Slowstart Window Modeling for Single Connection TCP-Tahoe

Jörgen Olsén, Ingemar Kaj

Abstract—**We present a stochastic model for the window dynamics of a TCP with a pure slow start flow-control algorithm. Our initial motivation was the increasing number of connections with recurring small file transfers that is more likely to be well characterized by a pure slow start model than most existing pure congestion avoidance models.**

Furthermore, with the increasing research done regarding the interaction between end-systems and the network it is possible to study different flow-control algorithms overall impact on network stability via characterizations of the single protocols behaviour. This tool-box of flow-control algorithms is increased with a characterization of the slow start algorithm. Comparisons are made of expected throughput rate and time out probability between a pure congestion avoidance model and a pure slow start model.

We present ns-2 simulations showing that our modeling approach characterize the slow start algorithm of TCP well.

Keywords—**TCP performance modeling, adaptive window congestion control, slow start algorithm, time out probability**

I. TCP MODELING IN THE SLOW START SET UP

A. Introduction and objective

A substantial part of the traffic control in the Internet is provided by the end users from the Transmission Control Protocol (TCP). It provides reliable delivery of data between a sender and a receiver and implements an adaptive flow-control algorithm. TCPs flow-control increases the data sending rate until the network shows signs of congestion. The purpose of the flow-control is to use a large but fair amount of the bandwidth and to be able to adjust to changing network conditions that could be caused by new competing flows or changes in the networking topology. Despite existing standards there is a wide variety of existing flow-control algorithms operating in the Internet [10]. In its current implementations the majority of TCP implementations detects network congestion primarily from packet loss indications and hence the most widely used way for the network to send overload signals back to the senders is by dropping packets. Use of explicit congestion notification, measurement of the variance in round-trip-times and other joint efforts between the end users and the network are in various stages of research and deployment [3], [7], [9], [22], [4], [19], [16].

During a connection the most widely used TCPs updates its sending speed in various phases of the transmission either according to the cautious *congestion avoidance algorithm* or to the more aggressive *slow start algorithm*. Most analytical TCP models have concentrated on the congestion avoidance phase with pure congestion avoidance models. Here we elaborate on these ideas by instead considering a pure slow start algorithm driven TCP. This behaviour could be motivated in a web like environment where small files are transfered followed by a silent period that causes the TCP to reset its window to the initial size. Even with the increasing research being put into the interaction between the end users and the network it is still important to characterize individual TCP algorithms since this is needed for those user-network dual approaches. In this paper we will concentrate on providing expressions for the network utilization a single TCP will experience. However, by using the ideas and methodologies introduced in [4], [19] it should be possible to see how the total overall network utilization and stability is impacted by this type of TCP algorithm and with combinations of this slow start algorithm and other flow-control algorithms.

B. Slow start dynamics

The TCP variant considered in this paper always uses the slow start algorithm. In slow start mode this variant of TCP increases the current window quickly until either an error occurs or the maximal window, w, is reached. If the current window is W and a packet acknowledgement, ACK, is received, TCP adds one unit to the window size. A connection starts in the slow start mode by sending a single packet into the network. Upon receiving an ACK for this packet one round-trip-time later two new packets are injected. The ACKs for these two packets in turn trigger the transmission of four new packets. In effect, the slow start algorithm doubles the sending rate once every round-trip-time.

Two mechanisms for detecting lost packets are used by

Jörgen Olsén is with the Department of Mathematics, Uppsala University, Box 480, S-75106 Uppsala, Sweden and at Telia Research AB, Farsta, Sweden. E-mail: jorgen@math.uu.se. **Ingemar Kaj** is with the Department of Mathematics , Uppsala University, Box 480, S-75106 Uppsala, Sweden. E-mail: ikaj@math.uu.se.

current TCP implementations from the Tahoe family. They are called *triple duplicate detection* (TD) and *time-out detection* (TO). When a packet is lost within a window, already sent packets in the same window with higher sequence number than the lost one and packets sent in the next round as a response to the successfully ACKed pre-loss packets might still get through to the receiver. The receiver will notice that a packet sequence number is missing and respond to these new packets by generating *duplicate ACKs* all acknowledging the packet before the first lost one, i.e. the last correctly received in-order packet. If a given number K, the *duplicate ACK threshold*, of such duplicate ACKs are received back at the sender this is supposed to indicate that a packet has been lost and the packet will be immediately re-sent using the so called *fast retransmit* procedure. On the other hand, if the sender receives less than K duplicate ACKs the sender must wait a predetermined time for its *retransmit timer* to expire and a *time-out* occurs. No matter which procedure detects the loss, the TCP Tahoe variant that we will consider will restart after a loss with its congestion window reset to the initial size.

C. TCP Throughput

Our main motivation for modeling the TCP window dynamics is to be able to derive an asymptotic throughput relation. We will need to set up a proper stochastic model of TCPs flow control and identify its essential features. Finally we will use mean value properties to arrive at TCP throughput formulas that depends on the packet loss rate p observed in the network.

While setting up our model we will make some assumptions. As a starting point we consider greedy senders, i.e. TCPs that always have data to send, hence the only restriction on the sending rate is the current window size. Relaxation of this assumption and the consequence if there is limited amount of data to transfer will be discussed elsewhere. During the whole transmission period we assume *constant round-trip-times*, equal for all packets and independent of the current window size, as customary in the literature [20], [19], [15]. We assume that the time it takes to feed the packet from the sender onto the link is negligible compared to the round-trip-time. Transmission is assumed to go on in terms of *rounds*. A round is initiated when TCP sends the current window of W packets. It ends one round-trip-time later when one or more ACKs are received, or when a loss is detected. If there are no lost packets the returning ACKs updates the window and the process is repeated. Since we assume constant round trip times that also implies that we assume constant time-out intervals lasting a given number T_0 of rounds and we treat T_0 as a fixed pre-configured parameter, however we do include multiple time-out back-offs in the model. We consider a Random Early Detection, *RED*, queueing environment and in the model this assumption is idealized to the property of packets being dropped independently of each other throughout the connection. Finally, we restrict ourselves to the data transfer part of the TCP transmission and do not model the connection set-up and tear-down.

We now turn to the identification of the congestion window size after ACKs for all pre-error packets have arrived at the sender. As described each ACK increases the congestion window by one. Since we assume that all packets are sent simultaneously at the beginning of each round the ACKs will arrive in a burst one round-trip-time later and increase the congestion window. Hence from one round to the next the window size changes from W to $2W$. If we denote the number of loss free transmission rounds by Y, and the number of packets sent during this time by A, we can use the knowledge of the behaviour of the TCP algorithm to find the relationship between A and Y. After Y rounds of successfully sent and acknowledged full windows the window W will have grown to the size 2^Y.

We could continue working in this well defined discrete setting and derive distributions of quantities needed for the TCP throughput. It has however proven more favorable to take another approach. In the following we will start by allowing any discrete number for A, accounting for packet losses during rounds, and then continue by approximating the discrete number of sent packets by a continuous quantity. This will then also imply a continuously growing congestion window whose distribution will be derived in the following.

D. General notations

We will now introduce some of the notation that will be used repeatedly in the rest of the paper. By a cycle we will mean the intervals formed by the reductions of the congestion window to the initial size after an error detection. Each cycle consists of a random number of rounds during which the congestion window is increased. With each cycle we associate the following quantities:

$A_n =$ the number of packets correctly transmitted and acknowledged before the first loss in cycle n,

$B_n =$ the number of packets correctly transmitted and acknowledged after the first loss until the end of cycle n,

$U_n =$ the size of the congestion window after acknowledgements of the A_n packets have arrived at the sender,

$Y_n =$ the number of rounds needed to send the A_n packets,

$Q(p) =$ the probability that a cycle ends with a time out and not a triple duplicate detection for a TCP experiencing a packet error probability p in the network.

II. General Throughput relation

In [15], [16] we use renewal theory to motivate and derive the throughput for a TCP Tahoe and TCP New Reno sender with a general distribution on A_n. That derivation is done with a pure congestion avoidance model in mind but the general throughput expression is equally valid for this slow start model. In our notation the throughput formulas from [15], [16] here takes the form

$$\begin{aligned}&\text{Throughput}_{\text{TCP}} = \\ &= \frac{E(A_n)+E(B_n)}{E(Y_n)+1+(\frac{T_0}{1-2p}-1)Q},\end{aligned} \tag{1}$$

where we have used an approximative correction term for multiple time-out back-offs valid for packet probability rates in the approximative range $p \le 0.3$. In order to calculate throughput for our TCP we will need to find the expected values of A_n, B_n and Y_n and the probability $Q(p)$.

III. Explicit TCP Throughput models

A. *Unbounded congestion window ($w = \infty$)*

We will consider A_n as the fundamental quantity in terms of which we derive the window control dynamics. We assume that each packet is lost with probability p independently of all other packets. Then, the distribution of A_n is geometric, $A_n \in \text{Ge}(p)$. Hence

$$P(A_n = k) = p(1-p)^k, \quad k \ge 0,$$

and the expected number of correctly transmitted and acknowledged packets equals

$$E(A_n) = \frac{1-p}{p}.$$

Since each ACKed packet increases the congestion window by one we have the simple relationship $U_n = 1 + A_n$. Clearly,

$$\begin{aligned}P(U_n = k) &= P(A_n = k-1) \\ &= p(1-p)^{k-1}, \quad k \ge 1\end{aligned}$$

and

$$E(U_n) = \sum_{k=1}^{\infty} k\, P(U_n = k) = \frac{1}{p}.$$

As a result of our RED queueing assumption the remaining packets sent after the first loss in cycle n are again subject to independent losses with probability p. Together with the packets sent due to ACKs for the pre-error packets, another $B_n \in \text{Bin}(U_n - 1, 1-p)$ packets are successfully transmitted and acknowledged during the same cycle. Due to the simple relationship between U_n and A_n in the unbounded case we have

$$B_n \in \text{Bin}(U_n - 1, 1-p) = \text{Bin}(A_n, 1-p).$$

In particular

$$E(B_n) = E(A_n)\,(1-p) = \frac{1}{p}(1-p)^2.$$

The distribution for B_n can be found for example by using the generating function.

$$g_{B_n}(t) = E(E(t^{B_n}|A_n)) = E((p+qt))^{A_n},$$

where $q = 1-p$. Inserting the generating function for a geometric stochastic variable we find that

$$g_{B_n}(t) = \frac{p/(1-qp)}{1-tq^2/(1-qp)}$$

which we recognize as the generating function for a $\text{Ge}(\frac{p}{1-qp})$ stochastic variable. Hence by the uniqueness theorem for generating functions we deduce that

$$B_n \in \text{Ge}(\frac{p}{1-qp}),$$

so

$$P(B_n = j) = \frac{p}{1-pq}\,(\frac{q^2}{1-pq})^j, j = 0, 1, ..$$

We continue by deriving the probability of time out, $Q(p)$. From [15] the relationship between $Q(p)$ and B_n for a TCP from the Tahoe family is found to be

$$Q(p) = P(B_n < K)$$

where K is the triple duplicate threshold. By choosing its canonical value of $K = 3$ we can calculate $Q(p)$ explicitly.

$$Q(p) = \sum_{k=0}^{2} P(B_n = k) = \frac{p(3 - 9p + 13p^2 - 9p^3 + 3p^4)}{(1 - p + p^2)^3}. \quad (2)$$

We finally model the Y_n number of rounds needed to transfer the A_n packets. For this part we will need to derive expected values for functions of A_n. By using a technique first introduced in [14] of approximating A_n's discrete distribution by a continuous one the calculations are greatly simplified. Hence we use the convergence in distribution

$$pA_n \overset{d}{\Rightarrow} V_n \quad p \to 0$$

where $V_n \in \text{Exp}(1)$ and assume in the rest of this section for simplicity that we have identically $A_n = \frac{V_n}{p}$. The consequence of approximating A_n's discrete structure by a continuous is that we also approximate the way the window changes. The discrete structure of doubling each round is approximated by a continuous function $2^{x-1/2}$ as shown in Figure(1). This function sweeps over approximately the same area as the discrete window during n rounds which is a desired property for our modeling purposes.

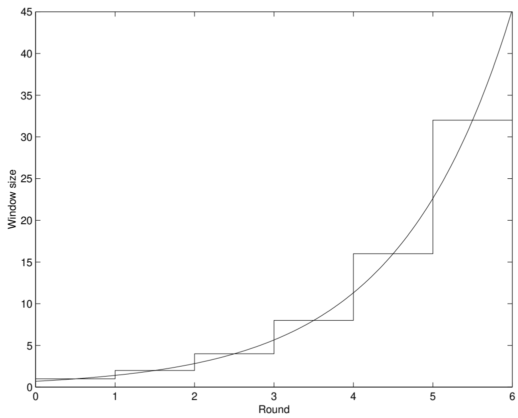

Fig. 1. Continuous window approximation

Now the approximate relationship between A_n and Y_n follows from

$$A_n = \int_0^{Y_n} 2^{x-1/2}\, dx = \frac{2^{Y_n} - 1}{\sqrt{2}\ln 2}.$$

Expressing Y_n in terms of A_n as

$$Y_n = \frac{\ln(A_n \sqrt{2}\, \ln 2 + 1)}{\ln 2}$$

allows us to derive the distribution for Y_n, namely

$$F_{Y_n}(y) = P(\ln(A_n \sqrt{2}\, \ln 2 + 1) \leq y\, \ln 2) = P(A_n \leq \frac{2^y - 1}{\sqrt{2}\, \ln 2}) = P(\frac{V_n}{p} \leq \frac{2^y - 1}{\sqrt{2}\, \ln 2}).$$

Since $V_n \in \text{Exp}(1)$ we have

$$F_{Y_n}(y) = 1 - e^{-p\frac{2^y - 1}{\sqrt{2}\, \ln 2}}.$$

The density of Y_n equals

$$f_{Y_n}(y) = \frac{p}{\sqrt{2}}\, 2^y\, e^{-p\frac{2^y - 1}{\sqrt{2}\, \ln 2}}, \quad (3)$$

and therefore

$$E(Y_n) = \int_0^\infty y\, f_Y(y)\, dy = \int_0^\infty y\, \frac{p}{\sqrt{2}}\, 2^y\, e^{-p\frac{2^y - 1}{\sqrt{2}\, \ln 2}}\, dy.$$

After changing variables, performing a partial integration and simplifying we end up with

$$E(Y_n) = \frac{e^{\frac{p}{\sqrt{2}\, \ln 2}}}{\ln 2} \int_{\frac{p}{\sqrt{2}\, \ln 2}}^\infty \frac{e^{-z}}{z}\, dz.$$

We choose to express this more compactly in terms of the Exponential integral, *Ei(k)*, as

$$E(Y_n) = \frac{e^{\frac{p}{\sqrt{2}\, \ln 2}}}{\ln 2}\, Ei(\frac{p}{\sqrt{2}\, \ln 2})$$

where $Ei(k) := \int_k^\infty \frac{e^{-t}}{t}\, dt$.

We now have all quantities needed to calculate the throughput. We use the general throughput expression from formula 1, insert our derived quantities, rewrite it slightly and end up with

$$\text{Throughput}_{\text{Unbounded}} = \frac{1 - p + (1-p)^2}{\frac{p\, e^{\frac{p}{\sqrt{2}\, \ln 2}}}{\ln 2}\, Ei(\frac{p}{\sqrt{2}\, \ln 2}) + p\,(1 + (\frac{T_0}{1-2p} - 1)Q(p))}. \quad (4)$$

B. *Bounded congestion window ($w < \infty$)*

The maximal window constraint that is effective in a normal TCP transfer due to sender, receiver or network constraints will now be included in the model. We thus make the further assumption that $U_n \leq w$ and derive the quantities needed for a throughput calculation under this constraint.

We first note that the distribution of A_n is not changed whereas U_n, B_n and Y_n all will be affected.

As long as the maximal window is not attained the window increases by one for each received ACK while it remains at w once the maximal window is reached. The probability function for U_n therefore has the same structure as in the unbounded case as long as the window is less than w and then has a point-mass at w. We again describe the window distribution $\{U_n\}$ using our fundamental quantity A_n as

$$U_n \stackrel{d}{=} (1 + A_n)\, 1_{\{A_n \le w-1\}} + w\, 1_{\{A_n > w-1\}}$$

where the probability of attaining the maximal window is

$$P(U_n = w) = \sum_{k=w-1}^{\infty} P(A_n = k) = (1-p)^{w-1}.$$

The distribution for U_n in the bounded case is therefore characterized by

$$P(U_n = k) = \begin{cases} p(1-p)^{k-1}, & 1 \le k < w \\ (1-p)^{w-1}, & k = w. \end{cases}$$

Now the expected value is found to be

$$\begin{aligned} E(U_n) &= \sum_{k=1}^{w-1} k\, p(1-p)^{k-1} + w(1-p)^{w-1} \\ &= \frac{1-(1-p)^w}{p}. \end{aligned}$$

We expect to get the same expected value as in the unbounded case when w tend to ∞ and to end up with the maximal window as p goes to zero. Indeed, it is simple to verify that $\lim_{w\to\infty} E(U_n) = \frac{1}{p}$ and $\lim_{p\to 0} E(U_n) = w$.

Turning to B_n we find that it has the same conditional distribution

$$B_n \in \mathrm{Bin}(U_n - 1, 1-p)$$

as in the unbounded case but the distribution and hence the generating function for $U_n - 1$ has changed due to the maximal window constraint.

Now

$$\begin{aligned} E(U_n) &= E(B_n) = E(U_n - 1)\,(1-p) \\ &= \frac{(1-p)^2}{p} - \frac{(1-p)^{w+1}}{p}. \end{aligned}$$

The distribution takes the form

$$P(U_n - 1 = k) = \begin{cases} p(1-p)^k, & 0 \le k \le w-2 \\ (1-p)^w, & k = w-1. \end{cases}$$

So, the probability generating function for $U_n - 1$ is

$$\begin{aligned} g_{U_n-1}(t) &= \sum_{n=0}^{\infty} t^n\, P(U_n - 1 = n) \\ &= \sum_{n=0}^{w-2} t^n\, p(1-p)^n + t^{w-1}(1-p)^w, \end{aligned}$$

and thus $g_{B_n}(t) = E[(p+qt)^{U_n-1}] = g_{U_n-1}(p+qt)$. Exchanging t for $p + qt$ in g_{U_n-1} gives

$$\begin{aligned} g_{B_n}(t) &= \sum_{n=0}^{w-2} (p+qt)^n\, pq^n \\ &+ (p+qt)^{w-1} q^w. \end{aligned}$$

Using the binomial theorem this can be expressed as

$$\begin{aligned} g_{B_n}(t) &= \sum_{n=0}^{w-2} pq^n \sum_{k=0}^{n} \binom{n}{k} (qt)^k p^{n-k} \\ &+ \sum_{k=0}^{w-1} \binom{w-1}{k} (qt)^k\, p^{w-1-k} q^w. \end{aligned}$$

Changing the order of summation we end up with

$$\begin{aligned} g_{B_n}(t) &= \sum_{k=0}^{w-2} \sum_{n=k}^{w-2} pq^n \binom{n}{k} (qt)^k p^{n-k} \\ &+ \sum_{k=0}^{w-1} \binom{w-1}{k} (qt)^k\, p^{w-1-k} q^w \end{aligned}$$

in which we identify the coefficient in front of t^k. We find that

$P(B_n = k) =$

$$\sum_{n=k}^{w-2} pq^n \binom{n}{k} q^k\, p^{n-k} + \binom{w-1}{k} q^k\, p^{w-1-k}\, q^w$$

for $0 \le k \le w-2$ and that it has a point-mass equal to q^{2w-1} at $k = w-1$.

Due to B_n's dependence on w it is more intricate to find the exact closed form solution for $Q(p)$ but we find an explicit expression for each maximal window w. From analytical inspection and 3-dimensional (p, w) dependent plotting of $Q(p, w)$ we observe that $Q(p, w)$ is rather independent of increasing w-values for w greater than, say 6. Since in a practical TCP implementation a maximal congestion window less than 6 packets seems rather unrealistic we argue that the unbounded timeout function can be used even in the bounded case for all reasonable maximal windows. Hence we approximate the function

$$\begin{aligned} Q(p, w) &= \sum_{k=0}^{2} \Bigg[\sum_{n=k}^{w-2} pq^n \binom{n}{k} q^k\, p^{n-k} \\ &+ \binom{w-1}{k} q^k\, p^{w-1-k}\, q^w \Bigg], \end{aligned}$$

valid for all $w \ge 4$, with our previously derived $Q(p)$-expression, formula 2.

Finally the distribution of Y_n is modeled in the bounded case. As long as the maximal congestion window has not been reached the Y_n dynamics is the same as in the unbounded case. However, in our continuous approximation we find that after $\frac{1}{2}+\frac{\ln w}{\ln 2}$ rounds, the congestion window has reached its maximal value w and $A_n > \frac{w-1/\sqrt{2}}{\ln 2}$. The remaining $A_n - \frac{w-1/\sqrt{2}}{\ln 2}$ packets are sent out at a constant max-rate of w packets per round. Putting all these pieces together we can express the distribution for Y_n in the bounded case as

$$\begin{aligned} Y_n \;=\; & \frac{\ln(A_n\sqrt{2}\,\ln 2+1)}{\ln 2}\,1_{\{A_n\le\frac{w-1/\sqrt{2}}{\ln 2}\}} + \\ & \Big((\frac{1}{2}+\frac{\ln w}{\ln 2}) + \frac{A_n-\frac{w-1/\sqrt{2}}{\ln 2}}{w}\Big)\,1_{\{A_n>\frac{w-1/\sqrt{2}}{\ln 2}\}}. \end{aligned}$$

To find $E(Y_n)$ we proceed along the same path as in the unbounded case by approximating the distribution of A_n. Looking at the three different parts of $E(Y_n)$ separately we find for the first part that

$$\begin{aligned} & E\Big(\frac{\ln(A_n\sqrt{2}\,\ln 2+1)}{\ln 2}\,1_{\{A_n\le\frac{w-1/\sqrt{2}}{\ln 2}\}}\Big) = \\ & E\Big(\frac{\ln(A_n\sqrt{2}\,\ln 2+1)}{\ln 2}\,1_{\{\frac{\ln(A_n\sqrt{2}\,\ln 2+1)}{\ln 2}\le\frac{1}{2}+\frac{\ln w}{\ln 2}\}}\Big). \end{aligned}$$

Using the already derived density function from formula 3 in the unbounded case we find this conditional expected value as

$$\int_0^{\frac{1}{2}+\frac{\ln w}{\ln 2}} y\,\frac{p}{\sqrt{2}}\,2^y\,e^{-\frac{p}{\sqrt{2}}\,\frac{2^y-1}{\ln 2}}\,dy.$$

By changing variables and integrating partially in the same way as in the unbounded case we end up with

$$\frac{e^{\frac{p}{\sqrt{2}\,\ln 2}}}{\ln 2}\Big(\int_{\frac{p}{\sqrt{2}\,\ln 2}}^{\infty}\frac{e^{-z}}{z}\,dz - \int_{\frac{pw}{\ln 2}}^{\infty}\frac{e^{-z}}{z}\,dz - e^{-\frac{pw}{\ln 2}}\,\ln(w\sqrt{2})\Big).$$

Once more we utilize the Exponential integral and rewrite this as

$$\frac{e^{\frac{p}{\sqrt{2}\,\ln 2}}}{\ln 2}\Big(Ei(\frac{p}{\sqrt{2}\,\ln 2})\;-\;Ei(\frac{pw}{\ln 2})\;-\;e^{-\frac{pw}{\ln 2}}\,\ln(w\sqrt{2})\Big).$$

For the second part we have

$$\begin{aligned} & E\Big((\frac{1}{2}+\frac{\ln w}{\ln 2})\,1_{\{A_n>\frac{w-1/\sqrt{2}}{\ln 2}\}}\Big) = \\ & = (\frac{1}{2}+\frac{\ln w}{\ln 2})P(A_n > \frac{w-1/\sqrt{2}}{\ln 2}) \\ & = (\frac{1}{2}+\frac{\ln w}{\ln 2}\,P(V_n > p\frac{w-1/\sqrt{2}}{\ln 2}) \\ & = (\frac{1}{2}+\frac{\ln w}{\ln 2})e^{-p\frac{w-1/\sqrt{2}}{\ln 2}}. \end{aligned}$$

Finally, for the third part we use the same approximation as we did in the unbounded case and let $A_n = \frac{V_n}{p}$, hence $f_{A_n}(y) = p\,e^{-yp}$ and we get

$$\begin{aligned} & \frac{1}{w}E\Big((A_n-\frac{w-1/\sqrt{2}}{\ln 2})\,1_{\{A_n>\frac{w-1/\sqrt{2}}{\ln 2}\}}\Big) \\ & = \frac{1}{w}\int_{\frac{w-1/\sqrt{2}}{\ln 2}}^{\infty}(y-\frac{w-1/\sqrt{2}}{\ln 2})\,p\,e^{-py}\,dy \\ & = \frac{1}{pw}\,e^{p\frac{w-1/\sqrt{2}}{\ln 2}}. \end{aligned}$$

Putting the separate parts together and simplifying we find that

$$E(Y_n) = \frac{e^{\frac{p}{\sqrt{2}\,\ln 2}}\Big(Ei(\frac{p}{\sqrt{2}\,\ln 2}) - Ei(\frac{pw}{\ln 2})\Big)}{\ln 2} + \frac{e^{-p\frac{w-1/\sqrt{2}}{\ln 2}}}{pw}. \quad (5)$$

In an error free network we expect the cycle to continue forever. Since $Ei(\frac{p}{\sqrt{2}\,\ln 2}) - Ei(\frac{pw}{\ln 2}) \approx \ln(\frac{pw}{\ln 2}) - \ln(\frac{p}{\sqrt{2}\,\ln 2})$ for small p, we have

$$\begin{aligned} & \frac{e^{\frac{p}{\sqrt{2}\,\ln 2}}\Big(Ei(\frac{p}{\sqrt{2}\,\ln 2})\;-\;Ei(\frac{pw}{\ln 2})\Big)}{\ln 2} + \frac{e^{-p\frac{w-1/\sqrt{2}}{\ln 2}}}{pw} \\ & \approx \frac{\ln w\;+\;\ln\sqrt{2}}{\ln 2} + \frac{1}{pw} \nearrow \infty, \quad \text{as} \quad p \to 0 \end{aligned}$$

as expected.

Now the throughput in the bounded case is found as

$$\begin{aligned} & \text{Throughput}_{\text{Bounded}} = \\ & \frac{1-p+(1-p)^2-(1-p)^{w+1}}{p\,E(Y_n)+p\,(1+(\frac{T_0}{1-2p}-1)Q(p))} \end{aligned} \quad (6)$$

with $E(Y_n)$ from formula 5 and $Q(p)$ from formula 2.

IV. Simulations

This section contains a discussion on simulation results from ns-2 [12] simulations of persistent file transfers. Comparisons are made with results from analytical modeling both for the slow start model in this paper and the congestion avoidance model analyzed in [15].

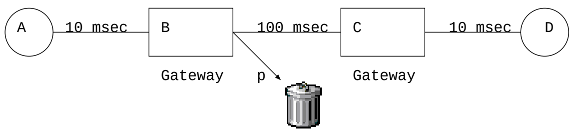

Fig. 2. Simulation scenario used in ns-2 simulations

In order to validate our model of the slow start algorithm of TCP Tahoe the existing ns-2 implementation of this protocol has been modified to always use the slow start mode. The simulations are performed using the well-known single bottleneck setup, as shown in Figure(2), which was also used in [15]. The idealized RED behaviour of dropping packets independently on the bottleneck link is implemented in the simulation via the combination of a drop-tail queue with a very large buffer followed by a device that discards packets independently of each other with a configurable probability p. Our aim is on describing the TCP protocols behaviour under loss behaviours observed in real networks, i.e. independent losses for RED. Since the focus in this paper is not on simulating real world like topologies and traffic scenarios needed to achieve the observed complex dropping behaviour we find the used simulation setup ideal for our purposes.

Figure(3) shows a comparison between ns-simulations of a pure slow start Tahoe, the throughput formula 6 for this algorithm and, to put it into perspective, the throughput formula from [15] for a Tahoe version that always uses the less aggressive congestion avoidance algorithm.

The following network parameters were observed in the simulations and are used in the throughput formulas: $RTT = 0.240$ sec, *maximum window* $w = 30$ packets, *TCP packet size* 500 bytes, *Time-Out length* $T_0 = 2.65$, *Time-out probability* $Q(p)$ from formula 2. The maximum window constraint of 30 packets per RTT results in a maximal attainable throughput of 62500 bytes per second.

We show the simulated and analytic throughput in Figure(3) for packet error probabilities in the range $p \leq 0.25$. This error rate range should be representative for real world scenarios [21] and the analytical formula predicts throughput accurately throughout this whole interval. For higher loss rates our model overestimates the throughput. From simulations we have seen that this is due to the fact that the impact of multiple back-offs is more severe than reflected in our model when packet loss rates are higher than approximately 0.25. Hence we underestimate the times needed for the lengthy multiple back-offs during these extreme loss rates.

We include the throughput formula for a pure congestion avoidance model from [15] into the graph for comparison. As expected, the throughput obtained for the slow start algorithm is higher than the congestion avoidance variant for the same packet error probabilities. This is of course natural since the slow start algorithm increases the sending rate more aggressively. Since everybody in practice can implement their own flow-control algorithm this says that in a network where everybody else is using the standardized congestion avoidance variant an implementation of the slow start variant would achieve higher throughput. However, this more aggressive variant is likely to bring the network into a congested state and introduce higher packet loss rates. An interesting question is therefore to see to what loss rate the network would settle down to for a combination of some congestion avoidance and some slow start algorithms. For this, we again argue that the modeling approaches presented in [17] and [4] together with throughput formulas from this paper and [15] could shed some light on this question.

We now turn to a comparison of the simulated and analytical time out probabilities. Figure(4) shows the probability $Q(p)$ that a cycle ends with a time out and not with a triple duplicate detection. We show this for simulations with the pure slow start model, analytical formulas for this algorithm and also include the analytical formulas for the congestion avoidance TCP variant derived in [15]. As the figure shows, in our idealized simulated RED environment, the time out probability is very well predicted by formula 2 for all possible packet error rates p. We now compare this with the time out probability for the pure congestion avoidance model. As the graph shows, the slow start algorithm has a much higher probability of avoiding time outs than the congestion avoidance variant for the same packet error rates. This is also something that should be taken into consideration while elaborating on the overall effect the slow start algorithm has on the network stability.

V. Conclusions and Future Work

We have shown a method to model the previously sparsely modeled slow start algorithm. Verifications have been done with the ns-2 simulator and we have achieved accurate predictions of throughput and probability of time outs. We believe that the characterization of the slow start algorithm in this paper together with the congestion avoidance model in [15], the study of impact on the throughput from different queueing mechanisms in the network performed in [16], ideas and methodologies introduced in [4], [19], [17] are good building blocks for considering the interaction between multiple TCP algorithms and the network.

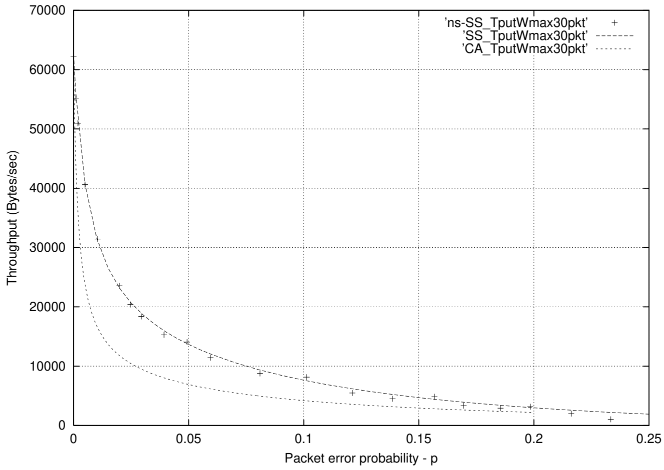

Fig. 3. Throughput - analytic formula 6 and simulated for Tahoe-SlowStart mode and analytic for Tahoe-CongestionAvoidance mode from [15]

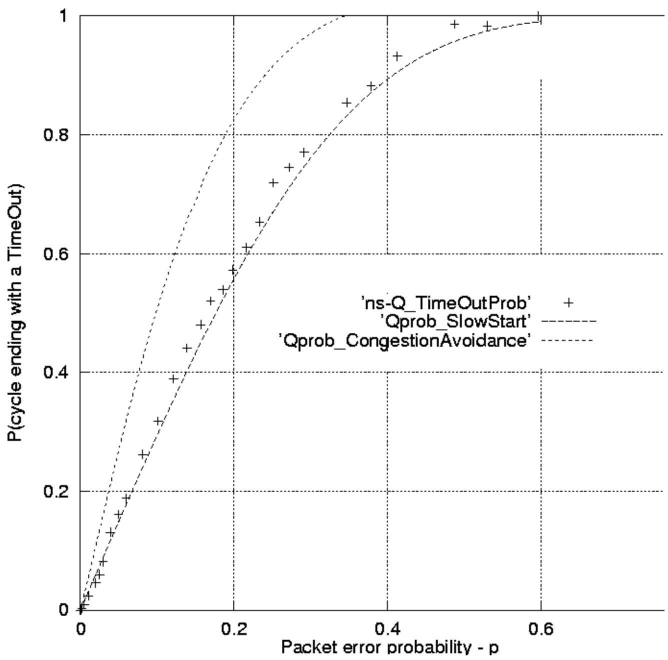

Fig. 4. Probability of Time Out - analytic formula 2 and simulated for Tahoe-SlowStart mode and analytic for Tahoe-CongestionAvoidance mode from [15]

References

[1] M. Allman, V. Paxson, W. R. Stevens, TCP Congestion Control, April 1999, RFC 2581.

[2] E. Altman, K. Avrachenkov, C. Barakat, A Stochastic Model of TCP/IP with Stationary Random Losses, Comp. Comm. Review 30 (October 2000), 231-242, ACM Sigcomm 2000.

[3] R. Braden et al, Recommendations on Queue Management and Congestion Avoidance in the Internet, April 1998, RFC 2309.

[4] T. Bu, D. Towsley, Fixed Point Approximations for TCP Behaviour in an AQM Network, ACM SIGMETRICS 2001.

[5] Neil Cardwell, Stefan Savage, Thomas Anderson, Modeling TCP Latency, Preprint, Department of Computer Science and Engineering, University of Washington, 2000

[6] K. Fall, S. Floyd, Simulation-based Comparisons of Tahoe, Reno and SACK TCP, Computer Communication Review, 26(3), July 1996.

[7] S. Floyd, TCP and Explicit Congestion Notification, ACM Computer Communication Review, V. 24 N. 5, October 1994, p. 10-23.

[8] S. Floyd, Congestion Control Principles, Internet Draft, ACIRI, January 2000.

[9] S. Floyd, A Report on Some Recent Developments in TCP Congestion Control, ACIRI 2000

[10] S. Floyd, J. Padhye, Identifying the TCP Behaviour of Web Servers, ACM Sigcomm, August 2001.

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

[12] S. Floyd, S. MCanne, ns-LBL Network Simulator, 1997. Obtain via http://www.isi.edu/nsnam/ns/

[13] G.R. Grimmet, D.R. Stirzaker, Probability and random processes, 2nd Ed, Oxford Science Publications, Oxford, 1992.

[14] I. Kaj, Stochastic modeling in broadband communications network, SIAM Monographs on Mathematical Modeling and Computation, 2001.

[15] I.Kaj, J. Olsén, Throughput modeling and simulation for single connection TCP-Tahoe, Proc. 17th International Teletraffic Congress ITC'01, 2001, Preprint verison available at http://www.math.uu.se/∼jorgen.

[16] I.Kaj, J. Olsén, Stochastic equilibrium modeling of the TCP dynamics in various AQM environments, Preprint, Department of Mathematics, Uppsala University, 2001.

[17] F. Kelly, Mathematical modelling of the Internet, Statistical Laboratory, Centre for Mathematical Sciences, University of Cambridge, available at http://www.statslab.cam.ac.uk/∼frank/

[18] A. Kumar, Comparative performance analysis of versions of TCP in a local network with a lossy link, IEEE/ACM Trans./ Networking 6 (August 1998) 485-498.

[19] S. Low, L. Peterson, L. Wang, Understanding TCP Vegas: A Duality Model, ACM SIGMETRICS 2001.

[20] J. Padhye, V. Firoiu, D. Towsley, J. Kurose, Modeling TCP throughput: a simple model and its empirical validation, Comp. Comm. Review 28 (October 1998) 303-314.

[21] V. Paxson, End-to-End Internet Packet Dynamics, Proc. ACM Sigcomm, September 1997.

[22] Stevens, W., RFC 2001: TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms, January 1997, RFC 2001.