A Comparison of Equation-Based and AIMD Congestion Control

Sally Floyd, Mark Handley, Jitendra Padhye
ACIRI

May 12, 2000

Abstract

This paper considers AIMD-based (Additive-Increase Multiplicative-Decrease) congestion control mechanisms that are TCP-compatible (i.e., that compete reasonably fairly with TCP), but that reduce their sending rate less sharply than does TCP in response to a single packet drop. The paper then briefly compares these smoother AIMD-based congestion control mechanisms with TFRC (TCP-Friendly Rate Control), which makes use of equation-based congestion control.

1 Introduction

This paper explores unicast, AIMD-based (Additive-Increase Multiplicative-Decrease) congestion control mechanisms that are TCP-compatible, in that they compete reasonably fairly with existing TCP, but that avoid TCP's halving of the congestion window in response to a single packet drop. TCP's congestion control mechanisms are a good choice for most current applications, as TCP is very effective at rapidly using bandwidth when it becomes available. However, for some applications the requirement for relatively smooth changes of the sending rate is more important than the ability to make opportunistic use of increases in available bandwidth. For such applications, a key reason not to use TCP's congestion control mechanisms is to avoid the abrupt halving of the sending rate in response to a single packet drop. This note explores TCP-compatible AIMD-based congestion control mechanisms that avoid this decrease in the sending rate by half in response to a single packet drop.

[FHPW00] presented TFRC (TCP-Friendly Rate Control), a mechanism for equation-based congestion control, where the sender directly adjusts its sending rate as a function of the packet loss rate reported by the receiver over some period of time. However, equation-based congestion control is not the only possible mechanism for TCP-compatible unicast congestion control with relatively smooth changes in the sending rate over time. Relatively smooth changes in the sending rate can also be accomplished by using AIMD congestion control with a moderate multiplicative decrease in the congestion window in response to a packet drop, or by using other families of congestion control algorithms with other functions for reducing the congestion window in response to a packet drop [BB00].

In this paper we let *AIMD(a, b) congestion control* refer to pure AIMD congestion control that uses an increase parameter a and a decrease parameter b. That is, after a loss event the congestion window is decreased from W to $(1-b)W$ packets, and otherwise the congestion window is increased from W to $W+a$ packets each round-trip time. We use the term *TCP(a, b) congestion control* to refer to TCP congestion control modified to use AIMD(a, b). Currently, TCP uses AIMD(1, 1/2) congestion control along with the congestion-control-related mechanisms of the retransmit timer and the exponential backoff of the retransmit timer in periods of high congestion. Given the long familiarity in the Internet with TCP, the most obvious choice for a congestion control mechanism that reduces its sending rate more smoothly than TCP would be TCP(a, b) with a decrease parameter b less than 1/2. Thus, a natural question in evaluating equation-based congestion control is to compare it with TCP(a, b) congestion control.

2 TCP-Friendly Rate Control

TFRC is an equation-based congestion control mechanism where the sender adjusts its sending rate as a function of the measured loss event rate. For most unicast flows that want to transfer data reliably and as quickly as possible, the best choice is simply to use TCP directly. TFRC is designed for applications that would prefer to maintain a slowly-changing sending rate, while still being responsive to network congestion over longer time periods (seconds, as opposed to fractions of a second). We describe TFRC briefly in this section; TFRC is described in detail in [FHPW00], and is implemented in the NS simulator.

In TFRC, the sender uses a response function that maintains a sending rate that is TCP-compatible, in that in steady-state it uses no more bandwidth than a conformant TCP with a comparable loss event rate, round-trip time, and MTU.

In TFRC, the sender begins with a slow-start procedure similar to TCP, roughly doubling its sending rate each round-trip time until congestion is encountered. Once a packet loss has been detected, the receiver estimates the loss event rate, where a *loss event* consists of one or more packets dropped within a single round-trip time. The TFRC sender uses the TCP response function

$$T = \frac{s}{R\sqrt{\frac{2p}{3}} + t_{RTO}(3\sqrt{\frac{3p}{8}})p(1+32p^2)} \tag{1}$$

from [PFTK98], which gives an upper bound on the sending rate T in bytes/sec, as a function of the packet size s, round-trip time R, steady-state loss event rate p, and the TCP retransmit timeout value t_{RTO}. The TFRC sender uses the response function (1) to adjust its sending rate as a function of the measured round-trip time and the reported loss event rate. The use of the TCP response function ensures that TFRC competes fairly with TCP over long time scales.

A key issue in the design of TFRC concerns the time constants used in estimating the loss event rate. These time constants determine TFRC's transient response to congestion, or to the sudden absence of congestion. TFRC estimates the loss event rate over the eight most recent loss intervals, where a *loss interval* is defined as the packets transmitted between two consecutive loss events. From [FHPW00], TFRC requires roughly five round-trip times to reduce its sending rate in half. In the absence of congestion, TFRC increases its sending rate by at most 0.28 packets per round-trip time.

3 AIMD congestion control

In this section we review the behavior of AIMD(a, b) congestion control.

3.1 The deterministic AIMD response function

To explore the AIMD response function for AIMD(a, b) congestion control, in this section we consider the deterministic AIMD model described in [Flo91, FF99], rather than the stochastic TCP model from [PFTK98]. The deterministic AIMD model assumes that a packet is dropped each time the congestion window reaches W packets, as shown in Figure 1. In contrast, the stochastic TCP model from [PFTK98] assumes that packets are dropped with a random probability p, and takes into account the role of TCP retransmit timeouts. The stochastic TCP model from [PFTK98] gives a considerably more accurate model for TCP; however, the deterministic AIMD model is useful for focusing on the role of the increase and decrease parameters a and b in AIMD congestion control.

For the deterministic AIMD model, we define a *congestion epoch* as a period beginning with a congestion window of $(1-b)W$ packets. The congestion window is increased additively by a packets per round-trip time up to a congestion window of W, when a packet is dropped. The congestion window

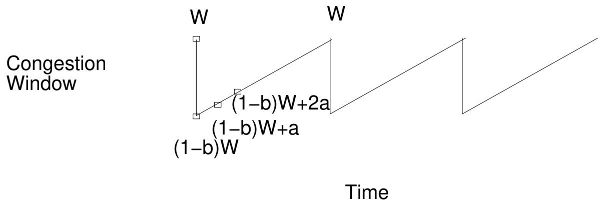

Figure 1: TCP's congestion window in steady-state.

is then decreased multiplicatively back to $(1-b)W$. Each congestion epoch consists of

$$\frac{b}{a}W + 1$$

round-trip times. We let S denote the sending rate in packets per RTT, and we let T denote the sending rate in packets per sec. For the deterministic model, the average sending rate over one congestion epoch is

$$S = \frac{2-b}{2}W \tag{2}$$

packets per RTT, or

$$T = \frac{2-b}{2R}W \tag{3}$$

packets per second.

This gives a total of

$$\left(\frac{b}{a}W + 1\right)S = \left(\frac{b}{a}W + 1\right)\frac{2-b}{2}W$$
$$\approx \frac{b(2-b)}{2a}W^2$$

packets in one congestion epoch, with one of these packets dropped at the end of the congestion epoch. The packet drop rate p is therefore

$$p = \frac{2a}{b(2-b)W^2 + a(2-b)W} \tag{4}$$

or, using an approximation,

$$p \approx \frac{2a}{b(2-b)W^2}. \tag{5}$$

Using the approximation in Equation (5), we get the following:

$$W \approx \sqrt{\frac{2a}{b(2-b)p}}. \tag{6}$$

We define the *AIMD response function* $T_{a,b,R,p}$ or T as the AIMD flow's steady-state sending rate T in packets per second in the deterministic model. The response function is a function of the decrease/increase parameters a and b, the round-trip time R, and the packet drop rate p. We let $\hat{T}_{a,b,R,p}$ or $\hat{T}$ denote the approximate solution to $T_{a,b,R,p}$. Substituting Equation (6) into Equation (3), we get the following:

$$\hat{T} = \frac{\sqrt{2-b}\sqrt{a}}{\sqrt{2b}R\sqrt{p}}. \tag{7}$$

The precise version of the AIMD response function in Equation (7), from Appendix A, is as follows:

$$T = \frac{pa(-2+b) + \sqrt{p\,(-2+b)\,a\,(pba - 8\,b - 2\,pa)}}{4\,pbR}$$

3.2 AIMD(a, b) and AIMD(1, $1/2$) congestion control

In this section we give rough guidelines for a and b parameters for which AIMD(a, b) congestion control is compatible with AIMD(1, $1/2$).

Applying the AIMD response function from Equation (7) to AIMD(1, $1/2$), we get the well-known version of the TCP response function [TCP]:

$$\hat{T}_{1,1/2,R,p} = \frac{\sqrt{1.5}}{R\sqrt{p}}. \tag{8}$$

In order for AIMD(a, b) to have the same long-term sending rate in relationship to the packet drop rate as AIMD(1, 1/2), we would like to have the same response functions:

$$T_{a,b,R,p} = T_{1,1/2,R,p}.$$

As a first step, we can specify values for a and b that give the same approximate response functions:

$$\hat{T}_{a,b,R,p} = \hat{T}_{1,1/2,R,p},$$

or

$$\frac{\sqrt{2-b}\sqrt{a}}{\sqrt{2b}R\sqrt{p}} = \frac{\sqrt{1.5}}{R\sqrt{p}}.$$

This is equivalent to the following:

$$a = \frac{3b}{(2-b)}. \tag{9}$$

Thus, the approximate solution of the deterministic AIMD model suggests that AIMD($3/7$, $1/4$) and AIMD($1/5$, $1/8$) should each compete reasonable fairly with AIMD(1, 1/2).

Abandoning the approximate response functions, we could also specify values for a and b that give the same values for the exact response functions in the deterministic AIMD model:

$$T_{a,b,R,p} = T_{1,1/2,R,p},$$

or equivalently,

$$\frac{-2\,pa + pba + \sqrt{p\,(-2+b)\,a\,(pba - 8\,b - 2\,pa)}}{4\,pbR}$$

$$= \frac{0.43\sqrt{(3p+8)}}{\sqrt{pR}} - \frac{0.75}{R}.$$

Solving this with the mathematics package *maple* [WMI] shows that for decrease factors of $b = 1/4$ or $b = 3/7$, the prescribed value for a using the exact response functions is similar to the prescribed value using the approximate response functions. For example, for a decrease factor of $b = 1/8$, the approximate AIMD response function specifies an increase factor a of 0.2 for compatibility with AIMD(1, 1/2), while the exact AIMD response function specifies an increase factor a ranging from 0.197 to 0.192 as the loss event rate p ranges from 0.001 to 0.5.

4 A comparison of TCP and TCP(a, b) congestion control, for TCP-compatible congestion control

AIMD(a, b) congestion control could be implemented either in a congestion-window-based version such as TCP, or in mechanisms such as RAP [RHE99], which implements a rate-based variant of AIMD. So far, our simulations with AIMD congestion control have all been with TCP, and not with a rate-based variant such as RAP. Because the RAP implementation in NS does not include slow-start, and does not model the congestion control mechanisms of TCP's retransmit timers, we have not run simulations using RAP's rate-based AIMD congestion control. For simulations with moderate packet drop rates, we expect that RAP would give basically the same performance as AIMD with the same increase/decrease parameters.

The deterministic AIMD model suggests that TCP($1/5$, $1/8$) should compete fairly with standard TCP. In this section we use simulations to explore the relative fairness of TCP and TCP(a, $1/8$) congestion control, for various values of a.

Figure 2 shows simulations of standard SACK TCP flows competing with SACK TCP($1/5$, $1/8$) over a 60Mbps link. In these simulations n TCP and n TCP($1/5$, $1/8$) flows share a common bottleneck. The graph shows the throughput over the last 60 seconds of simulation, normalized so that a value of one would be a fair share of the link bandwidth. The graph displays a mark showing the throughput for each flow. The dashed and solid lines show the mean throughput of the TCP and the TCP($1/5$, $1/8$) flows, respectively. The lower graph in Figure 2 shows the average packet drop rate for each simulation. The simulations in Figure 2 show that with a 2-3% packet drop rate, TCP($1/5$, $1/8$) flows receive only 70% of the bandwidth received by TCP flows.

Figure 3 shows standard SACK TCP flows competing with SACK TCP($1/5$, $1/8$) over a 15Mbps link, and Figure 4 shows the same simulations using TCP($2/5$, $1/8$). The simulations show that TCP($2/5$, $1/8$) comes closer than

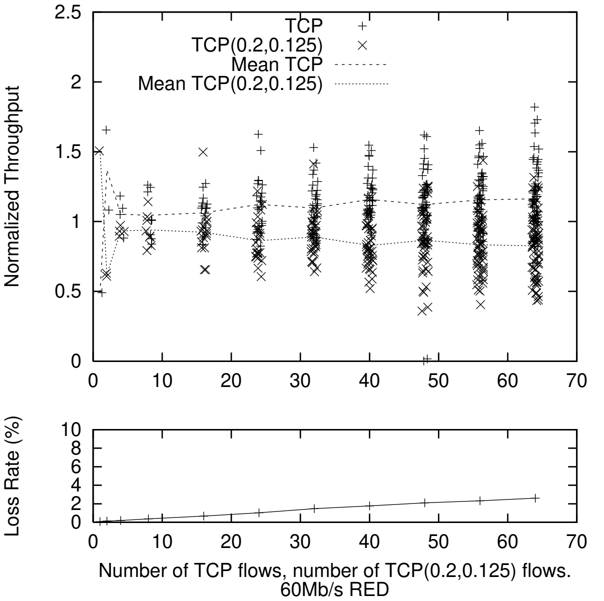

Figure 2: TCP competing with TCP($1/5$, $1/8$), with RED.

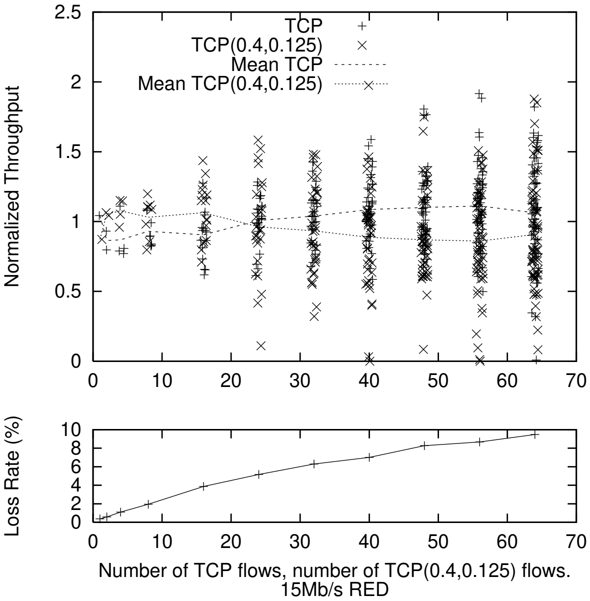

Figure 4: TCP competing with TCP($2/5$, $1/8$), with RED.

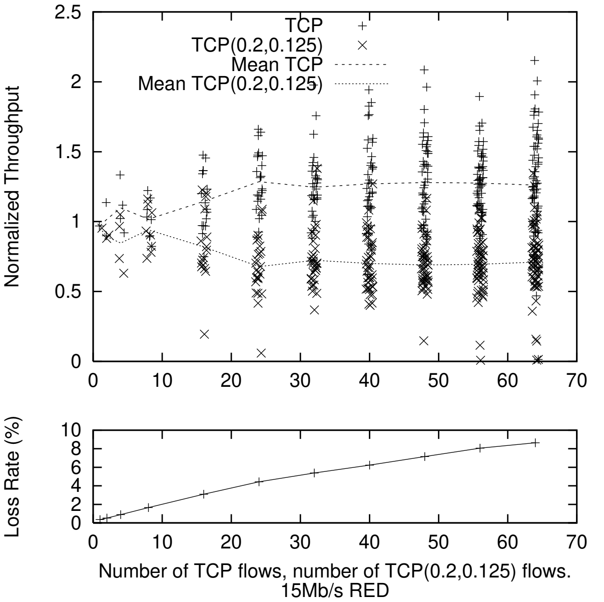

Figure 3: TCP competing with TCP($1/5$, $1/8$), with RED.

TCP($1/5$, $1/8$) to receiving the same bandwidth as TCP. The results are similar with Drop-Tail queue management, and with RED with ECN. We do not have a good explanation for this discrepancy between the theoretical predictions of the deterministic AIMD model and the actual simulations with TCP. One factor could be the assumption of regular, deterministic drops in the deterministic model; another factor could be the role played by retransmit timeouts for TCP in regimes with high packet drop rates.

5 A comparison of equation-based and AIMD congestion control

This section compares equation-based congestion control with the AIMD family of congestion control mechanisms. The most obvious advantage of TCP(a, b), relative to equation-based congestion control, is that AIMD congestion control is familiar and reasonably-well understood, in terms of fairness, stability, oscillations, and other properties. The most obvious disadvantage of TCP(a, b), relative to equation-based congestion control, is that equation-based congestion control has less abrupt changes in the sending rate; any congestion control based on AIMD inherently includes an oscillation in the sending rate.

Figure 5 shows TFRC competing with TCP($1/5$, $1/8$). The simulations in Figures 2 and 5 show similar results, that TCP($1/5$, $1/8$) is somewhat less aggressive than either TCP or TFRC. Figure 6 shows TFRC competing fairly with TCP($2/5$, $1/8$).

5.1 Transient response

The response of TCP(a, b) to a sudden increase in congestion is easily quantified. For example, a flow using TCP(a, b) congestion control would require $\log_{1-b} 0.5$ round-trip times of persistent congestion to reduce its sending rate by half. Thus, for $b = 1/8$, TCP(a, b) takes more than five round-trip

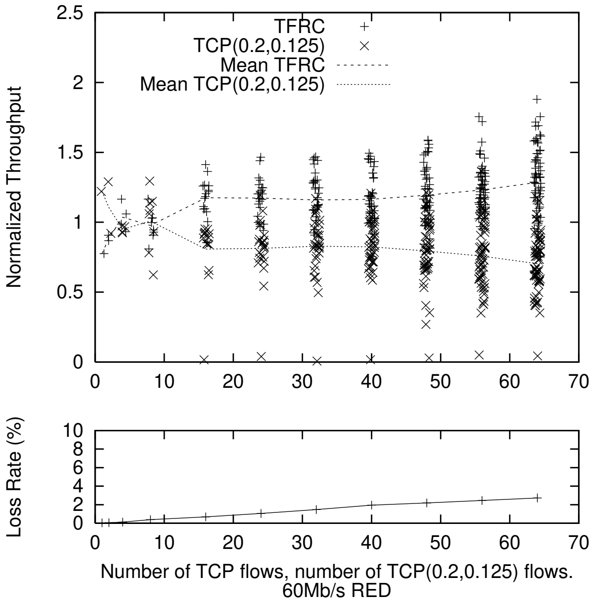

Figure 5: TFRC competing with TCP($1/5$, $1/8$), with RED.

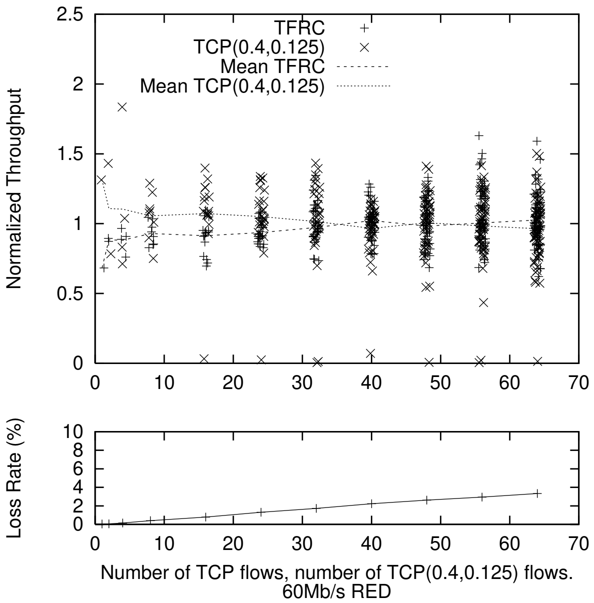

Figure 6: TFRC competing with TCP($2/5$, $1/8$), with RED.

times of persistent congestion to reduce the sending rate by half, while for $b = 1/4$, TCP(a, b) takes only three round-trip times.

Analysis and simulations in Appendix A.2 of [FHPW00] show that TFRC generally takes five round-trip times of persistent congestion to reduce the sending rate by half. Thus, TFRC responds roughly as promptly to a sudden increase in congestion as TCP(a, $1/8$).

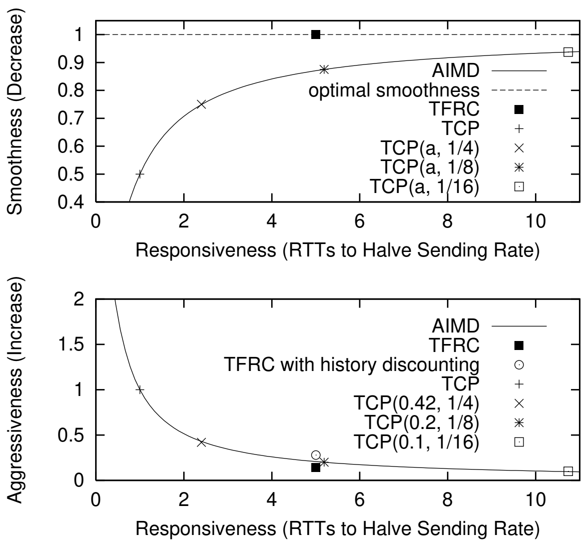

Figure 7: Tradeoffs between responsiveness, smoothness, and aggressiveness.

For AIMD(a, b) for a specific value for b, Equation 9 gives the value for a that results in the same response functions for AIMD(a, b) and AIMD(1, 1/2) in the deterministic steady-state environment. As the simulations above have shown, this does not necessarily imply that TCP(a, b) and TCP(1, 1/2) share bandwidth exactly equally in our simulation environments. Nevertheless, in this section we will use Equation 9 to give the a and b parameters for TCP-compatible congestion control.

In this section, we consider the tradeoffs between smoothness, responsiveness, and aggressiveness for TCP-compatible congestion control. We define *smoothness* as the largest reduction of the sending rate in one round-trip-time in the deterministic steady-state scenario. The closer the smoothness metric is to one, the smoother the sending rate over time in the steady-state scenario. For TCP(a, b) the smoothness metric is $1 - b$, and for TFRC the smoothness metric is 1.

We define *responsiveness* as the number of round-trip times of sustained congestion required to reduce the sending rate by half. Thus, a smaller responsiveness metric corresponds to a more prompt response to sustained congestion. For TCP(a, b) the responsiveness metric is $\log_{1-b} 0.5$, and for TFRC the responsiveness metric is 5.

We define *aggressiveness* as the maximum increase in sending rate in one round-trip time, in packets per second, given the absence of congestion. For TCP(a, b), the aggressiveness metric is simply the parameter a. Analysis and simulations in Appendix A.1 of [FHPW00] show that TFRC's increase rate is at most 0.14 pkts per RTT during normal conditions, and at most 0.28 pkts per RTT when history discounting has been invoked, giving aggressiveness metrics of 0.14 and 0.28, respectively.

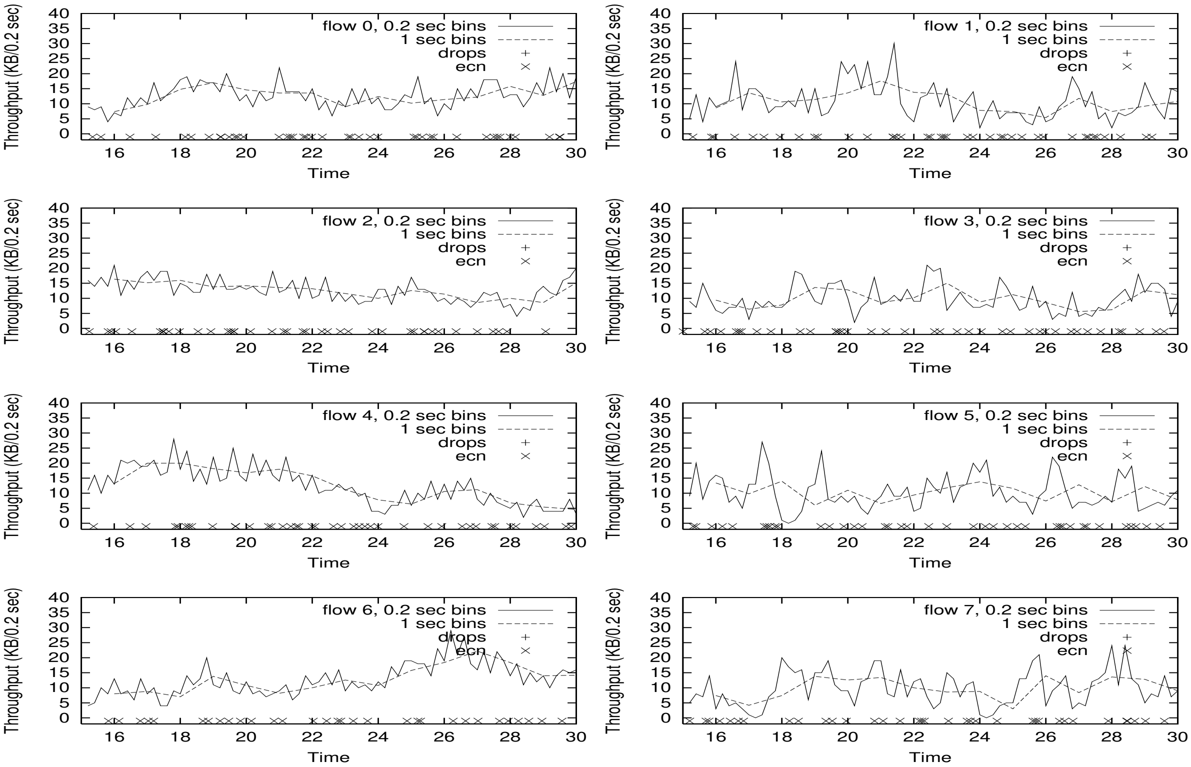

Figure 8: TCP[2/5, 1/8] (left column) and TCP (right column) flows, with RED and ECN, for $N = 16$.

The top graph of Figure 7 shows the tradeoffs between smoothness and responsiveness for TCP(a, b) congestion control, and compares these with TFRC. For the top graph of Figure 7, each dot on the line labeled AIMD corresponds to a specific value of b for TCP(a, b) congestion control. For the dot for each value of b, the x-axis shows the responsiveness and the y-axis shows the smoothness. The dark square on the graph shows the smoothness and responsiveness for TFRC. As the graph shows, TFRC gives the optimal smoothness, with no reduction in the sending rate from one round-trip time to the next in the deterministic steady-state scenario.

The bottom graph of Figure 7 shows the tradeoffs between aggressiveness and responsiveness for TCP(a, b) congestion control, and compares these with TFRC. Figure 7 shows that TFRC increases its sending rate, in the absence of congestion, roughly as aggressively as TCP($1/5, 1/8$) congestion control, while having superior smoothness.

5.2 Smoothness in steady-state

The sections above have shown that TCP($1/5$, $1/8$) and TCP($2/5$, $1/8$) both have similar transient behavior to that of TFRC, in terms of the response to persistent congestion. However, TFRC has a smoother sending rate; TCP(a, $1/8$) reduces its sending rate to 7/8-ths of its previous value in response to each packet loss event, and in steady-state TFRC avoids this reduction in the sending rate. In this section, we explore the differences between TFRC, TCP, TCP($1/5, 1/8$), and TCP($2/5$, $1/8$) in the variation of the sending rate over short time scales.

Figure 8 shows individual flows from a simulation with 16 TCP($2/5$, $1/8$) and 16 TCP flows competing on a 15 Mbps congested link. This simulation is from Figure 4, and has roughly a 4% packet drop rate. As shown in Figure 4, the TCP($2/5$, $1/8$) flows receive slightly more bandwidth than the TCP flows in this simulation.

Each graph shows one flow's throughput on the congested link during the second fifteen seconds of a 500-second simulation. The throughput is averaged over 0.2 sec intervals. This interval is twice as long as a typical round-trip time for this simulation.[1] At the bottom of each graph, there is a "+" for each packet dropped and an "x" for each packet marked in that flow. The queues use RED queue management with ECN. We have run these simulations with both RED and Drop-Tail queue management, with and without ECN, and for different values for the bandwidth of the congested link,

[1]The round-trip time in the absence of queueing delay ranges from 44 to 64 ms., and the average queueing delay at the congested router, for that packet drop rate, is 33 ms. The simulations in Figure 8 were run with RED queue management on the 15 Mbps congested link, with the RED parameters set as follows: *min_thresh* is set to 25 packets, *max_thresh* is set to five times min_thresh, max_p is set to 0.1, and the *gentle_* parameter is set to true.

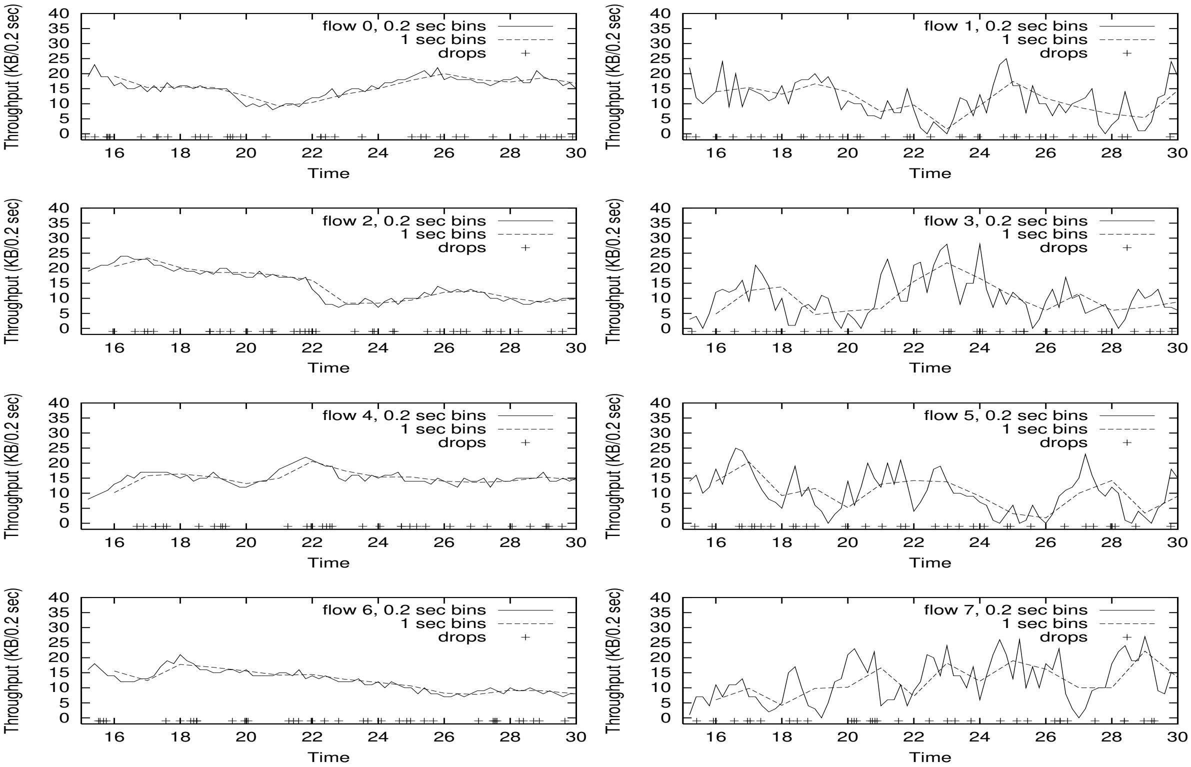

Figure 9: TFRC (left column) and TCP (right column) flows, with RED, for $N = 16$.

with similar results.

Figure 8 shows that the TCP($2/5, 1/8$) flows are somewhat less bursty than the TCP flows.

Figure 9 shows 16 TFRC flows and 16 TCP flows in the same simulation scenario, again with an average packet drop rate of 4%. The TFRC flows in Figure 9 are considerably smoother than the TCP flows, and than the TCP($2/5$, $1/8$) flows in Figure 8.

In order to better quantify the short-term burstiness of the different flows, the graphs in Figures 10 show the throughput ratio for the sending rate over adjacent intervals. Let T_i be the sending rate for a flow over the i-th time interval. We define a flow's *throughput ratio* for the i-th interval as

$$\frac{T_i}{T_{i-1}}.$$

A throughput ratio of 1 means that the flow's sending rate was the same over the two adjacent intervals, a ratio less than one means that the sending rate decreased over the two adjacent intervals, and a ratio greater than one means that the sending rate increased.

The upper left graph of Figure 10 shows a line for each of the 16 TFRC flows in the simulation from Figure 9, where each line gives a histogram of the throughput ratios for all of the 0.2-second intervals in that flow over the 500-second simulation. For each flow, the histogram of throughput ratios is plotted over 20 possible ranges from zero to two. For example, the fraction of throughput ratios that fall between 1.0 and 1.1 is plotted by a point with the x-coordinate of 1.05. The y-coordinate for that point gives the fraction of throughput ratios within the range [1.0, 1.1) for the 0.2-second intervals of that flow.

The upper left graph of Figure 10 shows that for the TFRC flows in the simulation with the throughput ratio computed over 0.2-second intervals, the throughput ratio was almost always at least 0.7, meaning that a TFRC flow's sending rate was rarely reduced to below 70% of its previous value from one 0.2-second interval to the next.

The four rows of Figure 10 give the throughput ratios for TFRC, TCP, TCP(2/5, 1/8), and TCP(1/5, 1/8) respectively, and the columns give the throughput ratios computed over 0.2-second, 1-second, and 10-second intervals. For the simulation with TCP(2/5, 1/8), the TCP(2/5, 1/8) flows are competing with TCP; the same holds for the simulations with TCP(1/5, 1/8). Figure 11 shows the cumulative distribution functions for these same distributions. Thus, Figures 10 and 11 contain roughly the same information in different forms.

The right column of Figures 10 and 11 shows that over 10-second intervals, TFRC is only slightly smoother than TCP. That is, TFRC and TCP are both unlikely to reduce their sending rate to less that 50% of its previous value from one 10-second interval to the next, in the steady-state conditions in

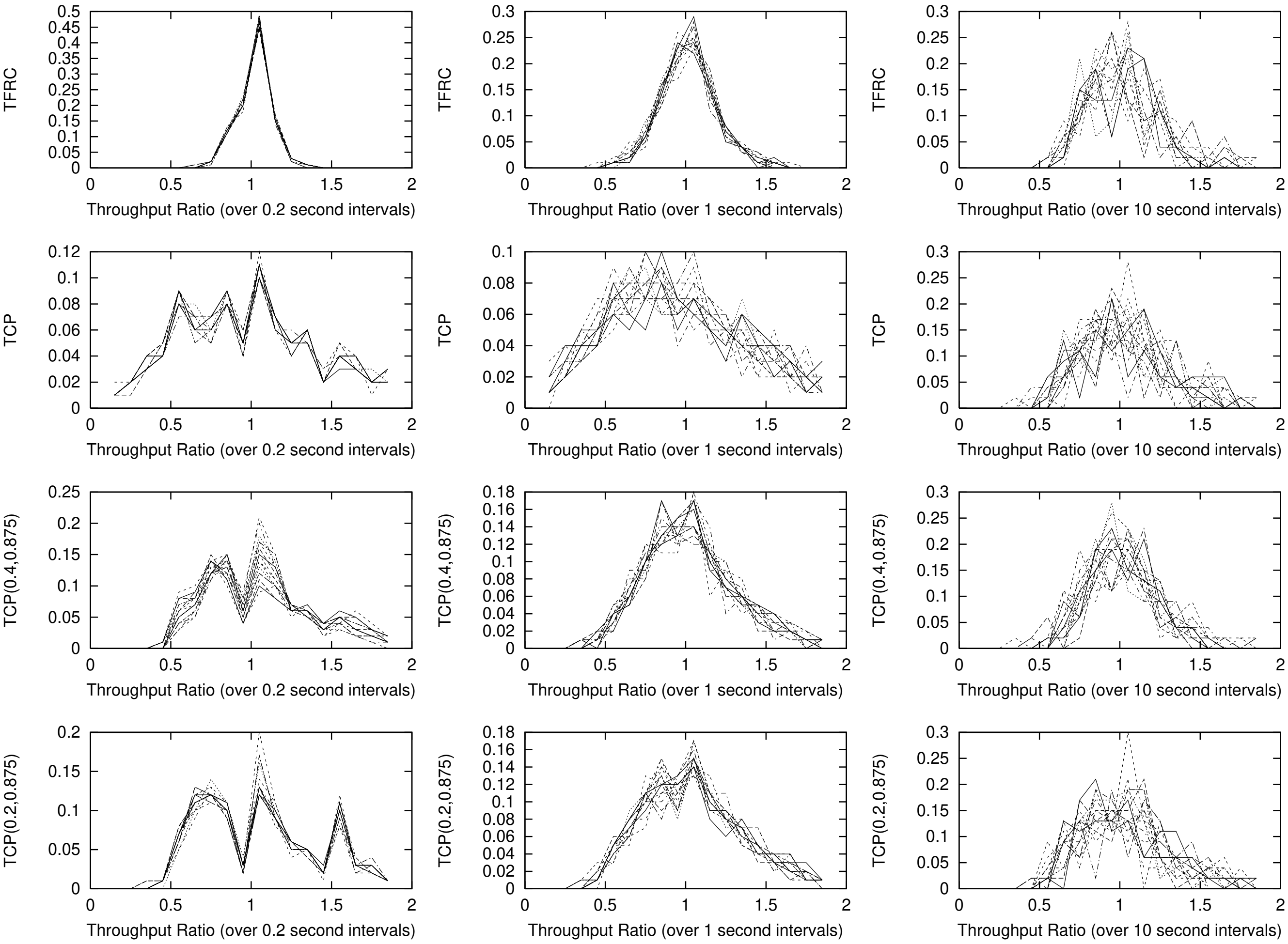

Figure 10: Histogram of throughput variation.

these simulations.

However, the middle column of Figures 10 and 11 shows that over 1-second intervals, TFRC is significantly smoother than TCP. In these simulations, TFRC is highly unlikely to reduce its sending rate to less that 50% of its previous value from one 1-second interval to the next, while for TCP a reduction this sharp happens for more than 10% of the 1-second intervals. This result concurs with the sending rates shown in Figure 9, where the TCP flows are more likely than the TFRC flows to reduce their sending rate by half from one one-second interval to the next.

Figures 10 and 11 show that over 0.2-second and 1-second intervals, the TCP(2/5, 1/8) and TCP(1/5, 1/8) flows are more smooth than the TCP flows, but less smooth than the TFRC flows. The throughput ratios of the TCP(2/5, 1/8) and TCP(1/5, 1/8) flows are quite similar, particularly over 1-second and 10-second intervals. However, the TFRC flows are significantly smoother than the TCP(2/5, 1/8) or TCP(1/5, 1/8) flows; for example, the TCP(2/5, 1/8) or TCP(1/5, 1/8) flows are considerably more likely to reduce their sending rate to less that 75% of its previous value from one 1-second interval to the next. Again, this concurs with the sending rates shown in Figures 8 and 9 for the TFRC and TCP(2/5, 1/8) flows.

All of the simulations in this section consist of a fixed number of long-lived flows. A open question would be to explore the throughput ratios of TFRC, TCP, and TCP(a, 1/8) in an environment with two-way traffic and many small flows, with an inherently burstier traffic mix. Also, in this section we have only explored the relative burstiness of TFRC, TCP, and TCP(a, 1/8) in an environment with a 4% steady-state packet drop rate. It would be interesting to explore the relative burstiness at different levels of congestion, reflected in different steady-state packet drop rates, and with different round-trip times. This should also be amenable to some simple but useful analysis.

All of the simulation scripts for the simulations in this paper will be available from [FHPW00].

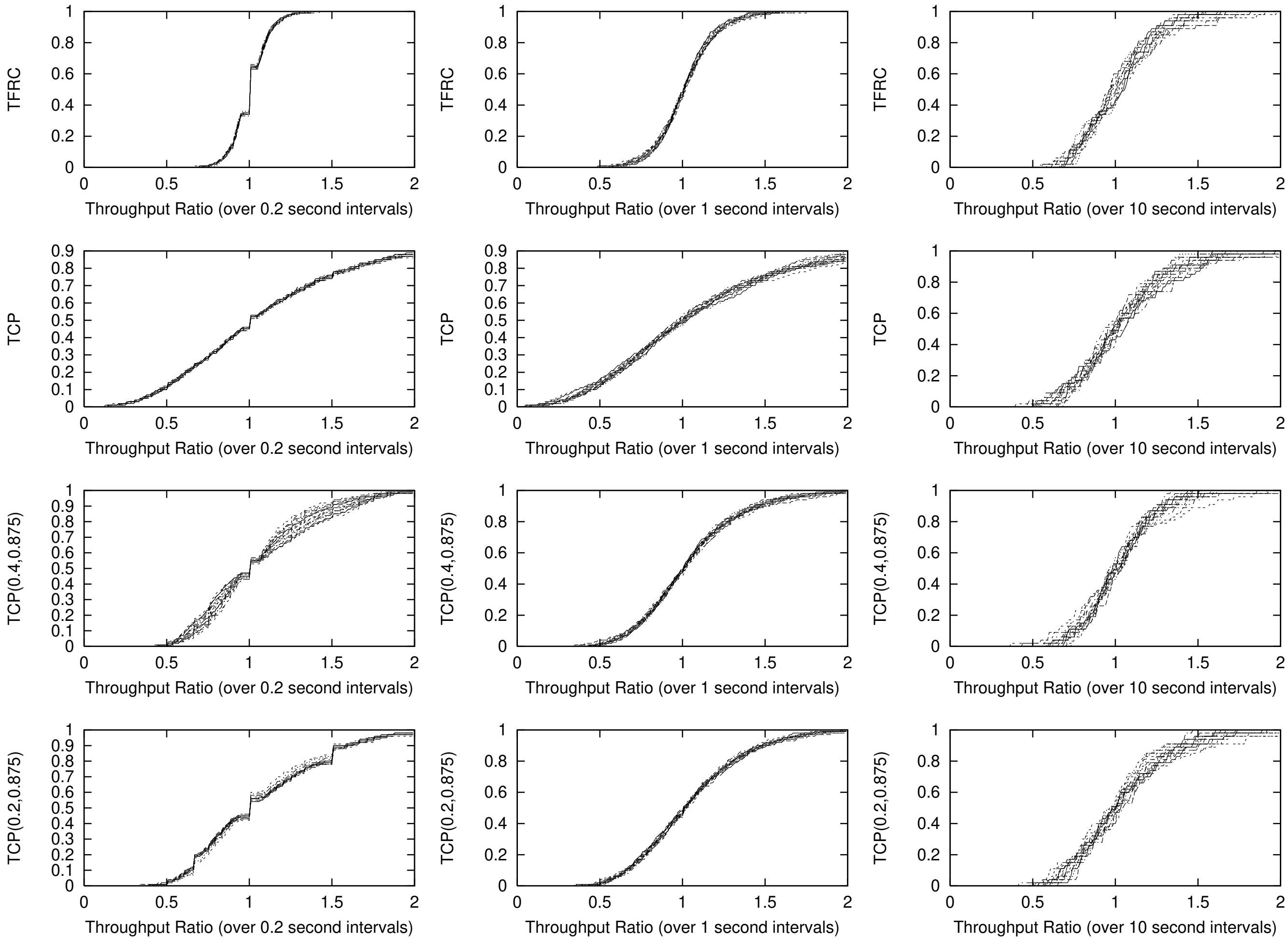

Figure 11: Cumulative distribution of throughput variation.

6 Non-TCP-compatible congestion control

In this section we digress briefly to consider the relative throughput and packet drop rates for non-TCP-compatible instances of TCP(a, b) congestion control.

6.1 Steady-state throughput

Consider a TCP(a, b) flow competing with standard TCP in a FIFO queue. Because the two flows are competing in a FIFO queue, they experience roughly the same packet drop rate p. Assume that both flows have the same round-trip time R. Then, from Equations (7) and (8), the bandwidth received by the TCP(a, b) flow is roughly

$$\frac{\left(\frac{\sqrt{2-b}\sqrt{a}}{\sqrt{2b}}\right)}{\sqrt{1.5}} = \frac{\sqrt{2-b}\sqrt{a}}{\sqrt{3b}} \qquad (10)$$

times the bandwidth received by the standard TCP.

6.2 Steady-state packet drop rates

Inverting Equation (8), consider a long-lived AIMD(1, 1/2) flow with round-trip time R, and a a throughput of roughly T packets/sec. The steady-state packet drop rate to control that AIMD(1, 1/2) flow would have to be roughly

$$p_{(1,1/2)} = \frac{1.5}{R^2T^2}.$$

In contrast, inverting Equation (7), if the flow was instead using AIMD(a, b) congestion control, the steady-state packet drop rate would have to be roughly

$$p_{(a,b)} = \frac{(2-b)a}{2bR^2T^2}.$$

This gives the ratio $p_{(a,b)}/p_{(1,1/2)}$ of the AIMD(a, b) packet drop rate over the AIMD(1, 1/2) packet drop rate as the following:

$$\frac{p_{(a,b)}}{p_{(1,1/2)}} = \frac{(2-b)a}{3b}. \qquad (11)$$

6.3 Simulations comparing TCP and TFRC with TCP(1, 1/8)

It has been suggested that one way to increases the smoothness of TCP congestion control would be to change the decrease parameter from 1/2 to 1/8, while leaving the increase of one packet per RTT unchanged. This section explores this proposal, and shows that TCP(1, 1/8) is significantly more aggressive that TCP.

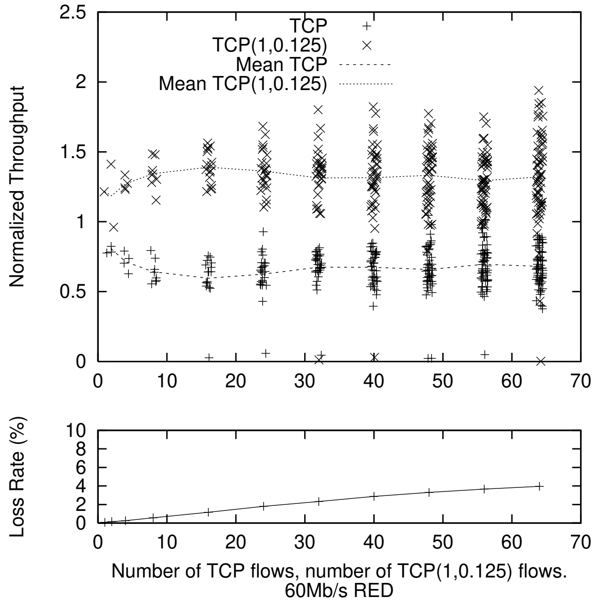

Figure 12: TCP competing with TCP($1, 1/8$), with RED.

The deterministic response functions in Equation (10) suggest that a TCP(1, $1/8$) flow would receive roughly $\sqrt{5} = 2.23$ times the bandwidth of a competing TCP flow. This is confirmed in simulations by Figure 12, which shows TCP flows competing with TCP(1, $1/8$) flows. In these simulations, at higher levels of congestion the TCP(1, $1/8$) flows get almost twice the bandwidth of a competing TCP flows.

Equation (11) suggests that a network with long-lived flows using AIMD(1, $1/8$) could be expected to induce a steady-state packet drop rate roughly 5 times that induced by AIMD(1, 1/2).

7 Summary of related work

There are a wide range of potential unicast congestion control mechanisms that provide a smoother sending rate than that given by TCP. This includes not only equation-based congestion control and AIMD-based congestion control, but a number of other proposals as well. We have compared TFRC with TCP(a, b) because AIMD is the default congestion control mechanism used in the current Internet in TCP today. We have not compared TFRC with other proposals for unicast congestion control.

The Rate Adaptation Protocol (RAP) described in [RHE99] uses an AIMD rate control scheme based on regular acknowledgements sent by the receiver. The sender uses these acknowledgements to detect lost packets and estimate the RTT. The sender adjusts the rate every RTT, depending on the presence or absence of loss in the most recent RTT. For mild to moderate packet drop rates, RAP should give the same behavior as TCP over timescales longer than a round-trip time.

Ott in [Ott99] considers AIMD congestion control in an environment with ubiquitous Explicit Congestion Notification [RF99], where it can be assumed that routers will be marking rather than dropping packets to indicate congestion. One section of [Ott99] assumes an environment with ubiquitous ECN where, in addition, compatibility with existing TCP implementations is no longer required, and considers a range of possible increase and decrease algorithms. For example, [Ott99] includes discussion of congestion control algorithms that would require a higher packet-marking rate than TCP-compatible congestion control, and that therefore could only be implemented in an environment with ECN. In this note we restrict our attention to TCP-compatible congestion control mechanisms.

Bansal and Balakrishnan in [BB00] consider binomial congestion control algorithms, where a *binomial algorithm* uses a decrease in response to a loss event that is proportion to a power l of the current window, and that otherwise uses an increase that is inversely proportional to the power k of the current window. AIMD congestion control is a special case of binomial congestion control that uses $l = 1$ and $k = 0$. [BB00] considers several binomial congestion control algorithms that are TCP-compatible and that avoid TCP's drastic reduction of the congestion window in response to a loss event. [BB00] specifies which binomial congestion control mechanisms do and do not share TCP's convergence to fairness in an environment of synchronized loss events. We would note that the binomial algorithms all show the same tradeoff between smoothness and responsiveness as AIMD. That is, the line for each of the binomial algorithms on Figure 7 is identical to the line for AIMD. However, for the binomial algorithms the tradeoff between aggressiveness and responsiveness would be a function of the current congestion window as well as of the parameters of the congestion control algorithm.

[CO98] explores changing the increase and decrease parameters for AIMD so that a single flow would get the same bandwidth that N TCP flows would have received.

8 Acknowledgements

This work has benefited from discussions with a number of people, including Hari Balakrishnan, Scott Shenker, Joerg

Widmer, and the End-to-end Research Group.

9 Conclusions

In this paper we have considered the family of AIMD(a, b) congestion control mechanisms, and compared them with the equation-based congestion control mechanism TFRC. We have shown that TCP($1/5$, $1/8$) and TCP($2/5$, $1/8$) compete fairly equally with TCP and with TFRC, while avoiding TCP's reduction of the sending rate in half in response to a single packet drop. At the same time, we have shown that TFRC changes its sending rate more smoothly that does TCP($1/5, 1/8$) or TCP($2/5, 1/8$), while having a similar transient response to a sudden increase in congestion.

References

[BB00] D. Bansal and H. Balakrishnan. "Behaviour of TCP-compatible Nonlinear Congestion Control", 2000. Private communication.

[CO98] J. Crowcroft and P. Oechslin. "Differentiated End-to-end Internet Services using a Weighted Proportional Fair Dharing TCP", Jul. 1998.

[FF99] S. Floyd and K. Fall. "Promoting the Use of End-to-End Congestion Control in the Internet". *IEEE/ACM Transactions on Networking*, Aug. 1999. URL http://www-nrg.ee.lbl.gov/floyd/end2end-paper.html.

[FHPW00] S. Floyd, M. Handley, J. Padhye, and J. Widmer. "Equation-Based Congestion Control for Unicast Applications: the Extended Version", March 2000. ICSI Technical Report TR-00-03, URL http://www.aciri.org/tfrc/.

[FHPW00] S. Floyd, M. Handley, J. Padhye, and J. Widmer. "TFRC, Equation-Based Congestion Control for Unicast Applications: Simulation Scripts and Experimental Code", 2000. URL http://www.aciri.org/tfrc/.

[Flo91] S. Floyd. "Connections with Multiple Congested Gateways in Packet-Switched Networks Part 1: One-way Traffic". *ACM Computer Communication Review*, 21(5):30–47, Oct. 1991. URL http://www-nrg.ee.lbl.gov/nrg-papers.html/.

[Ott99] T. Ott. "ECN Protocols and the TCP Paradigm", May 1999. URL ftp://ftp.research.telcordia.com/pub/tjo/ECN.ps.

[PFTK98] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. "Modeling TCP Throughput: A Simple Model and its Empirical Validation". *SIGCOMM Symposium on Communications Architectures and Protocols*, Aug. 1998.

[PFTK98] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. "Modeling TCP Throughput: A Simple Model and its Empirical Validation". *SIGCOMM Symposium on Communications Architectures and Protocols*, Aug. 1998. URL http://www.acm.org/sigcomm/sigcomm98/tp/abs_25.html.

[RF99] K. K. Ramakrishnan and Sally Floyd. "A Proposal to add Explicit Congestion Notification (ECN) to IP". Jan. 1999. RFC 2481.

[RHE99] R. Rejaie, M. Handley, and D. Estrin. "An End-to-end Rate-based Congestion Control Mechanism for Real-time Streams in the Internet". In *Proceedings of INFOCOMM 99*, 1999.

[TCP] "The TCP-Friendly Web Page". URL http://www.psc.edu/networking/tcp_friendly.html.

[WMI] WMI. "Waterloo Maple". URL http://www.maplesoft.com/.

A The deterministic AIMD response function: a more precise version

The approximate solution of the deterministic AIMD response function given in Equation (7) uses an approximation step in Equation (5). The more precise version of the deterministic AIMD response function can be useful. The precise expression for the packet drop rate in the deterministic AIMD model, given in Equation (4), is as follows:

$$p = \frac{2a}{b(2-b)W^2 + a(2-b)W}.$$

Solving from *maple* (or from simple algebra) gives the following:

$$W = \frac{-2\,pa + pba + \sqrt{p\,(-2+b)\,a\,(pba - 8\,b - 2\,pa)}}{2\,pb\,(2-b)}$$

The precise expression of the AIMD sending rate S in packets per RTT, using Equation (2), is as follows:

$$S = \frac{-2\,pa + pba + \sqrt{p\,(-2+b)\,a\,(pba - 8\,b - 2\,pa)}}{4\,pb} \tag{12}$$

A.1 The deterministic response function applied to AIMD($1, 1/2$)

Applying the deterministic response function to AIMD(1, $1/2$), we get the following sending rate S in packets per RTT:

$$S = \frac{0.43\sqrt{(3p+8)}}{\sqrt{p}} - 0.75.$$

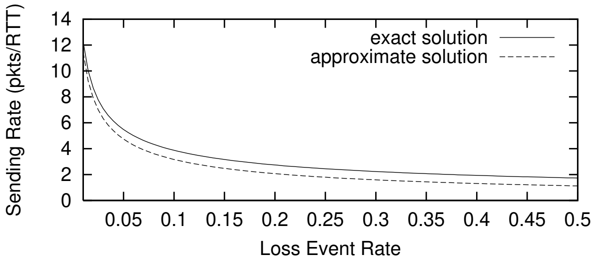

Figure 13: Exact vs. approximate solutions to the sending rate for AIMD(1, $1/2$) in the deterministic model.

Figure 13 compares the exact solution for the AIMD(1, $1/2$) sending rate given above to the approximate solution given earlier of $S = \sqrt{1.5}/\sqrt{p}$. For smaller values of p, the exact and approximate solutions for S in the deterministic model give similar results. However, for $0.1 < p$, the differences between the exact and the approximate solutions for S are more pronounced.