Discrete Event Modelling and Performance Evaluation of Versions of TCP: Lossy Links and Rate Feedback (extended abstract*)

Anurag Kumar,† S.G. Sanjay, and Aditya Karnik
Dept. of Electrical Communication Engg.
Indian Institute of Science, Bangalore 560 012, INDIA
email: anurag@ece.iisc.ernet.in

1 Introduction

The Internet protocol suite's Transmission Control Protocol (TCP) adapts IP's unreliable and nonsequential packet delivery into a reliable and sequential service. Apart from this basic function, TCP serves two other functions: (i) sender-receiver flow control, and (ii) congestion control. For both these functions, TCP uses an adaptive window mechanism that is triggered by packet losses and time-outs. In fact, the end-to-end congestion control exercised by TCP is the only congestion control mechanism in the current Internet packet transport technology. With the growing importance of the Internet as the most important vehicle for transporting store-and-forward information, it is important to understand the performance, the limitations, and the evolution of TCP's congestion control mechanism. While a lot of the work on TCP congestion control is measurement or simulation based, recently a considerable amount of research has been done on analytical performance evaluation of this important protocol. This work includes TCP performance over large bandwidth-delay networks [LM97], over networks with lossy links, as would arise in wireless networks, [Kum98, KH98], and over ATM's ABR service [SK98]; simple formulas for TCP throughput with random loss and propagation delays are reported in [PFTK98].

In our work we have used an analysis approach based on identifying certain Markov renewal epochs in the congestion window evolution process of a TCP session. The number of packets successfully transmitted during each Markov renewal cycle is viewed as a reward, and then the TCP throughput is computed as the reward rate in this Markov Renewal Reward process. In previously published work, we have applied this approach to analysing the following scenarios:

- A TCP connection over a local area link that introduces Bernoulli packet losses [Kum98].
- A TCP connection over a local area link that has a Rayleigh fading type behaviour, modelled as a two state Markov modulated packet loss process [KH98].
- A TCP connection over ATM/ABR over a wide area link with time varying link capacity, modelled as a two state Markov modulated process [SK98]. In this work it was assumed that the TCP window adaptation protocol is unchanged, that the ABR connection is end-to-end, and only the ABR source (below TCP) adapts its transmission rate to the rate feedback.

In this paper, we describe our analysis approach as applied to local and wide area situations, and provide results from a new application of this approach to the analysis of a wide-area TCP connection with rate controlled window adaptation, i.e., explicit rate feedbacks from the network are used in the congestion window adaptation. We provide analytical results when a (tagged) persistent rate controlled TCP session shares a wide-area bottleneck link with other elastic ephemeral sessions. Each such session brings a random amount of data to transfer, and the link rate is shared fairly between the tagged session and the ephemeral sessions; hence the tagged TCP session is fed back a time varying explicit rate, and adapts its window accordingly.

2 The Analysis Approach

The analysis needs to keep track of different variables in the local and wide area situations. In the next two subsections we outline the analysis approach for each case.

*IFIP Performance'99 (Special Session on TCP Performance), Istanbul, August 1999
†Work supported, in part, by research grants from Nortel Networks.

2.1 Local Network; Lossy Link

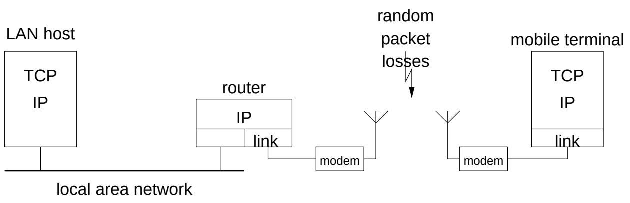

Figure 1: A LAN host with a TCP connection to a mobile host over a lossy link.

The motivating scenario is shown in Figure 1. A host on a fast LAN is doing a bulk transfer to another host attached to the LAN by a slower and lossy link. The end-to-end propagation delay is much smaller than the packet transmission times on either link. We identify random epochs $(T_1, T_2, \cdots, T_k, \cdots)$ as follows. Let $T_0 = 0$, and the connection starts off with the congestion window $W = 0$. As packets are transmitted and acked, the window grows (slow-start, if necessary, and then congestion avoidance). Eventually a loss occurs on the lossy link at the epoch l_1; we denote the window achieved at this epoch by X_1. Owing to the local area and high speed LAN assumptions, this window includes all increments due to any successful transmissions before the lost packet. Some random time later (depending on the version of the protocol and the recovery method used), at the epoch T_1, the normal congestion window algorithm resumes with a slow-start threshold of $\lceil \frac{X_1}{2} \rceil$. In this way we identify the sequence of random vectors $((X_0, T_0), (X_1, T_1), \cdots, (X_k, T_k), \cdots)$; see Figure 2. With Bernoulli packet loss [Kum98], or Markov modulated packet loss [KH98] and appropriate assumptions (the window increase during congestion avoidance is taken as a probabilistic increase), we can show that (i) $\{X_k\}$ is a Markov chain, and (ii) $\{(X_k, T_k)\}$ is a Markov renewal process. Denoting the number of successful packets in the kth cycle by a "reward" V_k, we find that the V_k depends only on $X_{(k-1)}$, and we have a Markov renewal reward process. Letting U denote the random variable for the cycle length, and taking expectations under the stationary distribution $\pi(\cdot)$ of $\{X_k\}$, the throughput in packets per second is given by

$$\gamma = \frac{E_\pi V}{E_\pi U}$$

The various versions of TCP are distinguished by the structure of the Markov chain $\{X_k\}$, and the cycle times $\{T_k\}$; e.g., for the original Van Jacobson algorithm (TCP "Old"Tahoe) $T_k = l_k + \text{timeout}$.

In spite of many approximations, the analysis is quite accurate as demonstrated in Figure 3 (taken from [Kum98]), where analytically obtained results are compared with those obtained from actual TCP code running over an emulated lossy link.

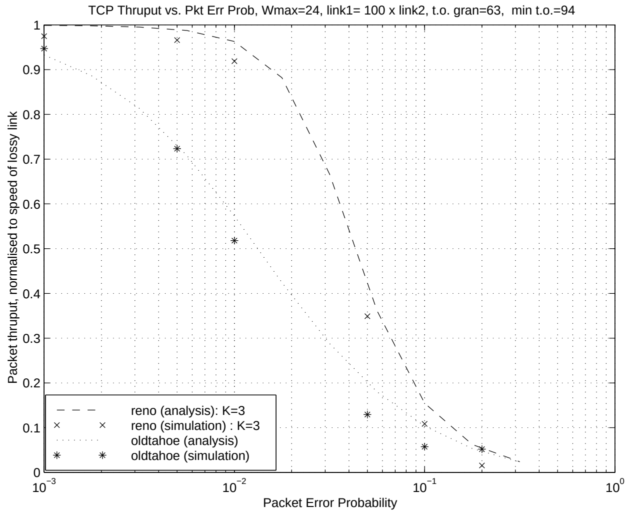

Figure 3: Throughput of TCP OldTahoe and TCP Reno vs. packet loss probability; comparison of simulation and analysis; (maximum congestion window is 24 packets; the LAN is 100 times faster than the lossy link; K is the duplicate ack threshold for fast retransmit).

2.2 Wide Area Network; Time Varying Link Capacity

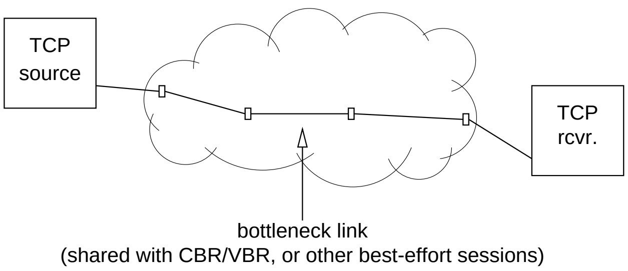

Figure 4: A TCP connection over the wide-area Internet, with a bottleneck link shared with CBR/VBR connections, or other (ephemeral) best-effort connections.

A different modelling approach is necessitated by the scenario depicted in Figure 4. Owing to the connection spanning a wide area, we need to model the round-trip propagation delay, Δ. This yields the model shown in Figure 5. The bottleneck rate variations are either due to CBR/VBR sessions sharing the link with the TCP connection [SK98], or because of other TCP connections sharing the same link and getting their fair share of the link bandwidth. We can model either TCP over ABR, where the ABR transmitter at the source adapts the packet emission rate according to the rate feedbacks (and hence there is buffering of packets at the

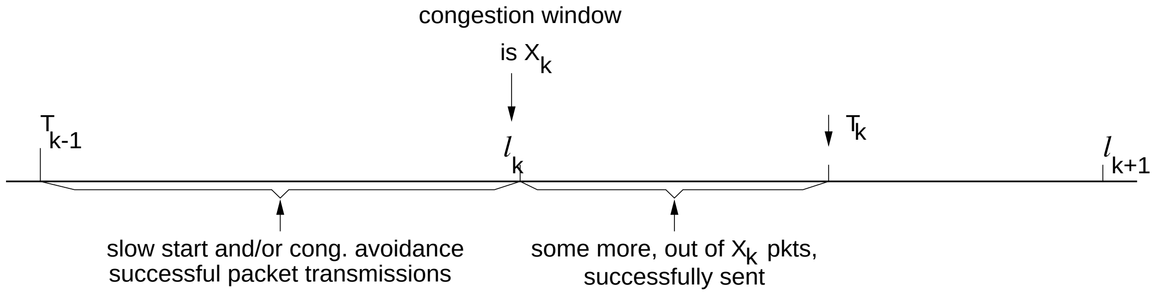

Figure 2: The epochs $\{T_k\}$ and the Markov chain $\{X_k\}$.

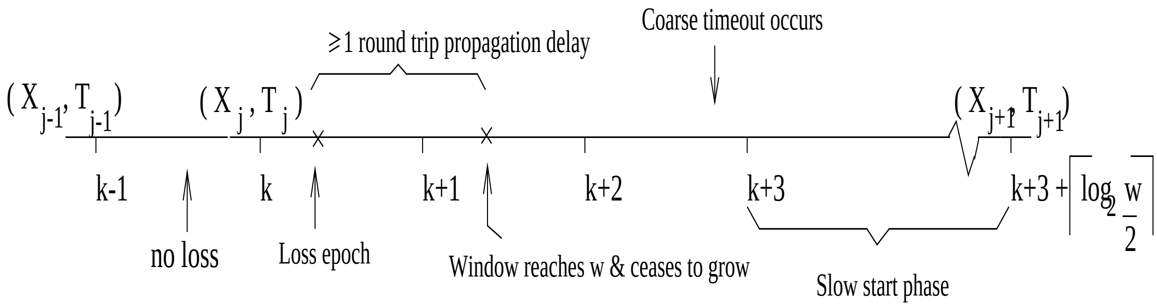

Figure 6: The embedded process $\{(X_j, T_j), j \geq 0\}$

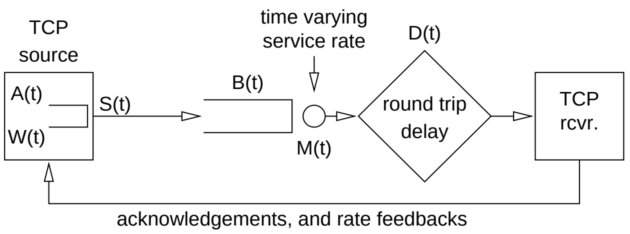

Figure 5: Model for a wide-area TCP connection over a bottleneck link with time varying service capacity.

source), or Rate Adaptive TCP (RATCP) with "native" IP rate feedback and TCP window adaptation according to this rate. In the former case, we need to keep track of the number of TCP packets buffered at the source ($A(t)$; see Figure 5), buffered at the bottleneck link ($B(t)$), and in the round-trip delay ($D(t)$). We also need to model the bottleneck link rate modulating process ($M(t)$), the delayed rate feedback ($S(t)$), and, in the case of RATCP, the congestion window process (($W(t)$). Analysis of such a continuous time multidimensional process would not be tractable. So we consider the processes embedded at multiples of Δ, denoted by $(t_0, t_1, \cdots, t_k, \cdots)$; any packet in the propagation delay at t_k would have departed the propagation delay by $t_{(k+1)}$, and any packet that arrives in the propagation delay in the interval $(t_k, t_{(k+1)})$ would still be in the propagation delay at $t_{(k+1)}$. Hence we do not need to keep track of the residual "service" times of packets in the delay queue.

Let the multidimensional process, embedded at the $\{t_k\}$ epochs be denoted by $\{Z_k\}$; e.g., in the case of TCP over ABR $Z_k = (A_k, B_k, D_k, M_k, S_k)$, where A_k = the number of packets in the source ABR buffer, B_k = the number of packets in the link buffer, D_k = the number of packets in the propagation delay, M_k = the link service rate, and S_k = the ABR source rate. We identify a sequence of random epochs $T_k \in (t_0, t_1, t_2, \cdots)$ as follows (see Figure 6):

$$T_0 = t_0 = 0 \text{ and } X_0 = Z_0 \tag{1}$$

For $k \geq 1$,

$$T_k = \begin{cases} T_{k-1} + \Delta & \text{if no loss occurs} \\ & \text{in } (T_{k-1}, T_{k-1} + \Delta) \\ T_{k-1} + (3+ & \text{if loss occurs} \\ \text{slow start duration})\Delta & \text{in } (T_{k-1}, T_{k-1} + \Delta) \end{cases} \tag{2}$$

and $X_k = Z_{T_k}$. We can argue that $\{X_k, k \geq 0\}$ is a Markov chain. Further, given T_k and X_k, the distribution of T_{k+1} can be computed without any knowledge of its past history. Hence, the process $\{(X_k, T_k), k \geq 0\}$ is a Markov Renewal Process (MRP). It is this MRP that is our model for the TCP connection over the link with time varying capacity.

Given the Markov Renewal Process $\{(X_k, T_k), k \geq 0\}$, we associate with the kth cycle (T_k, T_{k+1}) a "reward" V_k equal to the number of packets successfully transfered in the interval. Let $\pi(x)$ denote the stationary probability distribution of the Markov chain $\{X_k, k \geq 0\}$. Define $U_k = \frac{(T_{k+1} - T_k)}{\Delta}, k \geq 0$, as the

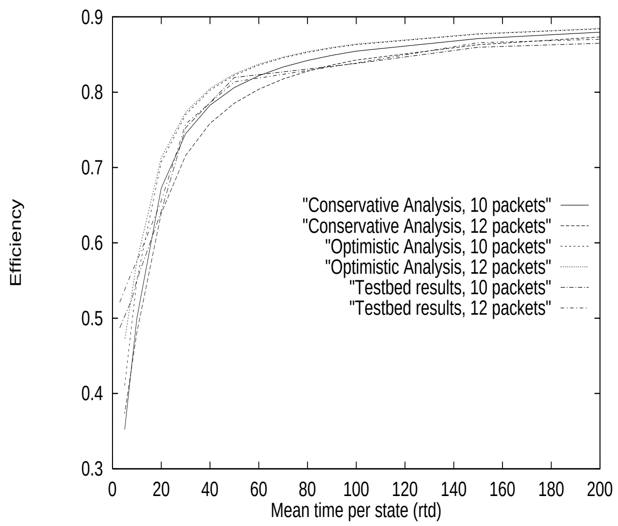

Figure 7: Analysis and Simulation results for throughput of TCP over ABR. Two state Markov modulated bottleneck link rate; the Markov chain spends the same mean time in each state. Efficiency (the ratio of TCP throughput to the average link rate) is plotted against the mean time the Markov chain spends in each state (expressed as a multiple of Δ). The round trip propagation delay is 40 packet transmission times at the fast link rate. The bottleneck link buffers are either 10 or 12 packets.

cycle length process. Then, by the Markov renewal-reward theorem, we have the throughput γ given by

$$\gamma = \frac{E_\pi V}{E_\pi U} \tag{3}$$

where $E_\pi(\cdot)$ denotes the expectation w.r.t. the stationary distribution $\pi(x)$.

In Figure 7 (taken from [SK98]) we show how the approximate analysis works in comparison to measurements on a test-bed. A conservative and an optimistic accounting of the reward in each cycle lead to corresponding estimates of throughput in the figure. It is seen that for slow variations of the link rate (slower than once every 10 Δ), the analysis matches well with the measurements.

3 Modelling and Analysis of Rate Adaptive TCP

We propose a procedure for incorporating rate feedbacks into the TCP window adaptation algorithm, and then model and analyse such a TCP connection (the tagged connection) over a bottleneck link shared with other TCP connections that arrive and depart.

We assume that, at time t, the bottleneck link serves the tagged TCP session at the fair rate $R(t)$, which is simply the link rate divided by the number of active sessions. Then, letting Δ denote the round trip time, and $\hat{\Delta}(t)$ the estimate of the round trip time, we can calculate an estimate of the desirable window

$$W^{rate}(t) = R(t - \Delta) \times \hat{\Delta}(t) + \beta, \tag{4}$$

where $W^{rate}(t)$ is the rate based window computed at the TCP source at time t, and β is a target buffer backlog. In the analysis we assume that the round-trip propagation delay Δ is known at the TCP transmitter. Thus the results we get are indicative of the best performance that is possible with rate adaptive windows.

In this work, we only consider the original TCP window adaptation (i.e., fast retransmit and fast recovery are not modelled). The rate adaptive window adaptation strategy is the following:

- The basic Slow Start and Congestion Avoidance algorithms remain unchanged, except that if during congestion avoidance W^{cong} becomes less than *ssthresh* (due to a W^{rate} feedback) then slow start is not reinitiated. This is reasonable, since it is as if the *ssthresh* has been adjusted downward, and we are now just entering congestion avoidance.

- Slow start is carried out either at connection startup, or at the restart after packet loss. We use the rate information for setting the slow start parameters: W^{cong} at timeout is set to 1, and the *ssthresh* is set to the value of W^{rate} at the timeout epoch. If during slow start $W^{rate} < W^{cong}$ then the window is dropped to W^{rate}, and congestion avoidance is entered.

- W^{cong} adapts to W^{rate} (at a given time t) in the congestion avoidance phase as, $W^{cong}(t+) = \min\{W^{cong}(t), W^{rate}(t)\}$. If the congestion window reduces as a result of $W^{rate}(t) < W^{cong}(t)$, then it means that more than the desirable number of packets are in the network. Acks following such a window reduction do not cause the window to increase until the number of unacknowledged packets corresponds to the rate window. This adds a phase of inactivity in the modified TCP. Normal congestion avoidance behaviour continues after the number of outstanding packets matches the new congestion window.

We call this modification of TCP, Rate Adaptive TCP (RATCP).

We analyse an RATCP session in the scenario shown in Figure 8. $M(t)$ is the number of other sessions sharing the link with the RATCP session. These sessions are assumed to adapt their rates ideally to the fair share of the bottleneck rate $\frac{C}{M(t)+1}$. The analysis approach discussed in Section 2.2 is used. The competing sessions are assumed to arrive at random epochs that are geometrically distributed over multiples of Δ, and bring exponentially distributed amounts

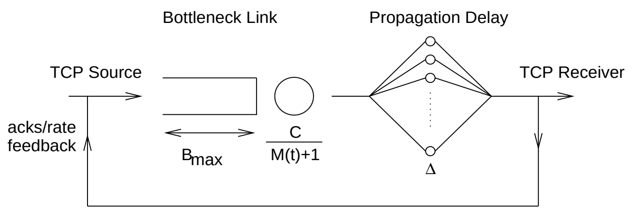

Figure 8: Closed Queueing network representation of the tagged TCP session. The bottleneck link is modelled as a finite buffer and a time-varying server. Round trip propagation time (fixed; Δ) is modelled as an infinite server.

of data to be transferred. With this we model the number of competing sessions embedded at the $\{t_k\}$ epochs (see Section 2.2) as a Markov chain. Then we define the embedded process (at the $\{t_k\}$ epochs) $Z_k = (B_k, D_k, W_{k-}^{cong}, W_k^{rate}, M_k)$, where B_k = the number of packets in the link buffer, D_k = the number of packets in the propagation delay, W_{k-}^{cong} = is the TCP congestion window just prior to the embedded epoch, W_k^{rate} = the window estimate based on rate feedback (see above), and M_k = the number of competing ephemeral sessions.

Note that owing to the assumption of an infinite rate link between the TCP source and the bottleneck link (see Figure 8), if W^{cong} is not rate limited then it corresponds to the number of unacknowledged packets in the network, i.e., $W_k^{cong} = B_k + D_k$. However, with the adaptation to the rate window (which reduces W^{cong}), W^{cong} may not correspond to the number of unacknowledged packets in the network. As explained above, in RATCP this causes the source to idle, i.e., W^{cong} does not increase and no new packets are inserted in the network till the excess packets are removed from the network; during this phase the left edge of the window moves up by one packet for each ack received, and the window stays constant. Further transmission is possible only after the right edge of the window moves past the last unacknowledged packet sent.

To study the evolution of the process $\{Z_k\}$, let $Z_k = z = (b, d, w^{cong}, w^{rate}, m)$, and we identify 3 cases that can occur in the next interval (t_k, t_{k+1}). With Z_k as defined, $W_{k+}^{cong} = \min\{w^{cong}, w^{rate}\}$. Now, observe that in congestion avoidance (slow-start is modelled separately),

1. If $W_{k+}^{cong} < b + d$, then $h = b + d - W_{k+}^{cong}$ packets need to be removed from the network before congestion avoidance resumes. Since the number of acks that can be received in (t_k, t_{k+1}) is $\leq d$, we first have the following two cases.

   - **Case 1:** $d < h \Rightarrow$ since not enough acks are received, the source is inactive throughout (t_k, t_{k+1}), and packet loss does not occur in (t_k, t_{k+1}); W_{k+}^{cong} is constant.

   - **Case 2:** $d > h \Rightarrow$ congestion avoidance commences after h acknowledgements are received; there may be loss in (t_k, t_{k+1}) after h acks are recived.

2. **Case 3:** $W_{k+}^{cong} = b + d \Rightarrow$ the congestion window does not drop, and congestion avoidance continues. There may be loss in (t_k, t_{k+1}).

The analysis proceeds as in Section 2.2 by obtaining the epochs T_k and the Markov chain $X_k = Z_{T_k}$. The transition probabilities of this Markov chain are obtained by examining each of the three cases described above. As in the previous analysis, we model the linear increase during congestion avoidance probabilistically. In the analysis, for simplicity, we do not consider adaptation to W^{rate} in the slow start phase, i.e., slow start terminates only after the congestion window reaches the slow start threshold. We model the losses in slow start, and also the extended slow start phase which occurs when the slow start threshold exceeds the capacity of the round trip pipe. We finally observe that $\{X_k, T_k\}$ is a Markov renewal reward process. Taking packet transmissions as rewards in evolution cycles, we calculate throughput as the reward rate. The details are provided in the thesis [Kar99].

To validate our analysis, we compared our analytical results with those obtained from a TCP test-bed. In this test-bed, the loopback driver in a Linux based PC is modified to provide the emulation of a serial link (buffering, transmission rate, and propagation delay). Loopback file transfers are then run within one machine, using actual Linux TCP code, modified according to RATCP. The simulation makes the following important assumptions in order to validate the analysis procedure:

- It is assumed that Δ is known at the TCP sender. In practice RATCP will have to make do with estimates of rtt. In ongoing work we will use the simulation to examine the effect of estimation errors.

- The exact rate feedback is artificially provided to the sender. In practice, some mechanism in the routers, and some number of bits in the IP packet headers will be needed to provide such feedback.

In Figure 9, we show the performance comparison between RATCP and TCP from the analysis and simulations. Efficiency is defined as the average throughput of the persistent session as a fraction of the average rate provided to it by the bottleneck link.The bottleneck link rate is 100KBytes/sec, Δ = 100ms, the link buffer size is 10 packets (500 byte packets), and the target buffer backlog, β, is 1 packet. The maximum number of sessions including the tagged TCP session

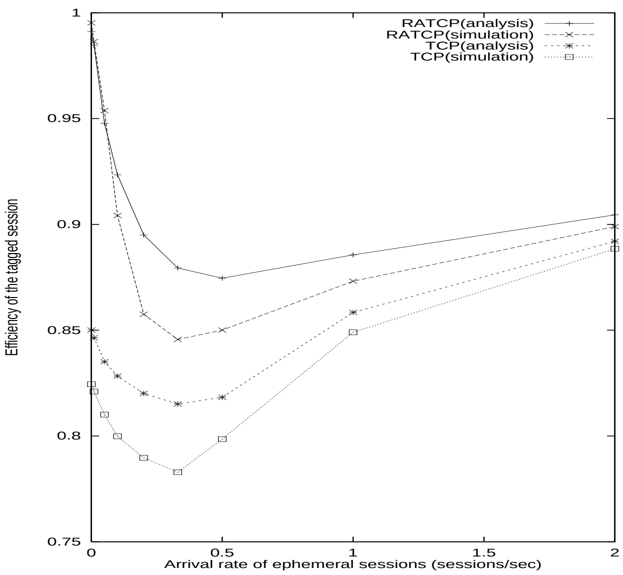

Figure 9: Efficiency variation with the ephemeral session arrival rate for RATCP and TCP; analysis and simulation.

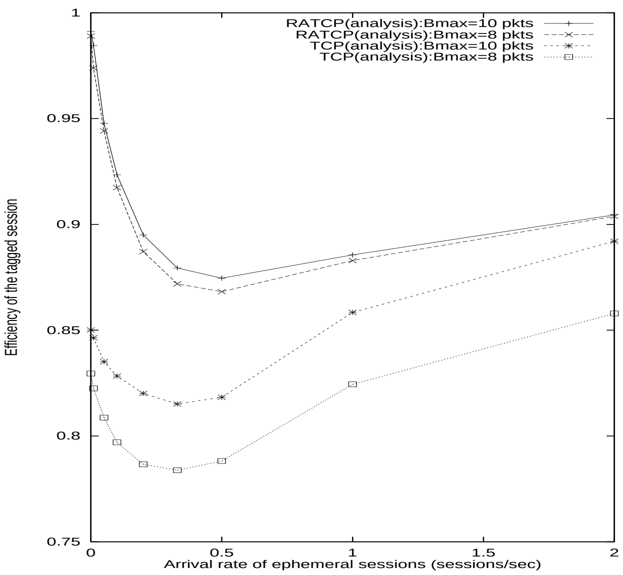

Figure 10: Efficiency variation of RATCP and TCP, with the mean session arrival rate for 2 buffers sizes: 10 pkts and 8 pkts; analysis results.

allowed on the link at any time is 4. This implies that, when the arrival rate of the ephemeral sessions is either very low or very high, the fair rate variations are slow, whereas for intermediate arrival rates the variations are fast. Note that the mean number of sessions on the link increases as the rate of arrival increases, but is bounded by 4.

It is seen that when the variations in the rate available to the persistent session are slow, i.e., when the rate of arrival of competing sessions is either very low or very high, analysis and simulation results match to within 1%. For rapid rate variations (intermediate arrival rates for the ephemeral sessions), the analysis overestimates the efficiency by about 3% to 5%. Thus, overall the analysis procedure captures the performance quite well.

We make the following observations:

- When the arrival rate of other sessions is low, RATCP yields an efficiency that is over 15% more than TCP. Since both RATCP and TCP recover conservatively from losses, the improvement with RATCP occurs because it suffers less losses. This is because RATCP adapts the window down owing to the explicit rate feedback, thus limiting the number of packets in the network, and hence reducing losses. TCP on the other hand builds a larger window, overshoots the optimal window, and must suffer a loss to adapt the window down.

- When there are rate variations comparable to the propagation delay, the performance of RATCP and TCP degrades due to mismatches between the sending rate and the available rate, leading to more losses. However, RATCP is able to contain packets losses to a smaller value as compared to TCP. Since the buffer backlog is only 1 packet in RATCP, it is not able to take advantage of the transient rate increases, and thus has a sharper decrease in the efficiency, with increasing ephemeral session arrival rate. We expect to see a better performance of RATCP in this region with a larger value of β.

- When the arrival rate of the competing sessions is higher, the mean number of sessions on the link increases. This implies that the rate available per session is small, and TCP needs to build a smaller window before a loss occurs. Thus the penalty for a packet loss is not as much as in the earlier case. In this region RATCP does not have a significant advantage over TCP.

- Figure 10 shows the analysis results for 2 buffer sizes: 8 packets and 10 packets. Because of the control over the number of packets in the network, the reduction in the buffers does not degrade the performance of RATCP as much as TCP. With the reduction in the buffer size, TCP incurs even greater losses.

References

[Kar99] Aditya Karnik. Performance of TCP congestion control with rate feedback:

TCP/ABR and rate adaptive TCP/IP. Master's thesis, ECE Department, Indian Institute of Science, Bangalore, India, January 1999.

[KH98] Anurag Kumar and Jack Holtzman. Comparative performance analysis of versions of TCP in a local network with a mobile radio link. *Sadhana, Proceedings of the Indian Academy of Sciences (Engineering)*, 23(1):113–129, February 1998.

[Kum98] Anurag Kumar. Comparative performance analysis of versions of TCP in a local network with a lossy link. *IEEE/ACM Transactions on Networking*, 6(4), August 1998.

[LM97] T.V. Lakshman and Upamanyu Madhow. The performance of TCP/IP for networkswith high bandwidth delay products and random loss. *IEEE/ACM Transactions on Networking*, 5(3):336–350, June 1997.

[PFTK98] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modeling TCP throughput: A simple model and its empirical validation. In *Proceedings ACM SIGCOMM'98*, 1998.

[SK98] S.G. Sanjay and Anurag Kumar. TCP over end-to-end ABR: A study of TCP performance with end-to-end rate control and stochastic available capacity. In *Proceedings of IEEE Globecom,98*. IEEE, November 1998.