Wireless Networks 9, 509–524, 2003
© 2003 Kluwer Academic Publishers. Manufactured in The Netherlands.

Stochastic Modeling of TCP in Networks with Abrupt Delay Variations

ALHUSSEIN A. ABOUZEID
Department of Electrical, Computer and Systems Engineering, Rensselaer Polytechnic Institute, 110 8th St., Troy, NY 12180, USA

SUMIT ROY
Department of Electrical Engineering, University of Washington, Box 352500, Seattle, WA 98195-2500, USA

Abstract. An analytical model of TCP (Transport Control Protocol) over an end-to-end path with random abrupt round-trip time (RTT) changes is presented. Modeling the RTT as a stochastic process, we analytically quantify and compare between the degree of degradation of the steady-state average throughput and window size due to spurious retransmissions for the different versions of TCP (Reno/NewReno versus Tahoe). The modeling methodology in this paper is used for evaluating different design alternatives for TCP for highly dynamic networks.

Keywords: transport control protocol, wireless networks, mobile ad-hoc networks, satellite networks, performance analysis

1. Introduction

TCP, the Internet Transport Control Protocol [15], is currently the most widely used transport protocol in packet networks with primary responsibility for congestion control. In the absence of any explicit information about the network configuration, TCP achieves its congestion control objective by attempting to drive the network to the point of full utilization (by increasing the rate at which it releases packets to the network) while continuously monitoring the network for signs of congestion and lowering the rate if congestion is detected. Packet loss is traditionally[1] used as a signal of congestion; however, this information to the TCP sender is only implicitly provided by the network, and hence TCP employs schemes to *infer* packet loss; either by the expiry of a retransmission timer leading to a Time-Out (TO) or the reception of multiple duplicate acknowledgments (ACKs) with the same expected packet number, termed Duplicate ACK detection (DA).

The original intent behind TCP's congestion control law is to assume that the *only* cause of any inferred packet loss is congestion. Thus situations where the packet loss inference is *noncongestion related* such as due to unreliable data links (as is characteristic for wireless channels), packet re-ordering or sudden large increase in propagation delays will needlessly trigger the congestion avoidance mechanisms in TCP and adversely affect the end-to-end throughput. Consequently there is a growing literature on modeling and performance evaluation of TCP performance where congestion loss is not dominant or, at least, not the only source of packet loss.

TCP performance is known to be relatively robust to *smooth round-trip time (RTT) variations* [2,3], attributable to the fact that TCP's RTT estimator successfully tracks such smooth changes; however, TCP performance is drastically affected by abrupt (step) changes in the RTT process which may result in incorrect inference of packet loss by a TCP sender. Paxson [14] shows that multi-path routing in the Internet can cause "route fluttering" (frequent route changes) which results in significant out-of-order packet delivery and/or frequent abrupt end-to-end RTT variations. Henderson et al. [8] and Allman et al. [2] show that packet reordering and abrupt delay variations might be caused by hand-offs of satellite links in Low Earth Orbit (LEO) as well as Geostationary Orbit (GEO) satellite networks.

Up to our knowledge, this paper presents the first analytical model of TCP operation over an end-to-end path experiencing randomly varying RTT. The RTT variations are modeled by a semi-Markov process; this allows analytical estimation of steady-state TCP throughput based on renewal-reward theory [16]. Our analytical model matched simulation results for a wide range of parameters with a maximum error of less than 20% (mean error of less than 10%). Further, the approach also provides some interesting new observations:

(i) OldTahoe outperformed Reno in a considerable number of cases, yielding an improvement in the throughput of up to 100% (setting the DA threshold to half the maximum path bandwidth-delay product improves Reno's performance to match that of OldTahoe);

(ii) Degradation of TCP performance compared to an ideal performance in environments with abrupt RTT changes becomes more evident at higher advertised window sizes.

LEO satellite networks constitute an example of an environment where the model in this paper is particularly ap-

[1] Proposals to use Explicit Congestion Notification (ECN) are expected to be standardized soon by the IETF. However, ECN will be used in addition to the current TCP packet loss detection schemes; thus, the results of our study also applies if ECN is used.

propriate. Previous simulation studies of Teledesic constellation [8] report that the end-to-end delay for a TCP session is characterized by periods of relatively slow variations punctuated by step increases. This behavior is attributed to the continuous motion of the satellites and the resulting changes in the number of hops traversed due to underlying route changes as a function of satellite topology. Higher delay variability is observed for satellite networks with smaller number of satellites (e.g., Iridium).

A second area of potential use is multi-hop radio networks with node mobility that introduce end-to-end RTT variability similar to LEO satellite systems. To motivate this, consider the following simple example: a long file transfer between a source (client) and destination (file server) pair where the server is fixed and the client is mobile. As the client moves away from the server, the connection requires multiple radio hops using intermediate nodes. Thus, the RTT delay between the source–destination pair varies as a function of the number of intervening hops. The RTT is thus relatively constant when the client is static at a specified intermediate node location, and changes abruptly (increases if client moves further away, and decreases if the client moves closer) upon client motion.

Note that this work does not attempt to separately model the characteristics of the data link layer and choice of routing protocols, all of which have potential impact on the performance of TCP in a multi-hop packet radio network setting. However, we suggest that the stochastic model presented here is sufficiently rich to capture the *collective* impact of all such factors affecting dynamic RTT variations. In particular, the choice of the model parameter values in section 6 is closely motivated by data obtained from simulations of the Iridium and Teledesic satellite network constellations [11,17] based on prior simulation studies of the multiple access and routing layers that incorporate satellite mobility.

This paper considers only loss-less handoffs/path changes as a precursor to future work that incorporates the effects of packet loss. In [1], we proposed *via simulations* a different model wherein in addition to the semi-Markov process describing the RTT, packet losses occur with probability p. Extension of our analytical results to the case of packet drops is deferred to future studies.

Broadly speaking, the analytical framework presented in this paper provides a context for the evaluation of window-based control protocols that may be *more adaptive* to abrupt variations of the RTT process. Recent works that have attempted to provide solutions to this problem include, notably, [12] that proposed use of time-stamps option to detect spurious packet retransmissions and then "undoing" unnecessary congestion control responses to such reordered or delayed packets. A heuristic approach was also proposed in [3]. D-SACK (an extension to the original SACK version of TCP) [6], provides a scheme for detecting duplicate (or spurious) packet reception at the receiver; this information is then sent to the TCP source thus allowing TCP sender to detect that it has erroneously triggered its congestion avoidance mechanisms. We emphasize that all these approaches can be used *in addition* to better tuning of TCP's retransmit timeout and fast-retransmit algorithms (e.g., see [5] for a summary), since their use alone does not completely obviate the occurrence of spurious retransmissions.

The paper is organized as follows. Section 2 summarizes the behavior of TCP and how RTT delay dynamics can cause incorrect inference of packet loss. Section 3 introduces the model of the RTT process to be used and its implications for TCP performance. Section 4 presents an analysis of TCP behavior for the case of a single abrupt RTT change. The results are then extended in section 5 to analyze two different versions of TCP; OldTahoe and Reno/NewReno in steady-state for the general M-state semi-Markov process presented in section 3. Equations allowing the numerical computation of the steady-state distribution of the congestion window size and throughput are derived using renewal-reward theory. Section 6 presents the model validation, a comparison between OldTahoe and Reno/NewReno and an analysis of the effect of the parameters of TCP's TO and DA algorithms on the end-to-end throughput. Section 7 concludes the paper outlining future extensions.

2. The impact of dynamic round-trip delay on TCP

We first briefly summarize TCP's normal window mechanism and then explain how (a) packet reordering and/or (b) abrupt RTT increase affect TCP's operation.

2.1. Normal operation

TCP Reno/NewReno detects congestion in one of two ways:

(1) the expiry of the round-trip timer, i.e., Time-out (TO), or

(2) the reception of multiple (typically three) duplicate ACKs (DA).

(1) With each packet transmission, the TCP sender sets a round-trip timer using an exponentially weighted moving average of the sample RTTs incremented by four times the standard deviation of the sample RTTs to minimize spurious retransmissions. If the timer expires before the reception of the ACKs, TCP assumes the network is congested and responds by setting the slow-start window threshold to half the window size at which the timer expired, resetting the congestion window to 1 and entering slow-start. During slow start, the window size is increased by one packet for every received ACK. This continues (assuming no further packet loss is detected) until the window reaches the slow-start threshold, after which TCP enters the congestion avoidance phase, and the window is increased by one packet for each window's worth of ACKs. The window size is always upper bounded by the maximum advertised window size.

(2) The *DA* mode of packet loss implicitly assumes that packet reordering in the network is infrequent; i.e., all packets are assumed to be received in the same order they were transmitted. Hence, when a TCP source receives three or more ACKs requesting the same next expected packet number, the source assumes that one of the packets in the sequence

of outstanding packets has been lost, indicating a mild level of congestion (compared to the TO case). It responds to a DA by attempting to recover this loss quickly by immediately retransmitting this packet (Fast Retransmit). The source also reduces its window size (typically by half) so as to avoid further congesting the network, but it continues in the congestion avoidance phase (i.e., does not trigger slow-start). TCP OldTahoe (the predecessor version of TCP Reno/NewReno) detects packet loss using TO's only.

2.2. Impact of abrupt RTT changes

Dynamic RTT delay may cause the following adverse effects on TCP:

(1) Spurious TO's;

(2) Spurious DA's;

(3) Loss of self-clocking.

(1) *Spurious TO* may occur if the round-trip delay suddenly increases and exceeds the current RTO value, causing the timer to expire. This causes TCP mechanism to (needlessly) retransmit packets that may have already been received at the destination. A more drastic effect (from an application layer perspective) is that the TCP source needlessly reduces its congestion window size and enters its slow start mechanism.

(2) *Spurious DA* may take place if the RTT variation on the forward path causes a reordering of packets that exceeds the DA threshold (default is 3).[2] The effect of spurious DA is less drastic than spurious TO since TCP does not enter the slow-start phase but it continues in the congestion avoidance phase. However, the window size is reduced (typically to half its value) and retransmission takes place.

(3) *Loss of self-clocking*: TCP relies on the arrival of ACKs for adjusting its rate of increase of the window size. Thus, while delay variation on the reverse path alone will not cause the receiver to generate duplicate ACKs, it may cause the ACKs to arrive out of order at the source. The effect of this is that TCP might lose its ACK clock, resulting in a highly bursty behavior and a slower rate of window increase [4].

It is worth noting that RTT decrease can only cause DAs (i.e., cannot cause TOs) while RTT increase can only cause TOs (and not TDs).

3. Modeling

In this section, we present a model for the RTT process as well as TCP's congestion control algorithm with supporting arguments and discussion of the modeling approximations involved.

[2] The OldTahoe version uses only TO to detect packet loss while the Reno and NewReno versions use the DA option.

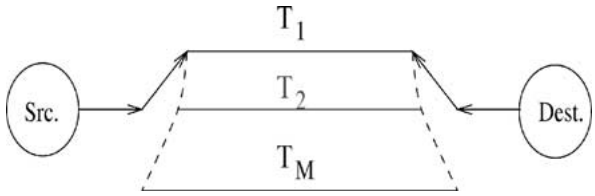

Figure 1. Abstract model for abrupt link changes.

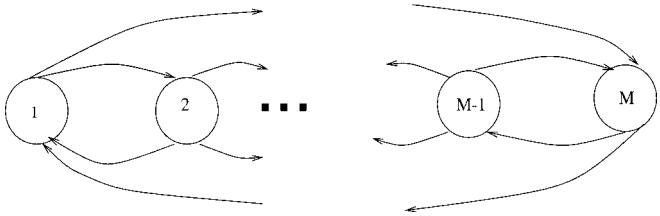

Figure 2. The embedded Markov chain of the RTT process.

3.1. A semi-Markov model for RTT

Let $T(t)$ denote the round-trip delay (RTT) between a source–destination pair at some time instant t. We assume that the set of possible values of $T(t)$ is finite of size M (see figure 1), and that $T(t)$ is a continuous-time stochastic process that transitions between these M RTT states labeled $1, 2, \ldots, M$. The state transitions are specified by an embedded Markov chain (see figure 2) with $M \times M$ transition probability matrix $\mathbf{H}$. More explicitly, assume that when the process is in state i, $1 \leqslant i \leqslant M$:

(i) The round-trip delay (in seconds) for a packet transmitted in this state is T_i.

(ii) The amount of time spent in state i, denoted X_i, prior to a transition into a different state has distribution F_i.

(iii) When the process leaves state i, it enters state $j \neq i$ with probability $\mathbf{H}_{i,j}$.

We will restrict the above model to the following two cases:

(iv) $X_i > T_i$ with probability 1.

(v) Causality: RTT changes only affect packets that are not yet transmitted. In other words, an ACK for a packet transmitted during state i will be received after an RTT T_i from its transmission.

Assumption (iv) implies that RTT step changes occur at a rate slower than one round-trip time, i.e., successive abrupt RTT changes are, with a high probability, separated by at least one RTT which is reasonable for many practical scenarios where hand-off/path change rate is normally slower than a packet RTT. Clearly, our model encompasses the quiescent scenario where the RTT process in state i varies *slowly* around T_i, in which case, T_i can be thought of as the mean RTT during state i. With (v), we attribute RTT changes as being due to forward propagation path only. While acknowledging that changes in the reverse path may result in loss of ACK clocking, we note that if these (reverse path) changes

occurred at a rate lower than one RTT, the ACK reordering effect becomes negligible (since TCP window mechanism would have enough ACKs within each round to maintain self-clocking [4]). Further, the assumption is motivated due to practical reasons; to relax this assumption, a model of the exact position (i.e., link or hop along a path composed of multiple hops) within a round-trip path that caused the step RTT change must be incorporated, which would limit the application of the model to specific networks.

A primary advantage of this set-up is its ability to capture both types of spurious packet loss inference – spurious DA's and spurious TO's. The analysis section will analyze these two cases in detail.

3.2. TCP modeling

We first summarize essential features of TCP window dynamics in the context of the proposed RTT model and the subsequent analysis.

3.2.1. Definitions

Definition 1 (Window round). A TCP window round (or, a round) corresponding to a window of size w is the *time duration* between successive window increments to w and $w + 1$. This duration is assumed equal to the current RTT.

Definition 2 (Busy period of a round). A busy period of a round with window size w is the *time duration measured from the beginning of the round* during which ACKs for previously transmitted packets arrive (hence TCP is busy transmitting packets) until all packets during this window round are transmitted. If μ is the packet transmission rate (packets/sec), then the duration of this period is equal to w/μ.

Definition 3 (Idle period of a round). In a window round, if the window size w is less than the link bandwidth delay product, then an idle period of duration $T(t) - w/\mu$ terminates the window round.

Definition 4 (Depth within a round). The time measured in packets from the beginning of the window round until an RTT state transition, if exists. Thus, for a window round during which the RTT is T_i and an abrupt RTT change takes place, the depth lies in the range $[1, \mu T_i]$.

3.2.2. Slow start

The window size during slow start increases by one for every received ACK that acknowledges the reception of new data until the window size reaches the slow start threshold value (which is set equal to half the window size at which the timer expired). This results in an exponential increase of TCP window at a rate equal to that of the arrival of ACKs.

3.2.3. Congestion avoidance

The rate of window size increase during congestion avoidance is assumed linear (rate of one packet every RTT) which is a good approximation to observed TCP behavior during congestion avoidance.

Thus TCP sender with a window size of w_1 starts with a burst of w_1 packet transmissions (of burst duration w_1/μ) and may terminate with an idle period if $w_1/\mu < T(t)$. The next round has a window size $w_1 + 1$. For steady state behavior (e.g., for long file transfer), the window size cannot grow infinitely, even in the absence of any packet loss detection events since it is bounded by TCP's maximum advertised window value, which is denoted by w_{m}, assumed constant throughout the lifetime of the session.

3.2.4. The retransmission timer

The TCP sender waits for a duration equal to the current retransmission timer value, denoted by RTO, before declaring a packet loss and subsequent retransmission of the packet. Too large an RTO results in loss of throughput, while too small an RTO may lead to needless retransmissions. The setting of the retransmission timer is thus necessarily adaptive and must reflect the current network conditions. The estimator proposed by [9] is widely used for TCP RTO estimation and works as follows. A smoothed RTT estimate (denoted *SRTT*) and a smoothed variance estimate (denoted *SVAR*) are computed from the sampled RTT measurements using an exponentially weighted moving average. The RTO is then set to

$$RTO = SRTT + kSVAR, \tag{1}$$

where the value of k in (1) is typically set to four [10].

TCP implementations make RTT measurements using a clock with a certain granularity G that ranges from very low values (fine grained timer) up to a coarse value of 500 msec. Also, in many implementations, a minimum RTO (denoted $RTO_{\min}$) is also used that can be as large as 1 sec.

The number of RTT samples per window round may also vary. Two options are proposed – one measurement per window round or using all available samples by making use of the time-stamps options. We model RTO operation as follows – assume the time-stamps option and a fine grained timer. Since we are considering step RTT changes whose rate is less than once per RTT, RTO will track closely the current RTT, except directly after an RTT change due to high instantaneous variance. The use of a coarse grain timer means that the time-out value will not reflect the small changes in the RTT and hence is equivalent to setting a large minimum RTO.

Let Y_i denote the RTO just before the transition *out of* state i. The above discussion then yields

$$Y_i = \max(T_i, G, RTO_{\min}). \tag{2}$$

When the time-stamps option is not used, RTO will be updated only once per RTT. Hence, the RTO might not track well the RTT even with the use of small G and $RTO_{\min}$, since there might not be enough window rounds between two step RTT changes to allow the RTO to converge towards the RTT. However, observe that in this case the RTO will be larger than the RTT (due to the variance term). Thus, our analysis based on (2) essentially assumes a more aggressive timer, resulting in the steady state throughput estimates that are a lower bound

to that observed via simulations. Note that for many practical configuration values, the RTT value will remain unchanged for many rounds, allowing (2) to hold even if the time-stamps option is not used. Extensions for the case where (2) does not hold is postponed for future research.

4. Analysis of TCP with a single step RTT transition

In this section, we consider the behavior of a TCP session for a *single (isolated) step transition* in RTT from T_i to T_j ($1 \leqslant i, j \leqslant M$) during congestion avoidance for a window round w_1. We analyze the behavior for the two cases of RTT change (decrease and increase) separately. The results of this section will be used in section 5 for deriving the *steady state* window size distribution and steady-state TCP throughput where the TCP session undergoes many such isolated RTT step change events.

4.1. RTT decrease

For RTT decrease to cause a reordering, the change (decrease) must occur during the busy period of a round so that packets with a higher sequence number will be acknowledged before the packets with a lower sequence number. An RTT decrease during the idle period of a round will not cause a DA since all packets transmitted during that round have the same RTT and hence will arrive in order during the next round. An RTT decrease will cause a spurious DA only if the number of packets received out of order at the receiver exceeds the threshold d required to trigger a DA (typically $d = 3$).

Consider a congestion window round w_1 (see figure 3). The packets transmitted during this round are numbered from $1, \ldots, w_1$ which can be divided into two bursts: the first is composed of packets $(1, 2, \ldots, c)$ that experience RTT T_i while the second is composed of packets $(c+1, c+2, \ldots, w_1)$ with RTT of T_j. The conditions for a spurious DA can be stated as follows:

(i) There are enough (d or more) packets in the second burst to trigger a DA (i.e., $c \leqslant w_1 - d$).

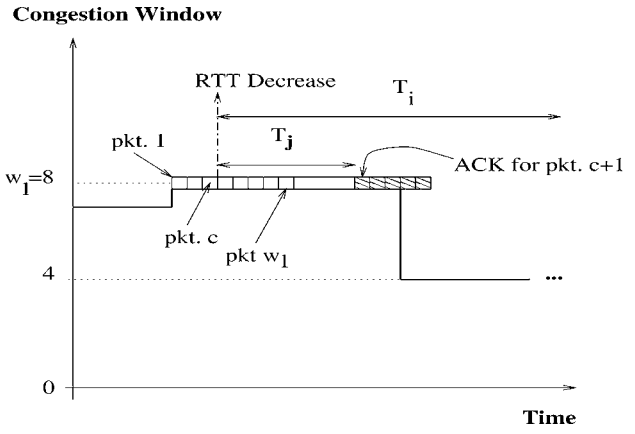

Figure 3. Sketch of an RTT decrease during the busy period of a round w_1 that causes a DA.

(ii) ACK for packet $c+d$ arrives before that of packet 1 (i.e., $T_j + d/\mu < T_i$).

(iii) ACK for packet $c + 1$ arrives after the transmission of packet w_1 (i.e., $T_j > (w_1 - c)/\mu$).

Let $I_c(w_1, T_i, T_j)$ be an indicator function if the above conditions are met for some value of c. Then,

$$I_c = \begin{cases} 1, & \text{if } c \leqslant w_1 - d \text{ and } \dfrac{w_1 - c}{\mu} < T_j < T_i - \dfrac{d}{\mu}, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

Let $P_{\mathrm{DA}}(w_1, T_i, T_j)$ denote the probability that a DA takes place, conditioned on an RTT change during the window round w_1. Let the depth be uniformly distributed in $[1, \mu T_i]$ (a discussion for this assumption is postponed to section 5.2). Then,

$$P_{\mathrm{DA}}(w_1, T_i, T_j) = \frac{\sum_{c=1}^{w_1} I_c(w_1, T_i, T_j)}{\mu T_i}. \tag{4}$$

4.2. RTT increase

RTT step increase causes a TO depending on the following:

(i) whether the RTT increase takes place during the idle or the busy period of the round;

(ii) the RTO prior to the time instant of RTT increase;

(iii) the RTT values before and after the RTT change.

Let Y_i denote the retransmission timer value (RTO) just before the RTT step increase from T_i to T_j. The two cases for RTT increase are considered separately below.

4.2.1. RTT increase during the idle period of a round

Figures 4 and 5 depict a scenario where an RTT step increase from T_i to T_j takes place during the idle period of a w_1 round. Three bursts of packet transmissions are shown. We number the first burst of packets as $(1, 2, \ldots, w_1)$ which are the packets transmitted during the w_1 round. Since the RTT changes after this burst of packets have been transmitted, the ACKs of these packets will arrive after duration T_i subsequent to transmission. The arrivals of the ACKs will cause packets $(w_1 + 1, w_1 + 2, \ldots, 2w_1 + 1)$, a total of $w_1 + 1$ packets, to be released during the busy period of round $w_1 + 1$. The RTO at the end of this burst of packets is still equal to Y_i since the ACKs with RTT equal to T_j will not have arrived yet at the source.

A TO takes place if the ACK for packet $w_1 + 1$ arrives after the scheduled time-out expiry. Note that the time-out period is measured from the instant the last ACK is received which is identical to the time packet $2w_1 + 1$ transmission is completed. The condition for one or more timeouts to take place is thus,

$$T_j > Y_i + \frac{w_1 + 1}{\mu}. \tag{5}$$

Since multiple timeouts may occur, our analysis considers the two cases – single TO and multiple consecutive TO's separately.

Case A: Single TO. With reference to figure 4, when the first TO takes place, the following occurs:

(i) the window size is reduced to 1,

(ii) the slow-start threshold is reduced to $\max(1, (w_1 + 1)/2)$, and

(iii) packet $w_1 + 1$ is retransmitted.

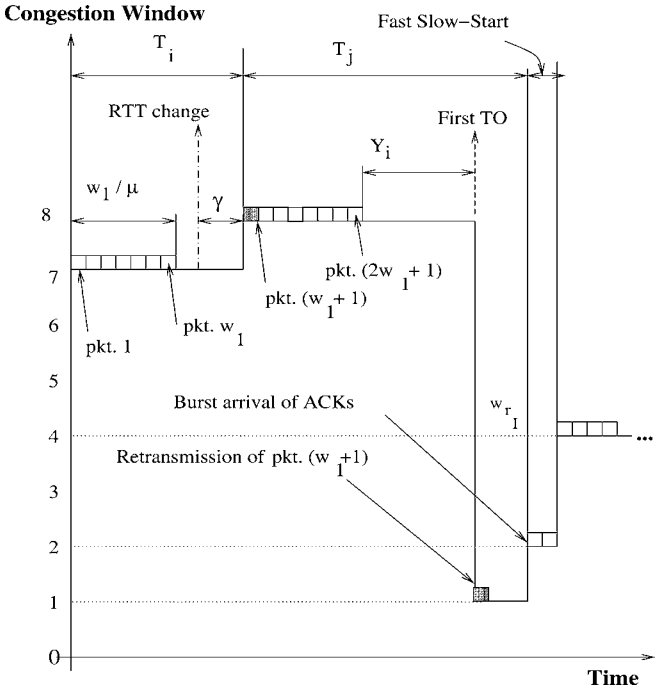

Figure 4. Sketch of an RTT increase during the idle period of a round w_1 that causes a single TO.

For a single TO, the original ACK for packet $w_1 + 1$ will arrive before the second timeout. If this is the case, then all ACKs for the original burst $w_1 + 1, \ldots, 2w_1 + 1$ arrive at an inter-packet rate of $1/\mu$. This results in an almost step increase of the window size from 1 till the slow-start threshold. We call this a "fast" slow-start since $\mu \gg 1/RTT$ for normal slow-start periods (normal slow start are those not involving spurious TO's). TCP then proceeds in the congestion avoidance phase.

Let w_{r_I} denote the window size just after the termination of the fast slow-start period (where I indicates a transition during the idle period of the round). Then, for the case of a single TO,

$$w_{r_I} = \max\left(1, \frac{w_1 + 1}{2}\right). \tag{6}$$

Case B: Multiple consecutive TO's. Consider now the case of multiple consecutive TO's (see figure 5). The TCP source will enter into a sequence of timeouts until the ACK for the first transmission of packet $w_1 + 1$ arrives. Due to the binary exponential back-off algorithm of TCP, the timer expiry period is doubled after each of the first six timeouts, and stays constant thereafter. Let K_I denote the number of consecutive TO's (where I indicates a transition during the idle period of the round). Recall that after the first TO, the threshold is set to $\max(1, (w_1 + 1)/2)$ while the congestion window is set to 1. When the second TO occurs, the threshold is hence set to $\max(1, (1 + 1)/2) = 1$ and the congestion window to 1. When the ACKs for the original transmission of packets $w_1+1, \ldots, 2w_1+1$ arrive, the first of these will end the slow start phase while the rest will advance the window size at a rate of one packet every window's worth of ACKs. Notice

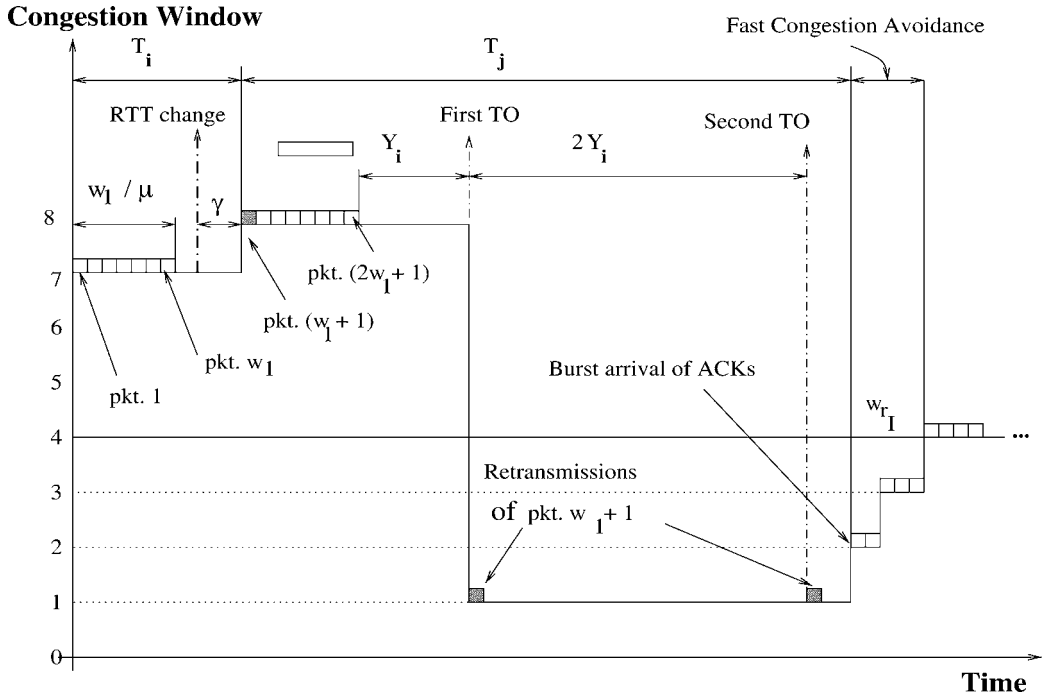

Figure 5. Sketch of an RTT increase during the idle period of a round w_1 that causes multiple consecutive TO's.

that the ACKs arrive in a burst separated by $1/\mu$; we call this a "fast" congestion avoidance phase.

Let w_{r_I} denote the final window size at the end of the "fast" congestion avoidance (which coincides with the reception of the ACK for packet $2w_1 + 1$). Then,

$$w_1 = \sum_{n=1}^{w_{r_I}} n = \frac{w_{r_I}(w_{r_I} + 1)}{2} \tag{7}$$

and hence

$$w_{r_I} = \frac{-1 + \sqrt{1 + 8w_1}}{2}. \tag{8}$$

Finally, combining the results for the two cases (i.e., (6) and (8)) yields

$$w_{r_I} = \begin{cases} \max\left(1, \dfrac{w_1 + 1}{2}\right), & K_I = 1, \\ \dfrac{-1 + \sqrt{1 + 8w_1}}{2}, & K_I > 1. \end{cases} \tag{9}$$

From the description of the Binary Exponential Back-off Algorithm and by inspecting the sketch in figure 5, it is straightforward to verify that K_I is given by

$$K_I = \begin{cases} k, & \text{if } (2^k - 1)Y_i < T_j - \dfrac{w_1 + 1}{\mu} \\ & \qquad < (2^{(k+1)} - 1)Y_i, \\ & \forall k\ 1 \leqslant k \leqslant 5, \\ k, & \text{if } 64(k-5)Y_i < T_j - \dfrac{w_1 + 1}{\mu} - 127Y_i \\ & \qquad < 64(k-4)Y_i, \\ & \forall k\ k > 5. \end{cases} \tag{10}$$

Notice that two different expressions are needed in (10) for the two ranges of k since the BEBO algorithm doubles the timeout period only for the first six consecutive timeouts while it remains constant thereafter.

In summary, there are two possible scenarios that follow an abrupt RTT increase during the idle period of a round:

(i) no time-outs, in which case the window size proceeds its normal operation of increase but at a slower rate; or

(ii) one or more timeouts as given by (5).

In the second case, there are two possibilities; (a) a single TO or (b) a sequence of TO's. If a single TO takes place, a "fast" slow-start phase ensues, which ramps up the window size to a value w_r given by (6). On the other hand, if b multiple TO's take place, a "fast" congestion avoidance phase takes place that ramps up the window size to a value w_r given by (8). The duration of these "fast" phases is $(w_1 + 1)/\mu$.

4.2.2. RTT increase during the busy period of a round

Figure 6 shows a sketch for the case of a step RTT increase just after the transmission of the cth packet during a busy period of a round with a congestion window w_1, where $c < w_1$. The analysis in this case is parallel to the above discussion. The condition of one or more TO in this case is,

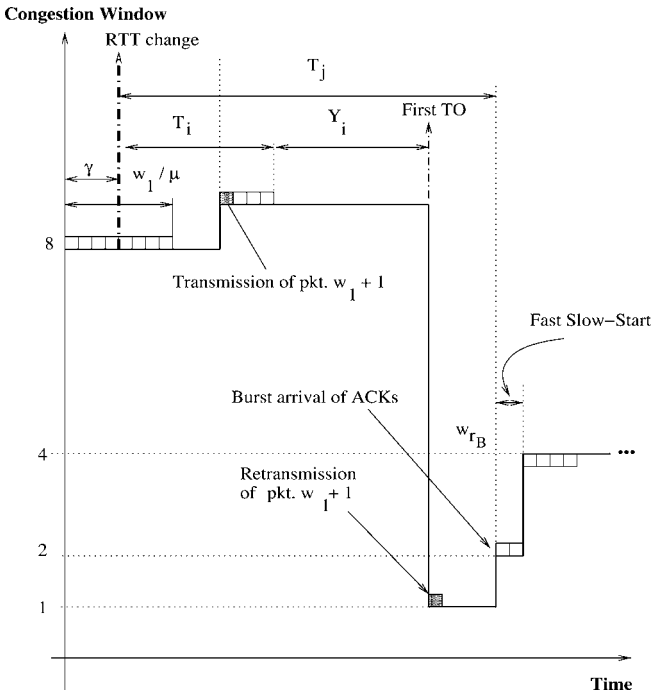

Figure 6. Sketch of an RTT increase during the busy period of a round w_1 that causes a single TO.

$$T_j > Y_i + T_i. \tag{11}$$

Let the window size just after the sequence of one or more TO's be denoted by w_{r_B}. In this case, w_{r_B} will depend on c, the exact location within the burst size at which RTT is changed. Assuming c to be uniformly distributed from 1 to w_1, the mean value is $w_1/2$. The motivations behind using a uniform distribution of c is discussed in the following section. Let K_B denote the number of consecutive TO's, then, similar to the derivation of (9), we have[3]

$$w_{r_B} = \begin{cases} \max\left(1, \dfrac{w_1/2 + 1}{2}\right), & K_B = 1, \\ \dfrac{-1 + \sqrt{1 + 4w_1}}{2}, & K_B > 1. \end{cases} \tag{12}$$

From the description of the Binary Exponential Back-off algorithm and by inspecting the sketch in figure 6, it is straightforward to verify (similar to the derivation of (10)) that

[3] Alternatively, the expected value of w_{r_B} can be derived by first finding an expression for w_{r_B} as a function of c and then finding the expected value. However, we use the approximate expression in (12) for simplicity and later validate the results with simulations.

K_B is given by

$$K_B = \begin{cases} k, & \text{if } (2^k - 1)Y_i < T_j - T_i \\ & \qquad < (2^{(k+1)} - 1)Y_i \\ & \forall k\ 1 \leqslant k \leqslant 5, \\ k, & \text{if } 64Y_i(k-5) < T_j - T_i - 127Y_i \\ & \qquad < 64(k-4) \\ & \forall k\ k > 5. \end{cases} \tag{13}$$

5. Steady-state analysis

5.1. The Embedded Markov Chain (EMC)

In the preceding section, we analyzed the behavior of TCP with a single step RTT change. Specifically, given a congestion window round w_1 and the depth during the round at which the RTT transition takes place, we derived expressions for the following; probability of a spurious DA or a spurious TO, number of consecutive TO's (in case of a TO), the congestion window size just after the fast slow start or fast congestion avoidance (denote w_r) and the duration of time following a spurious TO or DA until TCP resumes transmission in the congestion avoidance phase.

Based on the preceding analysis of the one-step transition (see figure 7), it is clear that, conditioned on the congestion window size and depth just before an RTT step transition, the evolution of the congestion window up till the next RTT transition is deterministic.

Let W_n denote the window size just before a transition out of the RTT state H_n. Let C_n denote the window round depth upon the nth transition. Then the sequence of values defined by $Z_n = (W_n, H_n, C_n)$, $n = 1, 2, \ldots, \infty$ forms an (embedded) Markov chain (EMC). The dimensions of the EMC are $w_{\mathrm{m}}^2 \cdot M$ (recall that M is the number of states and w_{m} is the TCP maximum advertised congestion window size).

Since the transitions of the RTT process $T(t)$ are independent of the TCP session evolution, and may take place anywhere during a TCP window round, the variable C_n must be included in the state description since it's omission would lead to ambiguity about the exact time of transition.

5.2. Reducing Markov chain dimensionality

The Markov chain requires the computation of the transition probability for each combination of window size, depth and state. Such a computation, while indeed possible, is almost as exhaustive as conducting a full simulation; the main complexity arising from the fact that the transition matrix must be computed not only for the congestion window size and RTT state, but also for the depth within each window round. One avenue for decreasing the EMC dimensionality without causing the above mentioned ambiguity is to assume that transitions of the RTT process are embedded at the instants of the window size changes. However, this assumption, in addition to contradicting the actual physical behavior, does not allow us to capture the important behavioral differences of TCP presented in the preceding section which depends on the exact instant at which a transition takes place.

Thus we revert to the following approximation that allows us to reduce EMC dimensionality while still capturing the detailed behavior of TCP presented in the preceding section. The resulting reduced EMC will be specified by $Z_n = (W_n, H_n)$ with dimensionality $w_{\mathrm{m}} \cdot M$. Assume that, conditioned on a transition during the window round w_1 out of state i, the depth is uniformly distributed in $[1, \mu T_i]$. Let $p(w_1, i)$ denote the probability, given that $Z_n = (w_1, i)$, that the transition takes place in the idle period of the round. Then,

$$p(w_1, i) = 1 - \frac{w_1}{\mu T_i}. \tag{14}$$

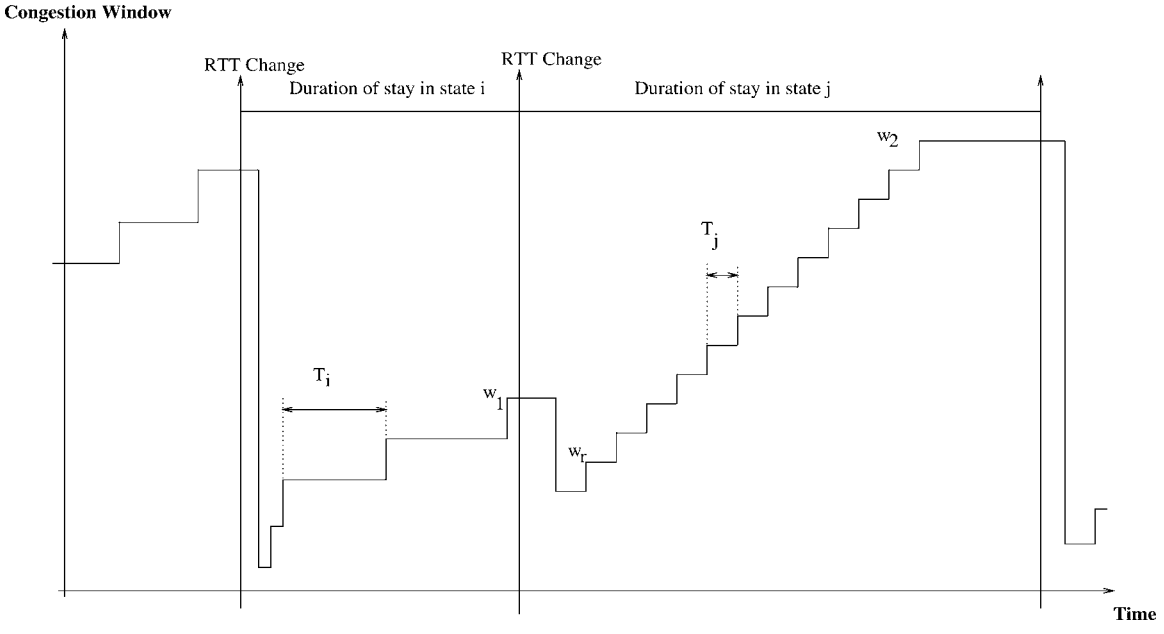

Figure 7. The EMC $Z_n = (W_n, H_n)$.

Although it is not likely that the depth distribution would be precisely uniformly distributed, we argue that (14) is a good approximation since, conditioned on a transition at window size is w_1, the probability that the transition takes place in the busy period is intuitively proportional to the ratio of the busy duration to the duration of the round. Since the busy duration is w_1/μ (see definition 2), (14) follows.

5.3. Approximating the fast phases

In section 4, we have shown that when one or more TO occurs, a "fast" slow start (in case of single TO) or "fast" congestion avoidance (in case of multiple TO's) is initiated that quickly increases the congestion window size to a value given by (9), (12) for RTT transition during idle and busy cases, respectively. The rate of window size increase during "fast" slow-start and "fast" congestion avoidance (μ) greatly exceeds that during the corresponding "normal" versions (1/RTT), thus justifying the nomenclature. Hence in the subsequent analysis, we neglect the *duration* of the fast phases;[4] in other words, after a sequence of one or more TO's, we assume that the window size is immediately increased to w_{r_I} (or w_{r_B}) and the TCP session enters congestion avoidance.

5.4. Analysis

Three parameters of the EMC $Z_n = (W_n, H_n)$ are of interest:

(i) the transition probability matrix $\mathbf{P}$,

(ii) the average number of packets transmitted upon each visit to an EMC state, and

(iii) the average time spent in each of the EMC states.

The last of these quantities is available from the MC describing the RTT process. The other two parameters need to be determined.

Let $\mathbf{P}(W_{n+1} = w_2, H_{n+1} = j | W_n = w_1, H_n = i) = \mathbf{P}(w_2, j | w_1, i)$ denote the probability of a transition from $Z_n = (w_1, i)$ to $Z_{n+1} = (w_2, j)$ and let $E[N|w_1, i, j]$ denote the expected number of transmitted packets conditioned on $Z_n = (w_1, i)$ and an RTT transition into j.

The analysis of the single state transition dynamics can be applied at each possible transition of the RTT process rendering expressions for the above two parameters of interest. From (5)–(11) for the analysis of RTT increase, notice the following: (a) at least one TO will take place if $T_j > Y_i + T_i$, (b) a single TO may take place, depending on whether RTT changes during the idle or busy durations of a round if $Y_i + (w_1 + 1)/\mu < T_j < Y_i + T_i$, and (c) a TO will *not* take place if $T_j < Y_i + (w_1 + 1)/\mu$. A TD may take place only in the case of RTT decrease (d) $T_j < T_i$. For the sake of analysis, we combine cases (c) and (d) together (by setting $P_{\mathrm{TD}} = 0$; $T_j > T_i$). Thus, we have three cases, the detailed derivation of each is shown in appendices A, B and C.

[4] The duration of the fast phases can be derived and included as an additive correction in the throughput calculations.

The analysis applies to both OldTahoe and Reno/NewReno versions with the difference that OldTahoe only uses TO's as a means of packet loss inference and hence, for OldTahoe, $P_{\mathrm{DA}} = 0$.

5.5. Average transmission rate

Let $\mathbf{s}(w_1, i)$, $1 \leqslant w_1 \leqslant w_{\mathrm{m}}$, $1 \leqslant i \leqslant M$ denote the steady state probability distribution of the embedded Markov chain Z_n. Then $\mathbf{s}(w_1, i)$ can be solved for numerically from the transition probabilities $\mathbf{P}(w_2, j|w_1, i)$ using any of the well-known standard techniques. The existence of a unique solution is discussed in the following section.

Suppose the current state is $z_1 = (w_1, i)$ and that the RTT process next transitions to j. Let $U(z_1, z_2)$ denote the time spent in transition between $z_1 = (w_1, i)$ and $z_2 = (w_2, j)$. Observe that $U(z_1, z_2) = X_j$. Let $E[R|w_1, i, j]$ denote the expected value of the reward rate for those states. From renewal reward theory,

$$E[R|w_1, i, j] = \frac{E[N|w_1, i, j]}{E[X_j]}. \tag{15}$$

Let $E[R|w_1, i]$ denote the expected value of the reward rate in state (w_1, i), then

$$E[R|w_1, i] = \sum_{j=1}^{M} E[R|w_1, i, j] H_{i,j}. \tag{16}$$

Let $E[U(z_1)]$ denote the average holding time in state $z_1 = (w_1, i)$. Then,

$$E[U(w_1, i)] = \sum_{j=1}^{M} E[X_j] H_{i,j}. \tag{17}$$

Finally, let $\delta(w_1, i)$ denote the proportion of time the TCP session spends in state $Z_n = (w_1, i)$. Then (see, for example, [7, chapter 5, equation (61)]),

$$\delta(w_1, i) = \frac{s(w_1, i) E[U(w_1, i)]}{\sum_{(w_2, j)} s(w_2, j) E[U(w_2, j)]} \tag{18}$$

and hence the average steady-state transmission rate $E[R]$ of the TCP session can be calculated as

$$E[R] = \sum_{(w_1, i)} E[R|w_1, i] \delta(w_1, i) \tag{19}$$

based on renewal reward theory [7].

5.6. Ideal average transmission rate

Let π_i, $i = 1, 2, \ldots, M$, denote the stationary distribution of the embedded (discrete time) Markov chain $\mathbf{H}$, and let θ_i denote the steady state probability of finding the process $T(t)$ in state i (which is also equal to the steady state proportion of time the process spends in state i), then, if the Markov chain is positive recurrent (e.g., see [16]),

$$\theta_i = \frac{\pi_i E[X_i]}{\sum_{j=1}^{M} \pi_j E[X_j]}. \tag{20}$$

Assume a TCP protocol that somehow differentiates between changes in round-trip delay and packet losses and thus avoids generating the spurious time-outs analyzed thus far. Hence, except for an initial slow-start phase at the on-set of the TCP session, the TCP session spends the rest of its lifetime in congestion avoidance phase with a window size w_{m}. The resulting steady-state average throughput, denoted $E[R_I]$ is thus given by

$$E[R_I] = w_{\mathrm{m}} \sum_{i=1}^{M} \frac{\theta_i}{T_i}. \tag{21}$$

5.7. Existence of steady state solution

Let the matrix of the transition probabilities of $Z_n = (W_n, H_n)$ be denoted by $\mathbf{P}$. Then $\mathbf{P}$ may have a *stationary* distribution vector $\mathbf{s}$ (of size $w_{\mathrm{m}} \cdot M$) that satisfies $\mathbf{sP} = \mathbf{s}$. We consider next the conditions for existence of such a unique steady state solution.

Consider first the transition probability matrix $\mathbf{H}$ of the RTT process. It is easy to see that, for many practical reasons, $\mathbf{H}$ will be composed of a single recurrent[5] class. For simplicity, we will also assume that the recurrent class is aperiodic (i.e., ergodic). This guarantees that there exists a unique steady state solution $\pi\mathbf{H} = \pi$ and all rows in π are identical.

Consider now the transition matrix $\mathbf{P}$; we now show that the EMC Z_n will be composed of at most two (recurrent or transient) classes of states, one of which must be recurrent (the transient class may or may not exist) thus guaranteeing the existence of a unique solution for $\mathbf{P}$.

It is well known that any finite-dimensional Markov chain will be composed of at least one recurrent class. The reason why there may exist another transient class in $\mathbf{P}$ can be explained as follows. Consider a Markov chain with $\mathbf{H}$ composed of two states, and assume that $T_1 < T_2 < 2T_1$. In this case, a single timeout may take place if the window size is less than a certain value given by (5). If the window size equals w_{m}, no DA's will take place (3). Thus, for this example, once the window size reaches any of the states (w_{m}, i), $i = 1, 2$, no TO's or DA's will take place and the window size will continue at w_{m}. Since there is a positive probability of the window size reaching w_{m} at some instant, the class of states (w_{m}, i), $i = 1, 2$ is a recurrent class and the class (w, i), $w < w_{\mathrm{m}}$, $i = 1, 2$ is a transient class. The same applies for higher dimension $\mathbf{H}$.

In summary, a sufficient (though not necessary) condition for the existence of a unique stationary solution $\mathbf{s}$ for $\mathbf{P}$ is that the embedded Markov chain $\mathbf{H}$ is recurrent and aperiodic (i.e., ergodic).

[5] Consider a TCP session over a path consisting of up to M inter-satellite hops where the major component of delay is due to propagation. Then the assumption of a single recurrent class is true if there *does not* exist a set of hop numbers such that the path is "locked in" without ever revisiting hop numbers outside this set. Such situations are pathological, thus supporting our assumption.

6. Results

We first validate the analytical results against simulations using the *ns* simulator (version 2.1b7a) [13] for the two version of TCP: OldTahoe and Reno. We then compare the performance of the two versions against the ideal TCP operation as a function of the various TCP parameters as well as the end-to-end path properties.

The choice of the parameter values for the numerical computations and simulations in this section are motivated by measurements from simulations of Teledesic [17] and Iridium [11] LEO satellite constellations reported in [8]. We summarize the relevant results from [8] here that motivate our choice of parameter values. For Teledesic constellation, the *one way* delay for a single session over a long period of time between two terminals located in New York and San Francisco was found to vary over a range of roughly 23–60 ms. End-to-end delays were characterized by slow variations punctuated by step increases and decreases in the delay that is *usually* less than 8 ms, attributed to handoffs somewhere along the route. The time duration between step changes was found to be in the order of tens of seconds. The effect of these abrupt changes was not considered in [8] since the link speeds used were low/moderate (1 Mbps), which, with 8 ms delay and 0.5 Kbyte packets, can not cause more than 2 packets reordering. Measurements for the Iridium constellation showed similar behavior but with higher variability in the delay, where step changes were found to be as large as 90 ms, with the delay varying from 20 to 75 ms. The difference in the delay behavior between the two constellations is largely attributed to the number of satellites in each constellation.

With the above reported values from [8] as guidelines, we select the parameter values for our experiments as follows; The RTT's (i.e., *two-way* delays) are selected in the range of 0.05–0.1 seconds. While [8] experiments with low speed links only (1 Mbps), we consider speeds in the range of 1–10 Mbps. Paper [8] considers constellations with a light traffic load. However, we expect that moderate and high loads will result in more frequent step changes and thus we experiment with range of values for the duration of time between step RTT changes ranging from one to ten times the RTT. The following section describes in more detail how each parameter is chosen.

6.1. Model validation

The numerical results were compared to those obtained from the *ns* simulator. In *ns*, we use a simulation script that changes the delay between a pair of TCP "agents" (a "source" and a "sink" [13]) during the simulation time according to a pregenerated sample of the RTT process which varies according to the MC statistics described in section 3. We use a fine-grained timer in all the experiments.

A total of 100 simulation experiments were performed for TCP Reno, and the same set was run for TCP OldTahoe. The parameters for each experiment are selected randomly,

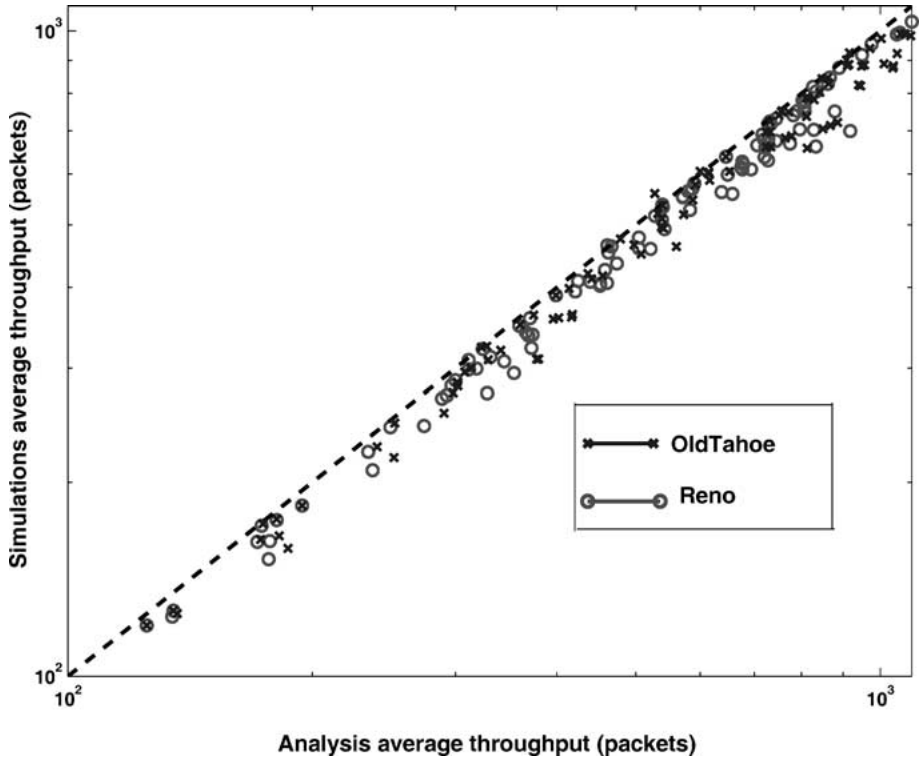

Figure 8. Validation of the analysis for TCP Reno/NewReno and OldTahoe.

as explained below. Once the parameters are chosen, they are kept fixed during the lifetime of the simulation. As motivated earlier, the parameters for these experiments are chosen as follows: link capacity $\mu = U[1, 10]$ Mbps (U denotes the uniform distribution), packet size of 1 Kbyte (and hence $\mu = U[125, 1250]$ packets/sec) and $T_i = U[0.05, 0.1]$. Since the script uses a pre-generated sample of the RTT's according to the semi-Markov model parameters, we select $w_m < \mu \min(T_i)$ to guarantee that the simulation does not introduce additional delays at the source agent's transmit queue that are not accounted for in the RTT model used to pre-generate the RTT samples. We thus choose $w_m = U[2, \mu \min(T_i)]$. The elements of **H** were selected randomly with the condition that each row in the transition matrix sums up to 1 and that the matrix is positive recurrent. The average holding time in each state is chosen as $X_i = T_i + V_i$ where V_i is exponentially distributed with mean $U[0, 9T_i]$. Thus, the duration of stay in each state is lower bounded by T_i, while the mean ranges from T_i to $10T_i$. The experiments consider a semi-Markov chain with $M = 3$ states. Each simulation experiment is composed of 5 simulation runs (with different random seeds) which provide 5 measurements of the steady-state throughput. The duration of each simulation run was set to $1000 \max(E[V_i]+T_i)$ to guarantee a sufficient duration of operation in steady state. The average of these five runs is plotted against the analytical result in figure 8. The same experiments (i.e., using the same parameters used for the Reno experiments) are repeated for OldTahoe, and the results are shown in figure 8. The experiments show the analytical estimates to match those of the simulations with reasonable accuracy as the points lie close to the ideal 45° line. The deviation was less than 20% for all experiments with most experiments reporting an error of within 10% error for a mean error of less than 10%. Another set of experiments was performed using 100 random parameters selected as $\mu = U[1, 100]$ Mbps, $w_m = U[4, 100]$, $T_i = U[w_m/\mu, 10w_m/\mu]$ and $E[V_i] = U[0, 9T_i]$. The results were similar to figure 8 and hence are omitted for space considerations.

It is worth noting that the analytical results are a (tight) lower bound to those obtained via simulations. This is expected due to the assumption used in obtaining (2), where we essentially assume an aggressive timer (refer to section 3.2.4 for explanation) resulting in a slight over-estimation of the probability of a spurious TO and thus a slightly lower throughput than the actual (i.e., measured) one.

6.2. OldTahoe versus Reno

Figure 9 shows that TCP OldTahoe provides equal or superior performance to TCP Reno in all experiments with improvement from 0% (no improvement) to almost 100% (i.e., throughput is doubled). An identical figure (and hence not shown in this paper) also applies for the goodput measurements. The data shows that 60% of the experiments using OldTahoe result in a slight improvement ($\leqslant$10%), 20% show a notable increase ($\geqslant$10%) and remaining 20% show a large increase of up to 100% in the throughput when using OldTahoe. We conclude that the use of OldTahoe is recommended[6] for highly dynamic networks (networks with frequent abrupt

[6] It is worth noting that this work did not consider the effect of packet loss in the presence of abrupt delay variations, which may take place in the context of a mobile user in a fixed terrestial cellular network with small cell size.

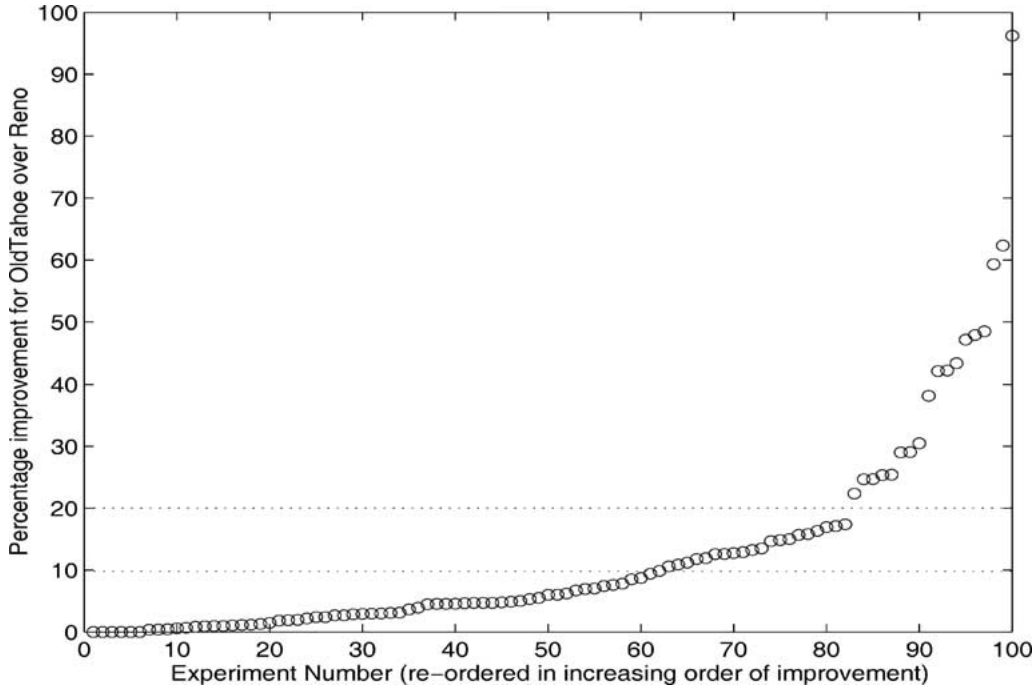

Figure 9. OldTahoe percentage improvement over Reno/NewReno.

RTT changes). While both versions are not immune to spurious TO's, Reno is also vulnerable to spurious DA's that result in significant loss of throughput when they occur.[7] OldTahoe can be thought of as equivalent to Reno with the DA threshold d (denoted *DUPACK* in *ns*) set to infinity. This indicates the need for an evaluation of the effect of d on TCP Reno's performance, which we discuss next.

6.3. Effect of the DA threshold d

The results presented in sections 6.3–6.5 are based on one set of parameter values given below:

$$H = \begin{bmatrix} 0.1 & 0.33 & 0.57 \\ 0.30 & 0.12 & 0.58 \\ 0.165 & 0.36 & 0.475 \end{bmatrix},$$
$$T = [0.058056 \quad 0.060383 \quad 0.081911],$$
$$V = [0.1067 \quad 0.1183 \quad 0.0042]$$

and $\mu = 1210$ packets/sec. The bandwidth-delay product over the smallest RTT for this experiment, denoted *BD*, is 70 packets.

Figure 10 shows the throughput as a function of the DA threshold value for different values of w_m. Three sets of curves are shown for $w_m = 60, 30, 25$. We notice that increasing the value of d increases the throughput until the throughput saturates; as expected, the saturation value is equal to the throughput provided by OldTahoe. However, the value at which the saturation takes place depends on w_m. For w_m larger than *half* the bandwidth-delay product, the saturation takes place at $d \approx BD/2$. However, when $w_m < BD/2$, the saturation takes place at $d \approx w_m$. Let d_{opt} denote the optimal threshold value setting. Then $d_{opt} = \min(w_m, BD/2)$.

The above result can be explained with reference to the derivation of (3) and figure 3; in both ranges of w_m, the maximum number of reordered packets is $w_m - 1$. However, in the case where $w_m > BD/2$, the maximum number of re-ordered packets satisfying *both* conditions (ii) and (iii) of section 4.1 is $BD/2$.

6.4. Effect of maximum advertised window w_m

Figure 11 shows the TCP throughput for the Ideal, Reno and OldTahoe cases as a function of w_m for the same parameters listed in the previous section with $d = 3$. As expected, increasing w_m increases the throughput in the three cases. However, we also notice that the deviation between the ideal throughput and that of OldTahoe increases by increasing w_m. A similar behavior exists for the deviation between OldTahoe and Reno. Thus, we conclude that the adverse effect of the RTT variations is more drastic for higher available bandwidth (indicated by higher w_m).

6.5. Effect of the transition rate

Consider the semi-Markov process for the RTT and imagine scaling the holding time in each state while keeping the transition probabilities and the RTT's unchanged. Specifically,

In [1], we proposed via simulations a modified model in which, in addition to the semi-Markov process describing the RTT, packet loss may take place with probability p independently. Extension of our analytical results to the case of random packet loss and abrupt delay changes is deferred to future studies.

[7] NewReno is identical to Reno except that NewReno responds to a DA only once per window round. Since our RTT model only allows one abrupt RTT change per round, the NewReno results will be identical to those of Reno.

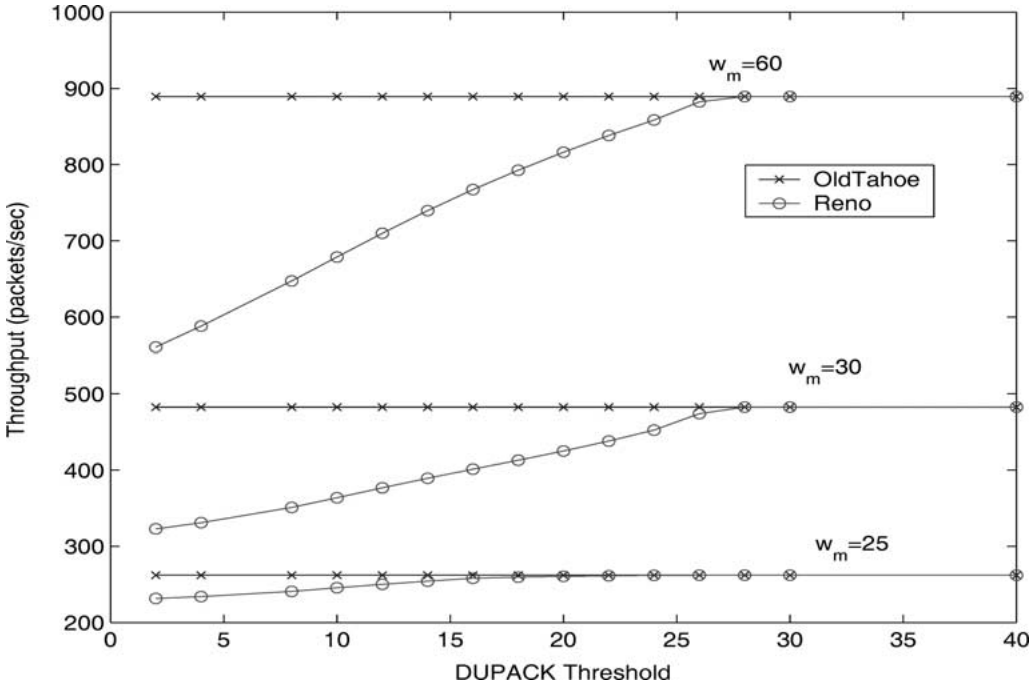

Figure 10. Effect of varying Reno's DUPACK threshold on the throughput.

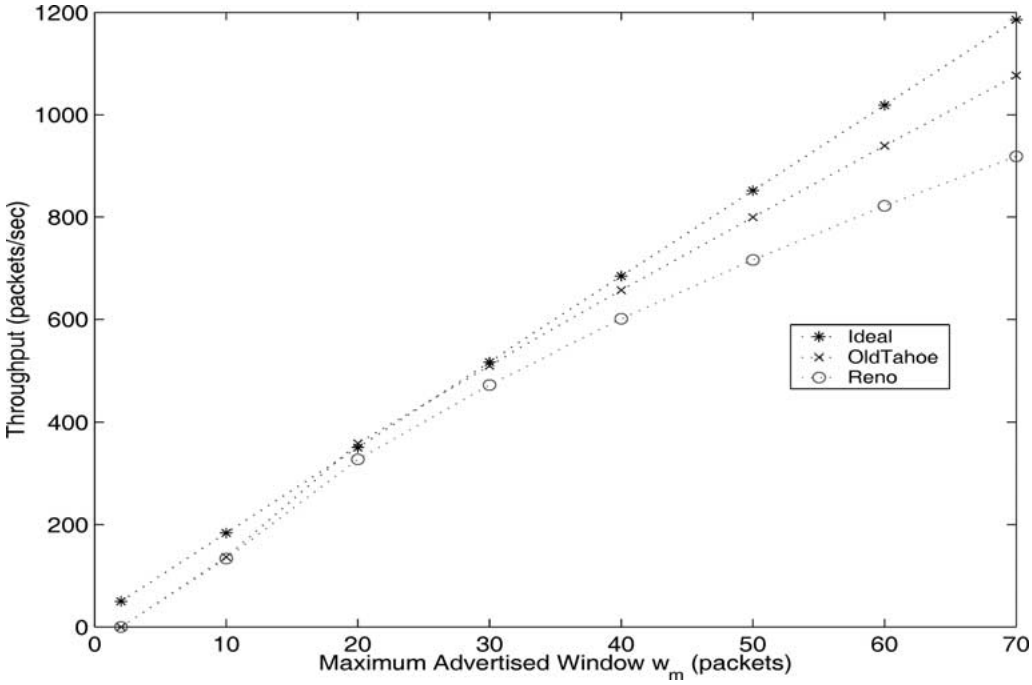

Figure 11. Effect of the maximum advertised window size.

consider varying $E[V_i]$ as $kE[V_i]$ (refer to section 6.1 for a definition of V_i) while keeping the rest of the parameters unchanged. Then this is equivalent to expanding ($k > 1$) or compressing ($k < 1$) the time scale at which transitions take place. We now consider the effect of this on TCP's behavior.

As may be expected, the behavior of the ideal TCP is not affected by changes in transition rates. However, we notice in figure 12 that Reno and OldTahoe both suffer from the fast changes.

OldTahoe shows a "knee" behavior, i.e., there exists a certain RTT transition rate below which OldTahoe's throughput will be close to ideal; transition rates above the knee value will cause a drastic and sudden decrease in OldTahoe's performance. The knee behavior can be attributed to the fact that high transition rate causes both DA and TO events to take place. As the rate is decreased (i.e., k increased), TO events become very infrequent and hence the DA events start to contribute the higher percentage of loss indications (and hence

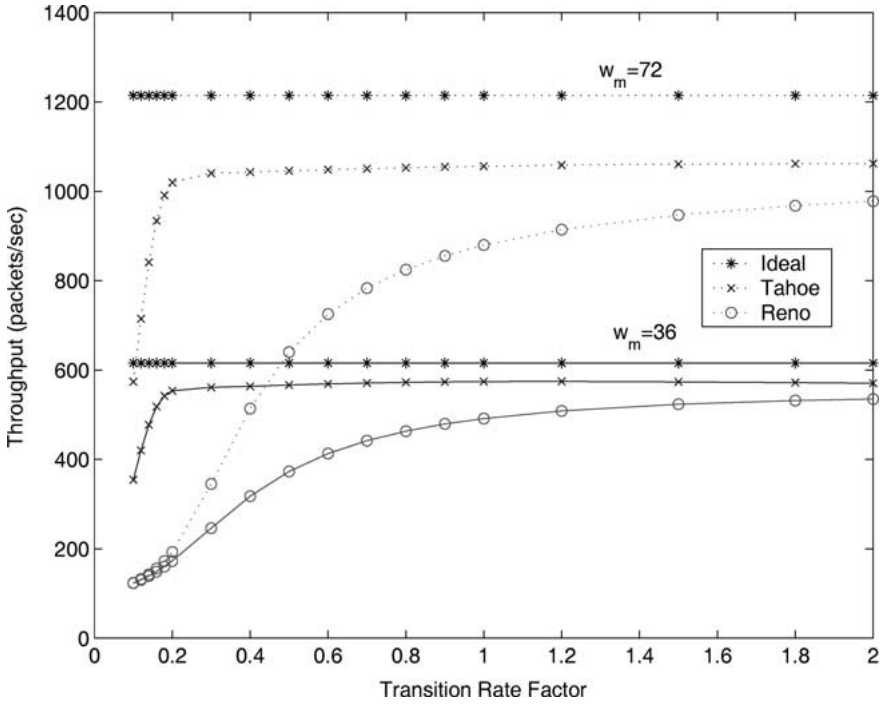

Figure 12. Effect of the transition rate.

OldTahoe's performance quickly improves). However, these DA's are separated by longer durations (due to the lower transition rate) and hence Reno's performance also improves. In the limit, the throughput from both TCP versions tend to that of the Ideal TCP.

7. Conclusion

In this paper, we have presented a model for the random and abrupt RTT transitions, by modeling the RTT process as a semi-Markov process. We then analyzed the behavior of two versions of TCP; the OldTahoe and the Reno/NewReno versions. We compared the performances of these versions against an Ideal TCP that does not respond to the spurious TO's or DA's.

We found that (i) our analytical model matched simulation results for a wide range of parameters with a maximum error of less than 20% (mean error of less than 10%), (ii) OldTahoe outperformed Reno in a considerable number of cases, yielding an improvement in the throughput of up to 100%, (iii) setting the DA threshold to half the maximum path bandwidth-delay product improves Reno's performance to match that of OldTahoe, (iv) the degradation of TCP thourghput performance as compared to ideal TCP becomes more evident at higher advertised windows, and (v) the rate at which the RTT changes has a drastic effect on TCP's throughput.

The analytical framework presented in this paper provides a model for the evaluation of different window-based control protocols that may be more adaptive to the variations of the RTT process. A number of recent works have attempted to provide solutions to this problem, including taking corrective actions if spurious TO's or spurious DA's are detected [3,12], or modifying TCP's retransmit timer to enable more accurate estimation of the RTT. The analytic evaluation of these schemes remain as avenues for future work.

Finally, we remark that the proposed analytical framework may also be used as a guide for optimizing the TO and DA schemes based on estimates of the ratios of spurious TO and DA events. However, for many practical applications, packet loss will also take place in addition to abrupt delay variations. In [1], we proposed via simulations a modified model in which, in addition to the semi-Markov process describing the RTT, packet loss may take place with probability p independently. Extension of our analytical results to the case of packet drops, and TO and DA refinements for this case is subject of our future work.

Appendix A. No TO, possible DA: $T_j < Y_i + (w_1 + 1)/\mu$

Let $\overline{F_j}(x) = \Pr(X_j > x)$ denote the complementary distribution function of the random variable X_j (which is the duration of stay in state j). (For an exponential distribution, $\overline{F_j}(x) = \mathrm{e}^{-\lambda_j x}$.) Let $F_j(x_1, x_2) = \Pr(x_1 \leqslant X_j \leqslant x_1 + x_2)$.

Consider the case where a transition does not cause a DA, and $w_2 > w_1$. Let $C_n = \gamma$ denote the depth upon the nth transition. Define

$$S = \frac{1}{T_i} \int_0^{T_i} F_j\big(T_j(w_2 - w_1) + \gamma, T_j\big)\, \mathrm{d}\gamma. \tag{A.1}$$

Then S is the probability, conditioned on a transition at w_1, that the next state transition occurs at window size w_2 (i.e., the duration of stay satisfies $T_j(w_2 - w_1) + \gamma \leqslant X_j \leqslant T_j(w_2 + 1 - w_1) + \gamma$), when there is no DA and $w_2 > w_1$. It was found that the numerical results

of the integrand in (A.1) were very close to those obtained with $\gamma = 0$ for various combinations of T_i and T_j and thus we henceforth set $\gamma = 0$. The results in this section will henceforth use this numerical approximation.

Following a similar analysis for all w_1, w_2, i and j pairs yields,

$$\mathbf{P} = \mathbf{H}_{i,j} \cdot \begin{cases} 0, & \text{if } w_2 < w_1/2; \\ P_{\mathrm{DA}} F_j\big(T_j(w_2 - w_1/2), T_j\big), & \\ \quad \text{if } w_1/2 < w_2 < w_1; & \\ P_{\mathrm{DA}} F_j\big(T_j(w_2 - w_1/2), T_j\big) & \\ \quad +(1 - P_{\mathrm{DA}}) F_j\big(T_j(w_2 - w_1), T_j\big), & \\ \quad \text{if } w_1 < w_2 < w_{\mathrm{m}}, & \\ \overline{F_j}\big(T_j(w_2 - w_1/2)\big) & \\ \quad +(1 - P_{\mathrm{DA}})\overline{F_j}\big(T_j(w_2 - w_1)\big), & \\ \quad \text{if } w_2 = w_{\mathrm{m}}, & \end{cases} \tag{A.2}$$

where $\mathbf{P}(w_2, j|w_1, i)$ and $P_{\mathrm{DA}}(w_1, i, j)$ are abbreviated as $\mathbf{P}$ and P_{DA}, the latter given by (4). The explanation of (A.2) is as follows. Since RTT transitions are independent from window evolution, the elements of $\mathbf{P}$ must be multiplied by the probability of a transition from i to j, i.e., $\mathbf{H}_{i,j}$. We now proceed to explain the different cases. Since in the case at hand there are no timeouts upon the nth transition, the window size on the $(n+1)$st transition must be at least equal to $w_1/2$; thus the probability that the window size is less than $w_1/2$ on the $(n+1)$st transition is zero. For $w_1/2 < w_2 < w_1$, a DA must take place to reduce the window size below the duration w_1, and of stay X_j in state j must be such that the transition takes place at w_2 (see the explanation of (A.1) above). If $w_1 < w_2 < w_{\mathrm{m}}$, then the two possibilities (with or without DA) may result in the next transition taking place at w_2. This is also the case when $w_2 = w_{\mathrm{m}}$, with the difference that the window size is bounded by w_{m}.

We now compute $E[N|w_1, i, j]$, the expected number of packets transmitted between the nth and the $(n+1)$st transitions, conditioned on $Z_n = (w_1, i)$ and that the transition of $T(t)$ is into state j.

$$E[N|w_1, i, j] = \sum_{n=0}^{\infty} \Pr[N > n|w_1, i, j]. \tag{A.3}$$

Conditioned on $C_n = \gamma$, it follows that the first packet transmission after the nth transition occurs at time $\gamma + 1/\mu$. Similarly, the gth packet transmission takes place at $\gamma + g/\mu$, for any $0 < g < w_1$ where w_1 is the window round. The next sequence of packets are transmitted during the round $w_1 + 1$, with the first packet in that sequence sent at $\gamma + T_j + 1/\mu$. In what follows, similar to the assumptions invoked for the analysis of the transition probability above, we will set $\gamma = 0$.

Let $t_g(w_1, i, j)$ denote the time till the gth packet transmission; note that $g = 0$ when $T(t)$ makes a state transition from i to j, conditioned on no timeouts. Let $E_1(w_1, i, j)$ and $E_2(w_1, i, j)$ denote $E[N|w_1, i, j]$ for the cases without and with a DA, respectively. Then,

$$\begin{aligned} &E_1(w_1, i, j) \\ &\quad = \sum_{g=0}^{\infty} \overline{F}_j\big(t_g(w_1, i, j)\big) \\ &\quad = \sum_{w=w_1}^{w_{\mathrm{m}}-1} \sum_{g=0}^{w} \overline{F_j}\left(\frac{g}{\mu} + T_j(w - w_1)\right) \\ &\qquad + \sum_{j=0}^{\infty} \sum_{g=1}^{w_{\mathrm{m}}} \overline{F_j}\left(\frac{g}{\mu} + T_j(w_m - w_1) + jT_j\right). \end{aligned} \tag{A.4}$$

It is easy to see that, since in the case of a DA the window size is halved,

$$E_2(w_1, i, j) = E_1(w_1/2, i, j) \tag{A.5}$$

and hence combining (A.4) and (A.5),

$$\begin{aligned} E[N|w_1, i, j] = {} & E_1(w_1, i, j)(1 - P_{\mathrm{DA}}) \\ & + E_1(w_1/2, i, j) P_{\mathrm{DA}}. \end{aligned} \tag{A.6}$$

Appendix B. No DA, at least one TO: $T_j > Y_i + T_i$

In this case, at least one TO will take place. The number of consecutive timeouts and the window size just after the sequence of TO's are given by (10), (9) for the case of a transition during the idle period and by (13), (12) for the transition during the busy period.

Thus, conditioned on a transition during the idle or the busy period of the round, the transition probability can be derived in a similar fashion to (A.2). Since the probability of a transition during the idle period is given by (14), the transition probability for the two cases can thus be combined. Hence, we have

$$\mathbf{P} = \mathbf{H}_{i,j} \cdot \begin{cases} 0, & \text{if } w_2 < w_{r_B}; \\ (1-p) F_j\big(T_j(w_2 - w_{r_B}), T_j\big), & \\ \quad \text{if } w_{r_B} < w_2 < w_{r_I}; & \\ (1-p) F_j\big(T_j(w_2 - w_{r_B}), T_j\big) & \\ \quad + p F_j\big(T_j(w_2 - w_{r_I}), T_j\big), & \\ \quad \text{if } w_{r_B} < w_2 < w_{\mathrm{m}}; & \\ (1-p)\overline{F_j}\big(T_j(w_2 - w_{r_B})\big) & \\ \quad + p\overline{F_j}\big(T_j(w_2 - w_{r_I})\big), & \\ \quad \text{if } w_2 = w_{\mathrm{m}}, & \end{cases} \tag{B.1}$$

where p above is a shorthand for $p(w_1, i)$ (see (14)). Observe that $w_{r_B} \leqslant w_{r_I}$ (see (9), (12)).

The derivation of $E[N|w_1, i, j]$ proceeds in a similar manner. Let $E_1(w_1, i, j)$ and $E_2(w_1, i, j)$ denote $E[N|w_1, i, j]$ for the two cases of transition during idle and busy, respectively. Then,

$$\begin{aligned} &E_1(w_1, i, j) \\ &\quad = \sum_{w=w_{r_I}}^{w_{\mathrm{m}}-1} \sum_{g=0}^{w} \overline{F_j}\left(\frac{g}{\mu} + T_j(w - w_{r_I})\right) \end{aligned}$$

$$+\sum_{j=0}^{\infty}\sum_{g=1}^{w_{\mathrm{m}}}\overline{F_j}\left(\frac{g}{\mu}+T_j(w_{\mathrm{m}}-w_{r_I})+jT_j\right). \quad \text{(B.2)}$$

The expression for $E_2(w_1, i, j)$ in this case is identical to (B.2) except for replacing w_{r_I} by w_{r_B}. Hence,

$$E[N|w_1, i, j] = pE_1(w_1, i, j) + (1-p)E_2(w_1, i, j). \quad \text{(B.3)}$$

Appendix C. No DA, possible single TO: $Y_i + (w_1+1)/\mu < T_j < Y_i + T_i$

In this case a single TO may take place if the RTT transition takes place during the idle period of the round, $K_I = 1$ and w_{r_I} is given by (9). Otherwise, TCP continues in the congestion avoidance phase. Hence,

$$\mathbf{P} = \mathbf{H}_{i,j}\cdot\begin{cases} 0, & \text{if } w_2 < w_{r_I};\\ pF_j\big(T_j(w_2-w_{r_I}), T_j\big), & \\ \quad \text{if } w_{r_I} < w_2 < w_1;& \\ pF_j\big(T_j(w_2-w_{r_I}), T_j\big) & \\ \quad +(1-p)F_j\big(T_j(w_2-w_1), T_j\big), & \\ \quad \text{if } w_{r_I} < w_2 < w_{\mathrm{m}};& \\ p\overline{F_j}\big(T_j(w_2-w_{r_I})\big) & \\ \quad +(1-p)\overline{F_j}\big(T_j(w_2-w_1)\big), & \\ \quad \text{if } w_2 = w_{\mathrm{m}}. & \end{cases} \quad \text{(C.1)}$$

Let $E_1(w_1, i, j)$ denote $E[N|w_1, i, j]$ in case of no TO. Then

$$\begin{aligned} &E_1(w_1, i, j)\\ &\quad=\sum_{w=w_1}^{w_{\mathrm{m}-1}}\sum_{g=0}^{w}\overline{F_j}\left(\frac{g}{\mu}+T_j(w-w_1)\right)\\ &\qquad+\sum_{j=0}^{\infty}\sum_{g=1}^{w_{\mathrm{m}}}\overline{F_j}\left(\frac{g}{\mu}+T_j(w_{\mathrm{m}}-w_1)+jT_j\right). \quad \text{(C.2)}\end{aligned}$$

The reward in the case of a TO is identical to (C.2) except that w_1 is replaced by w_{r_I}. Hence,

$$\begin{aligned} &E[N|w_1, i, j]\\ &\quad= E_1(w_1, i, j)(1-p) + E_1(w_{r_I}, i, j)p. \quad \text{(C.3)}\end{aligned}$$

Observe that the above expressions reduce to sums of geometric sequences and can be obtained in closed form.

References

[1] A. Abouzeid and S. Roy, TCP in networks with abrupt delay variations and random loss, in: *Proceedings of MILCOM'2001* (2001).

[2] M. Allman, J. Griner and A. Richard, TCP behavior in networks with dynamic propagation delay, in: *Proceedings of Globecom 2000* (November 2000).

[3] M. Allman and V. Paxson, On estimating end-to-end network path properties, in: *Proceedings of ACM SIGCOMM'99* (September 1999).

[4] J.C.R. Bennett, C. Partridge and N. Shectman, Packet reordering is not pathological network behavior, IEEE/ACM Transactions on Networking 7(6) (1999) 789–798.

[5] S. Floyd, A report on some recent developments in TCP congestion control, IEEE Communications Magazine 39(4) (2001) 84–90.

[6] S. Floyd, J. Madhavi, M. Mathis and M. Podolsky, An extension to the selective acknowledgement (SACK) option for TCP, Technical Report RFC 2883 (July 2000).

[7] R. Gallager, *Discrete Stochastic Processes* (Kluwer, Boston, MA, 1996).

[8] T.R. Henderson and R.H. Katz, Network simulation for LEO satellite networks, in: *Proceedings of 18th AIAA International Communications Satellite Systems Conference (ICSSC)* (April 2000).

[9] V. Jacobson, Congestion avoidance and control, in: *Proceedings of ACM SIGCOMM'88* (1988) pp. 314–329.

[10] V. Jacobson, R. Braden and D. Borman, TCP extensions for high performance, Technical Report RFC 1323 (May 1992).

[11] R. Leopold and A. Miller, The IRIDIUM communications system, IEEE Potentials 12(2) (1993) 6–9.

[12] R. Ludwig and R.H. Katz, The Eifel algorithm: Making TCP robust against spurious retransmissions, Computer Communications Review 30(1) (2000) 30–36.

[13] ns-Network Simulator (1995), `http://www.isi.edu/nsnam/ns/`

[14] V. Paxson, End-to-end routing behavior in the Internet, IEEE/ACM Transactions on Networking 5(5) (1997) 601–615.

[15] J. Postel, Transmission control protocol, RFC 793, IETF (September 1997).

[16] S. Ross, *Stochastic Processes*, 2nd ed. (Wiley, New York, 1996).

[17] M. Sturza, The Teledesic satellite system, in: *Proceedings of 1994 IEEE National Telesystems Conference* (1994) pp. 123–126.

Alhussein A. Abouzeid received the B.S. degree from Cairo University, Cairo, Egypt, in 1993 and the M.S. and Ph.D. degrees from University of Washington, Seattle, WA, in 1999 and 2001, respectively, all in electrical engineering. He is currently an Assistant Professor in the Department of Electrical, Computer and Systems Engineering (ECSE), Rensselaer Polytechnic Institute, Troy, NY. The industry interactions during his Ph.D. study included working with Microsoft Research, Redmond, WA (Ph.D. topic), Honeywell, Bellevue, WA (techniques for wireless access to airplanes), and at Hughes Research Labs, Malibu, CA (efficient networking protocol simulations for broadband satellite systems). He was a System Engineer with Alcatel Business Systems, Middle East Regional Office, Cairo, from 1994 to 1997 and a Software Engineer at the Information Technology Institute, Cairo, Egypt, during 1993.

E-mail: abouzeid@ecse.rpi.edu

Sumit Roy received the B.Tech. degree from the Indian Institute of Technology (Kanpur) in 1983, and the M.S. and Ph.D. degrees from the University of California (Santa Barbara), all in electrical engineering in 1985 and 1988, respectively, as well as an M.A. in statistics and applied probability in 1988. His previous academic appointments were at the Moore School of electrical engineering, University of Pennsylvania, and at the University of Texas, San Antonio. He is presently Professor of Electrical Engineering, University of Washington where his research interests center around analysis/design of communication systems/networks, with a topical emphasis on next generation mobile/wireless networks. He is currently on academic leave at Intel Wireless Technology Lab working on high speed UWB radios. His activities for the IEEE Communications Society includes membership of several technical committees and TPC committees for conferences, and he presently serves as an Editor for the IEEE Transactions on Wireless Communications.

E-mail: roy@ee.washington.edu