Refined PFTK-Model of TCP Reno Throughput in the Presence of Correlated Losses

Roman Dunaytsev, Yevgeni Koucheryavy, Jarmo Harju

Institute of Communications Engineering, Tampere University of Technology
P.O. Box 553, FIN-33101, Tampere, Finland
{dunaytse, yk, harju}@cs.tut.fi

Abstract. This paper presents a simple and accurate analytical model of TCP Reno throughput as a function of loss rate, average round trip time and receiver window size based on PFTK-model. The presented model refines previous work by careful examination of fast retransmit/fast recovery dynamics in the presence of correlated losses and taking into consideration slow start phase after timeout. The accuracy of the proposed model is validated against simulation results and compared with those of PFTK-model. Simulation results show that our model gives a more accurate estimation of TCP Reno throughput in the presence of correlated losses than PFTK-model.

1 Introduction

Transmission Control Protocol (TCP) is the de facto standard protocol for the reliable data delivery in the Internet. Recent measurements show that from 60% to 90% of today's Internet traffic is carried by TCP [1]. Due to this fact, TCP performance modeling has received a lot of attention during the last decade [2].

One of the most known and wide referenced analytical models of TCP throughput of a bulk transfer is the model proposed by J. Padhye et al. in [3], also known as PFTK-model. This model describes steady-state throughput of a long-lived TCP Reno bulk transfer as a function of loss rate, average round trip time (RTT) and receiver window size. It assumes a correlated (bursty) loss model that is better suited for FIFO Drop Tail queues currently prevalent in the Internet.

Unfortunately, this model does not capture slow start phase after timeout and uses simplified representation of fast retransmit/fast recovery dynamics in the presence of correlated losses as having negligible effect on TCP Reno throughput. As it will be shown later, such simplifications can lead to overestimation of TCP Reno throughput. Since new analytical TCP models are often compared with PFTK-model (e.g., [4], [5], [6]) and use its resultant formula (e.g., [7], [8]), such inaccuracy in throughput estimation can lead to inaccurate results or incorrect conclusions.

In this paper, we propose a simple and more accurate steady-state TCP Reno throughput prediction model. This is achieved by careful examination of fast retransmit/fast recovery dynamics in the presence of correlated losses and taking into consideration slow start phase after timeout.

The reminder of the paper is organized as follows. Section 2 describes assumptions we made while constructing our model. Section 3 presents a detailed analysis of the proposed model. Section 4 describes model validation experiments, presents an analysis of the accuracy of our model and the one proposed in [3]. Finally, Section 5 concludes the paper.

2 Assumptions

The refined model we develop in this paper has exactly the same assumptions about endpoints and network as the model presented in [3]. We assume that the sender uses TCP Reno congestion control algorithm based on [9] and always has data to send. Since we are focusing on TCP performance, we do not consider sender or receiver delays and limitations due to scheduling or buffering. Therefore, we assume that the sender sends full-sized segments whenever the congestion window ($cwnd$) allows, while the receiver window ($rwnd$) is assumed to be always constant. We model TCP behavior in terms of "rounds" as done in [3], where a round starts when the sender begins the transmission of a window of segments and ends when the sender receives an acknowledgement (ACK) for one or more of these segments. It is assumed that the receiver uses delayed acknowledgement algorithm according to [10]. When modeling data transfer, we assume that segment loss happens only in the direction from the sender to the receiver. Moreover, we assume that a segment is lost in a round independently of any segments lost in other rounds, but at the same time segment losses are correlated within a round (i.e., if a segment is lost, all the remaining segments in that round are also lost). Such bursty loss model is a simplified representation of IP-datagram loss process in routers using FIFO Drop Tail queuing discipline. We assume that the time needed to send a window of segments is smaller than the duration of a round; it is also assumed that probability of segment loss and the duration of a round are independent of the window size. This can only be true for flows that are not fully utilizing the path bandwidth (i.e., in case of high level of statistical multiplexing).

3 The Model

According to [9], segment loss can be detected in one of two ways: either by the reception at the sender of "triple-duplicate" ACK or via retransmission timeout expiration. Similarly to [3], let us denote the first event as a TD (triple-duplicate) loss indication, and the second as a TO (timeout) loss indication. As in [3], we develop our model in several steps: when the loss indications are exclusively TD (Section 3.1); when the loss indications are both TD and TO (Section 3.2); and when the window size is limited by the receiver window (Section 3.3).

3.1 TD Loss Indications

In this section, we assume that all loss indications are exclusively TD and that the window size is not limited by the receiver window. In this case, according to [3], the long-term behavior of TCP Reno flow may be modeled as a cyclic process, where a cycle (denoted in [3] as a TD Period, TDP) is a period between two TD loss indications. For the i-th cycle $(i = 1, 2, ...)$ let Y_i be the number of segments sent during the cycle, A_i be the duration of the cycle and W_i be the window size at the end of the cycle. Considering $\{W_i\}_i$ to be a Markov regenerative process with renewal reward process $\{Y_i\}_i$, we can define the long-term steady-state TCP throughput B as

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

Fig. 1 shows the evolution of congestion window size during the i-th cycle according to [3].

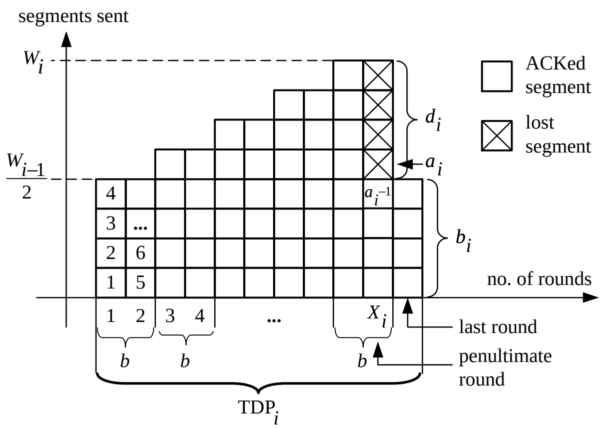


Fig. 1. Segments sent during the i-th cycle (TD Period) according to [3]

A cycle starts immediately after a TD loss indication, hence the current *cwnd* (expressed in segments) is set to $W_{i-1}/2$. The receiver sends one ACK for every b-th segment that it receives (according to [10], $b = 2$), so *cwnd* increases linearly with a slope of $1/b$ segments per round until the first segment loss occurs. Let us denote by a_i the first segment loss in the i-th cycle and by X_i the round where this loss occurs (see Fig. 1). According to the sliding window algorithm, after the segment a_i, $(W_i - 1)$ more segments are sent before a TD loss indication occurs and the current cycle ends.

Let us consider the evolution of congestion window size in the i-th cycle after the first TD loss indication. Taking into account the assumption about correlated losses within a round (i.e., if a segment is lost, so are all following segments till the end of the round), all segments following a_i in the round X_i (denoted in Fig. 1 as the penultimate round) are lost as well. Let us define d_i to be the number of segments lost in the round X_i and b_i to be the number of segments sent in the next (and the last) round $\left(X_i+1\right)$ of the i-th cycle (see Fig. 1). Similarly to [3], we assume that random variables b_i and d_i are uniformly distributed from zero to $\left(W_i-1\right)$ and from one to W_i correspondingly. Thus, taking into account that $b_i=W_i-d_i$ we have

$$E[b]=\frac{E[W]-1}{2}, \quad E[d]=\frac{E[W]+1}{2}. \tag{2}$$

After a TD loss indication the sender enters the fast retransmit/fast recovery phase and performs a retransmission of the lost segment. The slow start threshold ($ssthresh$) and the current value of $cwnd$ are updated according to [9] as

$$ssthresh=\max\left(FlightSize/2,\,2\right), \quad W'=ssthresh+N_{DupACK}, \tag{3}$$

where $FlightSize$ is the number of segments that has been sent, but not yet acknowledged; W' is the value of $cwnd$ during fast recovery phase; N_{DupACK} is the number of received duplicate ACKs.

Since $E\left[N_{DupACK}\right]=E[b]$, $E[FlightSize]=E[W]$ and using (2), we can determine $E[W']$ as

$$E[W']=E[ssthresh]+E\left[N_{DupACK}\right]=\frac{E[W]}{2}+\frac{E[W]-1}{2}=E[W]-\frac{1}{2}. \tag{4}$$

As $E[W']<E[FlightSize]$, it is expected that the sender will not send new segments in the fast recovery phase. After the successful retransmission of the segment a_i the sender will receive new ACK, indicating that the receiver is waiting for the segment $\left(a_i+1\right)$. As a result of receiving this new ACK, the phase of fast retransmit/fast recovery ends and according to [9] the new value of $cwnd$ is set as $W=ssthresh$, where $ssthresh$ is from (3). Since $FlightSize$ is still larger than the new value of $cwnd$, the sender cannot transmit new segments, therefore this ACK will be the single. As the sender will not be able to invoke the fast retransmit/fast recovery algorithms again, then it will wait for the expiration of retransmission timeout (RTO), which was set after the successful retransmission of the segment a_i (in accordance with [11], step 5.3). After the RTO expiration, the values of $cwnd$ and $ssthresh$ are set as $W=1$ and $ssthresh=\max\left(FlightSize/2,\,2\right)$, and the slow start phase begins.

Thus, in the presence of correlated losses and when the first loss is detected via a TD loss indication, the following sequence of steps is expected:

- initialization of the fast retransmit and fast recovery algorithms, retransmission of the first lost segment;
- awaiting for the RTO expiration, which was set after the successful retransmission of the first lost segment;
- initialization of the slow start algorithm.

Our observation is well agreed with the results from [12], showing that TCP Reno has performance problems when multiple segments are dropped from one window of segments and that these problems result from the need to wait for the RTO expiration before reinitiating data flow. Moreover, empirical measurements from [3] show that the significant part of loss indications (in average 71%) is due to timeouts, rather than TD loss indications.

In order to include the fast retransmit/fast recovery phase and the slow start phase, we define a cycle to be a period between two TO loss indications (besides periods between two consecutive timeouts). Therefore, a cycle consists of the slow start phase, congestion avoidance phase, fast retransmit/fast recovery phase and one timeout. An example of the evolution of congestion window size is shown in Fig. 2, where the congestion avoidance phase (TD Period in [3]) is supplemented with the slow start phase at the beginning and the fast retransmit/fast recovery phase with one timeout at the end.

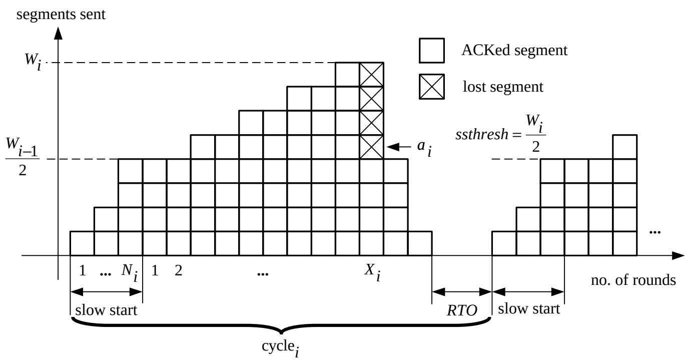


Fig. 2. Evolution of congestion window size during the i-th cycle, supplemented with the slow start phase at the beginning and the fast retransmit/fast recovery phase with one timeout at the end of the congestion avoidance phase

Observe that $Y_i = a_i + W_i$, thus we have

$$E[Y] = E[a] + E[W]. \tag{5}$$

The expected number of segments sent in a cycle up to and including the first lost segment is given in [3] as

$$E[a]=\sum_{k=1}^{\infty}(1-p)^{k-1}\cdot p\cdot k=\frac{1}{p}, \tag{6}$$

where p is the probability that a segment is lost, given that it is either the first segment in its round or the preceding segment in its round is not lost.

As in [3], let r_{ij} be the duration of the j-th round of i-th cycle ($i,\ j=1, 2, \ldots$). If we assume r_{ij} to be random variables independent of $cwnd$, then we have

$$E[A]=E[r]\cdot\left(E[N]+E[X]+2+\frac{E[RTO]}{E[r]}\right), \tag{7}$$

where $E[r]=\overline{RTT}$; $E[RTO]=\overline{RTO}$; $E[N]$ is the expected number of slow start rounds.

In order to derive $E[X]$ and $E[W]$, let us consider the evolution of $cwnd$ as a function of number of rounds. Similarly to [3], we assume that $W_{i-1}/2$ and X_i/b are integers. Therefore, we have

$$W_i=\frac{W_{i-1}}{2}+\frac{X_i}{b}-1, \quad i=1, 2, \ldots \tag{8}$$

Then the number of segments sent during the congestion avoidance (CA) phase of the i-th cycle can be defined as

$$Y_i^{CA}=\sum_{k=0}^{\frac{X_i}{b}-1}\left(\frac{W_{i-1}}{2}+k\right)\cdot b+b_i=\frac{X_i\cdot W_{i-1}}{2}+\frac{X_i}{2}\cdot\left(\frac{X_i}{b}-1\right)+b_i\ . \tag{9}$$

Combining (8) and (9), we obtain

$$Y_i^{CA}=\frac{X_i}{2}\cdot\left(W_i+\frac{W_{i-1}}{2}\right)+b_i\ . \tag{10}$$

Similarly to [3], we assume $\{X_i\}_i$ and $\{W_i\}_i$ to be mutually independent sequences of i.i.d. random variables. After the transformation of (8) we get

$$E[X]=b\cdot\left(\frac{E[W]}{2}+1\right). \tag{11}$$

Hence

$$E\left[Y^{CA}\right]=\frac{E[X]}{2}\cdot\frac{3\cdot E[W]}{2}+E[b]=\frac{3\cdot b\cdot E[W]}{4}\cdot\left(\frac{E[W]}{2}+1\right)+\frac{E[W]-1}{2}\ . \tag{12}$$

According to [7], the number of segments sent during the slow start (SS) phase can be closely approximated by a geometric series. At the same time it is known from [9], that the receiver sends an immediate duplicate ACK when out-of-order segment arrives (i.e., $b=1$). Then we can approximate

$$Y^{SS} = 1 + 2 + 2^2 + \ldots + 2^{N-1} = \sum_{k=1}^{N} 2^{k-1} = 2^N - 1 . \quad \textbf{(13)}$$

The required number of slow start rounds to send Y^{SS} segments can be expressed as

$$N = \log_2\left(Y^{SS}+1\right). \quad \textbf{(14)}$$

Taking into account, that in the slow start phase of the i-th cycle $cwnd$ growths exponentially from one to $ssthresh = W_{i-1}/2$, from (13) we have

$$\frac{E[W]}{2} = 2^{E[N]-1} . \quad \textbf{(15)}$$

Combining (13), (14) and (15), we obtain

$$E\left[Y^{SS}\right] = E[W]-1 . \quad \textbf{(16)}$$

By substituting (16) in (14) and taking into consideration (3), we get the expected number of slow start rounds as

$$E[N] = \max\left(\log_2 E[W], 2\right). \quad \textbf{(17)}$$

Based on (5), (12) and (16) and taking into account the retransmitted segment in the fast retransmit phase, the following system of equations can be defined as

$$\begin{cases} E[Y] = \dfrac{1}{p} + E[W] \\ E[Y] = E[W] - 1 + \dfrac{3\cdot b\cdot E[W]}{4}\cdot\left(\dfrac{E[W]}{2}+1\right) + \dfrac{E[W]-1}{2}+1 \end{cases} \quad \textbf{(18)}$$

Solving this system of equations for $E[W]$, we get

$$E[W] = -\left(\frac{2+3\cdot b}{3\cdot b}\right) + \sqrt{\frac{8+4\cdot p}{3\cdot b\cdot p} + \left(\frac{2+3\cdot b}{3\cdot b}\right)^2} . \quad \textbf{(19)}$$

In order to show that the slow start phase will enter in the congestion avoidance phase before the first segment loss occurs, we have to prove that $E\left[Y^{SS}\right] < E[a]$ (i.e., $E[W]-1 < \dfrac{1}{p}$). Solving this inequality, we get $\dfrac{3\cdot b}{p} + 9\cdot b\cdot p + 12\cdot b > 4$. The last inequality holds since $p > 0$ and $b \ge 1$.

By substituting (11) and (17) in (7), we have

$$E[A]=\overline{RTT}\cdot\left(\max\left(\log_2 E[W],\,2\right)+b\cdot\left(\frac{E[W]}{2}+1\right)+2+\frac{\overline{RTO}}{\overline{RTT}}\right). \tag{20}$$

Combining (1), (5), (6), (19) and (20), we obtain

$$B=\frac{\frac{1}{p}+E[W]}{\overline{RTT}\cdot\left(\max\left(\log_2 E[W],\,2\right)+b\cdot\left(\frac{E[W]}{2}+1\right)+2+\frac{\overline{RTO}}{\overline{RTT}}\right)}, \tag{21}$$

where $E[W]$ is given in (19).

3.2 TD and TO Loss Indications

A TO loss indication happens when segments (or ACKs) are lost and less than three duplicate ACKs are received. Note that in this case there will be no fast retransmit/fast recovery phase in a cycle. Similarly to [3], we define W_{ij} to be the window size at the end of the j-th cycle $(i,\ j=1,2,...)$, A_{ij} to be the duration of the j-th cycle, Z_i^{TO} to be the duration of a sequence of timeouts, Z_i^{TD} to be the duration of time interval between two consecutive timeout sequences, $S_i=Z_i^{TD}+Z_i^{TO}$. The number of transmitted segments during the last cycle and the duration of the last cycle can be approximated as $\left(E[Y]-1\right)$ and $\left(E[A]-\overline{RTT}\right)$ (where $E[Y]$ is from (5) and $E[A]$ is from (20)).

From [3] we can define long-term steady-state TCP throughput B as

$$B=\frac{E[n]\cdot E[Y]+E[R]-1}{E[n]\cdot E[A]+E\left[Z^{TO}\right]-\overline{RTT}}=\frac{E[Y]+Q\cdot\left(E[R]-1\right)}{E[A]+Q\cdot\left(E\left[Z^{TO}\right]-\overline{RTT}\right)}, \tag{22}$$

where $E[R]$ is the expected number of segments sent during timeout sequence; $E\left[Z^{TO}\right]$ is the expected duration of timeout sequence; $Q=1/E[n]$ is the probability that a loss indication ending a cycle is a TO.

The probability that a loss indication is a TO under the current congestion window size w is given in [3] as

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

which can be approximated for small values of p as

$$\widehat{Q}(w) \approx \min\left(1, \frac{3}{w}\right), \quad Q \approx \widehat{Q}\left(E[W]\right), \tag{24}$$

where $E[W]$ is given in (19).

According to [3], $E[R]$ can be defined as

$$E[R] = \frac{1}{1-p}. \tag{25}$$

Note that in contrast to [3], the duration of the first timeout from the sequence of consecutive timeouts is incorporated in the duration of a cycle. Therefore, the duration of the sequence of timeouts (excepting the first timeout) is

$$L_k = \begin{cases} \left(2^k - 2\right) \cdot RTO, & \text{when } k \in [2, 6], \\ \left(62 + 64 \cdot (k-6)\right) \cdot RTO, & \text{when } k \geq 7, \end{cases} \tag{26}$$

and the expectation of Z^{TO} is

$$\begin{aligned} E\left[Z^{TO}\right] &= \sum_{k=2}^{\infty} L_k \cdot p^{k-1} \cdot (1-p) = \\ &= \overline{RTO} \cdot \frac{2 \cdot p + 2 \cdot p^2 + 4 \cdot p^3 + 8 \cdot p^4 + 16 \cdot p^5 + 32 \cdot p^6}{1-p}. \end{aligned} \tag{27}$$

Combining (5), (20), (23) and (27), we obtain

$$B = \frac{\frac{1}{p} + E[W] + \widehat{Q}\left(E[W]\right)\frac{p}{1-p}}{\overline{RTT}\left(\max\left(\log_2 E[W], 2\right) + b\left(\frac{E[W]}{2} + 1\right) + 2 + \frac{\overline{RTO}}{\overline{RTT}}\right) + \widehat{Q}\left(E[W]\right)\left(\overline{RTO}\frac{f(p)}{1-p} - \overline{RTT}\right)}, \tag{28}$$

where

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

3.3 The Impact of Receiver Window Size

Let us denote by $W_{\max}$ the receiver window size and by $E[W_u]$ the unconstrained window size. Similarly to [3], we assume that $E[W_u] < W_{\max}$ leads to $E[W_u] \approx E[W]$ (where $E[W]$ is from (19)) and $E[W_u] \geq W_{\max}$ leads to $E[W] \approx W_{\max}$. Thus, using derivation from [3] and taking into account that

$$E\left[Y^{CA}\right]=\frac{3\cdot b\cdot\left(W_{\max}\right)^{2}}{8}-\frac{b\cdot W_{\max}}{4}+E\left[V\right]\cdot W_{\max}+\frac{W_{\max}-1}{2}, \tag{30}$$

we obtain the following system of equations

$$\begin{cases} E\left[Y\right]=\dfrac{1}{p}+W_{\max} \\ E\left[Y\right]=W_{\max}-1+\dfrac{3\cdot b\cdot\left(W_{\max}\right)^{2}}{8}-\dfrac{b\cdot W_{\max}}{4}+E\left[V\right]\cdot W_{\max}+\dfrac{W_{\max}-1}{2}+1 \end{cases} \tag{31}$$

Hence, the expected number of rounds when the window size remains constant is

$$E\left[V\right]=\frac{4+b\cdot p\cdot W_{\max}+2\cdot p}{4\cdot p\cdot W_{\max}}-\frac{1}{2}-\frac{3\cdot b\cdot W_{\max}}{8} \tag{32}$$

and

$$E\left[X\right]=\frac{b\cdot W_{\max}}{8}+\frac{4+b\cdot p\cdot W_{\max}+2\cdot p}{4\cdot p\cdot W_{\max}}-\frac{1}{2}. \tag{33}$$

Therefore

$$E\left[A\right]=\overline{RTT}\cdot\left(\max\left(\log_{2}W_{\max},2\right)+\frac{b\cdot W_{\max}}{8}+\frac{4+b\cdot p\cdot W_{\max}+2\cdot p}{4\cdot p\cdot W_{\max}}+\frac{3}{2}+\frac{\overline{RTO}}{\overline{RTT}}\right). \tag{34}$$

In conclusion, the complete expression of TCP throughput can be represented by the following expression

$$B=\begin{cases} \dfrac{\dfrac{1}{p}+E\left[W\right]+\hat{Q}\left(E\left[W\right]\right)\dfrac{p}{1-p}}{\overline{RTT}\left(\max\left(\log_{2}E\left[W\right],2\right)+b\left(\dfrac{E\left[W\right]}{2}+1\right)+2+\dfrac{\overline{RTO}}{\overline{RTT}}\right)+\hat{Q}\left(E\left[W\right]\right)\left(\overline{RTO}\dfrac{f\left(p\right)}{1-p}-\overline{RTT}\right)}, \\ \qquad\qquad \text{when } W_{\max}>E\left[W\right], \\ \dfrac{\dfrac{1}{p}+W_{\max}+\hat{Q}\left(W_{\max}\right)\dfrac{p}{1-p}}{\overline{RTT}\left(\max\left(\log_{2}W_{\max},2\right)+\dfrac{bW_{\max}}{8}+\dfrac{4+bpW_{\max}+2p}{4pW_{\max}}+\dfrac{3}{2}+\dfrac{\overline{RTO}}{\overline{RTT}}\right)+\hat{Q}\left(W_{\max}\right)\left(\overline{RTO}\dfrac{f\left(p\right)}{1-p}-\overline{RTT}\right)}, \\ \qquad\qquad \text{when } W_{\max}\le E\left[W\right]. \end{cases} \tag{35}$$

4 Model Validation through Simulation

In order to validate the proposed model and compare it with the one presented in [3], we compared the results obtained from the both analytical models against simulation results obtained from ns-2 [13]. We performed experiments using the well-known single bottleneck (“dumbbell”) network topology. In this topology all access links have a propagation delay of 1 ms and a bandwidth of 10 Mbps. The bottleneck link is configured as a Drop Tail link and has a propagation delay of 8 ms, bandwidth of 2 Mbps and a buffer size of 50 packets. To model TCP Reno connection we used Agent/TCP/Reno as a TCP Reno sender, Agent/TCPSink/DelAck as a TCP receiver with delayed acknowledgement algorithm and FTP as an application for transmitting infinite amount of data. We set TCP segment size to be 1460 bytes and maximum receiver window size ($W_{\max}$) to be 10 segments.

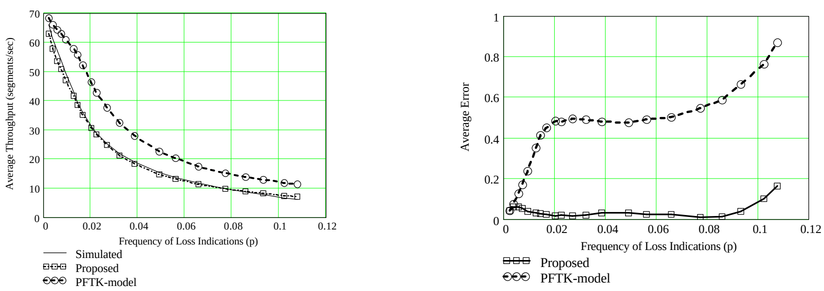
It has been noted in [14] that Web-traffic tends to be self-similar in nature and it was shown in [15] that superposition of many ON/OFF sources whose ON/OFF times are independently drawn from heavy-tailed distributions such as Pareto distribution can produce asymptotically self-similar traffic. Thus, we modeled the effects of competing Web-like traffic and high level of statistical multiplexing as a superposition of many ON/OFF UDP sources. Similarly to [16], in our experiments we set the shape parameter of Pareto distribution to be 1.2, the mean ON time to be 1 second and the mean OFF time to be 2 seconds. During ON times the UDP sources transmit with the rate of 12 kbps. The number of UDP sources was varied between 220 and 420 with the step of 10 sources. For each step we ran 100 simulation trials with a simulation time of 3600 seconds for each trial. In order to remove transient phase at the beginning of simulation, the collection of data was started after 60 seconds from the beginning of the simulation.

As in [3], in order to estimate the value of p, we used the ratio of the total number of loss indications to the total number of segments sent as an approximate value of p. Fig. 3a compares our model and the one presented in [3] against the simulation results. It easy to see, that the predicted values of throughput by the proposed model are much closer to the simulation results.

To quantify the accuracy of the both analytical models we computed the average error using the following expression from [3]:

$$\text{Average error} = \frac{\sum_{\text{observations}} \frac{\left|B(p)_{\text{predicted}} - B(p)_{\text{observed}}\right|}{B(p)_{\text{observed}}}}{\text{number of observations}}. \tag{36}$$

As shown in Fig. 3b, the proposed model has the average error smaller than 0.05 over a wide range of loss rates.



a) **b)**

Fig. 3. Average throughput (a) and average error (b) of the proposed and PFTK models

5 Conclusion

In this paper we developed an analytical model for predicting TCP Reno throughput in the presence of correlated losses. The model is based on the one proposed in [3] and improves it by taking into consideration a fast retransmit/fast recovery dynamics and slow start phase after timeout. The presented model has the average error smaller than 0.05 over a wide range of loss rates with the mean of 0.03, while the one, proposed in [3] performs well when the loss rate is quite small and significantly overestimates TCP Reno throughput in the middle-to-high loss rate range.

References

1. M. Fomenkov, K. Keys, D. Moore, and k. claffy. Longitudinal study of Internet traffic in 1998-2003. Technical Report, Cooperative Association for Internet Data Analysis (CAIDA), 2003.
2. J. Olsen. Stochastic modeling and simulation of the TCP protocol. PhD thesis, Uppsala University, Sweden, 2003.
3. J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modeling TCP Reno performance: a simple model and its empirical validation. IEEE/ACM Transactions on Networking, vol. 8, no. 2, pp. 133-145, April 2000.
4. S. Fortin and B. Sericola. A Markovian model for the stationary behavior of TCP. Technical Report RR-4240, IRISA-INRIA, France, September 2001.
5. B. Sikdar, S. Kalyanaraman, and K. Vastola. An integrated model for the latency and steady state throughput of TCP connections. Performance Evaluation, vol. 46, no. 2-3, pp. 139-154, October 2001.
6. O. Bogoiavlenskaia, M. Kojo, M. Mutka, and T. Alanko. Analytical Markovian model of TCP congestion avoidance algorithm performance. Technical Report C-2002-13, University of Helsinki, Finland, April 2002.
7. N. Cardwell, S. Savage, and T. Anderson. Modeling TCP latency. Proc. IEEE INFOCOM’00, vol. 3, Tel Aviv, pp. 1742-1751, March 2000.
8. S. Fu and M. Atiquzzman. Modeling TCP Reno with spurious timeouts in wireless mobile environment. Proc. 12-th International Conference on Computer Communications and Networks, Dallas, October 2003.
9. M. Allman, V. Paxson, and W. Stevens. TCP congestion control. IETF RFC 2581, April 1999.
10. R. Braden. Requirements for Internet hosts. IETF RFC 1122, October 1989.
11. V. Paxson and M. Allman. Computing TCP’s retransmission timer. IETF RFC 2988, November 2000.
12. K. Fall and S. Floyd. Simulation-based comparison of Tahoe, Reno and SACK TCP. ACM SIGCOMM Computer Communication Review, vol. 26, no. 3, pp. 5-21, July 1996.
13. UCB/LBNL/VINT. The network simulator - ns-2. http://www.isi.edu/nsnam/ns/
14. K. Park, G. Kim, and M. Crovella. On the relationship between file sizes, transport protocols and self-similar network traffic. Technical Report 1996-016, Boston University, USA, August 1996.
15. W. Willinger, M. Taqqu, R. Sherman, and D. Wilson. Self-similarity through high variability: statistical analysis of Ethernet LAN traffic at the source level. IEEE/ACM Transactions on Networking, vol. 5, no. 1, pp. 71-86, February 1997.
16. J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. A model-based TCP-friendly rate control protocol. Proc. NOSSDAV, Basking Ridge, pp. 137-151, June 1999.