(1-p)^k p}{1-(1-p)^w}, \tag{18}$$

[5]The delayed acknowledgment concept is not applied here, but we prove later that it does not affect the analysis of the Q^{ss}.

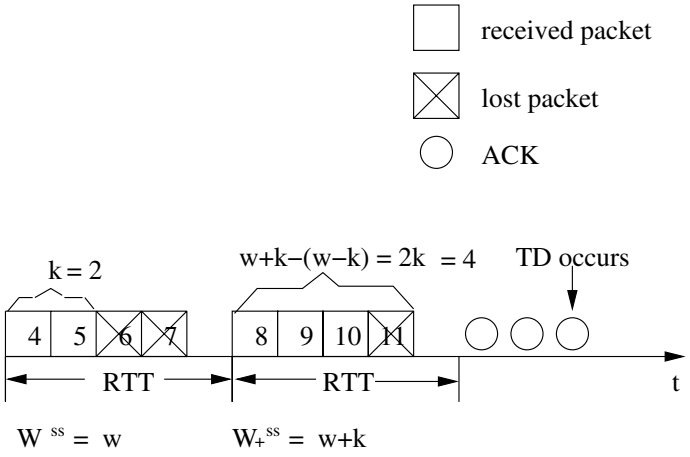

Fig. 1. A sample situation when TD happens.

be the probability that the first k packets have been successfully transmitted and ACKed in a round of w packets, provided that there might be one or more packets got lost. Now, let

$$h(m) = \sum_{i=0}^{2}(1-p)^i p \qquad \text{if } m \geq 3, \tag{19}$$

be the probability that no more than 2 packets have been transmitted successfully in a round of m packets. We then obtain Q^{ss} as:

$$Q^{ss}(W^{ss}) = \begin{cases} 1, & W^{ss} \leq 2 \\ \sum_{k=0}^{1} A(W^{ss},k) & \\ +\sum_{k=2}^{W^{ss}-1} A(W^{ss},k)h(2k), & \text{otherwise.} \end{cases} \tag{20}$$

Rewriting this in a simpler form, we have:

$$min\left(1, \frac{p(2-p) + (1-(1-p)^3)(1-p)^2(1-(1-p)^{W^{ss}-2})}{1-(1-p)^{W^{ss}}}\right). \tag{21}$$

As p approaches zero, Equation (21) reduces to:

$$Q^{ss} = \lim_{p\to 0} E[Q^{ss}(W^{ss})] = min\left(1, \frac{2}{E[W^{ss}]}\right). \tag{22}$$

In case of delayed acknowledgment, k successfully received packets generate $\lfloor k/2 \rfloor$[6] ACKs, and thus the size of the $cwnd$ increases to $\lfloor k/2 \rfloor + w$ and $\lfloor k/2 \rfloor + k$ packets are sent. Therefore Q^{ss} can be computed as:

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

which is the same with Equation (21) since

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

The expected time that TCP spends in the RTOs is given in [4] as:

$$E[Z^{TO}] = T_0 \frac{f(p)}{1-p} \tag{23}$$

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

where T_0 is the average duration of the first time-out, and

$$f(p) = 1 + p + 2p^2 + 4p^3 + 8p^4 + 15p^5 + 32p^6. \quad (24)$$

The time that TCP spends in the fast retransmit phase, n_t, depends on where the loss would happen [2]:

$$n_t = \begin{cases} 2RTT, & \text{if the lost packet is in the last} \\ & \text{three packets of the window} \\ RTT, & \text{otherwise} \end{cases} \quad (25)$$

Thus, when the congestion window size W^{ss} is bigger than three, the expected time, $E[n_t]$ is calculated as:

$$\begin{aligned} E[n_t] &= \frac{1-(1-p)^{W^{ss}-3}}{1-(1-p)^{W^{ss}}} \times 2RTT \\ &\quad + \frac{(1-p)^{W^{ss}-3}(1-(1-p)^3)}{1-(1-p)^{W^{ss}}} \times RTT \\ &= RTT \cdot \frac{2-(1-p)^{W^{ss}-3}-(1-p)^{W^{ss}}}{1-(1-p)^{W^{ss}}}. \end{aligned} \quad (26)$$

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

$$T_{loss} = (1-(1-p)^d)(Q^{ss}E[Z^{TO}] + (1-Q^{ss})E[n_t]), \quad (27)$$

where W^{ss} is

$$W^{ss} = min\left(W_m, \frac{E[Y^{ss}]+2}{g^2}\right). \quad (28)$$

D. Sending the Rest of the Packets

After the first packet loss, the transmission latency of the rest ($d - E[Y_{init}]$) packets is obtained by using the extended steady-state model derived in [8] as follows:

$$\begin{aligned} T_{rest} &= \frac{d-E[Y^{ss}]}{H} \\ &= \frac{dp-(1-(1-p)^d)(1-p)}{p \cdot H}. \end{aligned} \quad (29)$$

H is the throughput of a steady-state flow [8]:

$$H = \begin{cases} \frac{\frac{E[W^{TD}]g^2}{2} - 2 + \frac{1}{Q^{TD}(E[W^{TD}])}(\frac{1-p}{p} + (E[W^{TD}]-1)(1-p)) + 1}{\left(log_g(\frac{E[W^{TD}]}{2C_1}) + \frac{1}{Q^{TD}(E[W^{TD}])}(\frac{bE[W^{TD}]}{2}+b+1)\right)RTT + \frac{f(p)T_0}{1-p}} \\ \qquad \text{when } E[W^{TD}] < W_m \\ \frac{\frac{W_m g^2}{2} - 2 + \frac{1}{Q^{TD}(W_m)}(\frac{1-p}{p} + (W_m-1)(1-p)) + 1}{log_g(\frac{W_m}{2C_1})RTT + \frac{1}{Q^{TD}(W_m)}((\frac{b}{8}W_m + \frac{1-p}{pW_m}+1)+1)RTT + \frac{f(p)T_0}{1-p}} \\ \qquad \text{when } E[W^{TD}] \geq W_m \end{cases} \quad (30)$$

where Q^{TD}, the probability that a loss detection is a time-out (TO), is given as [8]:

$$Q^{TD} \approx min\left(1, \frac{3\sqrt{3}}{E[W^{TD}]}\right), \quad (31)$$

and $E[W^{TD}]$, the expected congestion window size in the congestion avoidance phase, is [8]:

$$\begin{aligned} E[W^{TD}] &= -\frac{2(b-2p)}{3} \\ &\quad + \sqrt{\frac{4(bp+2(1-p^2))}{3bp} + \left(\frac{2b-4p}{3b}\right)^2} \end{aligned} \quad (32)$$

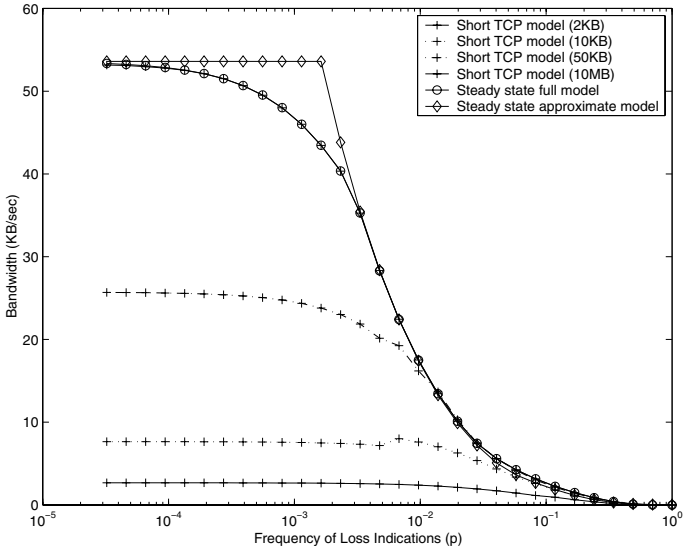

Fig. 2. Throughput predictions given by the short TCP model and the steady state model from [5]. The conditions are: $RTT = 200ms$, $MSS = 536bytes$, $w_1 = 1segment$, $T_0 = 1sec$, $W_m = 20segments$, $b = 2$.

E. Total Latency

Grouping (1), (17), (26) and (29) together and considering the delay (T_{delay}) caused by the delayed acknowledgment for the first packet (whose mean value is 100ms for the BSD-derived implementations), we now have the total expected latency:

$$\begin{aligned} T_{latency} &= E[T_{twhs}] + E[n]RTT + T_{loss} \\ &\quad + T_{rest} + T_{delay} - \frac{RTT}{2}. \end{aligned} \quad (33)$$

Note that the last term is due to the fact that only half of a round is needed to send the last window of packets. This short-lived TCP connection model is compared with our steady-state model in Fig. 2. It shows that as the transferred file size increases, the short TCP model approaches the steady state model. This is because when the connection has a large amount of data to send, TCP would spend most of its time in the steady-state. Also, Fig. 2 illustrates that as the loss rate increases, the throughput predicted by the short-lived TCP model approaches the one predicted by the steady-state model. This is because as the connection loses its packets at higher probability, the transient slow-start phase ends quickly and the remaining packets are sent in the steady-state phase.

IV. Model Validation through Simulation

We validated our proposed analytical model with simulation experiments. We performed all experiments in NS-2 [10] using the FullTCP agent. The FullTCP agent is modeled based on the 4.4BSD TCP implementation and can simulate all the important features of TCP Reno. The simulation topology used in all experiments is shown in Fig. 3.

Unlike in [1] where the Bernoulli loss model is used, in our experiments packets were getting lost according to the bursty

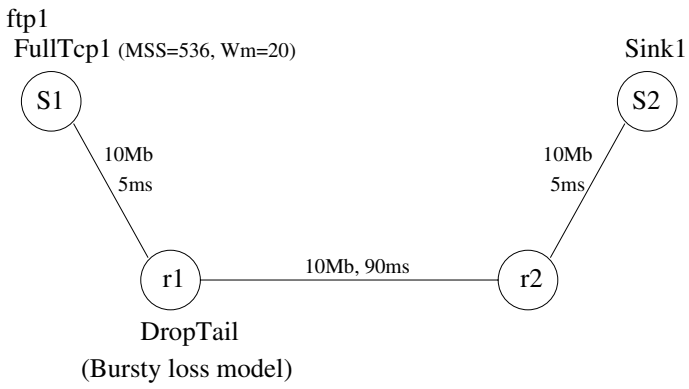

Fig. 3. The simulation topology

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

We used FTP [7] as the application for sending a controlled number of packets over a 10Mbps link. The experiments were designed such that the minimum RTT was 200ms.

A. Relation between the Transferred File Size and Latency

Fig. 4 shows the relationship between the latency and the transferred file size under no loss conditions. It compares the latency predictions given by our proposed model (Equation (33)) with those obtained by the models proposed in [1] and [2]. It is clear to see that our model's prediction values match the simulated values better than the values obtained by the other models. In order to numerically decide how well the proposed model approaches the simulated results, we computed the average error under different loss conditions. The formula for average error is defined as [4]:

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

in which $Th_{predicted}$ is the throughput predictions given by the models and $Th_{observed}$ is the throughput obtained from simulation. A smaller average error implies a better model accuracy. For this experiment, our model resulted in 5.83% average error, compared to 9.40% and 14.53% obtained by the models in [1] and [2] respectively.

Also, Fig. 4 indicates that all prediction errors fall in the range $[-RTT/2, RTT/2]$. We believe that this is due to the delayed acknowledgment mechanism. So if RTT is small, then the prediction errors become insignificant.

B. Relation between File Size, Loss Rate and Throughput

Figures 5, 6 and 7 compare the accuracy of our model with the one proposed in [1] in terms of throughput versus

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

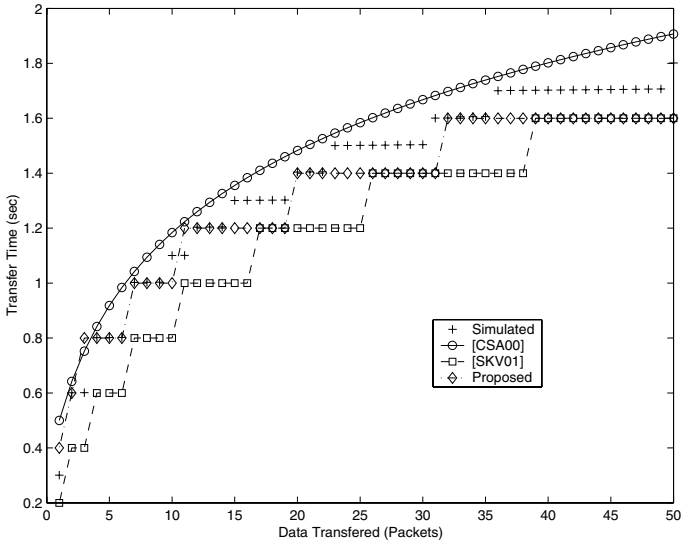

Fig. 4. Comparing the latency predicted by the short connection models for small transferred file size. The parameters are: $p = 0$, $RTT = 100ms$, $MSS = 536bytes$, $w_1 = 1segment$, $T_0 = 1sec$, $W_m = 20segments$.

TABLE I

COMPARISON OF THE PREDICTIONS AVERAGE ERROR.

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

transferred file size and loss rate. Table I compares the two models in terms of the average error.

As can be observed, when the transferred file size is small and the loss rate is low, our model gives more accurate predictions than the model from [1]. This is because we include the delay acknowledgment mechanism in our model. This is also due to the fact that we use g in our prediction expression rather γ which is used in [1]. However, when the file size is big and the loss rate is high, both of the models agree closely with our steady-state model, as expected.

V. CONCLUSION

In this paper, we developed an extensive stochastic model for predicting the performance of short-lived TCP flows in terms of the latency. We first constructed a new model for the slow-start phase based on the discrete evolutions of the congestion window. Then we integrated it with our improved steady-state model developed in [8]. We validated our model with simulation experiments. The results show that our model can predict the latency of short-lived TCP connections up to 20% more accurately than the models proposed in [1] and [2].

REFERENCES

[1] N. Cardwell, S. Savage, and T. Anderson, "Modeling TCP latency," in *Proc. IEEE INFOCOM '2000*, vol. 3, Tel Aviv, Israel, Mar. 2000, pp. 1742–1751.

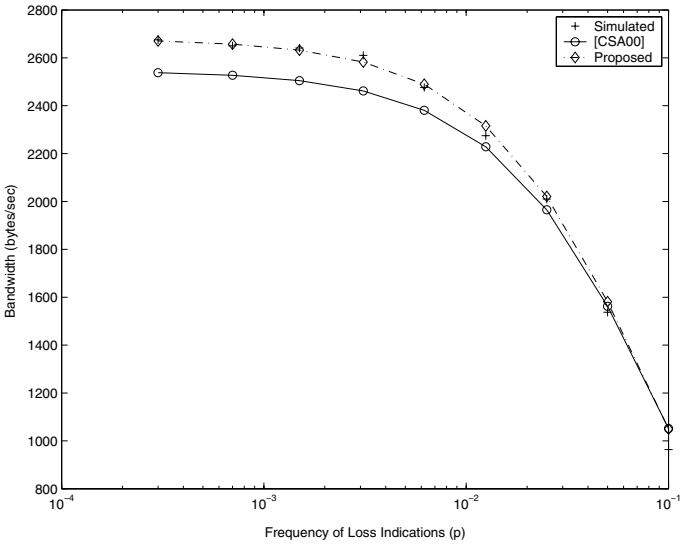

Fig. 5. Comparing the throughput predicted by the models for varying loss rate. The transferred file size is fixed at $2KB$. The parameters are: $RTT = 100ms$, $MSS = 536bytes$, $w_1 = 1segment$, $T_0 = 1sec$, $W_m = 20segments$.

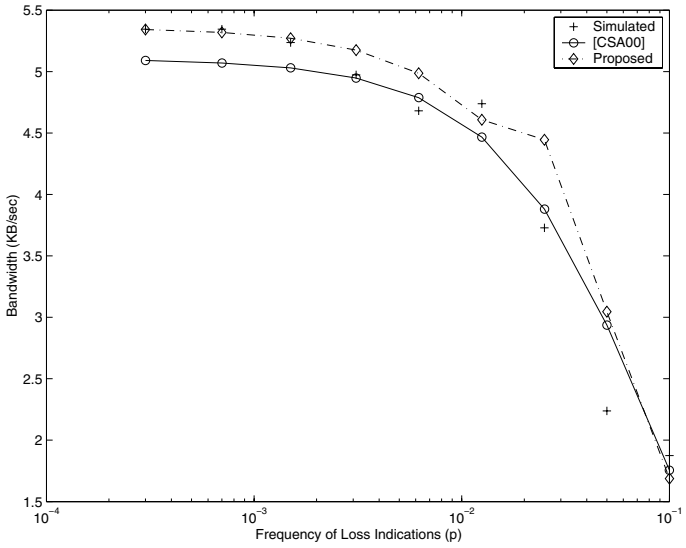

Fig. 6. Comparing the throughput predicted by the models for varying loss rate. The transferred file size is fixed at $6kB$. The parameters are: $RTT = 100ms$, $MSS = 536bytes$, $w_1 = 1segment$, $T_0 = 1sec$, $W_m = 20segments$.

[2] B. Sikdar, S. Kalyanaraman, and K. S. Vastola, "An integrated model for the latency and steady-state throughput of TCP connections," *Performance Evaluation*, vol. 46, no. 2-3, pp. 139–154, 2001.

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

[4] J. Padhye, V. Firoiu, D. F. Towsley, and J. F. Kurose, "Modeling TCP reno performance: A simple model and its empirical validation," *IEEE/ACM Trans. Networking*, vol. 8, no. 2, pp. 133–145, Apr. 2000.

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

[6] H. Balakrishnan, V. Padmanabhan, S. Seshan, R. H. Katz, and M. Stemm, "TCP behavior of a busy internet server. analysis and improvements," in *Proc. IEEE INFOCOM '98*, San Francisco, CA, Apr. 1998, pp. 252–262.

[7] M. Mellia, I. Stoica, and H. Zhang, "TCP model for short lived flows," *IEEE Commun. Lett.*, vol. 6, pp. 85–87, Feb. 2002.

[8] D. Zheng, "On the modeling of TCP latency and throughput," Master's thesis, Mississippi State University, Miss. State, 2002. [Online]. Available: http://titl.ece.msstate.edu/publications.html

[9] W. R. Stevens, *TCP/IP illustrated*. Reading, MA: Addison-Wesley, 1994, vol. 1.

[10] UCB/LBNL/VINT. (2002) The network simulator ns-2. [Online]. Available: http://www.isi.edu/nsnam/ns/

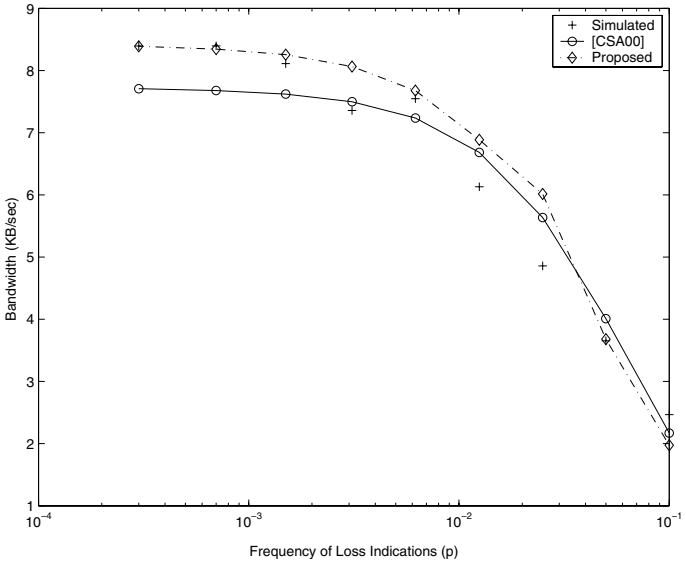

Fig. 7. Comparing the throughput predicted by the models for varying loss rate. The transferred file size is fixed at $11kB$. The parameters are: $RTT = 100ms$, $MSS = 536bytes$, $w_1 = 1segment$, $T_0 = 1sec$, $W_m = 20segments$.