Performance Analysis of TCP with Nonpersistent Sessions

Arzad A. Kherani and Anurag Kumar
Dept. of Electrical Communication Engg.
Indian Institute of Science, Bangalore, 560 012, INDIA
email: alam, anurag@ece.iisc.ernet.in

Abstract—**Most analytical models of TCP assume that the sessions have an infinite amount of data to transmit. In practice, it is well known that sessions arrive, transfer a finite amount of data and depart.**

This paper is about an analytical model for calculating the average bandwidth shares obtained by Additive Increase Multiplicative Decrease (AIMD) controlled *ephemeral* sessions that share a single (bottleneck) link. In particular, we consider the following traffic model: file transfer requests arrive in a Poisson process, each request requiring the transfer of an independent and exponentially distributed volume of data, which we view as a fluid. After transferring its required volume of data, each session departs.

We develop an analysis of this model, and we identify a Markov regenerative structure in the rate evolution process. We provide a complete and exact analysis that yields the average bandwidth share obtained by sessions. We obtain numerical results from the analysis and from an *ns* simulation of TCP controlled sessions, and discuss the region of applicability of such a modelling and analysis approach. We find that the detailed behavior of TCP's congestion avoidance phase is well characterised by our model. The analysis compares well with simulations for low offered loads and when the mean file transfer requests are large compared to the round-trip pipe. We observe that slow start has a considerable effect on performance of TCP controlled ephemeral sessions. We also find that our analytical model yields results that are close to those obtained from a *processor sharing* model with the link capacity taken to be 0.75 of its actual value.

Keywords— **TCP performance modelling, bandwidth sharing, processor sharing.**

I. Introduction

Traffic engineering and bandwidth dimensioning in the Internet requires effective models that can predict the performance as seen by network users. At the present time the Internet carries predominantly elastic traffic, and the bandwidth sharing between elastic sessions is controlled by TCP. Hence there is a need for developing performance models that can be used to calculate the throughputs obtained by TCP controlled elastic sessions.

This work was supported by a research grant from HFCL.

Most studies of implicit or explicit feedback control of elastic sessions consider only *persistent sources* which have an infinite backlog of data to send. In real applications, however, the TCP connections over a link are *ephemeral*, i.e., a new TCP connection starts, transmits some finite number of packets and closes the connection. This is the case, for example, when HTTP 1.0 is being used for downloading files, so that a new connection is established for each file transfer request, unlike HTTP 1.1 where existing connections can be reused for subsequent transfers.

In this work we are interested in the performance of TCP controlled bandwidth sharing in the situation where a single link is shared by TCP controlled sessions that arrive, transfer some data, and depart. We make the following mathematical assumptions: session arrival epochs constitute a Poisson process; each session needs to transfer a random volume of fluid data, which we will assume to be exponentially distributed; the sequence of transfer volumes is i.i.d. (independent and identically distributed). We also assume that the propagation delay for every session is the same.

An example of this situation is depicted schematically in Figure 1 where the destinations are downloading files from one or more servers through an intermediate bottleneck link. Users at the destination computers initiate file transfer sessions. The horizontal bars in the "server" indicate the remaining amount of data to be transferred for each session. The TCP mechanisms are used by the server to decide how fast to send the data for each of the ongoing

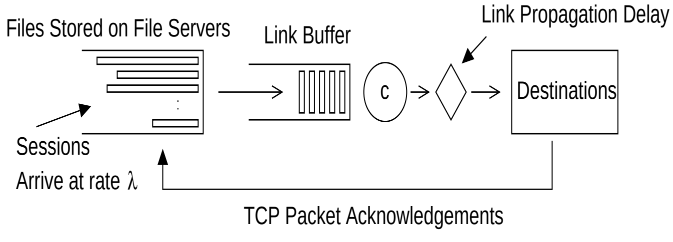


Fig. 1. TCP being used to share the bandwidth of a link among arriving and departing finite volume sessions.

sessions. The purpose of the link buffer is basically to take care of mismatches between the aggregate rate at which the server sends data and the rate at which the link can transmit it to the destination. Rate mismatches occur because the bottleneck bandwidth available to each session can only be "estimated" at the traffic source. Rate mismatches result in packet losses in the buffer, which cause the TCP transmitters to adjust their rates. The TCP transmitter builds up its rate, suffers loss and then drops the rate; thus *the active sessions share a time varying transmission rate*. When a session departs, the total shared rate decreases, and when a session arrives, loss occurs, the sessions drop their rates and begin to again probe for bandwidth. Our analytical model captures this behavior.

Assuming fluid file transfers, we model the Additive Increase Multiplicative Decrease (AIMD) behavior of the rate process during TCP's congestion avoidance phase. We identify a Markov renewal process embedded at the buffer overflow epochs, and use a "reward" rate analysis to obtain the average bandwidth share obtained by ongoing sessions. We provide several numerical results and compare them with a TCP simulation written in *ns*. Some interesting findings are the following:

- Detailed behavior of rate evolution during the congestion avoidance phase of TCP is well modelled by our analysis.
- We find that analysis compares well with simulations for low normalised loads and when the mean file transfer volumes are large compared to the volume of data in the round-trip pipe.
- It is important to capture the slow start phase of TCP when large propagation delays or small mean file sizes are used.
- We find that the average bandwidth share from our model is close to that obtained from a Processor Sharing (PS) model assuming that the link speed is 75% of the actual link speed. This is a considerable simplification, as the detailed analysis is quite complex, whereas the PS model has a simple closed form formula.

Some prior work related to ours is the following. In [1] and [2] the authors have considered a situation in which several Internet subscribers share a single backbone link, and they are attached to this link by lower speed access links. Each subscriber goes repeatedly through a download and "think" cycle. The authors' aim is to develop an approximate analysis for the throughput obtained by subscribers. Assuming zero propagation delay, and ideal bandwidth sharing (i.e., the entire bottleneck link rate is applied to the ongoing transfer and the rate is shared equally), a closed queueing network model applies. In [1] the authors also provide an approximate "correction" to the results obtained by assuming ideal bandwidth sharing to take care of TCP behavior. In [3] the authors consider a generalisation of the stochastic model that we have stated above (i.e., Poisson session arrivals and exponential transfer volumes) to the case of a network. Again assuming ideal bandwidth sharing, the authors show that the vector process of the number of ongoing sessions on each route is a Markov process. They show that a simple and natural condition is sufficient for the positive recurrence of this Markov chain. The case of a single link when there is no propagation delay on the link has been addressed in [4] using the processor sharing model.

The paper is organised as follows. In Section II we describe the evolution of the total rate process when session rates are controlled by TCP. In Section III we provide the assumptions under which we develop our analysis. In Section IV we specify and discuss the performance measure that we use. In Section V we develop the analysis. In Section VI we provide numerical results from the analysis and compare them with simulation results.

II. TCP Controlled Rate Evolution with Session Arrivals and Departures

The evolution of the total rate of TCP controlled sessions through a single link is controlled by various factors such as the round trip propagation delay, link bandwidth, TCP packet size, and link buffer size. A recent paper ([5]) reports the analysis of the rate evolution of *individual* TCP controlled *persistent* sessions with different round trip delays sharing a common bottleneck link. We consider a similar model but in a situation in which sessions arrive randomly, transfer a finite amount of data and then depart. We assume that all the sessions have the same round trip propagation delay.

Figure 2 shows a detailed evolution of a sample path of the total data arrival rate process at the link buffer when there are arrivals and departures. We first discuss the detailed evolution and then simplify it on the basis of certain assumptions in Section III. The figure neglects the time taken by an arriving session to reach the slow start threshold, and also assumes that the sessions are always able to recover losses without timing out and going into slow start. Hence a new TCP session starts with an initial rate of s packets per second (which corresponds to the initial slow start threshold of a session). The existing sessions continuously increase their data sending rate resulting in a rate mismatch between the link speed and the total data arrival rate to the link buffer. When the total data arrival rate exceeds the link bandwidth, the link buffer starts building up at a rate determined by the number of active sessions. This increase in buffer occupancy eventually results in a

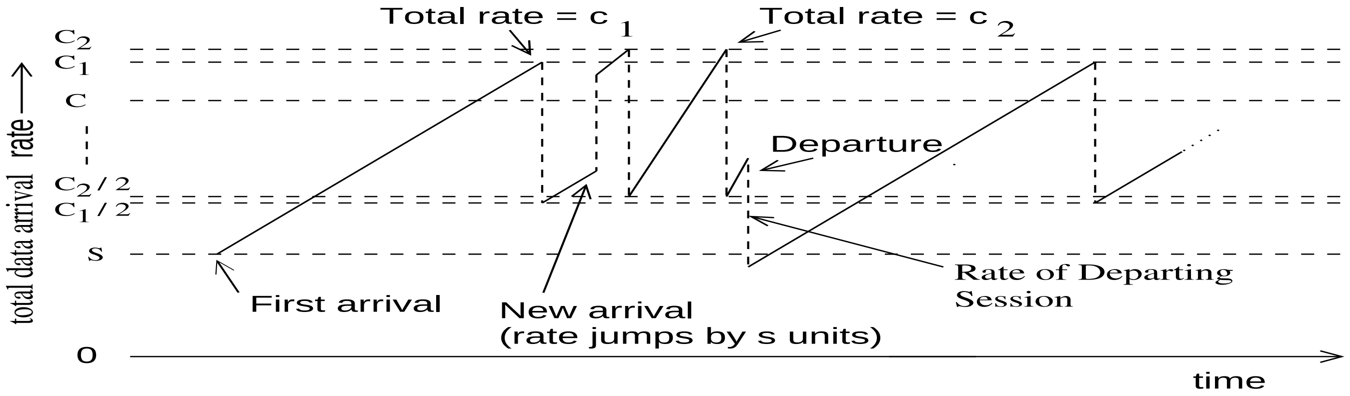


Fig. 2. A sample path of *total* rate evolution with arriving and departing sessions.

buffer overflow at a *total* data arrival rate determined by the number of sessions active; let c_n (packets per second) denote the total data arrival rate at which a buffer overflow occurs when n sessions are active (see Figure 2). In congestion avoidance phase each TCP session will increase it data sending rate by 1 packet per rtt (round trip time) hence the increase in data sending rate per rtt is more when the number of sessions are more thus, $c_{n+1} > c_n, n \geq 1$. In the figure only one session is shown to be active until the first arrival occurs. Note that the figure neglects the nonlinearity in the rate process owing to increase in RTT (as a result of increase in queuing delay) after the total rate crosses c, the link capacity. Also note that, since we ignore slow start, the total rate after an arrival that sees $n-1$ session active and a total rate R is $R+s$, if $R+s < c_n$, and is $c_n/2$ otherwise; in the latter case a loss is assumed to occur immediately upon the occurrence of the arrival. The figure also shows a departure; a departure has no effect on the rate of other active sessions but reduces the total rate, and slows down the increase rate of the *total* TCP controlled transmission rate.

The above process, even after ignoring slow-start, is too complex to analyse. Hence we have resorted to further simplifications for analytical tractability. Before we describe the model we analyse, we introduce some notation that will be used throughout the paper.

A. Notation Used

Traffic Model

λ the rate of the Poisson process modelling the arrival epochs of sessions

μ the parameter of the exponential distribution modelling the session file sizes; the mean file size is $1/\mu$.

Network Parameters

c link capacity (packets per second)

τ round trip link propagation delay (seconds); all sessions have the same τ.

b the number of data packets that one ack packet acknowledges

r_0 the value to which the total rate drops after an overflow; in implementations $r_0 = \frac{c}{2}$

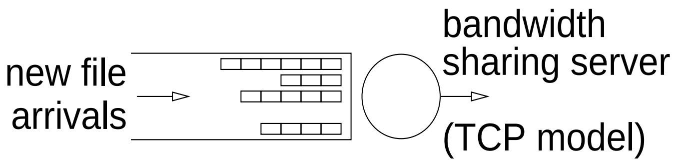
a the rate at which an *individual* session increases its data transfer rate in the congestion avoidance phase *(in packets per second per second)*; see Section III-A for the relationship between a, b and τ.

Processes in the Model

$N(t)$ the number of sessions at time t; each session $i, 1 \leq i \leq N(t)$, is transmitting its data at a time varying rate as long as it is in the system

$r_i(t)$ the data transfer rate of session $i, 1 \leq i \leq N(t)$, at instant t.

$R(t)$ the total data transfer rate at instant t; i.e., $R(t) = \sum_{i=1}^{N(t)} r_i(t)$



With the above definition of $R(t)$, our modelling approach is to view the files being transferred as being “queued” at a server that applies a time varying service rate to the files. This service rate $R(t)$ is bounded by the link service rate c. The problem is to analyse the random process $\{R(t)\}$ based on the way TCP sessions adapt their transmission rates. Notice that, in this viewpoint, $R(t) = c$, with each file getting an equal share, is just the Processor Sharing model.

III. Modelling Assumptions

In order to obtain an analytically tractable model, we make the following additional assumptions on the general model discussed above:

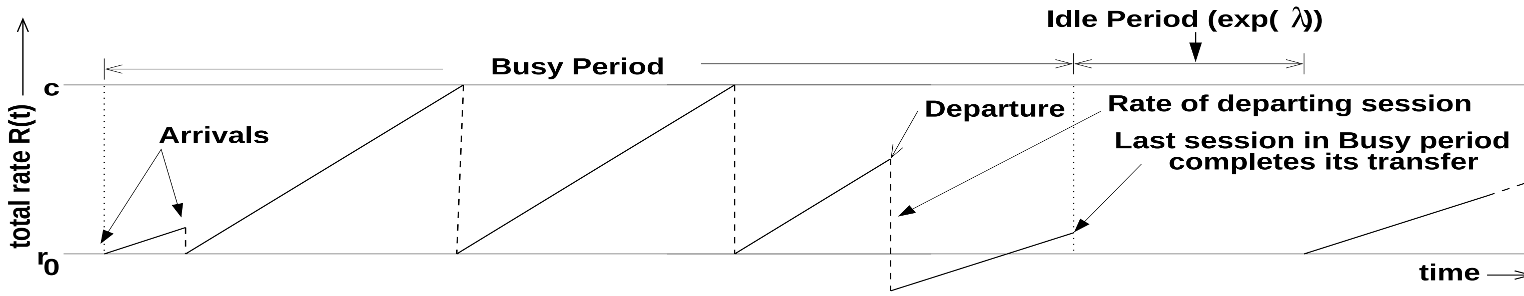


Fig. 3. A sample path of the *total* rate evolution with arriving and departing sessions under the assumptions made in Section III. Note that a *busy period* starts at Z_1 and another one starts at Z_5.

1. The delay-bandwidth product of the link is much larger than the link buffer size and hence, with reference to Figure 2, we assume that $c_n = c, \;\; n \geq 1$.
2. The initial slow-start of a session is neglected, and all overflows are recovered without entering slow-start; hence if an overflow occurs at t then $R(t+) = r_0 = \frac{c}{2}$
3. For an arriving session, the initial slow start threshold is infinity, i.e., with reference to Figure 2, $s = \infty$, and hence an arrival at t_a, when $n-1$ sessions are active, causes an immediate buffer overflow and the total rate of the sessions goes instantaneously to $c_n/2$. This, with Assumptions 1 and 2, implies that the effect of an arrival at t_a is:

$$N(t_a+) = n$$
$$R(t_a+) = \frac{c_n}{2} = \frac{c}{2}$$

4. The rate $r_i(t)$ given to any session i active at time t increases linearly with time with a slope of a units, i.e., $\frac{dr_i}{dt} = a$ for some constant a. We also assume that at any instant all the active sessions get equal rate, i.e., for any session i active at t, $r_i(t) = \frac{R(t)}{N(t)}$.
5. The effect of a departure of a session i at t_d is modelled as follows:

$$\begin{aligned} N(t_d+) &= N(t_d) - 1; \\ r_j(t_d+) &= r_j(t_d), j \neq i \\ R(t_d+) &= R(t_d)\left(1 - \frac{1}{N(t_d)}\right) \end{aligned}$$

6. If buffer overflow occurs at u, all the active sessions reset their transmission rates to $r_0/N(u)$ and continue sending their data.

An instance of the way total rate evolves under above assumptions is shown Figure 3. This figure will be discussed in detail in Section V

A. Justifications for the Assumptions

Assumption 2 is valid if the mean file size is much larger than the amount of data transferred in the initial slow start phase. We assume that the TCP congestion control is performing well in the sense that the TCP sessions go into slow start only at the beginning of their data transfer periods, and afterwards they do not require timeout for detecting losses. This is the case for example when TCP uses fast retransmit/recovery and the session arrival rates are small so that most of the losses are in the congestion avoidance phase. Hence after a buffer overflow, the sessions go into congestion avoidance phase thus justifying Assumption 6. Low arrival rates also imply that the transients die out and the steady state phenomenon of global synchronisation takes place.

Under the Assumption 2 of negligible slow start periods, we see that when an arrival occurs, it will see the other sessions in the congestion avoidance phase. Consider the initial slow start phase of the new arrival where the session can increase its data sending rate to a very high value. During this phase when it happens that the rate of the new session exceeds $c-r$ where r is the total rate of the *other* active sessions, the total rate will exceed c resulting in a buffer overflow (as $c_n = c$, for all n; Assumption 1). Hence the *total* rate will drop to $(c-r)/2 + r/2 = c/2 (= r_0)$. Since we neglect the duration of the initial slow start period, the effect of arrival is that it immediately causes a buffer overflow. This is the content of Assumption 3.

Under the above assumptions, the TCP sessions are in congestion avoidance phase during most of their sojourn time. In the congestion avoidance phase each session increases its window size by $1/b$ packets every τ where b is the number of packets that one ack packet acknowledges. Thus the slope a for TCP is $\frac{1}{b}$ packets per τ^2 i.e., $a = \frac{1}{b\tau^2}$.

B. Generality of the Model

It should be noted that the above model represents a class of additive increase multiplicative decrease congestion-feedback based rate control algorithms. Various algorithms of this class may differ with respect to the additive increase slope a, or the reset rate r_0. *For TCP,*

$r_0 = c/2$ *and* $a = \dfrac{1}{b\tau^2}$. In this paper in order to maintain the generality of the model we will continue to use the notation r_0 for the reset rate and a for the additive increase slope.

IV. Performance Measure

From the point of view of a user downloading a file, the relevant performance measure is the *file transfer throughput*, i.e., the service volume of the transfer (e.g., number of bytes) divided by its sojourn time in the system. Hence the *average session throughput* would be a useful performance measure.

Define

$\Phi_k =$ the volume of the k^{th} transfer (in data units),
$W_k =$ total time taken to complete the k^{th} transfer,
$\rho =$ the normalised offered load on the link, i.e., $\frac{\lambda(1/\mu)}{c}$.
Then the per session throughput τ is given by

$$\theta \; = \; \lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n} \frac{\Phi_k}{W_k} \; . \qquad (1)$$

Assuming existence of the stationary regime, and the associated moments, let (Φ, W) denote the stationary random vector of the transfer volume and the sojourn time of a session. It is clear that,

$$\theta \; = \; E\left(\frac{\Phi}{W}\right)$$

(Theorem IV.2 shows that $\frac{E\Phi}{EW}$ is a lower bound on $E\frac{\Phi}{W}$ for an M/G/1 PS queue).

For an M/G/1 PS queue the *average* time a session with a service requirement of θ spends in the system is $\frac{\theta}{1-\rho}$ ([6]). However, the *distribution* of the session's sojourn time in the system becomes very cumbersome [7]. This makes the above notion of throughput analytically intractable.

Another measure of performance is the *time averaged rate per ongoing session, conditioned on there being at least one ongoing session*, denoted by σ, i.e.,

$$\sigma := \lim_{t\to\infty} \frac{\int_0^t I_{\{N(u)>0\}} \frac{1}{N(u)} R(u) du}{\int_0^t I_{\{N(u)>0\}} du}$$

where $I_{\{N(u)>0\}}$ is 1 if at least one session is active at u and 0 otherwise. This performance measure can be obtained from analysis, whereas, as noted above, the more direct measure of session throughput averaged over sessions is intractable even for the simplest processor sharing model. See also other performance measures that have been used in [1], [8], [6] and [2].

If all the ongoing sessions get an equal share of the total rate $R(u)$ (Assumption 4 of Section III), we have

$$\sigma = \lim_{t\to\infty} \frac{\int_0^t r(u) du}{\int_0^t I_{\{N(u)>0\}} du}$$

where $r(u)$ *is defined as the rate given to an individual session active at* u. Now if the limits $\lim_{t\to\infty} \frac{1}{t}\int_0^t r(u)du$ and $\lim_{t\to\infty} \frac{1}{t}\int_0^t I_{\{N(u)>0\}} du$ both exist, then we can write

$$\sigma = \frac{\lim_{t\to\infty} \frac{1}{t}\int_0^t r(u)du}{\lim_{t\to\infty} \frac{1}{t}\int_0^t I_{\{N(u)>0\}} du} \qquad (2)$$

Thus our performance measure is equal to the time average rate per ongoing session divided by the fraction of time that there is at least one session in the system. We note here that for a Processor Sharing model, with $\rho := \frac{\lambda(1/\mu)}{c}$, and $\rho < 1$, the σ is given by

$$\sigma/c = \frac{1-\rho}{\rho} \log \frac{1}{1-\rho} \qquad (3)$$

We will use this formula when discussing the results from our TCP model. *Note that this formula is insensitive to the file size distribution.*

Comparison of θ and σ for the PS Model

It interesting is to study the relationship between σ and θ for the PS queue. We show that, in general, $\sigma \neq \theta$. However, numerically these performance measures can be quite close.

Theorem IV.1: For a PS queue (with $c = 1$) if $\rho < 1$, then with probability 1 (w.p.1)

$$\lim_{t\to\infty} \frac{1}{t} \int_0^t \frac{1}{N(u)} I_{\{N(u)\geq 1\}} \, du \; > \lambda \lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n} \frac{{\Phi_k}^2}{W_k}.$$

Proof: See the Appendix for a proof.
□

Corollary IV.1: For deterministic file sizes D, i.e., $\Phi_k \; = \; D, \forall k \geq 1$, w.p.1, $\sigma > \theta$.

Proof: We observe that

$$\lim_{n\to\infty} \frac{1}{n} \sum_{k\,=\,1}^{n} \frac{{\Phi_k}^2}{W_k} \; = \; E\left(\frac{\Phi^2}{W}\right).$$

Using Theorem IV.1,

$$\frac{1}{\rho} \lim_{t\to\infty} \frac{1}{t} \int_0^t \frac{1}{N(u)} I_{\{N(u)\geq 1\}} du > \; \frac{1}{E(\Phi)} E\left(\frac{\Phi^2}{W}\right).$$

For $\Phi_k = \; D$ we have

$$\frac{1}{E(\Phi)} \, E\left(\frac{\Phi^2}{W}\right) \; = \; D \, E\left(\frac{1}{W}\right).$$

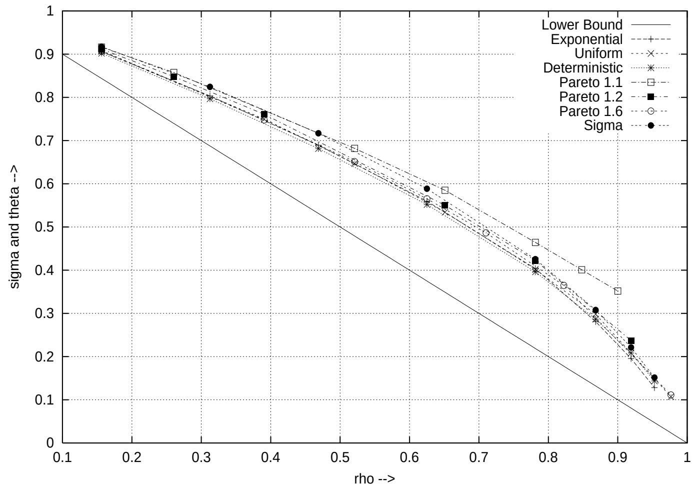


Fig. 4. σ and θ vs ρ for various file size distributions from simulation of a PS queue assuming fluid files

Hence

$$\frac{1}{\rho} \lim_{t\to\infty} \frac{1}{t} \int_0^t \frac{1}{N(u)} I_{\{N(u)\,\geq\,1\}}\, du \;>\; E\left(\frac{D}{W}\right)$$

i.e., $\quad \sigma > \theta.$

□

Corollary IV.1 provides an upper bound on θ for M/D/1 PS queue. Now we derive a simple lower bound on θ for an M/G/1 PS queue.

Theorem IV.2: For an M/G/1 PS queue with unit capacity server,

$$\theta \geq \frac{E\Phi}{EW} = 1 - \rho$$

Proof: We know (see [6]) that the expected sojourn time of a session with a service requirement of θ in a PS queue is given by

$$EW = \frac{\theta}{1-\rho}$$

irrespective of the file size distribution. Now,

$$E\frac{W}{\Phi} = E(E\frac{W}{\Phi}|\Phi) = \frac{1}{1-\rho}$$

Now, from Jensen's inequality,

$$E\frac{\Phi}{W} \geq \frac{1}{E\frac{W}{\Phi}} = (1-\rho) = \frac{E\Phi}{EW}$$

Hence the proof.

□

Figure 4 shows σ and θ for different file size distributions for a PS queue with unit capacity server. The values for θ are obtained from simulating the PS queue in which sessions arrive to the queue with a certain service requirement, obtain the service in processor sharing fashion (the file sizes being viewed as fluid) and depart; σ is obtained from Equation 3. The figure also shows the lower bound on θ for the M/G/1 PS queue derived in Theorem IV.2. Whereas the measure σ has been proved to be insensitive to the file size distribution, no such result regarding θ is known. It is seen, however, from the figure that the σ curve for the deterministic case is above the corresponding θ curve as the corollary claims. We see that θ is very close to σ, and also is not very sensitive to the file size distribution, except for Pareto with tail parameter $\alpha = 1.1$[1]. It appears that for finite second moment distributions, irrespective of the file size distributions, θ is between σ and $1-\rho$ and that σ is a good approximation for θ.

In the rest of paper we use σ as the performance measure.

V. Analysis of the Model

In order to develop the analysis we again refer to the sample path of the process $R(t)$ shown in Figure 3. The figure depicts a typical busy period during which sessions arrive, transfer a random amount of data and depart. Under our assumptions, a busy period always starts with an overflow due to the first arrival (epochs Z_1 and Z_5 are two such epochs). Buffer overflow also occurs when the total rate of existing sessions reaches the link capacity; e.g., epochs Z_3 and Z_4 in the figure. In the first busy period, another arrival occurs at Z_2. One of the departures in the busy period occurs shortly after Z_4, and the other departure ends the busy period.

Notation; for $k \geq 1$, define
Z_k to be the k^{th} buffer overflow epoch,
Y_k to be the number of ongoing sessions at epoch Z_k; note that $Y_k \geq 1$,
X_k to be the length of the k^{th} inter-buffer overflow cycle; i.e., $X_k = Z_{k+1} - Z_k$. □

At Z_k all the ongoing connections reset their transmission rate to r_0/Y_k (which is finite as $Y_k > 0$), and each connection starts increasing its rate with a slope a. Also because file sizes are exponentially distributed, the residual file sizes are again exponentially distributed, independent of the anything in the past. Hence, Z_{k+1} and Y_{k+1} depend only on Y_k. Thus $\{Y_k\}$ embedded at $\{Z_k\}$ is a Markov chain on $\{1, 2, \ldots\}$. *It follows that the process*

[1]This could be owing to the difficuly of simulation for this very heavy tailed distribution; since a substantial part of the distribution is in the tail, if the simulation is not run for very long the average of the sampled file sizes would be less than the nominal average, thus leading to a lower offered load and hence overestimation of the throughput.

$\{(Y_k, Z_k), k \geq 1\}$ *is a Markov renewal process, and the process* $R(t)$ *is Markov regenerative.* Clearly, the per session rate process $\{r(u)\}$ also has the same property. Assuming that the $\{Y_k\}$ is positive recurrent (it can be roughly argued that for this we need $\rho = \frac{\lambda(1/\mu)}{c} < 3/4$; however we leave a formal proof of this for future work). Let $\pi(n), n \geq 1$, denote the stationary probability distribution of $\{Y_k\}$.

Notation: in each cycle of the process $R(t)$ that starts with $Y_k = n$ sessions, we denote the (conditional) probability law by $P_n(\cdot)$ and the (conditional) expectation by $E_n(\cdot)$.□

Denote the generic marginal random variables for the processes $\{Y_k\}$, $\{Z_k\}$, and $\{X_k\}$ by Y, Z and X. Then the mean cycle time is given by:

$$EX = \sum_{n\geq 1} \pi(n) E_n X$$

Assuming that $EX < \infty$ (this is true since the cycle length is bounded by an interarrival time) we can now write the numerator of σ (see Equation 2) as

$$\lim_{t\to\infty} \tfrac{1}{t} \int_0^t r(u)du = \frac{E_\pi \int_0^X r(u)du}{EX}$$

Notice that $E_\pi \int_0^X r(u)du$ is the expectation of the total amount of data transferred in a cycle by the *longest lasting session in the cycle*; this longest lasting session may suffer an overflow at the end of the cycle, or, if the cycle ends in an idle period, the longest lasting session would be the last to leave in the busy period that ends in the cycle (see Figure 3).

Notation: let $V_k = \int_{Z_k}^{Z_{k+1}} r(u)du$, i.e., the volume of data transferred in the kth cycle by the longest lasting session. □ Then we have

$$\lim_{t\to\infty} \frac{1}{t} \int_0^t r(u)du = \frac{EV}{EX} = \frac{\sum_{n\geq 1} \pi(n) E_n V}{\sum_{n\geq 1} \pi(n) E_n X}$$

The denominator of σ (Equation 2) can be written as

$$\lim_{t\to\infty} \frac{1}{t} \int_0^t I_{\{N(u)\geq 1\}} du = \frac{E_\pi \int_0^X I_{\{N(u)\geq 1\}} du}{EX}$$

Notation: in the kth cycle, and for each $m \geq 1$, let $U_k^{(m)} = \int_{Z_k}^{Z_{k+1}} I_{\{N(u)\geq m\}} du$.□

It follows that

$$\lim_{t\to\infty} \frac{1}{t} \int_0^t I_{\{N(u)\geq m\}} du = \frac{EU^{(m)}}{EX} = \frac{\sum_{n\geq 1} \pi(n) E_n U^{(m)}}{\sum_{n\geq 1} \pi(n) E_n X}$$

Finally it follows that

$$\sigma = \frac{EV/EX}{EU^{(1)}/EX} = \frac{EV}{EU^{(1)}} = \frac{\sum_{n\geq 1} \pi(n) E_n V}{\sum_{n\geq 1} \pi(n) E_n U^{(1)}}. \quad (4)$$

Hence to obtain σ we need $E_n V$, $E_n U_{(1)}$ and $\pi(n)$. In the next sections we develop analysis that yields these quantities.

A. An Important Observation

We now make an important observation about the overflow epochs in a cycle. Consider a cycle that begins with n sessions (i.e., $Y = n$); each session begins with a rate $\frac{r_0}{n}$, and increases its rate linearly at the rate a. The departure of a session in a cycle does not disturb the rates of the other ongoing sessions. Notice that, with $Y = n$, overflows due to the total rate crossing the link capacity can occur only at those epochs u at which the individual rates equal the ratio of link speed c and $N(u)$. See Figure 5. Note that one possibility is that no overflow occurs and no arrival occurs, and each session completes its transfer before the total rate reaches c. In this case the cycle includes an idle period.

Notation: we denote these possible overflow epochs by $T_i^{(n)}, 0 \leq i \leq n-1$ where the subscript specifies that i sessions have departed, and the superscript specifies that there were n sessions at the start of cycle. We also define $T_{(-1)}^{(n)} = 0$ and $T_n^{(n)} = \infty$. See Figure 5. □

Thus $T_i^{(n)}$ is the time taken by $n - i$ sessions to build up their rate to c starting with an individual rate of $\frac{r_0}{n}$. Another view of $T_i^{(n)}$ is that if the cycle has to last beyond this epoch, then (a) no arrival should occur before this epoch, and (b) at least $j + 1$ departures should occur before $\{T_j^{(n)}\}$, $0 \leq j \leq i$. Yet another implication is that a buffer overflow leaves m sessions ongoing if and only if it occurs at $T_{n-m}^{(n)}$.

It is easy to see that $T_i^{(n)} = \frac{c}{a}(\frac{1}{n-i} - \frac{r_0}{cn})$. Note that this discretisation of $T_i^{(n)}$ is the result of Assumption 4 of Section III that all the active sessions at any instant have equal rates.

B. Analysis for Obtaining $E_n X$

Let the number of sessions at the beginning of a cycle be $n \geq 1$. We seek the complementary distribution $P_n(X > x)$, from which $E_n X$ can be obtained. For $0 \leq i \leq n$, and $x \in (T_{i-1}^{(n)}, T_i^{(n)}]$, we see that $X > x$ only if:

1. there is no arrival in $(0, x]$,
2. *and* there are at least $j + 1$ departures in $(0, T_j^{(n)}]$ for each $j = -1, ..., i - 1$.

Notation: Let $Q_1, Q_2, \ldots, Q_n$ denote i.i.d. exponential(μ) distributed random variables, representing the residual file sizes of the n sessions at the beginning of the cycle. Let $Q^{(n,1)}, Q^{(n,2)}, \ldots, Q^{(n,n)}$ denote the (ascending) ordered residual file sizes. Let, for $-1 \leq i \leq n-1$, $S_i^{(n)}$ denote the amount of data that an *individual* session can

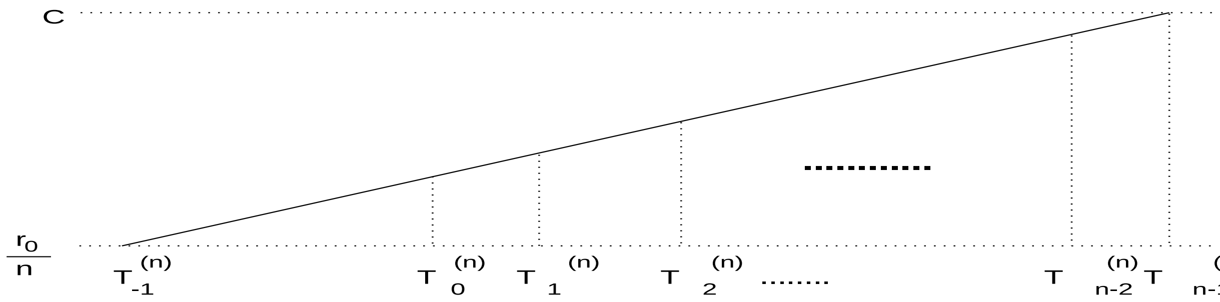


Fig. 5. Evolution of an individual connection's rate during a cycle, starting with n sessions, showing the possible overflow epochs $T_i^{(n)}, -1 \le i \le n-1$.

transfer in $(0, T_i^{(n)})$, given that $X > T_i^{(n)}$. Clearly, $S_i^{(n)} = \frac{r_0}{n} T_i^{(n)} + \frac{a}{2}(T_i^{(n)})^2$.□
Define, for $1 \le i \le n-1$

$$I_i^{(n)} = P_n\{Q^{(n,j)} \in (0, S_{j-1}^{(n)}), 1 \le j \le i+1\}$$

and also $I_0^{(n)} = 1$. It is then clear from the observations in Section V-A that, for $0 \le i \le n$, and $x \in (T_{i-1}^{(n)}, T_i^{(n)}]$,

$$P_n\{X > x\} = e^{-\lambda x}.I_{i-1}^{(n)}$$

Hence

$$\begin{aligned} E_n X &= \int_0^\infty P_n\{X > x\}dx \\ &= \sum_{i=0}^{n-1} \int_{T_{i-1}^{(n)}}^{T_i^{(n)}} P_n\{X > x\}dx + \int_{T_{n-1}^{(n)}}^{\infty} P_n\{X > x\}dx \\ &= \sum_{i=0}^{n-1} I_{i-1}^{(n)} \int_{T_{i-1}^{(n)}}^{T_i^{(n)}} e^{-\lambda x}dx + I_{n-1}^{(n)} \int_{T_{n-1}^{(n)}}^{\infty} e^{-\lambda x}dx \\ &= \frac{1}{\lambda}\sum_{i=0}^{n-1} I_{i-1}^{(n)} (e^{-\lambda T_{i-1}^{(n)}} - e^{-\lambda T_i^{(n)}}) + I_{n-1}^{(n)} \frac{e^{-\lambda T_{n-1}^{(n)}}}{\lambda} \end{aligned}$$

The second term in the second equality above is the case in which all sessions depart before reaching the link rate, and the cycle ends with a new busy period. Thus we need only $I_j^{(n)}, 0 \le j \le n-1$, to obtain $E_n X$. We will obtain these as a by-product of a subsequent calculation in Section V-C.

C. Analysis for Obtaining $E_n V$

Let the number of sessions at the beginning of a cycle be $n \ge 1$. We see that $V > d$ iff there is no overflow until the longest session has transferred d units of data.
Notation: $l(d)$ is the time a session, starting with rate r_0/n and increasing its rate with slope a, takes to transfer d units of data. It is easily seen that

$$l(d) = -\frac{r_0}{an} + \sqrt{\frac{r_0^2}{a^2 n^2} + \frac{2d}{a}}$$

□

Recall the notation $S_i^{(n)}$. It is clear that $0 \le V \le S_{n-1}^{(n)}$, as no session can last beyond $T_{n-1}^{(n)}$. Let, d be such that $S_i^{(n)} \le d < S_{i+1}^{(n)}$ for some $-1 \le i \le n-2$, . Then $P_n\{V > d\}$ is:
1. the probability of no arrival in $(0, l(d)]$,
2. *times* the probability of no overflow in $(0, l(d)]$ given no arrival in $(0, l(d)]$,
3. *times* the probability that at least one session has (residual) file transfer size requirement exceeding d given no arrival and no overflow in $(0, l(d)]$.

$$\begin{aligned} &P_n\{V > d \mid \text{no arrival in } (0, l(d)]\} = \\ &\quad \int_0^{S_0^{(n)}} .. \int_0^{S_i^{(n)} - x_i .. - x_1} n\mu e^{-n\mu x_1} .. (n-i)\mu e^{-(n-i)\mu x_{i+1}} \, . \\ &\quad [1 - (1 - e^{-\mu(d - x_{i+1} - \ldots - x_1)})^{n-(i+1)}] \, dx_{i+1} \ldots dx_1 \end{aligned}$$

which is the probability that, given no arrival till $l(d)$, there is no overflow in $(0, l(d)]$ and that not all of the sessions remaining after the $(i+1)^{th}$ departure go away before $l(d)$.
Notation: for $0 \le i \le n-1$, and $0 \le j \le n-i-1$, we define recursively,

$$I_{i,j}^{(n)} = \frac{n-i}{n-i-j}\left[I_{i-1,j}^{(n)} - e^{-(n-i-j)\mu S_i^{(n)}} I_{i-1,n-i}^{(n)}\right] \quad (5)$$

with

$$I_{0,j}^{(n)} = \frac{n}{n-j}(1 - e^{-(n-j)\mu S_0^{(n)}}) \quad (6)$$

□
It can then be shown that (we leave out the routine manipulations)

$$\begin{aligned} &P_n\{V > d \mid \text{no arrival in } (0, l(d)]\} = \\ &\quad I_i^{(n)} - \sum_{j=0}^{n-i-1} \binom{n-i-1}{j} (-1)^j e^{-j\mu d} I_{i,j}^{(n)} \end{aligned}$$

where $I_i^{(n)}$ is as defined in Section V-B and it can be seen that

$$I_i^{(n)} = I_{i,0}^{(n)}.$$

Finally we have

$$E_nV = \int_0^\infty P_n\{V > x\}dx = \int_0^{S_{n-1}^{(n)}} P_n\{V > x\}dx$$

It follows that

$$\begin{aligned} E_nV &= \sum_{i=0}^{n-1} \int_{S_{i-1}^{(n)}}^{S_i^{(n)}} P_n\{V > x\}dx = \\ &= \sum_{i=0}^{n-1} [I_i^{(n)} \int_{S_{i-1}^{(n)}}^{S_i^{(n)}} e^{-\lambda l(x)}dx - \\ &\quad \sum_{j=0}^{n-(i+1)} -1^j \begin{pmatrix} n-(i+1) \\ j \end{pmatrix} I_{i,j}^{(n)} \int_{S_{i-1}^{(n)}}^{S_i^{(n)}} e^{-\lambda l(x) - j\mu x}dx] \end{aligned} \tag{7}$$

D. Analysis for Obtaining $E_nU^{(m)}$

Observe that $P_n\{U^{(m)} > t\} = 0$ for $t \geq T_{n-m}^{(n)}$, and for $T_i^{(n)} \leq t < T_{i+1}^{(n)}$, $-1 \leq i < n-m$,

$$\begin{aligned} &P_n\{U^{(m)} > t\} = \\ &\quad e^{-\lambda t} \int_0^{S_0^{(n)}} .. \int_0^{S_i^{(n)} - x_i - .. - x_1} [n\mu e^{-n\mu x_1} ..(n-i)\mu e^{-(n-i)\mu x_{i+1}} \\ &\quad \sum_{k=m}^{n-i-1} \begin{pmatrix} n-i-1 \\ k \end{pmatrix} p(t)^k (1-p(t))^{n-i-k-1}] dx_{i+1}...dx_1 \end{aligned}$$

where $p(x) = e^{-\mu(d(x) - x_{i+1} - .... - x_1)}$. Hence $E_nU^{(m)} = \int_{x \geq 0} P_n\{U^{(m)} > x\}dx$. Manipulations similar to those already used above, finally yield

$$\begin{aligned} &E_nU^{(m)} = \\ &= \sum_{i=-1}^{n-(m+1)} \sum_{k=m}^{n-i-1} \begin{pmatrix} n-i-1 \\ k \end{pmatrix} \sum_{j=0}^{n-i-k-1} (-1)^j \\ &\quad \begin{pmatrix} n-i-1-k \\ j \end{pmatrix} I_{i,j+k} \int_{T_i^{(n)}}^{T_{i+1}^{(n)}} e^{-\lambda x - (j+k)\mu d(x)}dx \end{aligned} \tag{8}$$

We know that $\lim_{t\to\infty} \frac{1}{t} \int_0^t I_{\{N(u) \geq m\}} du = \frac{EU^{(m)}}{EX}$. *Hence Little's Theorem can be used to find the mean Sojourn Time of each session in the system.* This is a performance measure used in the paper [3].

E. Transition Probabilities for $\{Y_k\}$

Recall that the computation of σ requires the stationary distribution of the Markov chain $\{Y_k\}$. Let $p_{n,m} := P\{Y_{k+1} = m | Y_k = n\}$. Starting with n sessions, the cycle can end with m sessions with an overflow at $T_{n-m}^{(n)}$, provided no arrival occurs before this epoch, or the cycle can end with an arrival finding $m-1$ ongoing sessions. Clearly $p_{n,m} = 0$ for $m > n+1$ and for $m = 0$. It follows that

$$\begin{aligned} p_{n,m} = e^{-\lambda T_{n-m}^{(n)}} P_n\{X = T_{n-m}^{(n)} \mid \text{no arrival until } T_{n-m}^{(n)}\} \\ + P_n\{\text{an arrival sees exactly } m-1 \text{ sessions}\} \end{aligned}$$

Let the first part of $p_{n,m}$ be denoted by $p_{n,m}^{(d)}$, and the second part of $p_{n,m}$, owing to an arrival finding $m-1$ active sessions, be denoted by $p_{n,m}^{(a)}$. Notice that

$$\begin{aligned} &P_n\{X = T_{n-m}^{(n)} \mid \text{no arrival until } T_{n-m}^{(n)}\} \\ &\quad = P_n\{T_{n-(m+1)}^{(n)} < X \leq T_{n-m}^{(n)} \mid \text{no arrival until } T_{n-m}^{(n)}\} \\ &\quad = I_{n-m-1}^{(n)} - I_{n-m}^{(n)} \end{aligned}$$

Hence,

$$p_{n,m}^{(d)} = e^{-\lambda T_{n-m}^{(n)}} [I_{n-m-1}^{(n)} - I_{n-m}^{(n)}] \tag{9}$$

If an arrival has to see $m-1$ active sessions in a cycle starting with n sessions, we need the arrival to occur before $T_{n-m+1}^{(n)}$. Conditioned on an arrival occurring in $(x, x+dx)$, with $T_i^{(n)} < x \leq T_{i+1}^{(n)}$, $i \leq n-m$, we now find the probability that the arrival sees exactly $m-1$ active sessions. This probability is, (with $d(x) := \frac{r_0}{n}x + \frac{a}{2}x^2$),

$$\begin{aligned} &= \int_0^{S_0^{(n)}} .. \int_0^{S_i^{(n)} - x_i .. - x_1} n\mu e^{-n\mu x_1} ..(n-i)\mu e^{-(n-i)\mu x_{i+1}} \\ &\quad \begin{pmatrix} n-i-1 \\ m-1 \end{pmatrix} p(x)^{m-1} (1-p(x))^{n-m-i} dx_{i+1}...dx_1 \end{aligned}$$

where $p(x) = e^{-\mu(d(x) - x_{i+1} - .... - x_1)}$ is the probability that an exponential (μ) is greater than $d(x) - x_{i+1} - ... - x_1$. After some manipulations, it is seen that

$$\begin{aligned} &p_{n,m}^{(a)} = \\ &= \sum_{i=-1}^{n-m} \int_{T_i^{(n)}}^{T_{i+1}^{(n)}} \lambda e^{-\lambda x} \begin{pmatrix} n-i-1 \\ m-1 \end{pmatrix} \sum_{j=0}^{n-i-m} \begin{pmatrix} n-i-m \\ j \end{pmatrix} \\ &\quad (-1)^j e^{-(m+j-1)\mu d(x)} I_{i,m+j-1}^{(n)} dx \\ &= \sum_{i=-1}^{n-m} \begin{pmatrix} n-i-1 \\ m-1 \end{pmatrix} \sum_{j=0}^{n-i-m} (-1)^j \begin{pmatrix} n-i-m \\ j \end{pmatrix} \\ &\quad I_{i,m+j-1}^{(n)} \int_{T_i^{(n)}}^{T_{i+1}^{(n)}} \lambda e^{-\lambda x - (m+j-1)\mu d(x)}dx \end{aligned} \tag{10}$$

Hence, $p_{n,m} = p_{n,m}^{(d)} + p_{n,m}^{(a)}$, $m = 1, ...n+1$. $p_{n,m} = 0$ for $m = 0$, or $m > n+1$.

F. Calculating σ

As is seen from Equation (4), to calculate σ we need to compute $\pi(n)$, E_nV, $E_nU^{(1)}$ $n \geq 1$. To do this we first obtain $T_i^{(n)}$ and $S_i^{(n)}$, $0 \le i \le n-1$, $n \ge 1$. Then we compute $I_{i,j}^{(n)}$ using Equations (5) and (6). Using Equations (7) and (8) we then find $E_n V$ and $E_n U^{(m)}$, $n \ge 1$; for this we needed to use numerical integration to compute the integrals involved in the expressions in Equations (7) and (8). Using Equations (9) and (10) we then obtain $p_{n,m}, 1 \le m \le n+1, n \ge 1$. Using the transition probabilities $p_{n,m}$ we compute the stationary probability $\pi(n)$. These steps yield all the quantities required to find σ using Equation (4).

VI. Numerical Results

In this section we present results from our analysis and from an *ns* simulation where actual TCP connections with finite data transfer requirements share a common link as in Figure 1. The TCP packet size is $1000Bytes$ in all the numerical results presented in this section.

A. Verification of Modelling Assumptions

To verify our modelling assumptions (Section III) we check the validity of the model on a per cycle basis by obtaining the numerical values of quantities such as $T_i^{(n)}$ (sec), $S_i^{(n)}$ $(packets)$, $E_n X$ (sec), $E_n V$ $(packets)$ and $p_{n,m}$ from simulations and comparing the values thus obtained with the corresponding values obtained from analysis. For obtaining the above quantities from simulations, we start n TCP connections simultaneously, each connection has an exponentially distributed file to be transferred. The mean file size is $1/\mu$. Another session is scheduled to arrive at a time sampled from an exponential distribution with parameter λ. We reset the process as soon as an overflow occurs. For a fixed and particular set of parameters, we repeat the above process a large number of times and get the averages of the quantities listed in Table I. The parameters used for these results are:

Simulation and Analysis 1: $n = 5$, $c = 4Mbps$, link buffer 35 KB, $\tau = 0.2s$, $1/\lambda = 10s$, $1/\mu = 4000KB$,
Simulation and Analysis 2: $n = 2$, $c = 1Mbps$, link buffer 4 KB, $\tau = 0.3s$,$1/\lambda = 20s$, $1/\mu = 150KB$,
Simulation and Analysis 3: $n = 2$, $c = 1Mbps$, link buffer 8 KB, $\tau = 0.2s$,$1/\lambda = 5s$, $1/\mu = 100KB$,

It can be seen from Table I that the match between analysis and simulation is good on a per cycle basis, thus verifying the modelling assumptions we have made. The dashes in entries of the table indicate that either they were not obtained in simulation, or are not defined for the indices corresponding to that entry.

B. Results for Arriving and Departing Sessions

In this section we compare the values of σ obtained from simulations and from our analysis.

Choice of TCP Version
For every loss deteceted Tahoe enters slow start; since our model does not capture this feature, we use the Reno version of TCP in all the simulations reported here.

Choice of Link Buffer Size
In [9] the authors consider a long TCP-Tahoe connection on a link of capacity c and round trip propagation delay τ where the aim of slow start is to reach the slow start threshold quickly without losses; the ssthresh is equal to half the pipe size $((B + cT)/2)$. In the case of a receiver that acknowledges every data packet, they found that a buffer B larger than $cT/3$ is required.

Our analysis however neglects the queuing delays in the congestion avoidance phase after the packet arrival rate exceeds the link capacity. So large buffer sizes are not expected to give good match with our model. We also require that if after the initial slow starts the session again enter slow start, buffer should be enough to avoid any loss in this slow start. Hence in our simulations, we keep $B = \lceil \frac{cT}{3} \rceil$.

Parameters for Numerical Results
The model has several parameters: the session arrival rate: λ; the parameter of the exponentially distributed file sizes: μ (packets); the fixed RTT: τ (seconds); the link speed: c (packets sec^{-1}); the value to which the total rate is reset: r_0 (packets sec^{-1}); the per session sending rate increase rate: a (in packets sec^{-2}). We will take $r_0 = c/2$, and $a = 1/\tau^2$. By normalising time to τ, it is easily seen that the analysis depends on three parameters: $c\tau$, which is the fixed *RTT pipe* size in packets, $(\mu c\tau)^{-1}$, which is the mean file size in terms of the RTT pipe size, and $\rho = \lambda/(c\mu)$, the normalised offered load on the link. In view of these observations, in all the plots shown below we plot σ (normalised to c) versus ρ, with $c\tau$ and $(\mu c\tau)^{-1}$ as parameters when there are multiple plots on one axes.

Results:
Figures 6 and 7 show σ vs ρ plots for an RTT pipe size of 12.5 packets (for three values of c and τ), the mean file size being 64 RTT pipes (800KB) or 40 RTT pipes (500KB). In Figure 6, the three analysis plots overlap since the analysis is insensitive to the individual values of c and τ for fixed $c\tau$. We also notice that the value of σ obtained from analysis will become close to zero as ρ approaches 0.75. As a general observation, we note that since our analytical model is stable only for $\rho < 0.75$, the analysis results cannot be expected to match those obtained from simulation for $\rho > 0.75$. In fact, in the TCP simulation, for ρ close to 1 the number of sessions will be large, there will be

TABLE I

VERIFICATION OF MODELLING ASSUMPTIONS: COMPARISON OF VARIOUS EXPECTATIONS, PROBABILITIES AND EPOCHS OF A CYCLE OBTAINED FROM ANALYSIS AND SIMULATION FOR THREE SETS OF PARAMETERS (SEE TEXT, SECTION VI-A) AND FOR INITIAL VALUE OF $n = 5$ AND 2.

Quantity	Simulation 1	Analysis 1	Simulation 2	Analysis 2	Simulation 3	Analysis 3
n	5	5	2	2	2	2
$E_n X$	3.99	3.42	15.72	16.8	3.53	3.7
$E_n V$	361	357	174	162	83.33	76.5
$T_0^{(n)}(S_0^{(n)})$	4.26 (348)	3.38 (313.5)	3.3(150)	2.812 (131.8)	1.5 (66)	1.25 (59)
$T_1^{(n)}(S_1^{(n)})$	5.75 (551)	4.73 (518)	8.85(649)	8.44 (659)	4 (285)	3.75 (292)
$T_2^{(n)}(S_2^{(n)})$	8.23 (1031)	6.98 (961)	-(-)	-(-)	-(-)	-(-)
$T_3^{(n)}(S_3^{(n)})$	13.3 (2360)	11.5 (2227)	-(-)	-(-)	-(-)	-(-)
$T_4^{(n)}(S_4^{(n)})$	- (-)	24.98 (9061)	-(-)	-(-)	-(-)	-(-)
$p_{n,1}$	0.0044366	0.004622	0.660	0.698	0.4177	0.48
$p_{n,2}$	0.0081643	0.010381	0.278	0.233	0.4256	0.38
$p_{n,3}$	0.039042	0.038824	0.062	0.069	0.157	0.140
$p_{n,4}$	0.14889	0.15840	-	-	-	-
$p_{n,5}$	0.48625	0.54067	-	-	-	-
$p_{n,6}$	0.3132	0.2471	-	-	-	-

frequent losses and timeouts, the session windows will be small, and the AIMD behaviour that our model essentially captures will not be evident.

Figures 6 and 7 show that when the mean file size is large in relation to the RTT pipe size, analysis results are within $10 - 20\%$ of the simulation results, for $\rho < 0.4$. The errors that we see are owing to the analysis ignoring the initial slow start that every session must go through. A large file size reduces the error that arises from this approximation.

The observations from the above plots are emphasised by Figure 8, in which we fix the link speed (in bits per second) and the mean file size (in bytes), and vary the RTT. This varies the RTT pipe and the ratio of the mean file size to the the RTT pipe. For these values, the analysis does not vary much with the parameters. It can be seen that, for a particular (and small) ρ, as the ratio of the file size to the RTT pipe decreases the value of σ obtained from simulations drops.

These observations point towards the importance of modelling the slow start phase when considering ephemeral sessions (as opposed to persistent sessions). It appears that the regenerative structure in our model can be easily modified to include the slow start phase, and we are doing this in ongoing work.

C. Relationship with the PS Model

In a previous paper [4] we study a scenario similar to that in Figure 1 with a zero propagation delay link and a large buffer size. It is shown there that for small mean file sizes and zero propagation delay, σ for ephemeral TCP connections is very close to that for a processor sharing queue (Equation 3). For small but positive τ and small buffer, the $R(t)$ process alternates between r_0 and c. With $T_i^{(n)}$ close to zero for small τ, this is as if $R(u) \approx \frac{r_0+c}{2} I_{\{N(u)>0\}}$. Thus for $r_0 = c/2$, and δ very small, $R(u) \approx 0.75cI_{\{N(u)>0\}}$. Since all the sessions get an equal share of $R(u)$, this is like processor sharing with a service rate of $0.75c$. We compare the σ values obtained from our analysis with those obtained using $\rho = \lambda/(\mu(0.75c))$, and

$$\sigma/c = 0.75\frac{1-\rho}{\rho}\log\frac{1}{1-\rho} \tag{11}$$

Figure 9 shows this comparison for $c = 1Mbps$, $1/\mu = 500KB$, $\tau = 0.1, 0.2,$ and 0.3 sec, i.e., RTT = 12.5 packets, mean file size = 40 RTT pipes, or RTT =25 packets, mean file size = 20 RTT pipes, or RTT = 37.5 packets; mean file size = 13.3 RTT pipes. It can be seen from the figure that σ obtained from analysis is very close to that obtained using the PS formula. This observation is interesting for two reasons:

1. The model that we have here is complex to analyse, and

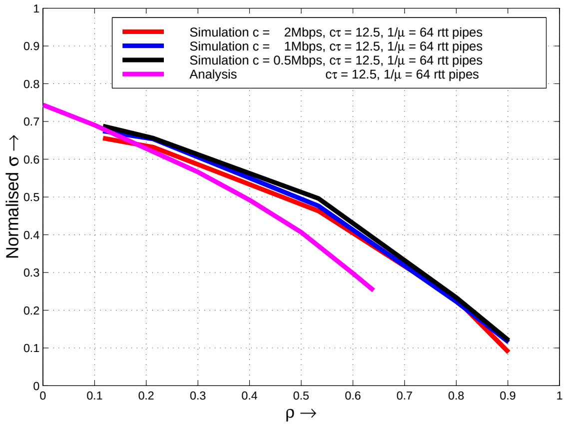


Fig. 6. σ from simulation and analysis for RTT pipe $(c\tau)$ = 12.5 packets; for various combinations of c and τ; $1/\mu$ = $800KB$= 800 packets = 64 RTT pipes.

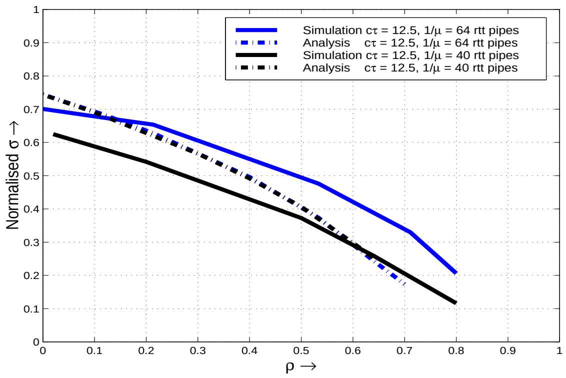


Fig. 7. σ from simulation and analysis for $c = 1Mbps$ and $\tau = 0.1sec$; i.e., RTT pipe = 12.5 packets; mean file size = 64 or 40 RTT pipes.

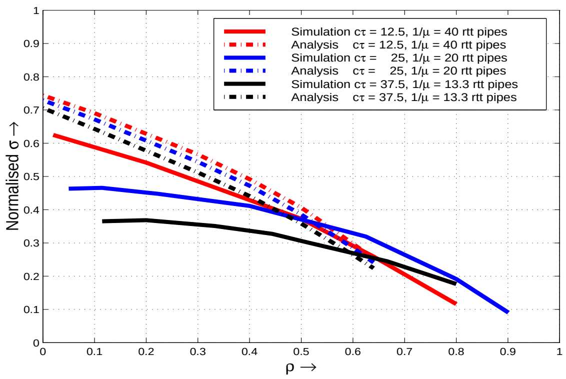


Fig. 8. σ from simulation and analysis for $1/\mu = 500KB$, $c = 1Mbps$ and $\tau = 0.1, 0.2, 0.3sec$; i.e., (RTT pipe, mean file size) = (12.5, 40), (25, 20), (37.5, 13.3).

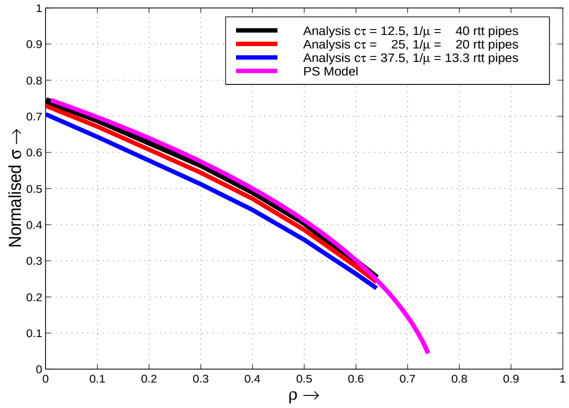


Fig. 9. Plot showing σ from analysis for $c = 1Mbps$, $1/\mu = 800KB$, $\tau = 0.1$ sec, or $\tau = 0.30$ sec (i.e., RTT pipe = 12.5 packets; mean file size = 64 RTT pipes, or RTT pipe = 37.5 packets; mean file size = 21.3 RTT pipes), and from the Processor Sharing formula (Equation 11) with service rate 0.75c.

being able to compute σ from a simple closed form formula is very useful.
2. The observation suggests when the file sizes are large compared to the RTT pipe, and ρ is not large, that TCP controlled bandwidth sharing may be approximately equivalent to processor sharing with a scaled down link capacity.

VII. Conclusions

We have developed an analytic method to evaluate the performance of a general AIMD algorithm for controlling bandwidth sharing among ephemeral sessions. We have found that the model captures the congestion avoidance phase of TCP quite well. The model provides a good estimate of the bandwidth shares provided by TCP when the mean transfer volumes are large compared to the RTT pipe size, and the normalised offered load is less than 0.4. Also the average bandwidth share calculated from the model is close to that obtained from the processor sharing model with a link capacity scaled by 0.75. Our investigations, however, point to the importance of capturing the slow start phase, as this is necessary for accurate prediction of the performance with ephemeral sessions when the file sizes are not large compared to the RTT pipe.

Our ongoing work includes incorporation of a slow start model into our fluid model of AIDM congestion control, and exploration of the possibility of insensitivity to file size distributions (suggested by the fact that the analysis yields results close to the Processor Sharing model).

APPENDIX

Proof of Theorem IV.1

Define a_k and d_k as the k^{th} session's arrival and departure epochs. Associate with session k the time function

$$f_k(u) = \begin{cases} \frac{1}{N^2(u)} & u \in [a_k, d_k) \\ 0 & \text{otherwise,} \end{cases}$$

Using Brumelle's Theorem [10], we have

$$\lim_{t\to\infty} \frac{1}{t}\int_0^t \sum_{k=1}^{\infty} f_k(u)du = \lambda \lim_{n\to\infty} \frac{1}{n}\sum_{k=1}^{n}\int_0^{\infty} f_k(u)du$$

i.e., using Equation 3,

$$\frac{1}{\rho}\sum_{n=1}^{\infty}\frac{\pi(n)}{n} = \frac{1}{E(X)}\lim_{n\to\infty}\frac{1}{n}\sum_{k=1}^{n}\int_{a_k}^{d_k}\frac{1}{N^2(u)}du \quad (12)$$

Consider the right hand side of Equation 12. Using the Cauchy-Schwartz inequality,

$$\int_{a_k}^{d_k}\left(\frac{1}{N(u)}\right)^2 du \int_{a_k}^{d_k} 1^2 du \geq \left(\int_{a_k}^{d_k}\frac{1}{N(u)}.1du\right)^2, \quad (13)$$

Since $W_k = d_k - a_k$, and, due to the processor sharing service, $X_k = \int_{a_k}^{d_k} \frac{1}{N(u)}du$, we can rewrite Equation 13 as

$$\int_{a_k}^{d_k}\left(\frac{1}{N(u)}\right)^2 du \geq \frac{X_k^2}{W_k}. \quad (14)$$

We now take our cue from the strict inequality condition in Equation 14 to prove that, w.p.1, and, for

$$U_k = \int_{a_k}^{d_k}\left(\frac{1}{N(u)}\right)^2 du \quad \text{and} \quad V_k = \frac{{X_k}^2}{W_k}.$$

Thus we need to show that, w.p.1,

$$\lim_{n\to\infty}\frac{1}{n}\sum_{k=1}^{n} U_k > \lim_{n\to\infty}\frac{1}{n}\sum_{k=1}^{n} V_k.$$

For fixed $\epsilon > 0$, let k_j index those (tagged) sessions that satisfy the following conditions:

1. A tagged session arrives to an empty queue,
2. A tagged session brings at least 44ϵ units of data,
3. Exactly one other session arrives during the sojourn of a tagged session
4. The only session to arrive during the sojourn of a tagged session arrives after the tagged session has transferred half of its data but before it has transferred 3/4 of its data.
5. The arriving session has so much of data to be transferred that it does not finish before the tagged session finishes.

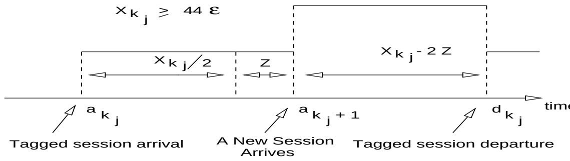


Fig. 10. A tagged session, and the single arriving session

A typical scenario for a tagged session is shown in Figure 10; in this figure the solid line is $N(u)$.
It is easily shown that [11], $U_{k_j} - V_{k_j} \geq \epsilon > 0$. Let $m(n) = |\{ k \leq n : \exists\, j \geq 1\ s.t.\ k = k_j \}|$ Using the fact that $U_{k_j} \geq V_{k_j} + \epsilon$, and $U_k \geq V_k$ for $k \neq k_j$, we have,

$$\lim_{n\to}\frac{1}{n}\sum_{k\,=\,1}^{n} U_k \geq \lim_{n\to\infty}\frac{1}{n}\sum_{k=1}^{n} V_k + \lim_{n\to\infty}\frac{m(n)}{n}\epsilon$$

Letting $G(.)$ denote the distribution function of the amount of data in a session, we have, w.p.1., $\lim_{n\to\infty}\frac{m(n)}{n} =$

$$\pi(0)\int_{44\epsilon}^{\infty}\int_{\frac{x}{2}}^{\frac{3x}{4}}\left(1-G(x-t)\right)\left(\exp{-\lambda 2(x-t)}\right)\lambda\exp{-\lambda t}\,dtdG(x).$$

Let, the right hand side integral be denoted by p, and observe that $p > 0$. Hence, w.p.1, we have

$$\lim_{n\to\infty}\frac{1}{n}\sum_{k\geq\,1}^{n} U_k \geq \lim_{n\to\infty}\frac{1}{n}\sum_{k=1}^{n} V_k + p\epsilon > \lim_{n\to\infty}\frac{1}{n}\sum_{k=1}^{n} V_k.$$

Hence the proof. •

ACKNOWLEDGEMENTS

The authors would like to acknowledge Munish Goyal, discussions with whom resulted in a simple expression (Equation 4) for the performance measure used.

REFERENCES

[1] T.V.Lakshman D.P.Heyman and A.L.Neidhardt, "New method for analyzing feedback protocols with applications to engineering web traffic over the Internet.," *Proceedings of ACM SIGMETRICS*, 1997.

[2] Arthur W. Berger and Yaakov Kogan, "Dimensioning bandwidth for elastic traffic in high-speed data networks," *Draft*, 1998.

[3] Tae-Jin Lee Gustavo De Veciana and Takis Konstantopoulos, "Stability and performance analysis of networks supporting services with rate control-could the Internet be unstable," *Proc. IEEE INFOCOM'1999*, April 1999.

[4] Arzad A. Kherani Anurag Kumar and R. Shobhanjali., "On processor sharing as a model for TCP controlled elastic sessions," *Submitted to GLOBECOM'2000*, 2000.

[5] Patrick Brown, "Resource sharing of TCP connections with different round trip times.," *Proc. IEEE INFOCOM'2000*, April 2000.

[6] Leonard Kleinrock, *Queueing Systems*, vol. 2, John Wiley and Sons, 1996.

[7] J.A.Morrison, "Response time distribution for a processor sharing system," *SIAM Journal of Applied Mathematics*, vol. 45, no. 1, pp. 152–167, 1985.

[8] J.W. Roberts and L. Massoulie, "Bandwidth sharing and admission control for elastic traffic," *ITC Specialist Seminar, Yokohama*, October 1998.

[9] T.V. Lakshman and Upamanyu Madhow, "The performance of TCP/IP for networks with high bandwidth delay products and random loss," *IEEE/ACM Transactions on Networking*, vol. 5, no. 3, pp. 336–350, June 1997.

[10] Ronald W.Wolf, *Stochastic Modelling and The Theory of Queues*, Prentice Hall, Englewood Cliffs, 1989.

[11] A.A.Kherani, ," M.S. thesis, Indian Institute of Science, January 1999.