The Window Distribution of Idealized TCP Congestion Avoidance with Variable Packet Loss

Archan Misra Teunis J Ott
archan@bellcore.com tjo@bellcore.com
Bell Communications Research
445 South Street
Morristown, NJ 07960

Abstract—**This paper analyzes the stationary behavior of the TCP congestion window performing ideal congestion avoidance when the packet loss probability is not constant, but varies as a function of the window size. By neglecting the detailed window behavior during fast recovery, we are able to derive a Markov process that is then approximated by a continuous-time, continuous state-space process. The stationary distribution of this process is analyzed and derived numerically and then extrapolated to obtain the stationary distribution of the TCP window. This numerical analysis enables us to predict the behavior of the TCP congestion window when interacting with a router port performing Early Random Drop (or Random Early Detection) where the loss probability varies with the queue occupancy.**

Keywords—**TCP, distribution, variable, loss.**

I. Introduction

In this paper, we present a quantitative analysis of the stationary behavior of the evolution of the TCP congestion window ($cwnd$) ([1]) when the packet loss probability is variable and depends on the (instantaneous) window of the TCP connection. It can, thus, be viewed as a generalization of the analysis in ([2]) where the drop probability was assumed constant. The mathematical model abstracts TCP behavior into a continuous cycle of "congestion avoidance", packet loss and "fast recovery". We disregard the details of fast recovery ([7]) of TCP and assume an idealized behavior, whereby a packet loss that occurs when the congestion window is n MSSs *instantaneously* reduces the congestion window (and the number of unacknowledged packets) to $\lceil \frac{n}{2} \rceil$ MSSs. The dynamics of window evolution can then be captured by a discrete-time Markov process with state-dependent conditional transition probabilities.

Mathematically speaking, we consider the stochastic process $(W_n)_{n=1}^{\infty}$, where W_n stands for the congestion window just after the n^{th} *good* acknowledgement packet (one that advances the left marker of TCP's sliding window) has arrived at the source. By disregarding time-outs and the behavior during fast recovery, this is a discrete-time Markov process with the following behavior:

$$P\{W_{n+1} = w + \frac{1}{w} | W_n = w\} = 1 - p(w) \qquad (1.1)$$

$$P\{W_{n+1} = \frac{w}{2} | W_n = w\} = p(w). \qquad (1.2)$$

where $p(w)$ is the packet loss probability when the congestion window is w.(The time index n in the above equations is referred to as *ack time* in this paper, since it increases only with the receipt of acknowledgements.) Let the maximum value of this loss probability, over all values of w, be denoted by p_{max} (hence $p_{max} \leq 1$). Our Markovian formulation holds when, given the current window size, packet losses are conditionally independent of past and future losses. As in ([2]), we will approximate this process by a more amenable continuous-space continuous-time process.

In ([2]), which assumed a constant loss probability p, the time axis was rescaled by a linear contraction with scale p, and space was rescaled by a linear contraction with scale $\sqrt{p}$, resulting in an effective rescaling given by $W(t) = \sqrt{p} W_{\lfloor \frac{t}{p} \rfloor}$. (The time index generated by the rescaling is called *subjective time* in this paper.) The resulting analysis derived the well-known 'square-root' behavior of TCP ([2],[9]): the average window of a persistent TCP connection is of the order of $\frac{1}{\sqrt{p}}$. We shall also engage in similar rescalings in this paper. While our space rescaling is still linear, the variable loss probability of our model requires the time rescaling to be non-linear, as explained in Section 2. The window evolution of this re-scaled process will be described by a differential equation (between events of packet loss); the intervals between these packet loss events will be shown to be independent and exponentially distributed random variables. This differential equation is then solved via numerical analysis; the stationary distribution of the TCP congestion window is approximated from this continuous process by appropriate corrections for the rescaling.

We verified the accuracy of our analysis by comparing our predictions with simulations which involved popular TCP versions (NewReno and Reno) and where packets were randomly dropped with a state-dependent loss probability. We also applied this model to predict the window distribution of a persistent TCP connection that interacts with a router port, which handles this single flow and performs *Early Random Drop* (ERD) ([10])). In an ERD port, the drop probability is a function of the *instantaneous* buffer occupancy (which we related to the instantaneous congestion window); we observed very good agreement between simulations results and analytical predictions. We also investigated the applicability of this model when the router port performed *Random Early Detection* (RED) ([11]), so that the loss probability is a function of the *average* queue length. Although our memoryless state-dependent Markovian loss model does not accurately reflect the complicated effect of queue averaging in RED, ([11]), surprisingly good predictions were obtained.

The paper is organized as follows. In section 2, we model the Markov process characterizing the TCP behavior and approximate its dynamics by an appropriate continuous-time,

This research was partially supported by NSF Grant NCR-9415552, in subcontract with the Pittsburgh Supercomputing Center.

continuous-valued process. In section 3, we derive the Kolmogorov differential equation governing the stationary behavior of the generalized process and present a numerical solution to this differential equation. In section 4, we provide numerical examples analyzing the window behavior of TCP with Early Random Drop and Random Early Detection queues and evaluate the effectiveness of our numerical techniques in predicting TCP behavior.

A. Related Work and Model Applicability

There has been a fairly large body of literature analyzing TCP window dynamics with constant drop probabilities. The square-root formula, which ignores the effects of TCP timeouts and fast recovery, has been rigorously derived in [2] and has also been separately reported through simpler analyses in [6] and [5] (the last publication also considers modifications to the formula resulting from losses of acknowledgement packets). Ott [8] stated that the throughput of persistent TCP (but not necessarily the congestion window) starts behaving like $\frac{1}{p}$, when timeouts become significant. The recent work by Padhye et al [3] provides a better estimate of throughput (especially at larger loss probabilities) by considering the effects of fast recovery and timeouts in greater detail; they also asserted the $\frac{1}{p}$ dependence when timeouts become significant. Kumar [4]) has presented an elaborate analysis of the performance of different TCP versions by considering the fast recovery and timeout dynamics of each version in great detail. All these papers, however, assume a constant drop probability p; our paper differs from these approaches in that it considers the case where the loss probability varies as a function of the window size.

We would also like to add a few words about the range of loss probabilities over which our analysis applies. Both [3] and [4] have noted the importance of timeouts in analyzing the performance of TCP versions like Tahoe, Reno and New Reno; our model, on the other hand, does not consider timeouts at all. Our analysis is accurate for such TCP versions only when loss probabilities are small enough and the delay-bandwidth product (including the buffering delay) high enough (approx. 10 and above) to ensure that timeouts are relatively rare events. The disproportionate impact of timeouts on TCP performance is due to the combined effects of coarse-grained timers and the integration of loss recovery mechanisms with congestion control in current TCP versions. When loss recovery is separated from adaptation to congestion (as in SACK TCP), timeouts begin to play a relatively less important role and the range of loss probabilities over which our analysis holds increases. As newer TCP implementations like SACK TCP and random drop mechanisms like RED become widespread in the Internet, our assumptions will become less restrictive.

II. Model Description

The TCP connection is assumed to send a large data file in the forward direction with the congestion window acting as the only constraint on the transmission of packets. It is assumed that the connection never goes into timeout, that the receive or advertized window never limits the number of unacknowledged packets, that data is always sent in equal-sized segments (one MSS) and that acknowledgements are never lost. The receiver generates an acknowledgement for every received packet (we shall also extend the analysis to model the phenomenon of delayed acknowledgements). Packet losses are assumed to be conditionally independent.

The stochastic process described by the conditional probabilities in equations (1.1) & (1.2) applies to TCP only if the time index corresponds to the arrival of *good acks*. This time, which we shall call *ack time* in the rest of the paper, is a positive-integer valued variable that increments by 1 whenever a good acknowledgement packet arrives at the source; it increases linearly with clock time only when the window size and round trip times are both constant. Let the cumulative probability stationary distribution for this process under this ack time be F_{ack}.

A. Time and State-space Rescaling

To derive a more amenable continuous-time continuous-valued random process from the process described by equations (1.1) & (1.2), we rescale both the time and state-space axes. This leads us to introduce the concept of *subjective time*, which is, roughly speaking, related to ack time through an invertible mapping. For the case considered in [2], where the loss probability was a constant p, the subjective time was derived from ack time by *linearly compressing* the time scale by a factor p, by using the relation $dt_{subjective} = p.dt_{ack}$. When the loss probability is not constant but state-dependent, a state-dependent (non-linear) scaling must be used.

We now present a generalized notion of subjective time by considering a continuous time stochastic process, $X(t)$ with a state-dependent failure rate $\lambda(x)$. We can now derive another process $Y(\tau)$ from $X(t)$ such that an increase of dt in the time index t of $X(t)$ corresponds to an increment of $\lambda(X(t))dt$ in the time index τ of $Y(\tau)$. A realization of the process Y will thus assume the same state-space values as the corresponding realization of X but at *different instants of time*. Subjective time can also be though of as a *history-and-state dependent rescaling* of the base (ack) time index[1]. **The importance of the process $Y(\tau)$ lies in the fact that $Y(\tau)$ will now have a constant failure rate in its own notion of time (proved in Appendix I).** The time index, τ, of the process $Y(\tau)$ is then known as *subjective time* in reference to the time index t of the process $X(t)$ and the two are related by the differential relation

$$d\tau = \lambda(X(t))dt \tag{2.1}$$

Subjective time can also be considered to be a variable stretching (or contraction) of the time index.

For the specific TCP process under consideration, our quantized increment in subjective time t is provided by the mapping

$$\Delta t \;=\; p(W_n)\Delta n \tag{2.2}$$

where Δt is the (real-valued) increment in subjective time, Δn is the (integer-valued) increment in ack time and $p(W_n)$ is the loss probability associated with the value of the window W_n at ack time n. *In other words, for a process defined under this*

[1]Readers familiar with Weighted Fair Queuing (WFQ) ([12]) may benefit from realizing that our subjective time formulation is analogous to that the definition of *virtual time* in WFQ; both attempt a state-dependent rescaling of time so that the process of interest has an invariant behavior in the new time scale.

subjective time, time advances at a variable rate, as an increase in the ack time index of 1 *corresponds to a state-dependent increase of* $p(W_n)$ *in the subjective time index.* Thus, $t(N)$, the subjective time immediately after sending packet number N, is expressed as $t(N) = \sum_{i=1}^{N} p(W_i)$. As $0 \leq p(W_n) \leq 1$, t is a real-valued sequence obtained by a contraction of the ack time index. As $p_{max} \downarrow 0$, the limiting subjective time index becomes a continuous variable. We shall see that, for this specific case, the process defined in subjective time has a failure rate that becomes Poisson and constant only asymptotically, as $p_{max} \downarrow 0$.

If $W'(t)$ represents the process W_n in subjective time t via the transformation in equation (2.2), its sample path between the events of packet failure can be modeled by the difference equation

$$\frac{\Delta W'}{\Delta t} = \frac{1}{p(W')W'} \tag{2.3}$$

As $p_{max} \downarrow 0$, the difference equation can be modeled by a corresponding differential equation with increasing accuracy. The differential equation would however, in the limit, be ill-behaved as the derivative goes to ∞ as $p_{max} \downarrow 0$. To obtain a well behaved process, we also need to rescale the state space of $W'(t)$. To rescale properly, we assume that $\frac{p(W)}{p_{max}} > \epsilon \ \forall W$ [2], (i.e., the ratio between the minimum and maximum loss probabilities is uniformly bounded away from 0). If we then rescale the state-space of the process $W'(t)$ by the multiplicative constant $\sqrt{p_{max}}$, the resulting process, which we call $W(t)$, obeys the functional relationship

$$W(t) = \sqrt{p_{max}}\, W_n \tag{2.4}$$

$$where\ n = n(t) = arg\ max\ j : \sum_{i=0}^{j} p(W_i) \leq t$$

This continuous-time and continuous valued process $W(t)$ will be the subject of our study and analysis for the rest of the paper.

Equation (2.2) implies that a loss probability of zero ($p(W_n) = 0$) results in *an increase in ack time but no increase in subjective time*. Subjective time thus loses information about the process behavior during those ack times when the system evolves deterministically without loss; the mapping in equation (2.2) is non-invertible if $p(W_n)$ is 0. Later in the paper, we shall see how to correct the stationary distribution of the process for portions of the state-space where the loss probability is 0; for the time being, assume that $p(W_n) \neq 0$ in the region of interest.

Proposition 1

We see that as $p_{max} \downarrow 0$, *the process defined by equations (1.1),(1.2) & (2.4), converges (path-wise) to a process whose window,* $W(t)$, *behaves as follows:*

There is a Poisson process with intensity 1**, the points of which are denoted by** $(\tau_n)_{n=1}^{\infty}$**. In between the points of this Poisson process, the window,** W**, evolves according to the equation**

$$\frac{dW}{dt} = \frac{p_{max}}{p(\frac{W}{\sqrt{p_{max}}})W} \tag{2.5}$$

At the points of the realization of the Poisson process, we have $W(\tau^+) = \frac{1}{2}W(\tau^-)$**.**

Proof:

The proof of the differential equation describing the window evolution between failure events is trivial and obtained by taking appropriate limits in equations (1.1), (2.2) & (2.3). The relationship at an instant of failure also follows easily from equations (1.2) and (2.3). Note that the derivative in equation (2.5) is always well-defined by virtue of our assumption that $p(W_N) > 0$ for the interval under consideration. The proof that the instants of failure become a realization of a Poisson process of intensity 1 is provided in Appendix I. It consists of showing that as $p_{max} \downarrow 0$, the number of packet transmission events in any finite interval T becomes infinitely large and the probability of loss of each transmitted packet is such that $Prob(\ no\ loss\ in\ an\ interval\ T) = e^{-T}$.

The process defined in Proposition 1 is an approximation of the re-scaled 'TCP' process; the approximation becomes asymptotically accurate as the loss probabilities become smaller. For a given loss probability function $p(W)$, we analyze the rescaled 'TCP' process by assuming that it exhibits the behavior of the limiting process outlined by Proposition 1. In other words, even for a finite loss probability, we assume that $W(t)$ is described by the differential equation (2.5), with an i.i.d and exponential distribution of times between packet drops. We can thus expect the numerical analysis outlined later to predict TCP window behavior more accurately as p_{max} becomes smaller.

B. Distribution in (Continuous) Ack Time

We shall see how to compute $F_{subj}(w)$, the stationary cumulative distribution of $W(t)$ in subjective time, later in section III. We now consider how to correct this distribution for the state-space and time rescalings, introduced in equation (2.4), assuming $F_{subj}(w)$ is already known.

The state-space scaling results in a simple linear transformation of the probability distribution. $F_{subj}(w)$ is corrected first to obtain $F_s(w)$, the cumulative stationary distribution in subjective time but without space-rescaling by the relationship $F_s(w) = F_{subj}(\sqrt{p_{max}}w)$.

Our desired distribution $F_{ack}(w)$ can then be obtained by noting that the state-dependent rescaling of subjective time (in equation (2.2)) introduces a sampling non-uniformity in the process $W(t)$. To see this non-uniformity, note that an acknowledgement arriving when the window is w occupies an interval of 1 in ack time but corresponds to an interval $p(w)$ in subjective time: a uniformly distributed sampling on the subjective time axis corresponds to a non-uniform sampling (with non-uniformity proportional to $p(w)$) in the ack time frame.

The sampling non-uniformity due to time-scaling is corrected, to obtain $F_{ack}(w)$, by dividing the probability density in subjective time, $dF_s(w)$, by the appropriate quantity $p(w)$. This is

[2]The above requirement may, in several cases, be more stringent than practically necessary. For example, when the variation of loss probability with window size is very gradual, the bulk of the distribution mass will often lie around some small value of p, say p^*. We can then use $\sqrt{p^*}$ as our space-rescaling factor; for the rescaled process to be well-behaved, we then merely need $\frac{p(W)}{p^*}$ to be bounded away from 0.

achieved by the transformation

$$dF_{ack}(w) = \frac{\frac{dF_s(w)}{p(w)}}{\int \frac{dF_s(\eta)}{p(\eta)}} \tag{2.6}$$

C. A Generalized Process

The analysis used to derive the stationary distribution of $W(t)$ is applicable to a more general class of processes. For example, any arbitrary process with a state-dependent failure rate can be reduced to a process with a constant Poisson failure rate by moving to an appropriate subjective time. Thus, we do not lose generality by considering only processes with constant failure rates.

Consider a general process $W(t)$, described by the differential equation

$$\frac{dW}{dt} = \frac{1}{q(W)} \tag{2.7}$$

in between the instants of failure of a Poisson process with rate λ; let q be a well-behaved function (finitely many discontinuities) such that $q(w) > 0 \, \forall \, w$. At the instants of failure of the Poisson process, the process evolution is given by $W(t^+) = A(W(t^-))$, where $A(w) : [0, \infty) \rightarrow [0, \infty)$ is a strictly increasing function of w such that $A(w) < w$, $\forall w > 0$, $A(0) = 0$. Since A is strictly increasing, it has an inverse function $a(w)$, such that $a(A(w)) = w$ and $a(w) > w$, $\forall w > 0$. The analysis presented later can be easily extended to consider this whole class of processes.

For the TCP-specific case at hand, we have $A(w) = \frac{1}{2}w$ (so that $a(x) = 2x$), the intensity λ of the Poisson process is 1 and the rate function $q(W) = \frac{p(\frac{W}{\sqrt{p_{max}}})W}{p_{max}}$.

In the next section, we shall formulate and solve the Kolmogorov equation for this generalized process. Our numerical examples will, however, solely deal with the TCP-specific process for simplicity.

III. The Stationary Kolmogorov Equation and its Solution

In this section we obtain the stationary distribution of the process, defined in section II, whose behavior is described by the equation $\frac{dW(t)}{dt} = \frac{1}{q(W(t))}$ in between the points of a Poisson process of rate λ. At the points of the Poisson process, $W(t)$ is obtained by $W(t^+) = A(W(t^-))$; let $a(x)$ be the inverse function of $A(x)$.

Proposition 2
The stationary cumulative distribution $F_{subj}(x)$ of the process in section II.C satisfies the differential equation

$$\frac{dF_{subj}(x)}{dx} = \lambda q(x)(F_{subj}(a(x)) - F_{subj}(x)) \tag{3.1}$$

Proof:
If $F_{subj}(x,t)$ is the cumulative distribution function at (subjective) time t, then the distributions at times t and $t + \Delta t$ can be related as

$$F_{subj}(x + \frac{\Delta t}{q(x)}, t + \Delta t) =$$
$$F_{subj}(x,t) + \lambda \Delta t (F_{subj}(a(x)) - F_{subj}(x))$$

The first term in the RHS of the above equation asserts that the process cannot increase by more than $\frac{\Delta t}{q(x)}$ in an interval of time Δt while the second term considers the probability of loss events that would cause the process value to reduce below x at time $t + \Delta t$. Since the stationary distribution $F_{subj}(x)$ is invariant in t, we get the resulting differential equation

$$\frac{dF_{subj}(x)}{dx} = \lambda q(x)(F_{subj}(a(x)) - F_{subj}(x)) \tag{3.2}$$

We were unable to obtain a closed form analytical solution for this differential equation. We however provide an open-form analytical expression for $F_{subj}(x)$ that translates into a rapidly converging numerical technique for evaluating the cumulative distribution. In passing, we note that the approximation of the TCP process results in the differential equation

$$\frac{dF_{subj}(x)}{dx} = q(x)(F_{subj}(2x) - F_{subj}(x)) \tag{3.3}$$

which will be used in the numerical tests to be presented later.

A. Solution of the Equation

Let G be the complementary distribution function defined by the relation $G(x) = 1 - F_{subj}(x)$. Equation (3.1) is equivalent to the equation

$$\frac{dG(x)}{dx} + \lambda q(x) G(x) = \lambda q(x) G(a(x)) \tag{3.4}$$

with the boundary conditions $G(0) = 1$, $G(\infty) = 0$. Let $Q(x) = \int_0^x \lambda q(u) du$ and define $G(x) = H(x) e^{-Q(x)}$ where $H(x)$ is an arbitrary function (to be evaluated). $H(x)$ is then seen to obey the differential equation

$$H(x) = H(z) - \lambda \int_x^z q(u) e^{Q(u)} G(a(u)) du \tag{3.5}$$

Now, suppose that $\lim_{x \uparrow \infty} H(x)$ exists and is equal to $\bar{H}$. $\bar{H}$ will exist only if the tail of the complementary distribution decays as $e^{-Q(x)}$. By evaluating the behavior of equation (3.4) for very large x (where $G(a(x))$ can be considered to be 0 with negligible error), we can easily see that this phenomenon of exponential decay is indeed true. Now, by letting $z \uparrow \infty$ in equation (3.5) and noting that $G(a(u)) = e^{-Q(a(u))} H(a(u))$, we have

$$H(x) = \bar{H} - \lambda \int_x^\infty q(u) e^{(Q(u) - Q(a(u)))} H(a(u)) du \tag{3.6}$$

with the boundary conditions $H(0) = 1$ and $H(\infty) = \bar{H}$.

By defining $J(u)$ as $J(u) = \lambda q(u) e^{Q(u) - Q(a(u))} = \lambda q(u) e^{-\int_u^{a(u)} q(\rho) d\rho}$, equation (3.6) reduces to

$$H(x) = \bar{H} - \int_x^\infty J(u) H(a(u)) du \tag{3.7}$$

By iterated expansion, $H(x)$ can be shown to obey the relation

$$H(x) = \bar{H} \sum_{k=0}^{\infty} (-1)^k \overbrace{\int_{u_1 > x} \cdots \int_{u_k > \beta u_{k-1}}}^{k-fold}$$

$$J(u_1)\ldots J(u_k)du_k\ldots du_1 \tag{3.8}$$

Appendix III provides a proof that the above infinite sum indeed converges to a limit when the function $q(x)$ is non-decreasing in x; this condition holds for the TCP process whenever the drop probability is a non-decreasing function of the window size.

B. Numerical Computation

Repeated substitution in equation (3.7) offers a numerical technique for evaluating $H(x)$. As $H(x)$ tends to a limit as $x \uparrow \infty$, it can be treated as a constant beyond a certain value x_{upper} (chosen such that the resulting error in computing $H(x)$ is at most a small value ϵ). We can then obtain an approximation for $H(x)$ by setting the value of $H(x)$ beyond x_{upper} to be a constant and computing $H(x)$ between $(0, x_{upper})$. After the algorithm converges, we can divide by $H(0)$ to satisfy the boundary conditions $H(0) = 1, H(\infty) = \bar{H}$.

The complete numerical procedure for computing $F_{subj}(x)$ is as follows:

1. Choose a small positive constant ϵ ($\epsilon > 0$), which indicates the accuracy of the computation.
2. Find x_{upper} such that $\int_{x_{upper}}^{\infty} J(u)\,du \leq \epsilon$.
3. Let $B_0(x) = 1$ for all x and let $B_i(x) = 1, \quad \forall x > x_{upper}, \ \forall\, i$.
4. Also compute $K(x) = \int_x^{\infty} J(u)\,du$ for $A(x_{upper}) \leq x \leq x_{upper}$. Denote $K(A(x_{upper}))$ by ζ.
5. For all values of i, let $B_i(x) = 1 - K(x)$, for $A(x_{upper}) \leq x \leq x_{upper}$.
6. Repeat the following iteration in the range $(0, A(x_{upper}))$ until the function converges below a specified bound:

$$B_i(x) = 1 - \int_x^{A(x_{upper})} J(u)\ B_{i-1}(\beta u)\ du\ -\ \zeta.$$

7. Let the final solution be denoted by $B(x)$.
8. Renormalize $B(x) = \frac{B(x)}{B(0)}$ to satisfy the necessary boundary conditions. $B(x)$ is then the numerical estimate for $H(x)$.
9. The complementary probability distribution is then obtained as
$$G(x) = B(x)e^{-Q(x)} \tag{3.9}$$
10. Compute $F_{subj}(x)$ from $F_{subj}(x) = 1 - G(x)$.

C. Correcting for Lossless Evolution

As noted in section II.A, the rescaled TCP process in subjective time cannot capture the dynamics of the window evolution when the loss probability is 0 (as subjective time freezes during these epochs). From a sample path point of view, the infinite derivative in equation (2.5) (Proposition 1) and the zero time increment in equation (2.2) imply that whenever the TCP process (in subjective time) enters an interval in the state-space corresponding to 0 loss, it instantaneously jumps from the lower to the upper end of the interval. In this subsection, we show how $F_{ack}(x)$ for the TCP process, obtained from the mapping in equation (2.6), can be corrected to incorporate the dynamics of the lossless evolution; the corresponding correction for the generalized process is then straightforward.

The correction for the density $f_{ack}(x)$ in ack time (after the correction for state-space rescaling has been completed) is computed by the *level crossing principle* which equates the rate at which the process evolves to the right of a value x to the rate at which the process transitions to the left. For the TCP process, this results in the equality

$$f_{ack}(x)\frac{1}{x} = \int_x^{2x} p(u)dF_{ack}(u). \tag{3.10}$$

This follows by noting that at a point x, TCP evolves to the right at a rate $\frac{1}{x}$ while it moves to the left at the rate governed by the loss rate in the interval $(x, 2x)$. By first obtaining the values of $F_{ack}(x)$ (upto a scaling constant) in the regions with non-zero loss probability, we can correct the solution for regions with zero loss probability using the equation (3.10)[3]. (If $F_{ack}(u)$ in the RHS of equation (3.10) is unknown for any u, it follows that $p(u) = 0$ also; the unknown region may thus be left out of the computation.)

The numerical recipe for correcting the distribution for the lossless region is.

1. For the region(s) where $p(x) = 0$, compute the density $f_{ack}(x)$ using the level crossing relation
$$f_{ack}(x) = x\int_x^{2x} p(u)f_{ack}(u)du \tag{3.11}$$
2. Renormalize $f_{ack}(x)$ by $\int f_{ack}(u)du$ over the entire state-space to ensure a well formed probability distribution function $f_{ack}(x)$.

IV. Simulations and Results

We now present numerical examples to compare our analytical results with those obtained via simulations. The simulations were carried out with the TCP Reno and NewReno versions in the *ns* simulator package. Although these versions differ in their fast recovery mechanisms and in the frequency of timeouts, the performance of the two versions was found to be almost identical for the relatively low loss environments studied in our simulations. To obtain adequate statistical confidence, simulation results were obtained by averaging over runs with multiple seeds; each run comprised at least 10^6 packet transmissions. While the entire simulation process would take $\sim 10-15$ minutes, the numerical computation over a fairly fine grid ($\sim$ 1000 points) took only about 30 secs (on a typical workstation).

A. TCP with Simple State-Dependent Loss

The results in Fig.1 correspond to the case when the packet drop probability depends directly on the window size. We

[3]The level-crossing equation (3.10) is actually valid for the entire range of the TCP process state-space (and not just where $p(x)$ is 0). It can easily be seen that the equation (3.3) (in subjective time) is equivalent to equation (3.10) (in ack time) by noting that the following set of relations: $p(x)dF_{ack}(x) = K.dF_s(x)$; $f_{ack}(x) = \frac{K.f_s(x)}{p(x)}$; $F_s(x) = F_{subj}(\sqrt{p_{max}}x)$; and $f_s(x) = \sqrt{p_{max}}f_{subj}(\sqrt{p_{max}}x)$, where K is a normalizing constant. The elaborate rescalings and computations in subjective time in this paper are necessary simply because there does not appear to be a simple way of solving equation (3.10) directly over the entire state space! Another way of looking at the subjective time formulation is therefore to think of it merely as a change of measure that replaces equation (3.10) with the more tractable version in equation (3.3).

achieve this effect by passing a TCP connection through a single queue with negligible link propagation and transmission delay (all outstanding packets are thus effectively resident in the queue), and independently dropping each arriving packet with a probability that varies with the queue occupancy. The drop probability in this example increases linearly with queue occupancy. It can be seen that the simulated behavior offers excellent agreement with the numerical prediction in this example. For comparison purposes, we include the distribution predicted by the 'square-root formula' in ([2]) assuming a constant drop probability; the constant value of the drop probability was taken to be the drop probability corresponding to the mean TCP window size obtained via simulation. As expected, the 'square-root' approximation predicts a much larger variation in the window size than the true distribution.

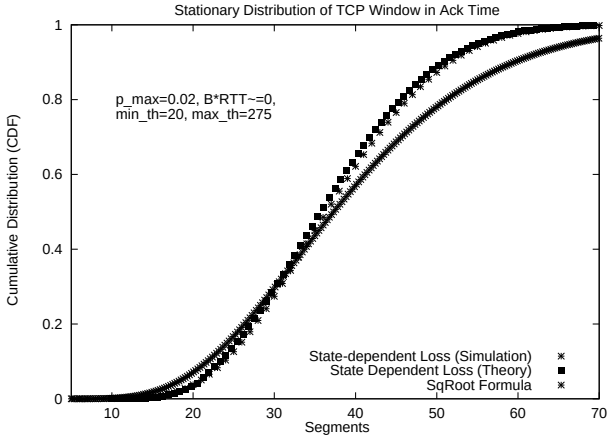

Figure 1: TCP Window Evolution and State-Dependent Loss

B. Predicting TCP behavior with Queue Management Techniques

One of the goals of our analysis is to predict the window distribution of a persistent TCP flow when it interacts with router queue management mechanisms like *Early Random Drop* (**ERD**) and *Random Early Detection* (**RED**), where the packet drop probability is not constant but varies with the queue occupancy. In the present paper, we only model the simple case where the persistent TCP connection is the only flow through the router port; we are currently investigating extensions to multiple TCP flows. While both ERD and RED involve variable drop probabilities that depend on the queue occupancy, they have significant differences (discussed in Appendix II), of which the two most important are:

- The drop probability in RED is a function of the moving-averaged queue occupancy while the drop probability in ERD is a function of the instantaneous queue length.
- Even if he queue occupancy remains constant, RED increases the drop probability for every accepted packet (which we call *drop biasing*) resulting in inter-drop gaps that are uniformly generated; ERD drops each packet with the same drop probability, resulting in inter-drop gaps that are geometrically distributed.

These differences make RED much harder to model than ERD: the use of averaged queue occupancies to determine drop probabilities destroys the state-dependent loss model (the drop probability is then a function of the past state behavior) while the drop biasing functionality negates the independent packet drop assumption. We circumvent these problem by (simplistically) assuming that the drop probability depends only on the instantaneous queue length and that each packet is dropped independently. We thus ignore the effect of queue averaging in RED; we shall however present a simple correction to account for the effect of drop biasing.

B.1 Relating the Loss Probability to Queue Occupancy

As already stated, we assume that the loss probability is determined by the instantaneous queue occupancy (for both RED and ERD); the loss probability for a given TCP window is derived by relating the queue occupancy to the TCP window. Neglecting the periods of fast recovery, the number of unacknowledged packets in flight, when the window is W_n, equals $\lfloor W_n \rfloor$, or in an approximate sense, W_n. If B (pkts/sec) is the service rate of the (bottleneck) queue and the round-trip delay (ignoring the queuing delay) is RTT (sec), then $B.RTT$ packets are necessary to fill the transmission pipe. *Assuming that this pipe is always full*[4], the occupancy of the queue is given by the residual number of unacknowledged packets, so that we have

$$Q_n = W_n - B.RTT \tag{4.1}$$

For our experiments, the loss function is given by the traditional model of RED behavior, i.e., $p(Q) = 0$ for $Q \leq min_{th}$, $p(Q) = p_{max}$ for $Q \geq max_{th}$ and $p(Q) = \frac{Q - min_{th}}{max_{th} - min_{th}} p_{max}$ for $min_{th} < Q < max_{th}$. The loss probability as a function of the window size is then given by $p(W - B.RTT)$[5].

While the above model cannot capture the queue averaging function of RED, we can make a simple correction to approximate the effect of drop biasing in our model. We note that for a given value of drop probability p, the uniform distribution of inter-drop gaps in RED implies that the average gap is $\frac{1}{2p}$; the geometric distribution of gaps (resulting from an independent loss model) implies an average gap of $\frac{1}{p}$. For the RED simulations, we accordingly modify our analytical drop function such that our average agrees with that of RED, i.e., for a given queue occupancy q, we make $p_{model}(q) = 2p_{red}(q)$.

B.2 Experimental Results

Illustrative results of our validation experiments are provided in figures 2 and 3, which plot the numerically predicted cumulative distribution of the TCP window against that obtained from simulations. Figure 2 shows that our numerical analysis provides an excellent match with simulation when the queue implements the ERD algorithm. The distribution predicted by the square-root formula is also provided for comparison. Figure 3 consists of two graphs, the top one for a RED queue with $B.RTT \approx 0$

[4]This assumption holds only if the buffer never underflows (which, in turn, can hold only if the time taken by the buffer to drain min_{th} packets is longer than RTT.

[5]The reader will note that the ack arriving at the source at *time* n (when the window is W_n) corresponds to a packet generated a round-trip time earlier when the window was $W_{n'}$; the loss probability of the packet acked at n should thus be $p(W_{n'})$. However, as $cwnd$ increases by a maximum of 1 segment in a round-trip time $W_{n'} \approx W_n$, so that the loss probability of the packet acked at n can be assumed to be $p(W_n)$ with negligible error.

and the bottom one with $B.RTT = 5$. The top graph isolates the effect of approximating the RED averaging process from the performance obtained when this approximation is combined with the assumption of a full pipe (equation (4.1)). The two graphs show, somewhat surprisingly, that the numerical predictions (with the correction for drop biasing) provide fairly close agreement with the simulated distribution when the queue implements RED. The closeness of the fit is somewhat unexpected since the averaging effect in RED queues typically last over 500 packets; we expected this memory to significantly degrade the accuracy of our modeling.

C. Incorporating Delayed Acknowledgements

Our model of TCP window evolution has so far assumed that TCP receivers generate an acknowledgement for every arriving packet. Many implementations, however, use delayed acknowledgements to slow the rate of window expansion or alleviate congestion on the reverse link. We can model this artifact by noting that if the receiver sends one ack for every K packets received, then the TCP window grows from W to $W + \frac{1}{W}$ for every K packets transmitted. An approximation to this behavior is achieved by supposing that the TCP window grows by only $1/K^{th}$ of its value for every packet transmitted i.e. by modifying the window evolution equation to $W_{n+1} = W_n + \frac{1}{K.W_n}$.

Numerical results verify the effectiveness of this correction in accounting for the phenomenon of delayed acknowledegments. The graphs in figure 3 contain the comparisons between analysis and simulations when a TCP connection performing delayed acknowledgements is combined with the RED queue management algorithm while figure 4 shows the comparisons when a TCP performing delayed acknowledgements interacts with the ERD queue management algorithm. For the ERD queue, we also provide the theoretical distribution obtained by applying the correction for delayed acknowledgements in the square-root formula [2].

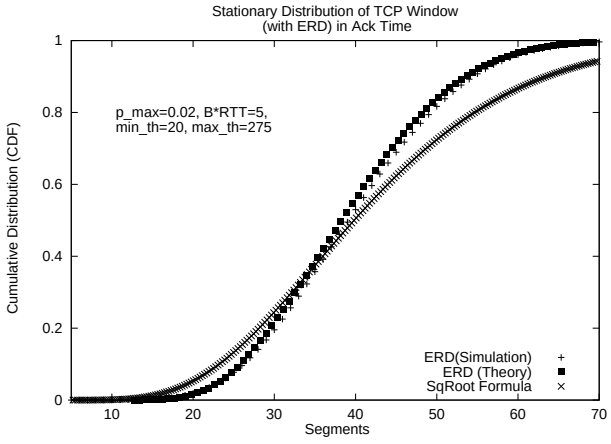

Figure 2: Behavior of TCP Window with Early Random Drop (and External Delay)

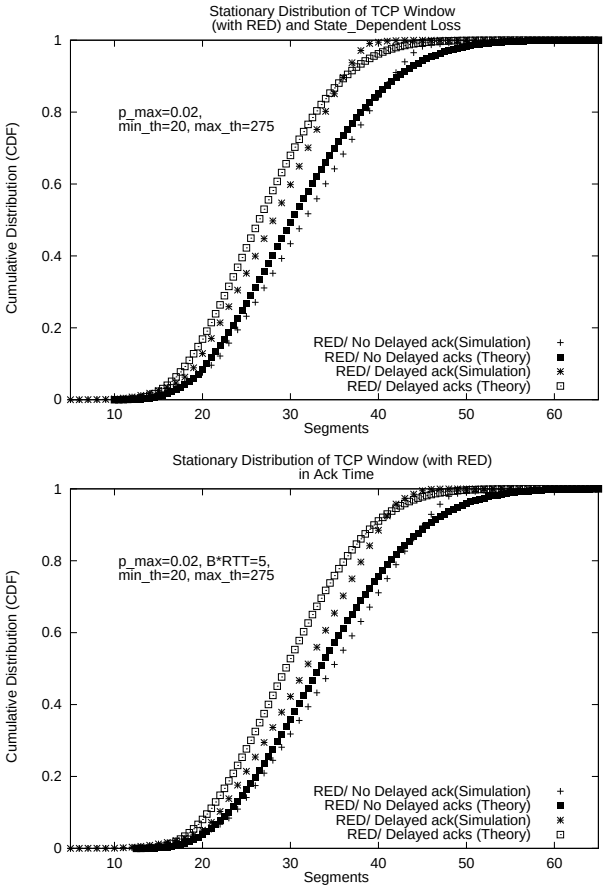

Figure 3: Behavior of TCP Window with Random Early Detection (with & without external delay and delayed acks)

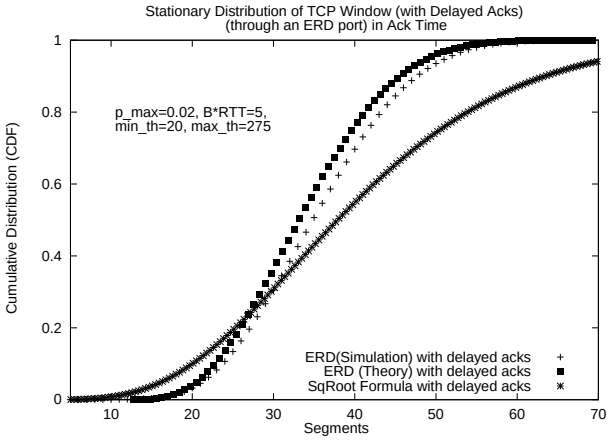

Figure 4: Behavior of TCP Window with Delayed Ack and Early Random Drop

V. Conclusion

In this paper, we present a technique for analyzing and predicting the window distribution of a persistent TCP connection over a path where packet losses occur with variable (state-dependent) probability and where packet drops are conditionally uncorrelated events. The key to an effective numerical procedure for predicting the distribution is the change in the index to subjective time, which is a history-dependent rescaling of the time index.

Comparisons with simulation results suggest that this technique is fairly accurate in predicting obtaining TCP window behavior of current TCP versions when the loss probabilities are low and timeouts are relatively rare events. In particular, we find that this analysis can be used to predict the window behavior of a single persistent TCP flow interacting with queue management algorithms like Early Random Drop and Random Early Detection. While the accuracy of the predictions was expected for Early Random Drop, the fit for the case of Random Early Detection was surprisingly good.

We are currently attempting to utilize this approach to extend the analysis to the case of multiple persistent TCP connections interacting with ERD/RED queues. By obtaining reasonably accurate distributions of individual TCP windows, we hope to derive an accurate characterization of the composite queue dynamics (including its mean and variance).

APPENDICES

I. POISSON NATURE OF PACKET DROP EVENTS

We prove here that the subjective time formulation results in an inter-loss interval that is exponentially distributed with mean 1 and is independent of past and future intervals. For the TCP process under consideration, this property is asymptotically true as the loss probabilities tend towards 0.

Let us find the probability $P\{X_i > T\}$, i.e., the probability at least subjective time T elapses between the $(i-1)^{th}$ packet loss and the i^{th} packet loss. We renumber the packets: packet j (temporarily) denotes the j^{th} packet after the one that was the i^{th} loss. Since the congestion window is increasing after the i^{th} loss, there exists with probability 1 a (random!) N such that

$$p_1 + p_2 + \cdots + p_N < T \leq p_1 + p_2 + \cdots + p_{N+1}.$$

The probabilities p_j are also random.

The probability of interest is that none of the first N packets are lost. Since N is random, this probability equals

$$\sum_{n=1}^{\infty} P\{N=n\}\ E[\prod_{j=1}^{n}(1-p_j)|N=n]. \qquad \text{(I.1)}$$

As long as (with probability one) $\max\{p_j, 1 \leq j \leq n\}$ is almost zero, the expression (I.1) is close to

$$\sum_{n=1}^{\infty} P\{N=n\} E[e^{-\sum_{j=1}^{n} p_j}|N=n].$$

Since $0 < T - \sum_{j=1}^{N} p_j \leq p_{N+1}$, we see that as long as (for example)

$$E[\max\{p_j, 1 \leq j \leq N+1\}] \downarrow 0, \qquad \text{(I.2)}$$

$P\{X_i > T\} \to e^{-T}$.

Since the above proof is also independent of the size of the packet that caused the i-th packet loss, we see that if condition (I.2) holds[6], the inter-loss intervals (in subjective time) are not only exponentially distributed, but also independent of past and future intervals. This establishes the fact that the loss events are realizations of a Poisson process of intensity 1 in subjective time.

[6]Note that the condition (I.2) may hold even if p_{max} is not small. All we really need is the congestion window *almost always* stays small enough for $p(w)$ to be small.

II. DIFFERENCES BETWEEN ERD AND RED

In this appendix, we discuss the differences between the Early Random Drop (ERD) and the Random Early Detection (RED) algorithms, which are important in understanding the applicability of our loss model. The important differences are:

- RED operates on the average (and not the instantaneous) queue length. The drop probability, p, is thus a function of the weighted average (Q_{avg}) of the queue occupancy i.e., p is a function not just of Q_n but of $(Q_n, Q_{n-1}, Q_{n-2}, \ldots)$ with an exponential decay. Q_{avg} closely mirrors the instantaneous occupancy only if the queue varies slowly.
- To prevent large inter-drop durations, RED increases the drop probability for every accepted packet. (This property, which we call *drop biasing*, is achieved by using a variable, cnt, which increases with every successive accepted packet; the true dropping probability is then given by $\frac{p(Q)}{1-\ cnt.p(Q)}$. This results in an inter-drop period that is uniformly distributed between $(1, \ldots, \lfloor \frac{1}{p(Q)} \rfloor)$, as opposed to the independent drop model in ERD which results in geometrically distributed inter-drop periods.
- Some RED configurations have a sharp discontinuity in drop probability: when the average queue exceeds max_{th}, $p(Q)$ becomes 1 so that all incoming packets are deterministically dropped. This contrasts with our assumption that random drop occurs throughout the entire range of the buffer occupancy. Our analysis applies to RED queues only if the TCP process almost never builds up queues that exceed max_{th}.

III. PROOF OF CONVERGENCE OF $H(x)$

To see that $H(x)$ in equation (3.8) indeed converges to a limit, let us define $C(x)$ by $C(x) = \int_x^{\infty} J(u)du$. Now, assume that there exists a $\beta > 1$, such that $A(x) \leq \frac{x}{\beta}\ \forall\ x$ (i.e., $a(x) \geq \beta x$).This is a stronger requirement than $A(x) < x$; in the case of the TCP model, $\beta = 2$. Now since $q(u)$ is a non-decreasing function of u,

$$\int_u^{a(u)} q(\rho)d\rho \geq \int_u^{\beta u} q(\rho)d\rho\ \ \geq \frac{\beta u\ -\ u}{\beta u}\int_0^{\beta u} q(\rho)d\rho$$

so that

$$\int_u^{a(u)} q(\rho)d\rho\ \geq \frac{\beta-1}{\beta}\int_0^{\beta u} q(\rho)d\rho \qquad \text{(III.3)}$$

Hence, $C(x) \leq \lambda \int_x^{\infty} q(u)e^{-\gamma Q(\beta u)}\ du$ where $\gamma\ =\ \frac{\beta-1}{\beta}$. Thus,

$$C(x) \leq \lambda \int_x^{\infty} q(\beta u)e^{-\gamma Q(\beta u)}\ du \qquad \text{(III.4)}$$

$$\leq \lambda(1-\gamma)\int_{\beta x}^{\infty} q(u)e^{-\gamma Q(u)}\ du \qquad \text{(III.5)}$$

$$\leq \frac{\lambda(1-\gamma)}{\gamma}e^{-\gamma Q(\beta x)} \qquad \text{(III.6)}$$

This shows that $C(x)$ is upper bounded by $C(0)$. (Note that for the case of TCP, $\beta = 2$ and $\lambda = 1$, so that $C(0) = 1$; in other cases, C(0) is some finite value.) Now, consider a random variable with density $f(x) = \frac{J(x)}{C(0)}$ and let $X_1, X_k, \ldots, X_k$ be k i.i.d

realizations of this random variable and let $X_{(1)}, X_{(2)}, \ldots, X_{(k)}$ be the order statistic. Then,

$$\overbrace{\int_{x_1>x} \ldots \int_{x_k>a(x_{k-1})}}^{k-fold} J(x_1)\ldots J(x_k)dx_k...dx_1 = \frac{\{C(x)\}^k}{k!} \times Prob(X_{(j)} > a(X_{(j-1)}) \; for \; j\epsilon(2,\ldots,k) | X_j > x, \; \forall j). \tag{III.7}$$

Hence, if we denote the sum of the first l terms in the RHS of equation (3.8) as $H_l(x)$, we see that

$$| H(x) - H_l(x) | \leq \bar{H} \sum_{j=l}^{\infty} \frac{C(x)^j}{j!}, \tag{III.8}$$

which proves that $H(x)$ is indeed convergent.

REFERENCES

[1] V Jacobson, "Congestion Avoidance and Control", SIGCOMM 1988.

[2] T Ott, M Matthis and J Kemperman, "The Stationary Behavior of Idealized Congestion Avoidance", ftp://ftp.bellcore.com/pub/tjo/TCPwindow.ps, August 1996.

[3] J Padhye, V Firoiu, D Towsley and J Kurose, "Modeling TCP Throughput: a Simple Model and its Empirical Validation", Proceedings of Sigcomm '98,September 1998.

[4] A Kumar, "Comparative Performance Analysis of Versions of TCP in a Local Network with a Lossy Link", IEEE/ACM Transactions on Networking, August 1998.

[5] T V Lakshman, U Madhow and B Suter, "Window-based Error Recovery and Flow Control with a Slow Acknowledgement Channel: a Study of TCP/IP Performance", Proceedings of Infocom '97, April 1997.

[6] S Floyd, "Connections with Multiple Congested Gateways in Packet-Switched Networks Part 1: One-way Traffic", Computer Communication Review, Vol.21, No.5, October 1991.

[7] V Jacobson, "Modified TCP congestion avoidance algorithm", April 30, 1990, end2end-interest mailing list.

[8] T Ott, "Is Random Early Drop fair?", ftp://ftp.isi.edu/end2end/end2end-interest-1996.mail (Nov. 19, 1996).

[9] M Mathis, J Semke, J Mahdavi and T Ott, "The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm", Computer Communications Review, July 1997.

[10] E Hashem, "Analysis of Random Drop for Gateway Congestion Control", MIT-LCS-TR-506.

[11] S Floyd and V Jacobson, "Random Early Detection Gateways for Congestion Avoidance", IEEE/ACM Transactions on Networking, August 1993.

[12] A Parekh and R Gallager, A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Single-Node Case, IEEE/ACM Transactions on Networking, June 1993