Available at www.ComputerScienceWeb.com
POWERED BY SCIENCE DIRECT®

Computer Networks 41 (2003) 193–210

COMPUTER NETWORKS

www.elsevier.com/locate/comnet

Transient behaviors of TCP-friendly congestion control protocols ☆

Y. Richard Yang, Min Sik Kim, Simon S. Lam *

Department of Computer Sciences, The University of Texas at Austin, Austin, TX 78712-1188, USA

Received 17 July 2002; accepted 31 July 2002

Responsible Editor: I.F. Akyildiz

Abstract

We investigate the fairness, smoothness, responsiveness, and aggressiveness of TCP and three representative TCP-friendly congestion control protocols: GAIMD, TFRC, and TEAR. The properties are evaluated both analytically and via simulation by studying protocol responses to three network environment changes. The first environment change is the inherent fluctuations in a stationary network environment. Under this scenario, we consider three types of sending rate variations: smoothness, short-term fairness, and long-term fairness. For a stationary environment, we observe that smoothness and fairness are positively correlated. We derive an analytical expression for the sending rate coefficient of variation for each of the four protocols. These analytical results match well with experimental results. The other two environment changes we study are a step increase of network congestion and a step increase of available bandwidth. Protocol responses to these changes reflect their responsiveness and aggressiveness, respectively.

Keywords: Congestion control; TCP-friendliness

1. Introduction

In a shared network such as the Internet, end systems should react to congestion by adapting their transmission rates to share bandwidth fairly, to avoid congestion collapse, and to keep network utilization high [1]; the robustness of the Internet is due in large part to the end-to-end congestion control mechanisms of TCP [2]. However, while TCP congestion control is appropriate for applications such as bulk data transfer, other applications such as streaming multimedia would find halving the sending rate of a flow to be too severe a response to a congestion indication as it can noticeably reduce the flow's user-perceived quality [3].

In the last few years, many unicast congestion control protocols have been proposed and investigated [3–13]. Since the dominant Internet traffic is TCP-based [14], it is important that new congestion control protocols be *TCP-friendly*. By this, we mean that the sending rate of a non-TCP flow should be approximately the same as that of a

☆ An abbreviated version of this paper was published in Proceedings of IEEE INFOCOM 2001, Anchorage, Alaska, April 2001.

* Corresponding author. Tel.: +1-512-471-9531; fax: +1-512-471-8885.

E-mail address: lam@cs.utexas.edu (S.S. Lam).

1389-1286/03/$ - see front matter
PII: S1389-1286(02)00374-2

TCP flow under the same conditions of round-trip time (RTT) and packet loss rate [4,15].

Evaluations of these protocols, however, have been focused mainly on protocol fairness in stationary environments. Two methods were proposed to establish the fairness of a protocol. The first is Chiu and Jain's phase space method [16], which can be used to show that a protocol will converge asymptotically to a fair state, ignoring such operational factors as randomness of the loss process and timeouts. The second method is to show that the long-term mean sending rate of a protocol is approximately the same as that of TCP. However, it has been observed in experiments [11,12,17] that flows with TCP-friendly long-term mean sending rates can still have large rate variations when loss rate is high.

Furthermore, fairness is only one of several desirable properties of a congestion control protocol. We identify four desired properties: (1) *fairness:* small variations over the sending rates of competing flows; (2) *smoothness:* small sending rate variations over time for a particular flow in a stationary environment; (3) *responsiveness:* fast deceleration of protocol sending rate when there is a step increase of network congestion; and (4) *aggressiveness:* fast acceleration of protocol sending rate to improve network utilization when there is a step increase of available bandwidth.

The objective of this paper is to evaluate these properties by studying the transient behaviors of several congestion control protocols under three network environment changes. Proposed congestion control protocols in the literature fall into two major categories: AIMD-based [6–8,12,13] and formula-based [3–5,9,11]. For our study, we select TCP [2] and GAIMD [12] as representatives of the first category. GAIMD generalizes TCP by parameterizing the congestion window increase value and decrease ratio. That is, in the congestion avoidance state, the window size is increased by α per window of packets acknowledged and it is decreased to β of the current value whenever there is a triple-duplicate congestion indication. In our evaluation, we choose $\beta = 7/8$ because it reduces a flow's sending rate less rapidly than TCP does. For $\beta = 7/8$, we choose $\alpha = 0.31$ so that the flow is TCP-friendly [12]. In what follows, we use GAIMD to refer to GAIMD with these parameter values. We select TFRC [13] as a representative of the formula-based protocols. In addition to these three protocols, we select TEAR [13] which uses a sliding window to smooth sending rates.

The first environment change we study is the inherent network fluctuations in a stationary environment. We evaluate three types of sending rate variations: smoothness, short-term fairness, and long-term fairness. For a stationary environment, we observe that smoothness and fairness are positively correlated. To quantify the smoothness of a flow, we derived an analytical expression for the sending rate coefficient of variation (CoV) for each of the four protocols. We found that our analytical results match experimental results very well. We observe that with increasing loss rate, smoothness and fairness become worse for all four protocols. However, their deteriorating speeds are different. In particular, at 20% loss rate, TFRC CoV increases to be the highest. TEAR maintains a relatively stable smoothness and fairness performance, but it scores the lowest in experiments on responsiveness and aggressiveness (see below). Also, while TFRC and TEAR have smoother sending rates than those of TCP and GAIMD, they have undesirable fairness behaviors at high loss rate, i.e., TFRC sending rate dropping to almost zero and TEAR sending rate being too high compared with TCP.

The second environment change we study is a step increase of network congestion. Protocol responses to this change reflect their responsiveness. In our experiments, TCP is the most responsive of the four protocols. However, TCP overshoots and has to recover from its overshot state. We also found an undesirable behavior of TEAR. This shows that our evaluation framework can be a valuable tool for evaluating congestion control protocols and detecting undesirable protocol behaviors.

The third environment change we study is a step increase of available bandwidth. Protocol responses to this change reflect their aggressiveness. In our experiments, we found that TCP is the most aggressive of the four protocols to use newly available bandwidth. Again TCP overshoots. TFRC with history discounting and GAIMD have

similar aggressiveness. TEAR is the least aggressive to utilize newly available bandwidth.

The balance of this paper is organized as follows. In Section 2 we discuss our evaluation methodology. In Section 3 we evaluate protocol responses in stationary environments. In Section 4, we evaluate protocol responses to a step increase of network congestion. Protocol responses to a step increase in available bandwidth are shown in Section 5. Our conclusion and future work are in Section 6.

2. Evaluation methodology

2.1. Loss models

Network loss process is a major factor in determining the performance of a congestion control protocol. In our simulations, we use four simple and representative loss models. We distinguish between loss models for high multiplexing environments and low multiplexing environments. By high multiplexing environment, we mean that loss is relatively insensitive to the sending rate of the flow under study. This is intended to be a model for backbone routers. By low multiplexing environment, we mean that loss is somewhat sensitive to the sending rate of a flow.

Our first loss model is deterministic periodic loss. Though this model may be unrealistic, it is simple and protocol responses for this model are representative and clear.

The second loss model is Bernoulli loss. In this model, each packet is lost with probability p, which is independent and identically distributed (i.i.d.) for all packets. We consider this model as one representative for high multiplexing environments. For example, in today's Internet, packets are dropped by routers without regard to which flows they belong to when buffers overflow. Though packet losses can be correlated, a number of studies [18–20] show that loss bursts in the Internet are short and any loss correlation does not span long, typically less than one RTT.

The third loss model for high multiplexing environments is the loss process when background traffic consists of ON/OFF sources, which can generate web-like traffic (short TCP connections and some UDP flows). In our experiments, we set the mean ON time to be 1 s, and the mean OFF time to be 2 s. During ON time each source sends at 500 Kbps. The shape parameter of the Pareto distribution is set to be 1.5. The number of ON/OFF sources in our experiments is 5.

The fourth loss model is the loss process when N flows are competing with each other. We consider this loss model as a representative for low multiplexing environments.

2.2. Simulation configurations

Our network topology is the well-known single bottleneck ("dumbbell") as shown in Fig. 1. In this topology, all access links have a delay of 10 ms, and they are sufficiently provisioned to ensure that packet drops due to congestion occur only at the bottleneck link from R1 to R2. The bottleneck link is configured to have a bandwidth of 2.5 Mbps and a propagation delay of 30 ms. We repeat each simulation twice by configuring the bottleneck link as either a drop-tail or a RED link. For drop-tail link, we set a buffer size of 50 packets with packet size 1000 bytes. The parameters of RED link are scaled as in [11]. For most cases, the results for drop-tail and RED are similar. Therefore, the reported results are for drop-tail link unless we state otherwise.

We use GAIMD based on TCP/Reno. Most of our reported results on TCP are based on TCP/Reno unless we explicitly point out. TFRC is based on the code from *ns* June 12th, 2000 snapshot. In our initial set of TEAR experiments, we used the code from the authors' web site. However, we found that the timeout mechanism described in their paper [13] was not implemented. Therefore, we modified their code to implement timeout. For most of the experiments, differences between

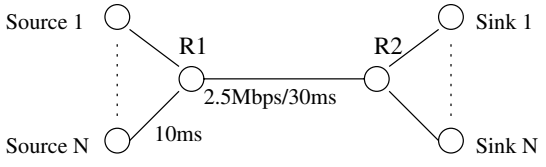


Fig. 1. Network topology.

the modified and unmodified versions are small. However, there are big differences in some experiments; in those cases, we will point them out.

To avoid phase effects [21] that mask underlying dynamics of the protocols, we introduce randomizations by setting the *overhead* parameter of TCP, GAIMD, and TFRC to a small non-zero value.

3. Responses to stationary fluctuations

We first investigate protocol responses in stationary environments. The properties we study in this section are smoothness and fairness.

3.1. Performance metrics

We use CoV to measure protocol smoothness and fairness. First, we clarify three types of CoV.

3.1.1. Three types of coefficient of variation

The definition of CoV depends on measurement timescale: the longer the timescale, the smaller the CoV is. For our purpose, we measure smoothness and short-term fairness at a timescale of RTT; we define long-term fairness at a timescale of multiple RTTs.

(1) *Smoothness* $\mathrm{CoV}_{\mathrm{time}}$. Consider any solid dot in Fig. 2(a), which represents the sending rate during a RTT of a specific flow. We define $\mathrm{CoV}_{\mathrm{time}}$ as the CoV of this time series. We observe that $\mathrm{CoV}_{\mathrm{time}}$ measures the smoothness of a flow.

(2) *Short-term fairness* $\mathrm{CoV}_{\mathrm{sf}}$. Consider the solid dots in Fig. 2(b), which are samples of the sending rates of several competing flows during the same RTT. The coefficient of variation $\mathrm{CoV}_{\mathrm{sf}}$ of this data series measures short-term fairness among competing flows.

(3) *Long-term fairness* $\mathrm{CoV}_{\mathrm{lf}}$. Instead of measuring the sending rates of competing flows during the same RTT, we can measure their sending rates during multiple RTTs. Therefore, we define long-term fairness $\mathrm{CoV}_{\mathrm{lf}}$ as the CoV over the sending rates of competing flows in a longer time period.

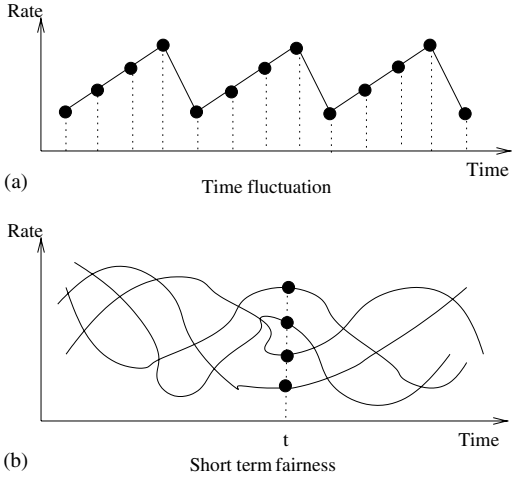


Fig. 2. $\mathrm{CoV}_{\mathrm{time}}$ and $\mathrm{CoV}_{\mathrm{sf}}$.

With the definitions above, next we discuss their relationships. First consider the relationship between smoothness and fairness at a *given* timescale. Assuming competing flows are i.i.d. (the flows will then have the same mean sending rate if the measurement interval is infinity) and ergodic, we know that time distribution and population distribution are equal, that is,

$$\mathrm{CoV}_{\text{time samples}} = \mathrm{CoV}_{\text{population samples}}. \tag{1}$$

Thus we observe that generating smoother traffic (measured by time samples) improves fairness (measured by population samples).

Next, consider the relationship between short-term and long-term CoV. It is intuitive that long-term CoV will be smaller than short-term CoV. Define an epoch as a time interval long enough such that the sending processes of a flow between epochs are independent and identically distributed. Let S_j denote the flow's average sending rate during the jth epoch, and define $R(n) = \sum_{j=1}^{n} S_j/n$ as its average sending rate in n epochs. Since we assume the random variables $\{S_j\}_{j=1}^{n}$ are i.i.d., by the central limit theorem, we know that the distribution of $R(n)$ can be approximated by normal distribution when n is large:

$$\mathrm{CoV}[R(n)] \approx \frac{\mathrm{CoV}[\{S_j\}_{j=1}^{n}]}{\sqrt{n}}. \tag{2}$$

3.1.2. Metrics

In our evaluations, instead of using $\mathrm{CoV}_{\mathrm{sf}}$ to measure short-term fairness, we follow [22] and use fairness index F, defined as $(\sum x_i)^2/(K\sum x_i^2)$, where $\{x_i\}_{i=1}^K$ are the sending rates of competing flows. Let X denote the underlying random variable of samples $\{x_i\}_{i=1}^K$. We observe that $F \approx E[X]^2/E[X^2]$. Rearranging, we have

$$F(X) \approx \frac{1}{1+\mathrm{CoV}(X)^2}. \tag{3}$$

In summary, the performance metrics we use in this section are $\mathrm{CoV}_{\mathrm{time}}$, which measures smoothness; F, which measures short-term fairness; and $\mathrm{CoV}_{\mathrm{lf}}$, which measures long-term fairness. However, the detailed behavior of a flow cannot be fully characterized by these metrics. Moreover, our analytical results are derived for specific loss models. Therefore, to gain intuition, we will also show sending rate traces and the fluctuations of the bottleneck queue length for some simulations.

3.2. Analytical results

We present our analytical results on $\mathrm{CoV}_{\mathrm{time}}$ for TCP, GAIMD, TFRC, and TEAR. The derivations of these results are presented in the appendix.

3.2.1. AIMD

At low loss rate, assuming Poisson loss arrival, we derive $\mathrm{CoV}_{\mathrm{time}}$ for AIMD (including GAIMD and TCP Reno as special cases) to be (see Appendix A):

$$\mathrm{CoV}_{\mathrm{time}}^{\mathrm{AIMD}} = \sqrt{\frac{1-\beta}{1+\beta}} \tag{4}$$

where β is the reduction ratio of congestion window size when there is a congestion indication.

Plugging $\beta = 1/2$ for TCP into Eq. (4), we have

$$\mathrm{CoV}_{\mathrm{time}}^{\mathrm{TCP}} = \sqrt{\frac{1}{3}} \approx 0.58. \tag{5}$$

Plugging $\beta = 7/8$ for GAIMD into Eq. (4), we have

$$\mathrm{CoV}_{\mathrm{time}}^{\mathrm{GAIMD}} = \sqrt{\frac{1}{15}} \approx 0.26. \tag{6}$$

When loss rate is high, both GAIMD and TCP Reno will be in timeout states most of the time. Modeling timeout as a Markovian process, we derive $\mathrm{CoV}_{\mathrm{time}}$ to be (see Appendix B):

$$\mathrm{CoV}_{\mathrm{time}}^{\mathrm{AIMD}} = \sqrt{\frac{64(t-1)+32p+16p^2+8p^3+4p^4+2p^5+p^6}{64-32p-16p^2-8p^3-4p^4-2p^5-p^6}} \tag{7}$$

where p is packet loss rate, and t is the ratio of timeout interval to RTT.

Plugging $p = 20\%$ and $t = 4$ into the expression above, for GAIMD and TCP Reno, we have

$$\mathrm{CoV}_{\mathrm{time}}^{\mathrm{AIMD}} \approx 1.7. \tag{8}$$

3.2.2. TFRC

At low loss rate, assuming Bernoulli loss, we derive $\mathrm{CoV}_{\mathrm{time}}$ for TFRC to be (see Appendix C):

$$\mathrm{CoV}_{\mathrm{time}}^{\mathrm{TFRC}} \approx 0.22. \tag{9}$$

At high loss rate (about 20%), we derive in the appendix that $\mathrm{CoV}_{\mathrm{time}}$ for TFRC will be between 0.8 and 2.4.

3.2.3. TEAR

At low loss rate, assuming Poisson loss, we derive $\mathrm{CoV}_{\mathrm{time}}$ for TEAR to be (see Appendix D):

$$\mathrm{CoV}_{\mathrm{time}}^{\mathrm{TEAR}} \approx 0.21. \tag{10}$$

3.3. Simulation results

3.3.1. High multiplexing environments

We start our simulation with periodic loss. Fig. 3 shows flow sending rate traces when the loss rate is 5%. For this figure, the horizontal axis is measured in the number of RTTs, and the vertical axis is the flow sending rate during a RTT. This simple experiment shows that the sending rates of TFRC and TEAR are smoother than those of TCP and GAIMD at low loss rate. Fig. 4 shows flow sending rate traces under Bernoulli loss model at the same loss rate. Comparing Fig. 3 with Fig. 4, we observe that because of the randomness of Bernoulli loss, all four protocols exhibit much

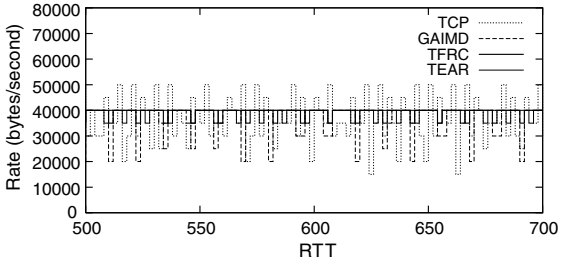


Fig. 3. Sending rates (periodic loss, $p = 5\%$).

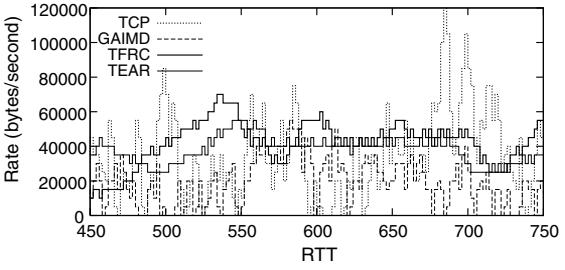


Fig. 4. Sending rates (Bernoulli loss, $p = 5\%$).

larger fluctuations even at this relatively low loss rate.

The result for 20% Bernoulli loss is even worse. We observe from Fig. 5 that at 20% loss, the sending rate of TFRC drops to almost 0 and the average sending rate of TEAR is much higher than those of TCP and GAIMD. Therefore, at high loss rate, the behaviors of neither TFRC nor TEAR are desirable.

Fig. 6 summarizes $\mathrm{CoV}_{\mathrm{time}}$ from simulations for all four protocols when Bernoulli loss rates are varied from 0.5% to 20%. We make the following observations:

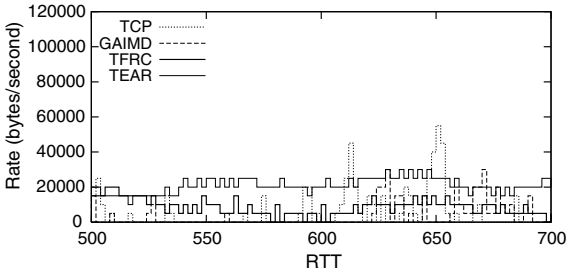


Fig. 5. Sending rates (Bernoulli loss, $p = 20\%$).

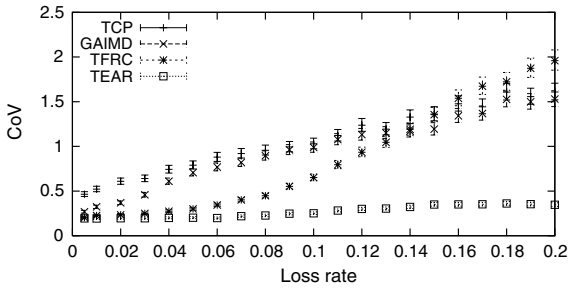


Fig. 6. $\mathrm{CoV}_{\mathrm{time}}$ of sending rates (Bernoulli loss).

- For all protocols, the overall trend is that $\mathrm{CoV}_{\mathrm{time}}$ increases with increasing loss rates. In other words, the smoothness of the protocols reduces with increasing loss rate.
- At the low loss rate of 1%, TCP has the largest $\mathrm{CoV}_{\mathrm{time}}$ of 0.51, which indicates that TCP smoothness at low loss rate is the worst. GAIMD $\mathrm{CoV}_{\mathrm{time}}$ at this loss rate is the second largest with a value of 0.3. TFRC and TEAR have similar $\mathrm{CoV}_{\mathrm{time}}$ at this loss rate with values of 0.23 and 0.22, respectively. We observe that these experimental values, 0.51, 0.3, 0.23 and 0.22, are close to the analytical predictions of 0.58 from Eq. (5), 0.26 from Eq. (6), 0.22 from Eq. (9), and 0.21 from Eq. (10), respectively.
- At 8% loss rate, GAIMD $\mathrm{CoV}_{\mathrm{time}}$ increases to be the same as that of TCP. This is not surprising since at high loss rate, timeout dominates AIMD approaches. From Eq. (8), we anticipate a $\mathrm{CoV}_{\mathrm{time}}$ of 1.73 at 20% loss rate. We observe that this analytical prediction is close to the measured value of 1.6.
- TFRC $\mathrm{CoV}_{\mathrm{time}}$ stays low for up to 4% loss rate. Then it increases very fast and exceeds TCP and GAIMD at 15% loss rate. At 20% loss rate, $\mathrm{CoV}_{\mathrm{time}}$ of TFRC increases to 2, which is in the analytical prediction range of 0.8–2.4. Since TFRC has the highest $\mathrm{CoV}_{\mathrm{time}}$ at high loss rate, it indicates that TFRC smoothness and fairness become the worst at high loss rate.
- TEAR keeps a low $\mathrm{CoV}_{\mathrm{time}}$ of 0.2–0.4 across the range of measured loss rates. These experimental values agree with the analytical prediction of 0.21. These results show that TEAR can maintain a relatively stable smoothness and fairness performance over a wide range of loss rates.

Besides Bernoulli loss model, for high multiplexing environments, we have also conducted experiments with the ON/OFF loss model. We found that under ON/OFF loss the smoothness of a protocol is slightly worse than that under Bernoulli loss. To understand the reason for the higher fluctuations, we investigated the bottleneck queue length. We found that ON/OFF background traffic causes large fluctuations of bottleneck queue length. Therefore, we can expect large fluctuations in sending rates of responsive competing traffic.

3.3.2. Low multiplexing environments

We again start from the simplest environment. Fig. 7 shows flow sending rate traces when a single flow is sending across the bottleneck link. From simulation configuration, we know that the bandwidth delay product is about 30 packets and the bottleneck link has a buffer size of 50 packets, therefore, there can be only 80 outstanding packets. When the congestion window size of a TCP/GAIMD flow exceeds 80 packets, a packet will be dropped and the flow's window size will be reduced. However, since most of the time the congestion window allows sending at a rate that is higher than the bottleneck link speed, the achieved sending rate is limited by ACK arrivals. Therefore, TCP/GAIMD sending rates are stable at the bottleneck link speed until they experience packet loss and the window size drops below 30 packets.

As in the previous ON/OFF case, to understand the reason for sending rate fluctuations, we plot in Fig. 8 the fluctuations of the bottleneck queue length. We make the following observations: (1) The queue length under TCP exhibits large variations. TCP builds up the queue very quickly; when the queue is full and a packet is dropped, TCP backs off, and the queue drains to be empty very quickly. (2) GAIMD behavior is similar, but the fluctuations of queue length are much smaller than those of TCP. (3) TEAR queue behavior is similar to TCP—fast ramp up to the peak and quickly drain out, but the cycle of TEAR queue fluctuation is about two times of TCP. We notice that during half of the cycle TEAR queue is almost empty. The reason, as we will see in Section 5, is that TEAR is very slow in accelerating its sending rate. (4) Similar to GAIMD, TFRC does not drain the queue to be empty, and its queue length fluctuations are smaller. Similar to TEAR, the cycle of TFRC queue length fluctuation is much longer than those of TCP and GAIMD.

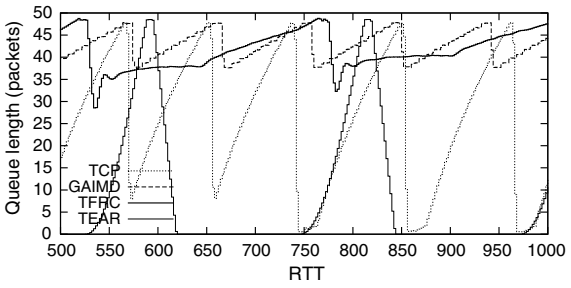


Fig. 8. Queue length (1 flow).

Following the simplest low multiplexing environment, we next consider an environment with several competing congestion avoidance flows. Fig. 9 shows flow sending rate traces when 7 TCP flows are competing with the flow under study. We observe that TFRC and TEAR can maintain relatively smooth sending rates, while the sending rates of TCP and GAIMD fluctuate.

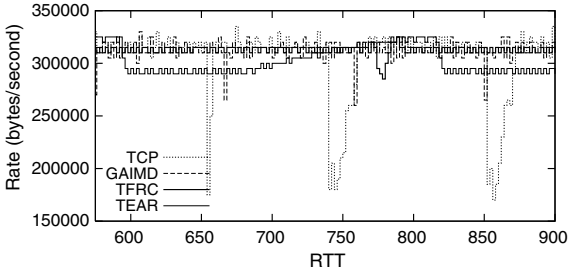


Fig. 7. Sending rates (1 flow).

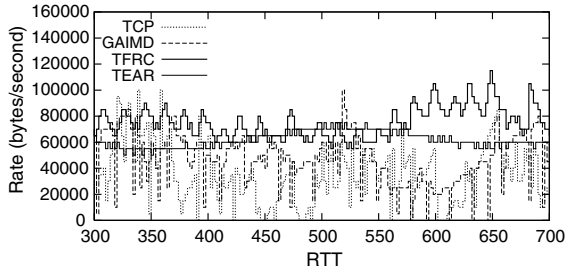


Fig. 9. Sending rates (1 flow + 7 TCP).

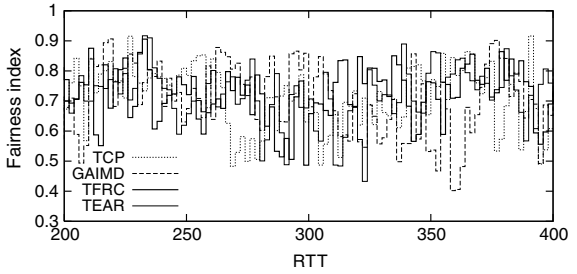


Fig. 10. Fairness index (1 flow + 7 TCP).

Since in this environment 8 TCP-friendly flows are competing against each other, we investigate the fluctuations of short-term fairness index. Fig. 10 plots fairness index at each RTT. As we stated in Section 3.1.1, short-term fairness is correlated to sending rate smoothness. We know that TCP smoothness metric $\mathrm{CoV}_{\mathrm{time}}^{\mathrm{TCP}}$ is 0.58, and the 7 TCP flows dominate in this experiment. Therefore, plugging $\mathrm{CoV} = 0.58$ into Eq. (3), we predict a short-term fairness index of $F = 1/(1 + 0.58^2) = 0.7$. From Fig. 10, we see that the simulation results fluctuate around the analytical value. We have also repeated this simulation with RED link, which has a different loss model; the result is similar.

Fig. 11 presents this experiment from another perspective: the fluctuations of the bottleneck queue length. Comparing Fig. 11 with Fig. 8, we observe that the queue behavior in this experiment is similar to the queue behavior of a single TCP flow, but the fluctuations have shorter cycles. Therefore, we get higher fluctuations in sending rates.

Protocol behaviors are different in an environment consisting of flows that belong to the same protocol. Fig. 12 shows flow sending rate traces when the 7 competing flows belong to the same protocol as the flow under study. Comparing with Fig. 9, we observe that in single protocol environment the smoothness of GAIMD, TFRC, and TEAR improves.

Fig. 13 investigates the fluctuations of short-term fairness. It is particularly interesting to notice that the short-term fairness indices of TFRC and TEAR are close to 1, and the indices do not exhibit large variations. Therefore, it shows that TFRC and TEAR have better short-term fairness performance than that of TCP.

We next compare our analytical results of short-term fairness index with those from simulations. First consider TFRC and TEAR. We know that their $\mathrm{CoV}_{\mathrm{time}}$ are about 0.21. Plugging this value into Eq. (3), we have $F = 1/(1+ 0.21^2) = 0.96$, which is close to 1. As for GAIMD, we know its $\mathrm{CoV}_{\mathrm{time}}$ is 0.26. Plugging this value into Eq. (3), we have $F = 1/(1 + 0.26^2) = 0.93$, which is slightly higher than the experimental result in Fig. 13.

Fig. 14 investigates the fluctuations of the bottleneck queue length for this experiment. The

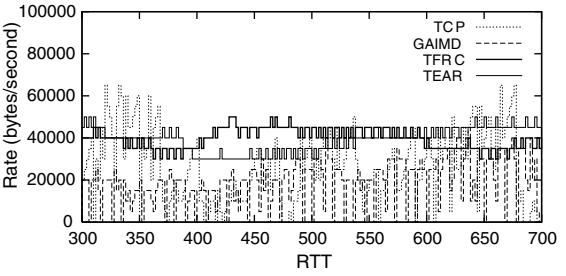


Fig. 12. Sending rates (same protocol, 8 flows).

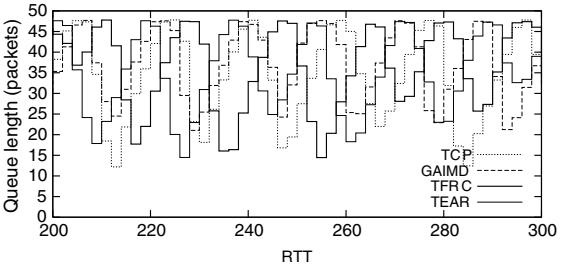


Fig. 11. Queue length (1 flow + 7 TCP).

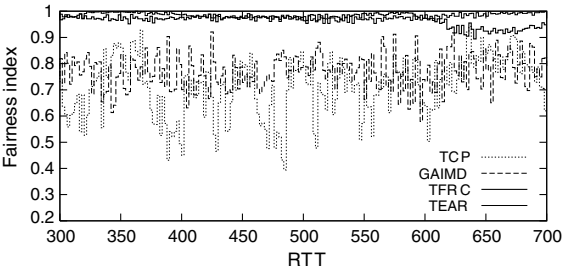


Fig. 13. Fairness index (same protocol, 8 flows).

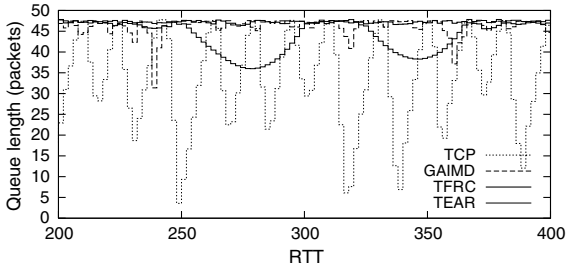


Fig. 14. Queue length (same protocol, 8 flows).

queue behaviors are different between Figs. 11 and 14. In particular, when all flows belong to TFRC or TEAR, they can maintain a high and stable queue length. The queue length of GAIMD is also relatively high and stable. Therefore, for these three protocols, we can expect smaller delay jitter and smoother sending rates than those of TCP.

In the last experiments in stationary environments, we evaluate the long-term fairness $\mathrm{CoV}_{\mathrm{lf}}$. Fig. 15 shows the simulation results for $\mathrm{CoV}_{\mathrm{lf}}$ of 32 flows belonging to the same protocol when Bernoulli loss rates are varied from 0.5% to 17%. In this simulation, TCP is based on TCP/SACK, and the measurement interval is 15 s. We observe that this figure is very similar to Fig. 6 in terms of trend and relative orders among the four protocols. This is not surprising given the relationship we observed from Eq. (2). In another experiment, we have also evaluated $\mathrm{CoV}_{\mathrm{lf}}$ when the measurement interval is 60 s, which is four times longer. We observed that, at low loss rate, $\mathrm{CoV}_{\mathrm{lf}}$ with 60 s measurement interval is about half of that with 15 s measurement interval. These results validate the relationship in Eq. (2).

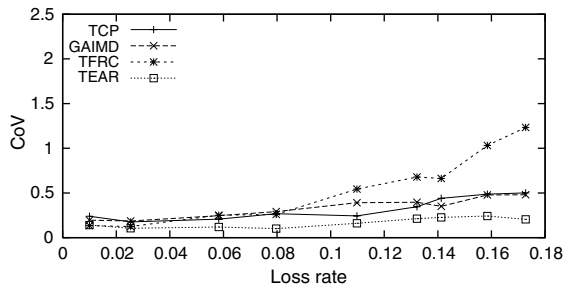


Fig. 15. $\mathrm{CoV}_{\mathrm{lf}}$ (same protocol, 32 flows, 15 s average).

4. Responses to step increase of congestion

In this section, we evaluate protocol responsiveness. In the terminology of control theory, what we study are protocol responses to a step increase function.

We first define the metric. Our metric to measure protocol responsiveness is the number of RTTs D for a protocol to decrease its sending rate to half under a persistent congestion, i.e., one loss indication for each RTT. This metric has also been used in [11,17].

4.1. Analytical results

Since TCP takes only one RTT to reduce its sending rate to half, its responsiveness $D_{\mathrm{TCP}} = 1$. For GAIMD, $D_{\mathrm{GAIMD}} = \log_{7/8} 0.5 \approx 5$. As for TFRC, from [11], we have $D_{\mathrm{TFRC}} = 5$. For TEAR, denote W as the steady state window size just before persistent congestion. We know that TEAR sending rate is $3W/4$ per RTT before persistent congestion, and that all of the eight entries in its history window are $3W/4$. After five consecutive congestion indications, the eight entries in its history window become $\{W/32, W/16, W/8, W/4, W/2, 3W/4, 3W/4, 3W/4\}$. Therefore, TEAR sending rate after five loss indications will be reduced to half, and we have $D_{\mathrm{TEAR}} = 5$.

4.2. Simulation results

We start our simulation with periodic loss model. Fig. 16 shows protocol responses when loss rate is increased from 1% to 4% at RTT 1000, which is indicated as a vertical line in the figure. Clearly, TCP is the fastest of the four protocols to respond to loss rate increase; GAIMD follows; TFRC, and TEAR have similar responding speed and are obviously slower than TCP and GAIMD. However, we observe that TCP over-reacts and drops its sending rate to almost 0. Due to this behavior, *TCP takes as long to reach its new stable state as the other three protocols.*

Next we consider protocol responses to Bernoulli loss rate change. Fig. 17 shows protocol responses when at RTT 1000 Bernoulli loss rate is increased from 0.5% to 100%, i.e., all data packets

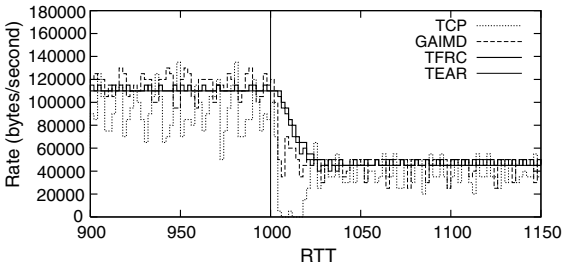


Fig. 16. Sending rates (periodic loss, $p = 1\% \rightarrow 4\%$).

are dropped at the bottleneck link from the sender to the receiver. Since no data packet can go through, we expect that a responsive protocol would reduce its sending rate to almost 0. Among the four protocols, TCP responds at the highest speed and reduces its sending rate to almost 0. GAIMD is the second; it also reduces its rate to almost 0. TFRC is the third with a reasonable responding speed. The responding speed of TEAR is very slow. Furthermore, what we show here is the behavior of TEAR with timeout mechanism added. In our initial experiments, where TEAR timeout mechanism is not implemented, TEAR does not reduce its sending rate at all. Another potential problem with TEAR is that TEAR puts all control functionality in the receiver. Therefore, if the feedback channel from the receiver to the sender is totally congested, the sender will keep on sending at previous rate. To rectify this potential problem, we suggest that TEAR sender should reduce its sending rate if no feedback has arrived for a certain amount of time.

Protocol responses in the previous experiment are mainly determined by their timeout and self-clocking mechanisms (either at the sender or at the receiver); they show different responses when data packets can still go through the bottleneck link. Fig. 18 tests the protocols when Bernoulli loss rate is increased from 1% to 4% at RTT 1000. Since these loss rates are relatively low, we expect a TCP-friendly protocol to reduce its sending rate to half. From Fig. 18 we observe that TCP is the fastest to respond, and GAIMD follows. However, TCP drops below the new target state and recovers slowly from its over-reaction. On the other hand, GAIMD, TFRC, and TEAR have slower response speed than that of TCP, but none of them has over-reaction.

Instead of controlling loss rate directly, next we test the protocols by introducing new flows into a steady environment. We first consider the case when eight new TCP flows start at time 1000 when one flow of the protocol under study and seven competing TCP flows are in steady state.

In this experiment, we find the queue behavior is particularly interesting. Fig. 19 shows queue length traces at the bottleneck link. At about RTT 1025, the queue lengths for all four protocols exhibit large dips. This suggests that the old flows back off due to the introduction of new flows. Thus, the bottleneck queue length decreases.

Next, we study protocol responses in a single protocol environment. Fig. 20 shows the response of a flow as eight new flows start at time 1000 when the flow and the seven competing flows are in steady state. It is clear from this figure that TCP and GAIMD respond very fast, but both protocols overshoot to 0. TFRC and TEAR reduce their speeds slower. However, they do manage to reduce gradually to the new states.

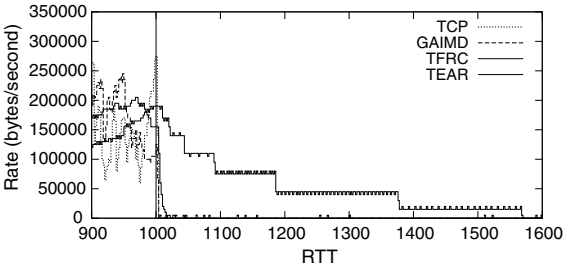


Fig. 17. Sending rates (Bernoulli loss, $p = 0.5\% \rightarrow 100\%$).

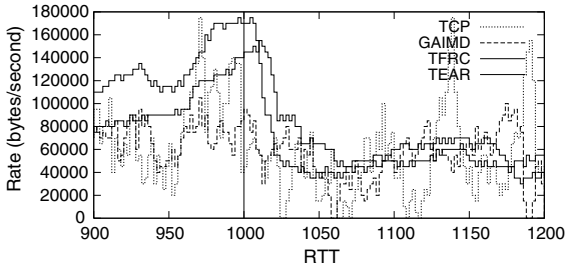


Fig. 18. Sending rates (Bernoulli loss, $p = 1\% \rightarrow 4\%$).

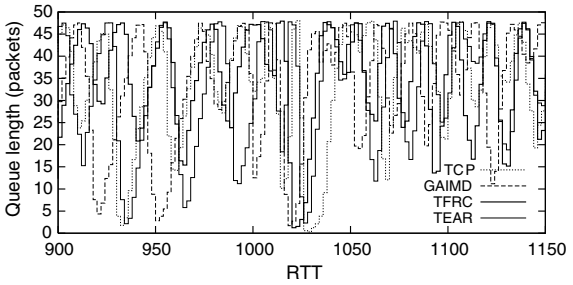


Fig. 19. Queue length (1 flow + 7 TCP → 1 flow + 15 TCP).

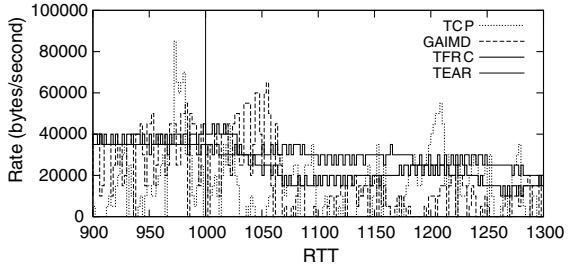


Fig. 20. Sending rates (same protocol, 8 → 16 flows).

We next study the behavior of the bottleneck queue. We expect that the queue will be in overload state for a longer period of time for a less responsive protocol. Fig. 21 investigates queue lengths before and after we increase the number of flows. In this figure, increasing the number of flows generates a large dip of queue length for TEAR around RTT 1050. This dip indicates that the responses of TEAR flows are much longer delayed.

As another measure of protocol transient behaviors when network congestion is increased, we consider fairness indices before and after the disturbance. Fig. 22 shows the fluctuations of fairness

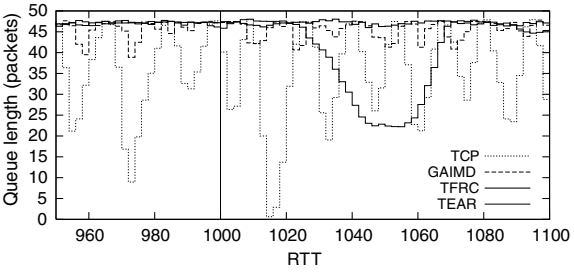


Fig. 21. Queue length (same protocol, 8 → 16 flows).

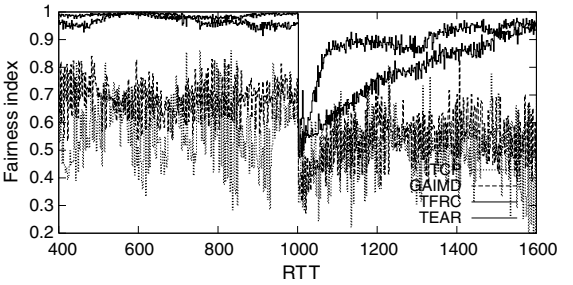


Fig. 22. Fairness index (same protocol, 8 → 16 flows).

indices when the number of flows belonging to the same protocol is doubled. The fairness indices of TFRC and TEAR reduce from near to 1 to 0.5 right after the increase. Afterwards, their short-term fairness indices gradually increase to 1 and become stable at a value close to 1. The slow increase of TEAR's index indicates that the new TEAR flows are slow in increasing their sending rates, which is also observed in Fig. 21. As we have already observed in Section 3, the fairness indices fluctuate for both TCP and GAIMD. We also notice that their fairness indices reduce after the number of flows is doubled. This decrease of fairness index is a result of increased loss rate.

5. Responses to step increase of bandwidth

We evaluate protocol aggressiveness in this section. In the terminology of control theory, what we will study are protocol responses to a step increase function of available bandwidth.

As in the previous section, we first define our metric. Our single-number-of-merit metric to measure protocol aggressiveness is a protocol's increasing speed I per RTT. Since protocol increasing speed depends on other factors such as feedback interval, what we derive is an upper bound. However, we notice that the metric I can be used to optimize application performance.

5.1. Analytical results

TCP increases its rate by 1 per RTT in congestion avoidance state. Thus its aggressiveness metric I_{TCP} equals to 1; likewise $I_{\mathrm{GAIMD}} = \alpha = 0.31$.

As for TFRC, from [11], we have $I_{\mathrm{TFRC}} = 0.12$ without history discounting and $I_{\mathrm{TFRC}} = 0.22$ with history discounting. For TEAR, we know that, when there is no loss, the receiver increases its estimation of the sending rate from $\sum_{i=0}^{t}(W+i)/(t+1) = W + (t/2)$ at RTT t to $W + ((t+1)/2)$ at RTT $t+1$. Since the weight of the most recent epoch is $1/6$, the upper bound of I_{TEAR} is $1/12$.

5.2. Simulation results

We again start with periodic loss. Fig. 23 shows protocol responses when loss rate is decreased from 4% to 1% at RTT 1000. It is obvious from this figure that TCP is the fastest to utilize new bandwidth. GAIMD and TFRC with history discounting have similar increasing speed but GAIMD is slightly faster. TEAR is the slowest to increase its sending rate. For this experiment, we observe that the relative order of the protocols to increase their sending rates conforms to our analytical result.

Fig. 24 shows another experiment where periodic loss rate is decreased from 3% to 2%. In this experiment, the history discounting mechanism of TFRC is not activated, and we observe that TEAR and TFRC become similar.

Next, we consider Bernoulli loss model. In Fig. 25, we reduce Bernoulli loss rate from 10% to 0% at $t = 1000$. Since no loss event occurs when $p = 0$, TFRC uses history discounting and increases its sending rate faster than usual. We observe that TCP is the fastest, and GAIMD is faster than TFRC. TEAR is very slow compared with the other three protocols.

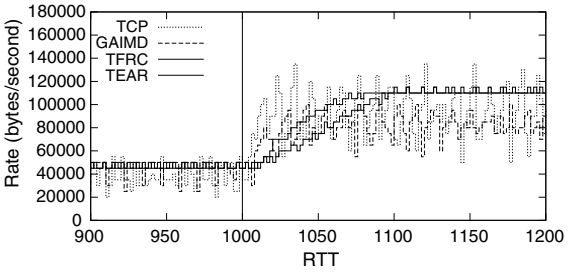


Fig. 23. Sending rates (periodic loss, $p = 4\% \rightarrow 1\%$).

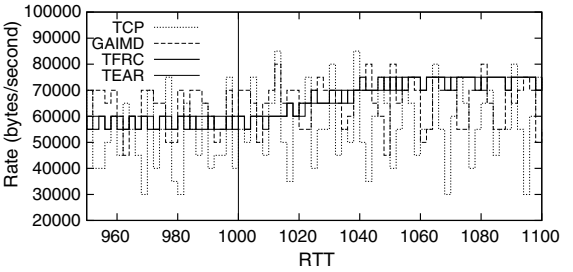


Fig. 24. Sending rates (periodic loss, $p = 3\% \rightarrow 2\%$).

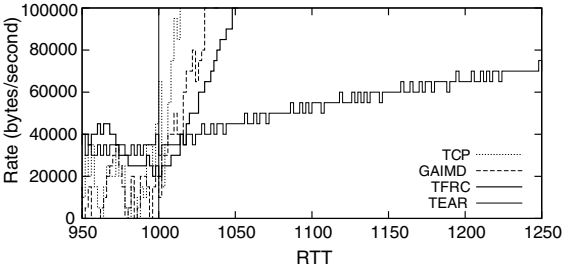


Fig. 25. Sending rates (Bernoulli loss, $p = 10\% \rightarrow 0\%$).

Instead of testing the extreme case when there is no loss at all, Fig. 26 shows protocol responses when we reduce Bernoulli loss rate from 4% to 1%. As is shown in the interval $(1000, 1050)$, TCP is the fastest in increasing its sending rate. GAIMD, TFRC, and TEAR follow it.

Instead of controlling the loss rate directly, next we test the protocols by stopping some flows in a steady environment to increase available bandwidth to remaining flows.

We first consider the case when 8 of the 15 TCP flows stop at time 1000 in Fig. 27. Since we decrease the total number of flows from 16 to 8, the

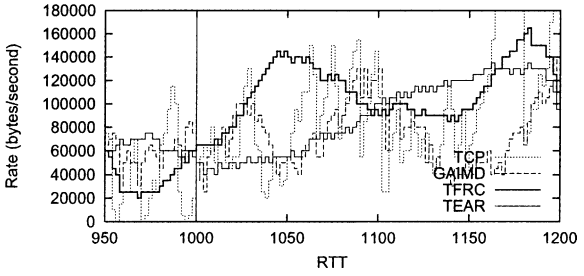


Fig. 26. Sending rates (Bernoulli loss, $p = 4\% \rightarrow 1\%$).

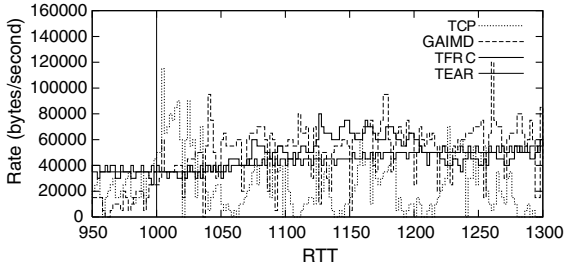


Fig. 27. Sending rates (1 flow + 15 TCP → 1 flow + 7 TCP).

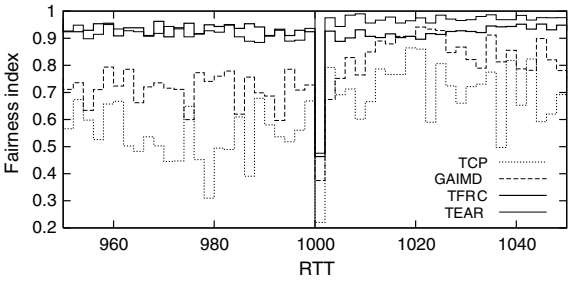


Fig. 28. Fairness index (same protocol, 16 → 8 flows).

sending rates of remaining flows should be doubled. In this figure, TCP and GAIMD respond almost instantaneously, while TFRC and TEAR take much longer to utilize the newly available bandwidth. From a control theory perspective, it appears TEAR is over-damped.

Next we study protocol responses in a single protocol environment. In this experiment, 8 of the 16 flows stop at time 1000. Fig. 28 shows the adaptation of fairness indices. While fairness indices of TFRC and TEAR do not change when the

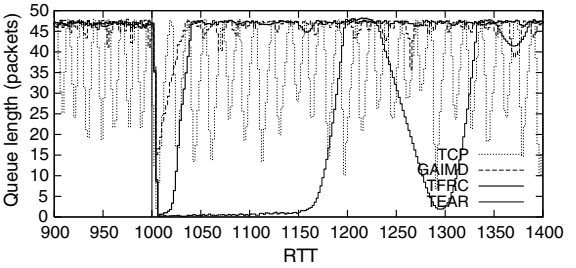


Fig. 29. Queue length (same protocol, 16 → 8 flows).

number of flows decreases, those of TCP and GAIMD increase almost instantaneously. This conforms to the behaviors in Fig. 22, where TCP and GAIMD fairness indices decrease when the number of flows is increased.

Another point to notice is how fast each protocol occupies newly available bandwidth. Fig. 29 shows queue length traces for this experiment. For all four protocols, queue lengths drop to zero when the 8 flows stop. TCP takes only 20 RTTs to fill the queue again. GAIMD and TFRC takes 42 and 45 RTTs, respectively. TEAR takes more than 200 RTTs because its sending rate increasing speed is very slow.

6. Conclusion and future work

We studied analytically and via simulation the transient behaviors of TCP, GAIMD, TFRC, and TEAR. Table 1 summarizes our quantitative results. The first row shows fairness measured by the short-term fairness index F discussed in Section 3. From this metric, we infer that TFRC and TEAR

Table 1
Summary of quantitative results

	TCP	GAIMD	TFRC	TEAR
Fairness	$F = 0.7$ at low loss	$F = 0.9$ at low loss	$F \approx 1$ at low loss; sending rate drops to 0 at high loss	$F \approx 1$ at low loss; sending rate too high at high loss
Smoothness (CoV_{time})	0.58 at 2% loss, 1.7 at 20% loss	0.26 at 2% loss, 1.7 at 20% loss	0.22 at 2% loss, 2 at 20% loss	0.2 at 2% loss, 0.4 at 20% loss
Responsiveness (D)	1 RTT	5 RTTs	5–6 RTTs	5–6 RTTs, slower if no feedback
Aggressiveness (I)	1.0/RTT	0.31/RTT	0.12/RTT w/o history discounting; 0.22/RTT w/ history discounting	0.08/RTT; slower with delayed feedback

have better fairness performance at low loss rate than TCP and GAIMD. However, they have undesirable behaviors at high loss rate, i.e., TFRC sending rate dropping to almost zero and TEAR sending rate being too high compared to TCP. The second row summarizes protocol smoothness measured by $\mathrm{CoV}_{\mathrm{time}}$. From this metric, we observe that TFRC and TEAR have better smoothness performance at low loss rate. While TEAR can maintain a stable smoothness performance up to 20% loss rate, TFRC becomes the worst of the four protocols at 20% loss rate. The third row summarizes protocol responsiveness measured by D defined in Section 4. From this metric, we see that TCP is the most responsive among the four protocols. The other three have similar responsive speed. The last row summarizes protocol aggressiveness measured by I defined in Section 5. This metric shows that TCP is the fastest protocol in utilizing extra bandwidth. TFRC, with history discounting, is slightly slower than GAIMD. TEAR is the slowest to increase sending rate.

Some issues that we have not studied include the impact of different RTTs, and the transient behaviors of congestion control protocols in diffserv environments. Also, it will be interesting to investigate the impact of variations of sending rates (especially for TCP and GAIMD) on protocol responsiveness and aggressiveness. We defer these issues to a future study.

Acknowledgements

Research sponsored in part by NSF grant no. ANI-9977267, ONR grant no. N00014-99-1-0402, and TARP grant no. 003658-0439-2001. Experiments were performed on equipment procured with NSF grant no. CDA-9624082.

Appendix A. AIMD mean and CoV when loss rate is small

Consider the window size W_n just after a packet loss. We can express the congestion window update rule as

$$W_{n+1} = \beta W_n + \alpha X_n. \tag{A.1}$$

Notice that we use α instead of α/RTT for ease of typing.

From [23–25], we know that the above stochastic difference equation has a stationary solution,

$$W_n^* = \alpha \sum_{k=0}^{\infty} \beta^k X_{n-1-k}. \tag{A.2}$$

Moreover, even if W_0 starts from an arbitrary value, it will converge almost surely to the above stationary distribution, i.e.,

$$\lim_{n\to\infty} |W_n - W_n^*| = 0. \tag{A.3}$$

Now, we can evaluate the mean and variance of W_n. Follow the convention, denote $m_1 = E[X_i]$, $m_2 = E[X_i^2]$ and $R(k) = E[X_n X_{n+k}]$, then

$$E[W_n^*] = m_1 \frac{\alpha}{1-\beta}. \tag{A.4}$$

As for variance, we have

$$E[(W_n^*)^2] = \frac{\alpha^2}{1-\beta^2}\left(R(0) + 2\sum_{k=1}^{\infty} \beta^k R(k)\right). \tag{A.5}$$

However, the above expectations are event average, we are more interested in time average. Through Palm inverse formula [26], we have

$$E[g(W(t))] = \frac{1}{m_1} E^0\left[\int_0^{X_0} g(W(t))\,\mathrm{d}t\right]. \tag{A.6}$$

First we consider $E[W(t)]$, and we have

$$E[W(t)] = \frac{1}{m_1} E^0\left[\int_0^{X_0} (\beta W_0 + \alpha t)\,\mathrm{d}t\right] \tag{A.7}$$

$$= \frac{1}{m_1} E^0\left[\beta W_0 X_0 + \frac{\alpha}{2} X_0^2\right] \tag{A.8}$$

$$= \frac{1}{m_1} E^0\left[\alpha\beta \sum_{k=0}^{\infty} \beta^k X_{-1-k} X_0\right] + \frac{\alpha}{2m_1} E^0[X_0^2] \tag{A.9}$$

$$= \frac{\alpha}{m_1}\left[\frac{1}{2}R(0) + \sum_{k=1}^{\infty} \beta^k R(k)\right]. \tag{A.10}$$

Next, we consider $E[W(t)^2]$, we have

$$E[W(t)^2] = \frac{1}{m_1}E^0\left[\int_0^{X_0}(\beta W_0 + \alpha t)^2\,\mathrm{d}t\right] \tag{A.11}$$

$$= \frac{1}{m_1}E^0\left[\beta^2 W_0^2 X_0 + \alpha\beta W_0 X_0^2 + \frac{\alpha^2}{3}X_0^3\right] \tag{A.12}$$

$$= \frac{1}{m_1}E^0\left[\alpha^2\beta^2\left(\sum_{k=0}^{\infty}\beta^k X_{-1-k}\right)^2 X_0 + \alpha^2\beta \times\left(\sum_{k=0}^{\infty}\beta^k X_{-1-k}\right)X_0^2 + \frac{\alpha^2}{3}X_0^3\right]. \tag{A.13}$$

Next, we consider a special case where the $\{X_i\}$ are i.i.d. In this case, we assume the loss interval follow an exponential arrival with parameter λ.

$$E[(W_n^*)] = \frac{\alpha m_1}{1-\beta}, \tag{A.14}$$

$$E[(W_n^*)^2] = \frac{\alpha^2}{1-\beta^2}\left[m_2 + \frac{2\beta m_1^2}{1-\beta}\right], \tag{A.15}$$

$$\mathrm{CoV}[W_n^*] = \frac{1-\beta}{1+\beta}, \tag{A.16}$$

$$E[W(t)] = \frac{\alpha}{m_1}\left[\frac{m_2}{2} + \frac{\beta m_1^2}{1-\beta}\right] \tag{A.17}$$

$$= \frac{\alpha}{\lambda}\frac{1}{1-\beta} \tag{A.18}$$

and the average sending rate is

$$T = \frac{E[W(t)]}{\mathrm{RTT}} \tag{A.19}$$

$$= \frac{\alpha}{\lambda \mathrm{RTT}}\frac{1}{1-\beta}. \tag{A.20}$$

Next, we consider $E[W(t)^2]$. From the above, we know that

$$E[W(t)^2] = \alpha^2\beta^2\left(\frac{m_2}{1-\beta^2} + \frac{2\beta}{1-\beta^2}\frac{m_1^2}{1-\beta}\right) \tag{A.21}$$

$$+ \alpha^2\beta\frac{m_2}{1-\beta} \tag{A.22}$$

$$+ \frac{\alpha^2}{3}\frac{m_3}{m_1}. \tag{A.23}$$

Plug in the parameters, and we have

$$\mathrm{Var}[W(t)] = \frac{\alpha^2}{(1-\beta^2)\lambda^2} \tag{A.24}$$

and

$$\mathrm{CoV}[W(t)] = \sqrt{\frac{1-\beta}{1+\beta}}. \tag{A.25}$$

Appendix B. AIMD CoV in timeout states

In this section, we derive the CoV formula for AIMD when timeout loss indications dominate. From [12], we can see that this will happen when loss rate is above 20%.

Fig. B.1 shows a Markovian state transition diagram. Each state represents one back-off factor. We have also analyzed a model considering slow-start effect, but the result is similar. Finding the steady state distribution for this Markovian chain, we have

$$p_{2^i} = (1-p)p^i, \quad \text{where } 0 \leqslant i \leqslant 5, \qquad p_{64} = p^6. \tag{B.1}$$

Denote t as the ratio of the timeout value and RTT. We know that in state 1, the sender will send

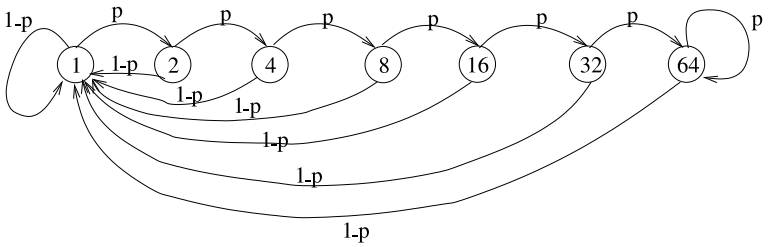


Fig. B.1. Timeout state transition diagram.

one packet in t RTTs, in state 2, the sender will send one packet in one of the $2t$ RTTs, etc. Therefore, the probability of sending a packet in a RTT will be $\pi_1 = \sum_{k=0}^{k=6} p_{2^k}(1/(2^k t))$.

Therefore, the CoV will be

$$\text{CoV} = \sqrt{\frac{64(t-1) + 32p + 16p^2 + 8p^3 + 4p^4 + 2p^5 + p^6}{64 - 32p - 16p^2 - 8p^3 - 4p^4 - 2p^5 - p^6}}. \tag{B.2}$$

Appendix C. TFRC CoV

At low loss rate, the sending rate of TFRC per RTT becomes

$$R(t) = \sqrt{\frac{3s}{2}} \tag{C.1}$$

where s is the weighted average of 8 loss intervals $\{s_i\}_{i=1}^{8}$,

$$s = \sum_{k=1}^{8} w_i s_i. \tag{C.2}$$

First, without considering the smooth effect, we consider the CoV of the sending process

$$R'(t) = \sqrt{\frac{3s_i}{2}}. \tag{C.3}$$

Denote p as the per packet loss probability, and $q = 1 - p$. Consider the Markovian model in Fig. C.1. The reason that the departure rate at state i is proportional to $\sqrt{i}$ is that we assume Bernoulli loss model, therefore, the higher the sending rate is, the faster the rate to leave the state.

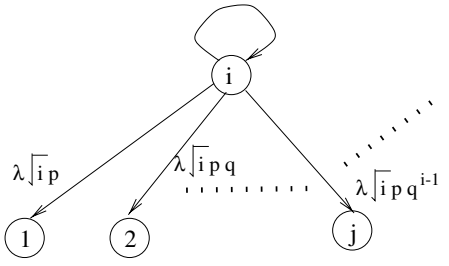


Fig. C.1. TFRC low loss rate state model.

Solve the local balance equation:

$$P_i \lambda \sqrt{i} p q^{i-1} = P_j \lambda \sqrt{j} q^{j-1}. \tag{C.4}$$

We have that the probability in each state is given by

$$P_i = c \frac{q^i}{\sqrt{i}} \tag{C.5}$$

where c is the normalizing factor so that the summation of P_i is 1.

With the distribution, we can find $E[R']$ as

$$E[R'] = \sum_{k=1}^{\infty} \lambda \sqrt{i} c \frac{q_i}{\sqrt{i}}, \tag{C.6}$$

$$E[(R')^2] = \sum_{k=1}^{\infty} \lambda^2 i c \frac{q_i}{\sqrt{i}}. \tag{C.7}$$

From the above two expressions, we can derive the expression for $\text{CoV}[R']$. Numerically calculating the value, we have $\text{CoV}[R']$ have values between 0.66 and 0.54 when the loss rate varies from 0.5% to 9%.

Approximate the smoothed version of TFRC CoV by

$$\text{CoV}[R] = \left(\sum_{k=1}^{8} w_i^2 \right) \text{CoV}[R']. \tag{C.8}$$

We know that $\text{CoV}[R]$ varies between 0.25 and 0.21.

Under high loss rate, as a first approximation, the sending rate will be proportional to the inverse of $p^{3/2}$. Similar to the above approach, we can calculate the $\text{CoV}[R']$ at about 2.2 when loss rate is about 20%, and the smoothed version will have CoV of about 0.8.

As another approximation, we consider the case when the sending rate is proportional to the inverse of $p^{5/2}$ (which is the middle of 3/2 and 7/2), we can calculate the $\text{CoV}[R']$ at about 6.4 when loss rate is about 20%, and the smoothed version will have CoV of about 2.4.

Appendix D. TEAR CoV

TEAR is based on TCP with a sliding window to smooth sending rate. Be specific, the per RTT sending rate of TEAR will be

$$T_{\mathrm{TEAR}} = \sum_{i=1}^{8} w_i \frac{\sum_{k=0}^{W_i - W_{i-1}/2} \left(\frac{1}{2} W_{i-1} + k\right)}{W_i - \frac{1}{2} W_{i-1} + 1}. \quad \text{(D.1)}$$

Consider any term in the above summation:

$$Y_i = \frac{\sum_{k=0}^{W_i - W_{i-1}/2} \left(\frac{1}{2} W_{i-1} + k\right)}{W_i - \frac{1}{2} W_{i-1} + 1} \quad \text{(D.2)}$$

$$= \tfrac{1}{2} W_i + \tfrac{1}{4} W_{i-1}. \quad \text{(D.3)}$$

Therefore,

$$\mathrm{Var}_{\mathrm{time}}[T_{\mathrm{TEAR}}] = \sum_{i=1}^{8} w_i Y_i \quad \text{(D.4)}$$

$$= 0.07708 \mathrm{Var}[W_i] \quad \text{(D.5)}$$

where $\mathrm{Var}[W_i]$ is the variance of the window sizes just before a loss indication arrives.

Therefore,

$$\mathrm{CoV}_{\mathrm{time}}[T_{\mathrm{TEAR}}] = 0.37 \mathrm{CoV}[W_i]. \quad \text{(D.6)}$$

From Eq. (A.16), we have

$$\mathrm{CoV}_{\mathrm{time}}[T_{\mathrm{TEAR}}] = 0.37 \times 0.58 = 0.21. \quad \text{(D.7)}$$

References

[1] S. Floyd, K. Fall, Promoting the use of end-to-end congestion control in the Internet, IEEE/ACM Transactions on Networking 7 (4) (1999) 458–472.

[2] V. Jacobson, Congestion avoidance and control, in: Proceedings of ACM SIGCOMM'88, 1988.

[3] W.-T. Tan, A. Zakhor, Real-time Internet video using error resilient scalable compression and TCP-friendly transport protocol, IEEE Transactions on Multimedia 1 (2) (1999) 172–186.

[4] J. Mahdavi, S. Floyd, TCP-friendly unicast rate-based flow control, note sent to the end2end-interest mailing list, 1997.

[5] T. Turletti, S.F. Parisis, J.-C. Bolot, Experiments with a layered transmission scheme over the Internet, Research Report No 3296, INRIA, November 1997.

[6] D. Sisalem, H. Schulzrinne, The loss-delay based adjustment algorithm: A TCP-friendly adaptation scheme, in: Proceedings of NOSSDAV'98, 1998.

[7] S. Cen, C. Pu, J. Walpole, Flow and congestion control for Internet streaming applications, in: Proceedings of Multimedia Computing and Networking 1998, 1998.

[8] R. Rejaie, M. Handley, D. Estrin, RAP: An end-to-end rate-based congestion control mechanism for realtime streams in the Internet, in: Proceedings of IEEE INFOCOM'99, vol. 3, 1999.

[9] J. Padhye, J. Kurose, D. Towsley, R. Koodli, A model based TCP-friendly rate control protocol, in: Proceedings of NOSSDAV'99, 1999.

[10] L. Vicisano, L. Rizzo, J. Crowcroft, TCP-like congestion control for layered multicast data transfer, in: Proceedings of IEEE INFOCOM'99, vol. 3, 1999.

[11] S. Floyd, M. Handley, J. Padhye, J. Widmer, Equation-based congestion control for unicast applications, in: Proceedings of ACM SIGCOMM 2000, 2000.

[12] Y.R. Yang, S.S. Lam, General AIMD congestion control, in: Proceedings of the 8th IEEE International Conference on Network Protocols, Osaka, Japan, 2000.

[13] I. Rhee, V. Ozdemir, Y. Yi, TEAR: TCP emulation at receivers—flow control for multimedia streaming, Tech. Rep., Department of Computer Science, North Carolina State University, Raleigh, NC, April 2000.

[14] K. Thompson, G.J. Miller, R. Wilder, Wide-area Internet traffic patterns and characteristics, IEEE Network 11 (6) (1997) 10–23.

[15] B. Braden, D. Clark, J. Crowcroft, B. Davie, S. Deering, D. Estrin, S. Floyd, V. Jacobson, G. Minshall, C. Partridge, L. Peterson, K. Ramakrishnan, S. Shenker, J. Wroclawski, L. Zhang, Recommendations on Queue Management and Congestion Avoidance in the Internet, RFC 2309, April 1998.

[16] D.-M. Chiu, R. Jain, Analysis of the increase and decrease algorithms for congestion avoidance in computer networks, Computer Networks and ISDN Systems 17 (1) (1989) 1–14.

[17] S. Floyd, M. Handley, J. Padhye, A comparison of equation-based congestion control and AIMD-based congestion control, work-in-progress. Available from <http://www.aciri.org/tfrc>, 2000.

[18] J.-C. Bolot, End-to-end packet delay and loss behavior in the Internet, in: Proceedings of ACM SIGCOMM'93, 1993.

[19] M. Yajnik, S. Moon, J. Kurose, D. Towsley, Measurement and modelling of the temporal dependence in packet loss, in: Proceedings of IEEE INFOCOM'99, 1999.

[20] Y. Zhang, V. Paxson, S. Shenker, The stationarity of Internet path properties: Routing, loss, and throughput, Tech. Rep., ACIRI, May 2000.

[21] S. Floyd, V. Jacobson, On traffic phase effects in packet-switched gateways, Internetworking: Research and Experience 3 (3) (1992) 115–156.

[22] R. Jain, K.K. Ramakrishnan, D.-M. Chiu, Congestion avoidance in computer networks with a connectionless network layer, Tech. Rep. DEC–TR–506, DEC, August 1987.

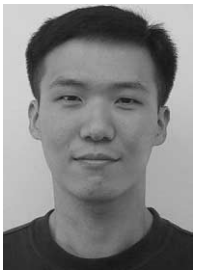
[23] A. Brandt, The stochastic equation $Y_{n+1} = A_n Y_n + B_n$ with stationary coefficients, Advances in Applied Probability 18 (1986) 211–220.

[24] P. Glasserman, D.D. Yao, Stochastic vector difference equations with stationary coefficients, Journal of Applied Probability 32 (1995) 851–866.

[25] E. Altman, K. Avrachenkov, C. Barakat, A stochastic model of TCP/IP with stationary random losses, in: Proceedings of ACM SIGCOMM 2000, 2000.
[26] F. Baccelli, P. Bremaud, Elements of Queueing Theory: Palm-Martingale Calculus and Stochastic recurrences, Springer, Berlin, 1994.

Y. Richard Yang received the B.E. degree in computer science and technology from Tsinghua University, Beijing, China, in 1993, and the M.S. and Ph.D. degrees in computer science from the University of Texas at Austin, in 1998 and 2001, respectively.

Since 2001, he has been with the Department of Computer Science, Yale University, New Haven, CT, where he is an Assistant Professor. His current research interests are heterogeneous networks, mobile ad-hoc networks, and network security.



Min Sik Kim received the B.S. degree in computer engineering from Seoul National University, Seoul, Korea, in 1996. He is currently pursuing the Ph.D. degree in computer science at the University of Texas at Austin. His research interests are in the area of Internet streaming media and congestion control.

Simon S. Lam received the BSEE degree with Distinction from Washington State University, Pullman, in 1969, and the M.S. and Ph.D. degrees in engineering from the University of California at Los Angeles (UCLA) in 1970 and 1974, respectively.

From 1971 to 1974, he was a Postgraduate Research Engineer at the ARPA Network Measurement Center, UCLA, where he worked on satellite and radio packet switching networks. From 1974 to 1977, he was a Research Staff Member at the IBM T.J. Watson Research Center, Yorktown Heights, New York. Since 1977, he has been on the faculty of the University of Texas at Austin, where he is Professor and Regents Chair in Computer Sciences. He served as Department Chair from 1992 to 1994. His current research interests are in network protocol design and analysis, distributed multimedia, and Internet security services.

He served on the editorial boards of IEEE/ACM Transactions on Networking, IEEE Transactions on Software Engineering, IEEE Transactions on Communications, Proceedings of the IEEE, and Performance Evaluation. He was Editor-in-Chief of IEEE/ACM Transactions on Networking from 1995 to 1999.

He currently serves on the editorial board of Computer Networks. He organized and was Program Chair of the inaugural ACM SIGCOMM Symposium held at the University of Texas at Austin in 1983. He is a founding Steering Committee member of the IEEE International Conference on Network Protocols.

Dr. Lam received the 1975 Leonard G. Abraham Prize for the best paper published in IEEE Transactions on Communications, and the 2001 William R. Bennett Prize for the best paper published in IEEE/ACM Transactions on Networking, both from the IEEE Communications Society. He is a Fellow of IEEE (elected 1985) and also a Fellow of ACM (elected 1998).