A Spectrum of TCP-Friendly Window-Based Congestion Control Algorithms

Shudong Jin, Liang Guo, *Student Member, IEEE*, Ibrahim Matta, *Member, IEEE*, and Azer Bestavros, *Member, IEEE*

***Abstract*—The increasing diversity of Internet application requirements has spurred recent interest in transport protocols with flexible transmission controls. In window-based congestion control schemes, increase rules determine how to probe available bandwidth, whereas decrease rules determine how to back off when losses due to congestion are detected. The control rules are parameterized so as to ensure that the resulting protocol is TCP-friendly in terms of the relationship between throughput and loss rate. This paper presents a comprehensive study of a new spectrum of window-based congestion controls, which are TCP-friendly as well as TCP-compatible under RED. Our controls utilize history information in their control rules. By doing so, they improve the transient behavior, compared to recently proposed slowly responsive congestion controls such as general additive-increase and multiplicative-decrease (AIMD) and binomial controls. Our controls can achieve better tradeoffs among smoothness, aggressiveness, and responsiveness, and they can achieve faster convergence. We demonstrate analytically and through extensive *ns* simulations the steady-state and transient behavior of several instances of this new spectrum.**

***Index Terms*—congestion control, fairness, TCP-compatibility, TCP-friendliness, transient behavior.**

I. Introduction

TCP uses additive-increase and multiplicative-decrease (AIMD). It probes available bandwidth by increasing the congestion window size linearly, and responds to increased congestion (indicated by packet losses) by decreasing the window size multiplicatively. Recently proposed congestion control mechanisms include generalizations of TCP-like window-based schemes [1]–[4] and equation-based schemes [5]–[7]. A common objective of these schemes is to reduce the high variability of TCP's transmission rate. Such high variability may limit network utilization. In addition, it is not desirable for emerging applications such as real-time streaming applications on the Internet.

A new transport protocol should implement congestion control mechanisms that interact well with TCP [8]. That is, it should maintain *TCP-compatibility*, or fairness across connections using different protocols. To provide such fairness, one solution is to satisfy *TCP-friendliness*, which means the (λ, p) relationship $\lambda \sim 1/(R\sqrt{p})$ should hold, where λ is the throughput of a flow, p is the loss rate, and R is the round-trip time (RTT).

In addition to TCP-friendliness, *smoothness*, *aggressiveness*, and *responsiveness* [1], [9] are important indices of congestion control performance. Smoothness indicates the variability in transmission rate. Aggressiveness indicates how fast a connection probes extra bandwidth by opening up its window. Responsiveness measures how fast a connection reacts to increased congestion by decreasing its window size. Smoothness characterizes the steady-state behavior of congestion control protocols, whereas both aggressiveness and responsiveness characterize transient behavior. An important observation is that there are tradeoffs among smoothness, aggressiveness, and responsiveness [1], [9]. Comparisons of TCP, general AIMD [1], [3], TFRC [5], and TEAR [2] have shown that typically higher smoothness means less aggressiveness and responsiveness.[1]

A. Motivation

Our work is motivated by the need for new controls that have high smoothness in steady state and high aggressiveness/responsiveness when network conditions change drastically. To that end, we explore the design space between window-based and equation-based congestion control schemes. Previous window-based schemes do *not* use history while equation-based schemes do so. History information can be useful to improve the behavior of previous window-based schemes such as AIMD. For example, the congestion window size in the past is not only an indicator of the current congestion level of the network, but also a good predictor of the congestion state for the future. Furthermore, previous window-based schemes provide smoothness of transmission rate but sacrifice aggressiveness. We answer the question of whether we can provide high smoothness in steady state as well as better transient behavior when network conditions change drastically (e.g., when there is a sudden increase in available bandwidth).

B. Contribution

This paper presents a thorough study of TCP-like window-based congestion control schemes that utilize history information, in addition to current window size. These schemes are fundamentally different from memoryless AIMD [1], [3] and

Manuscript received December 14, 2001; revised May 22, 2002; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor V. Paxson. This work was supported in part by the National Science Foundation under Grant ANI-0095988, Grant ANI-9986397, Grant EIA-0202067, and Grant ITR ANI-0205294. The work of the first author was also supported by an IBM Ph.D. Research Fellowship.

The authors are with the Department of Computer Science, Boston University, Boston, MA 02215 USA (e-mail: jins@cs.bu.edu; guol@cs.bu.edu; matta@cs.bu.edu; best@cs.bu.edu).

Digital Object Identifier 10.1109/TNET.2003.813046

[1]In feedback control systems, of which congestion control is an example, there is inevitable tension between stability and responsiveness. In our context, we use smoothness as a quality measure of stability, and both aggressiveness and responsiveness as measures of responsiveness. Note, in the control-theory literature, responsiveness usually means how fast the system reaches a target state (rise time), whereas we use aggressiveness and responsiveness to distinguish between how fast the window is increased and decreased, respectively, to reach a target window size.

binomial schemes [4]. The only history used in our schemes is the window size at the time of detecting the last loss. Such a small step allows a much broader exploration of TCP-friendly congestion controls than memoryless AIMD and binomial schemes. To this end, we propose a spectrum of window-based congestion controls possessing high smoothness in steady state, while reacting promptly to sudden changes in network conditions. We analyze the smoothness, transient behavior, and performance tradeoffs of this new spectrum of controls, of which our recently proposed square-increase/multiplicative-decrease (SIMD) [10] is an instance. In SIMD, the congestion window size increases super-linearly, in proportion to the *square* of the time elapsed since the detection of the last loss event (alternatively, the increase is inversely proportional to the window size at the time of last loss detection). Thus, SIMD has high aggressiveness and fast convergence to fairness.

Our work is the first step toward exploring a new design space between memoryless window-based congestion control schemes and equation-based schemes which use more history information. Compared to memoryless window-based schemes, our controls improve the transient behavior by using history. Compared to equation-based schemes, our controls have several unique properties: the self-clocking nature of window-based schemes, and simple modifications to TCP's implementation.

The remainder of this paper is organized as follows. We propose our controls in Section II, and define our TCP-friendly controls in Section III. We analyze the tradeoffs among smoothness, aggressiveness, and responsiveness in Section IV. The convergence properties of our SIMD instance is studied in Section V. Our results from extensive simulations using the *ns* simulator [11] are presented in Section VI. We revisit related work in Section VII and, finally, conclude the paper.

II. Window-Based Congestion Control Using History

A TCP-like window-based congestion control scheme increases the congestion window as a result of the successful transmission of a window of packets, and decreases the congestion window upon the detection of a packet loss event. We call such a sequence of window increments followed by one window decrement a *congestion epoch*. A window-based congestion control scheme defines one control rule for window increase, and another rule for window decrease. AIMD uses the following control rules:

$$\text{Increase: } w_{t+1} \leftarrow w_t + \alpha, \quad \alpha > 0$$
$$\text{Decrease: } w_t \leftarrow w_t - \beta w_t, \quad 0 < \beta < 1.$$

where w_t is the window size at time t (in RTTs). That is, for AIMD, the window size is increased by a constant when a window of packets are transmitted successfully, and it is decreased by a constant factor instantaneously when a packet loss event is detected.[2] Binomial controls [4] generalize AIMD and use the following control rules:

$$\text{Increase: } w_{t+1} \leftarrow w_t + \alpha / w_t^k, \quad \alpha > 0$$
$$\text{Decrease: } w_t \leftarrow w_t - \beta w_t^l, \quad 0 < \beta < 1.$$

[2]We use $\mathrm{AIMD}(\alpha, \beta)$ to refer to the general AIMD with additive constant α and multiplicative decrease parameter β. The term TCP AIMD refers to $\mathrm{AIMD}(1, 0.5)$ or standard TCP. For simplicity, we also use AIMD for the general case.

That is, binomial controls generalize additive-increase by increasing inversely proportional to a power k of the current window, and generalize multiplicative-decrease by decreasing proportional to a power l of the current window.

We say that AIMD and binomial controls are memoryless since the increase and decrease rules use only the current window size w_t and constants (α, β, k, and l). Neither of them utilizes history information. We argue that the window size at the end of the last congestion epoch is useful, not only as an indicator of the current congestion level of the network, but also as a good predictor of the congestion state for the next epoch. Thus, our proposed scheme maintains such a state variable $w_{\max}$, which is updated at the end of *each* congestion epoch. In addition, let w_0 denote the window size after the decrease. Given a decrease rule, w_0 can be obtained from $w_{\max}$, and *vice versa*. For example, for AIMD, $w_0 = (1 - \beta) w_{\max}$. Henceforth, for clarity, we use both $w_{\max}$ and w_0.[3]

Such history information can then be used to improve the transient behavior of the control. We propose to adopt the following window increase function:

$$w(t) = w_0 + ct^u, \quad u, c > 0 \tag{1}$$

where $w(t)$ is the continuous approximation of the window size at time t (in RTTs) elapsed since the window started to increase. By definition, $w_0 = w(0)$. This window increase function is equivalent to the following window increase rule:[4]

$$w_{t+1} \leftarrow w_t + \alpha / (w_t - w_0)^k, \quad \alpha > 0 \tag{2}$$

where $k > -1$ and α is independent of t. In particular, $u = 1/(k+1)$ and $c = ((k+1)\alpha)^u$.

We are interested in congestion control schemes that have various window size increase patterns (different u's, or equivalently, different k's). Consider three cases. First, if $-1 < k < 0$, the congestion window increases super-linearly. The window is increased cautiously just after the detection of packet loss, and the increase becomes more and more aggressive when no more loss occurs. Second, if $k = 0$, the window increases linearly, i.e., additive increase. The aggressiveness does not change with time. Third, if $k > 0$, the window increases sublinearly. The connection approaches the previously probed window size fast, but it becomes less aggressive beyond that. These various schemes possess different degrees of aggressiveness, and may satisfy different applications. For example, super-linear increase

[3]When the slow-start phase of TCP ends and the congestion avoidance phase starts, we have the first value of w_0, i.e., the current window size. Then the first value of $w_{\max}$ is obtained.

[4]Equivalence of window increase function (1) and window increase rule (2): Using linear interpolation and continuous approximation, from (2), we have

$$\frac{dw(t)}{dt} = \frac{\alpha}{(w(t) - w_0)^k}.$$

This gives us

$$(w(t) - w_0)^k dw(t) = \alpha dt,$$

and then by integrating both sides, we have

$$\frac{(w(t) - w_0)^{k+1}}{k+1} = \alpha t + C.$$

Notice that the constant $C = 0$ since when $t = 0, w(t) = w_0$. We then rewrite it as (1):

$$w(t) = w_0 + ((k+1)\alpha t)^{1/(k+1)}.$$

can support applications that need to quickly acquire bandwidth as it becomes available.

Therefore, we consider the following control rules:

$$\begin{aligned}&\text{Increase: } w_{t+1} \leftarrow w_t + \alpha(w_{\max})/(w_t - w_0)^k,\ \alpha(w_{\max}) > 0\\ &\text{Decrease: } w_t \leftarrow w_t - \beta w_t^l, \quad 0 < \beta < 1.\end{aligned} \tag{3}$$

Note that we write α as a function of $w_{\max}$ since this is required in the derivation of TCP-friendliness. In the remainder of this paper, we simply write α for clarity. We use the same decrease rule as binomial controls, thus we do not use history in it.[5] For the increase rule, we consider $k > -1$, since otherwise the window size increases exponentially or faster and we consider it unstable. For the decrease rule, we consider $l \leq 1$, since otherwise $(w_t - \beta w_t^l)$ can be negative when w_t is large enough.

We illustrate this family of controls as the (k, l) space in Fig. 1. In [13], we show that the spectrum inside the shaded area satisfies the convergence-to-fairness property under the synchronized feedback model used by Chiu and Jain [15].

Before further elaboration, we state several main properties of our controls. First, we show that our controls can be TCP-friendly by appropriately defining α as a function of the constant β and the state variable $w_{\max}$. We elaborate on this in Section III. Second, our controls enable different tradeoffs among smoothness, aggressiveness, and responsiveness. We elaborate on this in Section IV. Third, our controls can have better convergence behavior as we show in Section V using SIMD [10] as an instance. For SIMD, $k = -0.5$ and $l = 1$.

We need to point out that our controls are radically different from binomial controls [4]. Binomial controls generalize AIMD, but they are still in the memoryless space. Therefore, binomial controls cannot be simply situated on the spectrum in Fig. 1.

III. TCP-Friendliness

We show that our control scheme using the control rules in (3) can be TCP-friendly. The notion of TCP-friendliness refers to the relationship between throughput and packet loss rate. We consider a random loss model, where the losses are Bernoulli trials; packets are dropped uniformly with a fixed probability.

In Appendix A, assuming such a random loss model, and without considering the effect of TCP's timeout mechanisms, we explain the use of the following definition of α to make our congestion control scheme TCP-friendly:

$$\alpha = \frac{3}{2(k+1)\left(1 - \frac{1}{k+2}\beta w_{\max}^{l-1}\right)}\left(\frac{\beta}{\Gamma\left(\frac{1}{k+1}+1\right)}\right)^{k+1} w_{\max}^{kl+l-1} \tag{4}$$

where the Gamma function $\Gamma(\cdot)$ is a constant. According to Section II, c in (1) is defined as a function of α and we have

$$c = \left(\frac{3}{2\left(1 - \frac{1}{k+2}\beta w_{\max}^{l-1}\right)}\right)^{\frac{1}{k+1}} \frac{\beta}{\Gamma\left(\frac{1}{k+1}+1\right)} w_{\max}^{l-\frac{1}{k+1}}. \tag{5}$$

[5]The use of history in the decrease rule was explored in [12]. Their control, LIMD/H, uses the history of losses *across* "measurement periods" to adapt its backoff strategy, but its increase rule is still additive. Our schemes use history in the increase rule *within* the congestion epoch to improve aggressiveness and convergence-to-fairness.

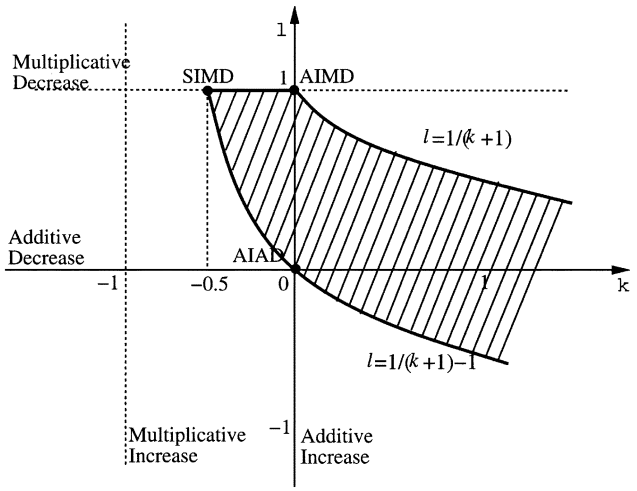


Fig. 1. Spectrum of TCP-friendly congestion controls using history.

When the window size variation is small, i.e., the window decrease is small, $\beta w_{\max}^l \ll w_{\max}$, we can simplify α and c as

$$\alpha \approx \frac{3}{2(k+1)}\left(\frac{\beta}{\Gamma\left(\frac{1}{k+1}+1\right)}\right)^{k+1} w_{\max}^{kl+l-1} \tag{6}$$

$$c \approx \left(\frac{3}{2}\right)^{\frac{1}{k+1}} \frac{\beta}{\Gamma\left(\frac{1}{k+1}+1\right)} w_{\max}^{l-\frac{1}{k+1}}. \tag{7}$$

That is, α is a constant factor of $w_{\max}^{kl+l-1}$ and c is a constant factor of $w_{\max}^{l-(1/k+1)}$.

Table I gives several special cases. We give their control rules and the window increase functions. When $k = 0$ and $l = 1$, from (4) we have $\alpha_{\text{AIMD}} = 3\beta/(2-\beta)$. If $\beta \ll 1$, $\alpha_{\text{AIMD}} \approx 3\beta/2$. It degenerates to the memoryless TCP-friendly AIMD control [1], [3]. When $k = -0.5$ and $l = 1$

$$\alpha_{\text{SIMD}} = \frac{3\sqrt{\beta}}{\left(1 - \frac{2\beta}{3}\right)\sqrt{2w_{\max}}}. \tag{8}$$

If $\beta \ll 1$, $\alpha_{\text{SIMD}} \approx (3\sqrt{\beta})/(\sqrt{2w_{\max}})$. In this case, the window size decreases multiplicatively upon the detection of packet loss, but increases in proportion to the *square* of the time elapsed since the detection of the last loss event (cf. Table I). We call this control square-increase/multiplicative-decrease.

Another way of illustrating TCP-friendliness is to compare our controls with binomial controls. In [4], the authors show that binomial controls are TCP-friendly. We observe that for every instance of the binomial controls, there is a corresponding point along the line where $k = 0$ and $0 \leq l \leq 1$ in Fig. 1 which roughly gives the same control rules. For example, the point $k = l = 0$ (marked as "AIAD" in Fig. 1) corresponds to the special case inverse-increase/additive-decrease (IIAD) of binomial controls. IIAD has the following control rules:

$$\begin{aligned}&\text{Increase: } w_{t+1} \leftarrow w_t + \frac{3\beta}{2w_t}\\ &\text{Decrease: } w_t \leftarrow w_t - \beta.\end{aligned}$$

The only difference between IIAD and our AIAD is in the window increase factor: in IIAD, the factor is inversely proportional to the *current* window size w_t, while in AIAD, the

TABLE I
SEVERAL SPECIAL CASES OF OUR TCP-FRIENDLY CONGESTION CONTROLS USING HISTORY

(k,l)	Increase rule	Decrease rule	Increase function
$k=0, l=1$, AIMD	$w_{t+1} \leftarrow w_t + \frac{3\beta}{2-\beta}$	$w_t \leftarrow w_t - \beta w_t$	$w(t) = w_0 + \frac{3\beta}{2-\beta}t$
$k=-\frac{1}{2}, l=1$, SIMD	$w_{t+1} \leftarrow w_t + \frac{3\sqrt{\beta}}{\sqrt{2}(1-2\beta/3)}\sqrt{\frac{w_t-w_0}{w_{\max}}}$	$w_t \leftarrow w_t - \beta w_t$	$w(t) = w_0 + \frac{9\beta}{8(1-2\beta/3)^2}\frac{1}{w_{\max}}t^2$
$k=0, l=\frac{1}{2}$	$w_{t+1} \leftarrow w_t + \frac{3\beta}{2\sqrt{w_{\max}}-\beta}$	$w_t \leftarrow w_t - \beta\sqrt{w_t}$	$w(t) = w_0 + \frac{3\beta}{2\sqrt{w_{\max}}-\beta}t$
$k=0, l=0$, AIAD	$w_{t+1} \leftarrow w_t + \frac{3\beta}{2w_{\max}-\beta}$	$w_t \leftarrow w_t - \beta$	$w(t) = w_0 + \frac{3\beta}{2w_{\max}-\beta}t$

factor is a constant whose value is inversely proportional to $w_{\max}$.[6] Notice that $w_{\max}$ records the maximum window size in the previous congestion epoch, thus its value is proportional to the time average of w_t if the TCP congestion window has reached steady state. In other words, IIAD and AIAD controls are equivalent in steady state. However, when there is a sudden increase in network bandwidth, AIAD's linear increase rule is more aggressive than the IIAD's sublinear increase rule.

The above observation applies to all instances of binomial controls, with only one exception at $k=0$, $l=1$, i.e., AIMD control, where our control algorithm degenerates precisely to general AIMD. However, as shown earlier, for the whole shaded area in Fig. 1, our controls can be *adjusted* to be TCP-friendly [cf. (4)]. This gives the needed flexibility to *control the transient behavior*. For example, as shown in the next section, by exploiting the history information $w_{\max}$, SIMD control is able to increase the window super-linearly (more aggressively than AIMD) and shows much better transient behavior, without affecting TCP-friendliness.

In this paper, due to space limitation, we only present results for SIMD, AIMD, and AIAD as instances in the spectrum of Fig. 1.

IV. TRADEOFFS AMONG SMOOTHNESS, AGGRESSIVENESS, AND RESPONSIVENESS

In this section, we consider important properties of congestion controls other than TCP-friendliness. These are smoothness, aggressiveness, and responsiveness. Smoothness measures the variability in a connection's window size over time. High variability is not desirable. Aggressiveness measures how fast a connection probes bandwidth as it becomes available by opening up its window. Higher aggressiveness, implying potentially higher utilization, is desirable. Responsiveness measures how fast a connection decreases its window size in response to increased congestion. High responsiveness is desirable.

Smoothness can be observed at different time scales [1]. We consider short time scales since long-term smoothness can be affected by other dynamics in the system. We define smoothness as the variation of the window size of a connection during one congestion epoch. In particular, we use the coefficient of variation of window size in one congestion epoch as a measure of short-term smoothness. Note that the coefficient of variation is not necessarily an accurate measure of smoothness, but it is adequate to give insight into the tradeoffs. We define aggressiveness as the inverse of the time needed for the connection to increase the window size, in response to a step increase of available bandwidth [9]. That is, the available bandwidth is increased by a factor of m. We define responsiveness as the inverse of the number of loss events required for the connection to decrease its window by a substantial amount, in response to a step increase of congestion [9]. That is, a decrease of available bandwidth by a factor of m.

Table II gives the approximate expressions of smoothness, aggressiveness, and responsiveness for AIMD, IIAD, SIMD, and AIAD controls. More details are given in [13]. Intuitively, the smoothness index is proportional to the window decrease divided by the average window size. Aggressiveness is determined by the window size increase function. Responsiveness is determined by the decrease rule.

Numerical results in Fig. 2 show the tradeoffs among smoothness, aggressiveness, and responsiveness. Results for AIAD are not shown here since they are similar to those of IIAD except that AIAD has higher aggressiveness. Fig. 2(a) shows the inverse of aggressiveness of AIMD, SIMD, and IIAD as the coefficient of variation varies. Their special cases, TCP $\mathrm{AIMD}(1/5, 1/8)$, $\mathrm{AIMD}(1/10, 1/16)$, $\mathrm{IIAD}(1, 2/3)$, and $\mathrm{SIMD}(1/16)$ are also shown by points. Note that $\mathrm{AIMD}(1/5, 1/8)$ and $\mathrm{AIMD}(1/10, 1/16)$ are parameterized according to the TCP-friendly condition $\alpha = 3\beta/(2-\beta)$. The inverse of aggressiveness is computed as the number of RTTs necessary to double the window size, i.e., $m = 2$. Fig. 2(b) shows the inverse of responsiveness of AIMD, IIAD, and SIMD as the coefficient of variation varies. The inverse of responsiveness is computed assuming the target window size is half of the current window size, i.e., $m = 2$.

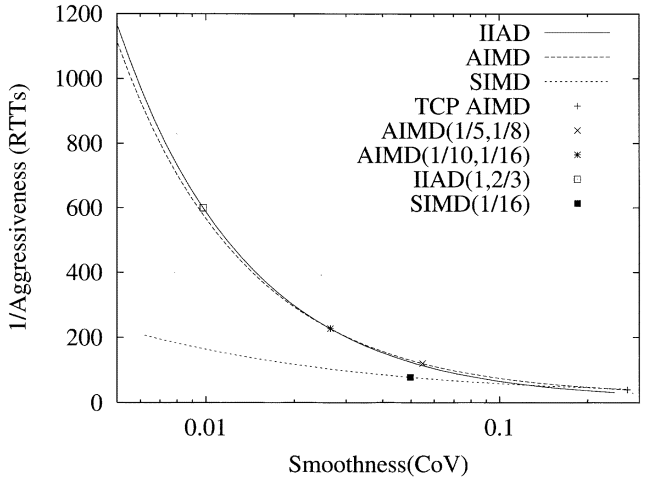
From this figure, we can see that SIMD has much higher aggressiveness (fewer RTTs) than the others, especially when high smoothness (low coefficient of variation) is needed. Meanwhile, SIMD has a slight loss of responsiveness. In particular, SIMD shows up to order of magnitude better aggressiveness at less than about 1.7 times lower responsiveness for about the same smoothness value. For example, we can predict that $\mathrm{AIMD}(1/20, 1/30)$, $\mathrm{SIMD}(1/30)$, and $\mathrm{IIAD}(1, 2/3)$ have comparable smoothness when the average window size is 20. However, $\mathrm{SIMD}(1/30)$ can react to a substantial increase of available bandwidth much faster. The smoothness–aggressiveness relationship can also be inferred from Table II. For both AIMD and IIAD, aggressiveness varies in proportion to the coefficient of variation. For SIMD, aggressiveness varies as the square root of the coefficient of variation. Thus, when the transmission rate is very smooth, SIMD has much higher aggressiveness than AIMD and IIAD.

We should note that, we have not considered the effect of the self-clocking property of window-based schemes in our analysis of responsiveness. When there is a burst of packet losses,

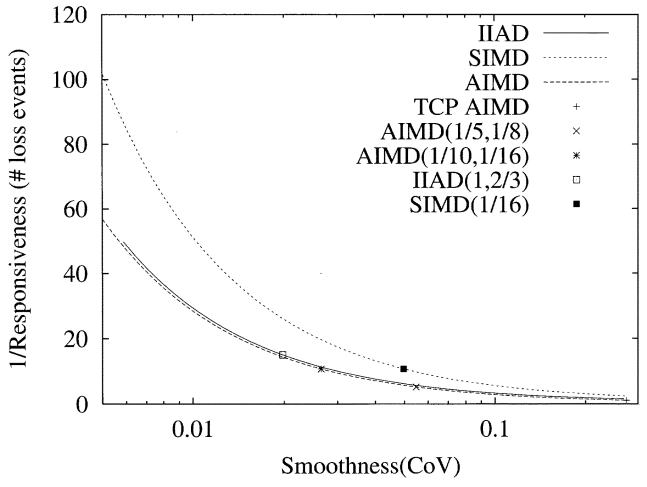
[6]Unlike our history-based AIAD control, memoryless AIAD increases its window by an amount that is constant over all congestion epochs.

TABLE II
SMOOTHNESS, AGGRESSIVENESS, AND RESPONSIVENESS COMPARISONS OF AIMD, IIAD, SIMD, AND AIAD

	Smoothness	*1/Aggressiveness*	*1/Responsiveness*
AIMD	$\frac{0.41\beta}{1-\beta/2}$	$\frac{m-1}{\beta}\frac{2W}{3}$	$\log_{(1-\beta)}\frac{1}{m}$
IIAD	$\frac{0.41\beta}{W-\beta/2}$	$\frac{(m^2-1)W^2}{3\beta}$	$\frac{W(1-1/m)}{\beta}$
SIMD	$\frac{0.73\beta}{(1-2\beta/3)^2}$	$\sqrt{\frac{m-1}{\beta}\frac{2\sqrt{2}W}{3}}$	$\log_{(1-\beta)}\frac{1}{m}$
AIAD	$\frac{0.41\beta}{W-\beta/2}$	$\frac{2(m-1)W^2}{3\beta}$	$\frac{W(1-1/m)}{\beta}$



(a)



(b)

Fig. 2. Tradeoffs of smoothness, aggressiveness, and responsiveness. For (a), we assume available bandwidth is doubled. For (b), we assume the window is reduced to half, i.e., $m = 2$. The initial average window size W before bandwidth changes is 20. (a) Aggressiveness versus smoothness. (b) Responsiveness versus smoothness.

since the connections are acknowledgment (ACK) clocked, it is possible that the congestion window size is reduced to one due to a retransmission timeout regardless of which control is used. Therefore, SIMD's slight loss of responsiveness is even less noticeable in such scenarios. This observation is validated by our simulations in Section VI-B2.

If we use a larger factor m for the sudden increase and decrease of available bandwidth, the advantage of SIMD's aggressiveness is more pronounced [14]. We can observe from Table II that, for SIMD, aggressiveness is inversely proportional to the square root of m, and for AIMD and IIAD, aggressiveness is inversely proportional to m or even m^2, respectively. Therefore, larger m makes SIMD more favorable.

Remark: In the spectrum of controls in Fig. 1, SIMD is the one whose aggressiveness grows the fastest. SIMD has the best tradeoff between smoothness in steady state and aggressiveness during transient periods. As k increases, the spectrum of controls have worse tradeoffs.

V. CONVERGENCE TO FAIRNESS AND EFFICIENCY

In this section, we first show the convergence of our SIMD instance. Then we show that SIMD has better convergence behavior than AIMD.

We adopt the synchronized feedback assumption [15]. This assumption is not realistic in real networks, and our analysis is not a proof of convergence if this assumption does not hold. However the analysis still provides an intuitive way to gain insights. To show that multiple users with synchronized feedbacks using our control scheme converge to fairness, we use the vector space used by Chiu and Jain [15] to view the system state transitions as a trajectory. For ease of presentation, we show a two-user case. It is straightforward to apply the same technique to the multiple-user case to reach the same conclusion.

As shown in Fig. 3, any two-user resource allocation can be represented by a point $X(x_1, x_2)$, where x_i is the resource allocation (normalized by total capacity) for the ith user, $i = 1, 2$. We define the fairness index as $\max(x_1/x_2, x_2/x_1)$. If the fairness index is closer to unity, the resource allocation is more fair. The line $x_1 = x_2$ is the "fairness line." The line $x_1 + x_2 = 1$ is the "efficiency line." The goal of control schemes is to bring the system to the intersection of the fairness line and the efficiency line. When the system is under-utilized, assuming $x_1 \leq x_2$ without loss of generality, AIMD increases the resource allocation of both users by a constant. Fig. 3(a) shows the trajectory to X' parallel to the fairness line. This movement improves fairness, i.e., reduces the fairness index. Then both users use multiplicative decrease, which does not change fairness. Hence, as the system evolves, AIMD brings the resource allocation point toward the fairness line, finally oscillating around the efficiency line.

For SIMD control, we first observe Table I. We can see that the window size of a connection increases in proportion to $1/w_{\max}$ or $1/x_i$ here for $i = 1, 2$. Thus, as shown in Fig. 3(b), the increase trajectory emanates from $X(x_1, x_2)$ with slope x_1/x_2. Indeed, at any point between the two lines emanating from the origin with slopes x_1/x_2 and x_2/x_1, the resource allocation X' is more fair than X as it reduces the value of the fairness index. Therefore, the increase phase of SIMD improves fairness. Since like AIMD, SIMD uses multiplicative decrease, the decrease phase of SIMD does not change fairness. Hence, SIMD converges to fairness and efficiency.

We also analytically compare the convergence time of SIMD, AIMD, and binomial control schemes. We still assume synchronized feedback and use Fig. 4(a) to illustrate the process of convergence. For ease of analysis, we choose the variables to be the actual window sizes (w_1, w_2). The convergence time consists of two parts: T_1, the time it takes the control mechanism to bring an arbitrary initial point (W_1, W_2), where $W_1 \leq W_2$ and

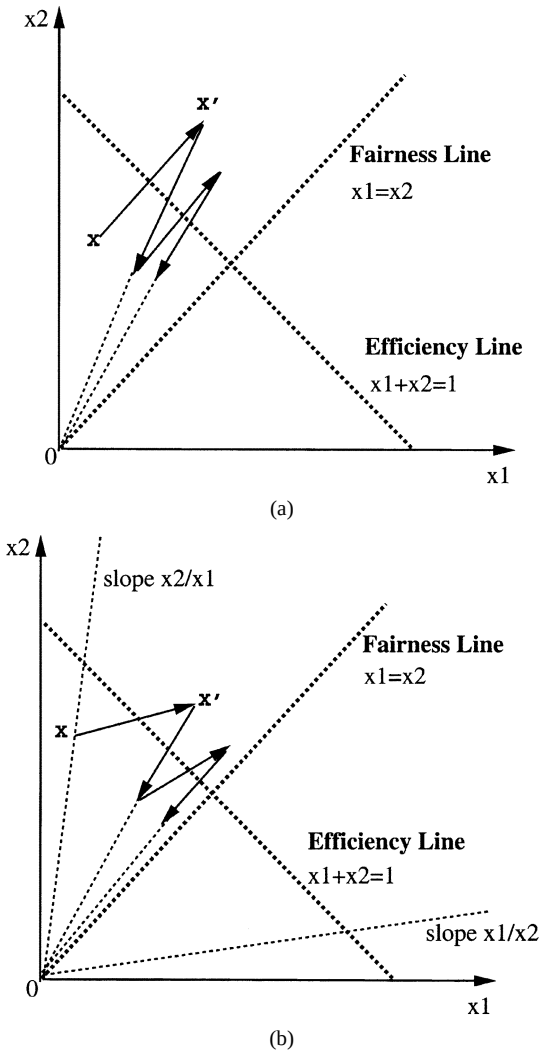


Fig. 3. Convergence of AIMD and SIMD. (a) AIMD trajectory. (b) SIMD trajectory.

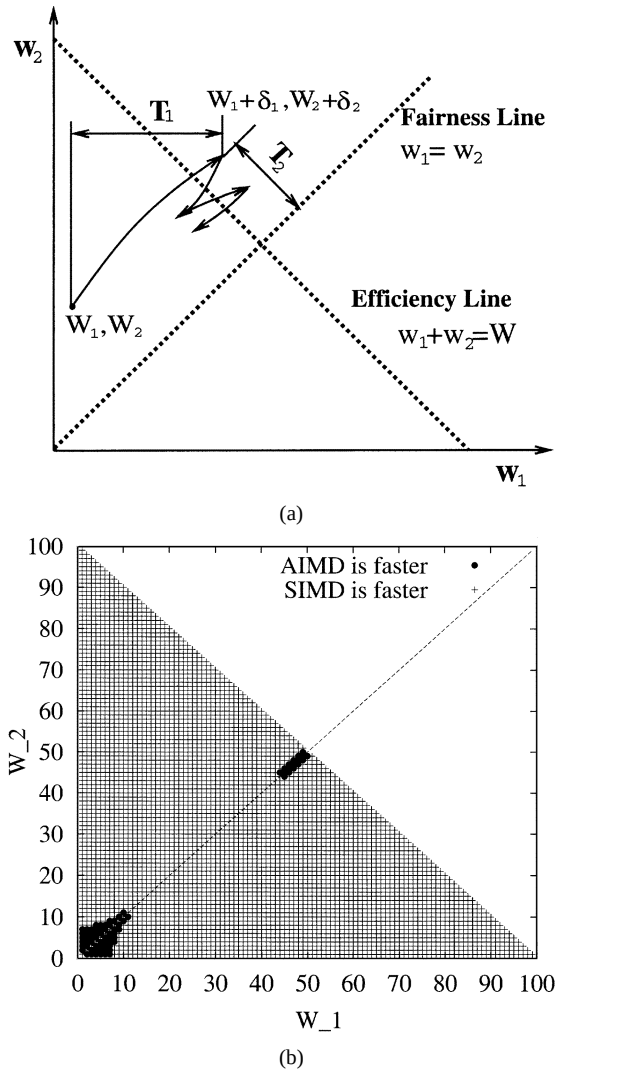


Fig. 4. Comparison of convergence speed. (a) Metrics definition. (b) AIMD versus SIMD.

$W_1 + W_2 < W$, close to the efficiency line $w_1 + w_2 = W$,[7] and T_2, the time until the difference between the two user windows stays within a certain small bound, i.e., $|w_1 - w_2| < \epsilon$. T_1 and T_2 are measured in RTTs. We also denote the difference between the two user windows after T_1 as Δ. Due to space limitation, we only present the main results here in Table III. The detailed analysis can be found in [13].

We numerically solve the above equations for different initial points. Fig. 4(b) shows the regions for which SIMD with $\beta = 1/16$ converges faster/slower (i.e., $T_1 + T_2$ is smaller/larger) than TCP-friendly AIMD with $\beta = 1/16$ for $\epsilon = 1$ and $W = 100$. In most cases, SIMD converges faster than AIMD. Numerical results also show that IIAD with $\alpha = 1$ and $\beta = 2/3$ is much slower than AIMD and SIMD in all cases.

VI. Simulation Results

We use the *ns* simulator [11] to validate that with RED [16] queue management, our proposed controls, most notably SIMD, are TCP-friendly and TCP-compatible. In addition, we compare our controls to standard TCP [17], generalized AIMD [3], and IIAD [4], in terms of smoothness, responsiveness, and aggressiveness. In most simulations, we also include AIAD. In addition, we investigate the way two homogeneous flows converge to their bandwidth fair share and show that our SIMD algorithm outperforms other algorithms. Details about the implementation of SIMD in the *ns* simulator are described in [14].

Unless explicitly specified, in all of the experiments, RED is used as the queue management policy at the bottleneck link. The bottleneck queue configuration and other simulation parameters are listed in Table IV.

The bottleneck queue size and RED queue parameters are tuned as recommended in [18]. The "gentle_" option of the RED queue is turned on as recommended in [19]. We choose $\beta = 1/16$ for SIMD and AIMD (and, thus, $\alpha \approx 1/10$ for AIMD to ensure TCP-friendliness). For IIAD, $\alpha = 1$ and $\beta = 2/3$. For AIAD, $\beta = 2/3$. For ease of presentation, in the rest of this section, we will call these implementations by their family name, e.g., AIMD for $\mathrm{AIMD}(1/10, 1/16)$ when there is no confusion. We use SACK [20] for congestion detection. We also obtained similar results for other mechanisms such as Reno and NewReno. We assume no delayed acknowledgments.

[7]Note that for ease of analysis we assume a small buffer is used at the bottleneck, i.e., packets start to get dropped once the efficiency line is reached. However, adding more buffer space does not qualitatively change the conclusions.

TABLE III
PERFORMANCE MEASURES ON CONVERGENCE TO FAIRNESS AND EFFICIENCY.

Algorithm	T_1 (RTT)	Δ	T_2 (RTT)
TCP	$\frac{W-W_1-W_2}{2}$	$W_2 - W_1$	$\frac{W}{4}\log_{1/2}\frac{\epsilon}{\Delta}$
AIMD	$\frac{(W-W_1-W_2)(2-\beta)}{6\beta}$	$W_2 - W_1$	$\frac{(2-\beta)W}{6}\log_{1-\beta}\frac{\epsilon}{\Delta}$
IIAD	$\frac{1}{12\beta}((\frac{(W_2^2-W_1^2)}{W})^2 - 2(W_1^2+W_2^2) + W^2)$	$\frac{W_2^2-W_1^2}{W}$	$\frac{W}{3}\log_{1-2\beta/W}\frac{\epsilon}{\Delta}$
SIMD	$\frac{2}{3}(1-\frac{2\beta}{3})\sqrt{\frac{2}{\beta(1-\beta)}}\sqrt{\frac{W_1W_2(W-W_1-W_2)}{W_1+W_2}}$	$(2-\frac{W}{W_1+W_2})(W_2-W_1)$	$\frac{\sqrt{2}W}{3}\log_{1-2\beta}\frac{\epsilon}{\Delta}$

TABLE IV
NETWORK CONFIGURATION

Description	*Value*
Packet size	1000 bytes
Maximum window	128 packets
TCP version	SACK
TCP timer granularity	0.1 seconds
RED queue limit Q	2.5 × B/W delay product
DropTail queue limit	1.5 × B/W delay product
RED parameters	min_{th}: 0.15Q, max_{th}: 0.5Q, w_q:0.002
	max_p:0.1, wait_ on, gentle_ on

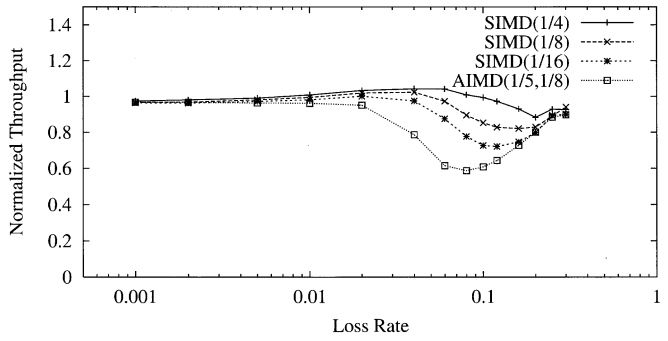


Fig. 5. TCP-friendliness.

A. TCP-Friendliness and Compatibility

1) TCP-Friendliness: We conduct the following experiment to test the TCP-friendliness of our SIMD control. A single flow under investigation is traveling through a single fat link with infinite bandwidth and buffer size. However, the link drops an incoming packet uniformly with probability p. We vary the loss rate p and compare the normalized long-term throughput of SIMD (with respect to standard TCP measured over 3000 RTT) for different β values and plot them in Fig. 5. For comparison, we also plot the throughput of $\mathrm{AIMD}(1/5, 1/8)$.

We notice that all of the curves have a dip when the loss rate is moderate. A close look at the TCP-friendly equation [21], shown at the bottom of the page, can reveal one possible explanation of this abnormality. When loss rate is low, TCP mainly stays in the *congestion avoidance* stage, and AIMD control dominates the equation, while when loss rate is very high, TCP spends most of its time retransmitting packets, and the *exponential backoff* control dominates the equation. Since all controls studied in this paper use the same timeout mechanism as standard TCP, and they carefully calibrate the values of their parameters during congestion avoidance to match standard TCP, they can achieve comparable throughput as standard TCP for very high and low loss rates. However, for the loss regime in between, it becomes hard, if not impossible, to obtain α and β values that would approximate well both congestion avoidance and exponential backoff components of the TCP-friendly equation [3].

Nevertheless, in the worst case with loss rate around 15%, $\mathrm{SIMD}(1/16)$, which is the worst among all SIMD controls considered, can achieve at least 75% throughput as standard TCP, and performs much closer to standard TCP than $\mathrm{AIMD}(1/5, 1/8)$.[8] Given the fact that most parts of the Internet are experiencing less than 5% loss rate [22], our control is TCP-friendly under these conditions.

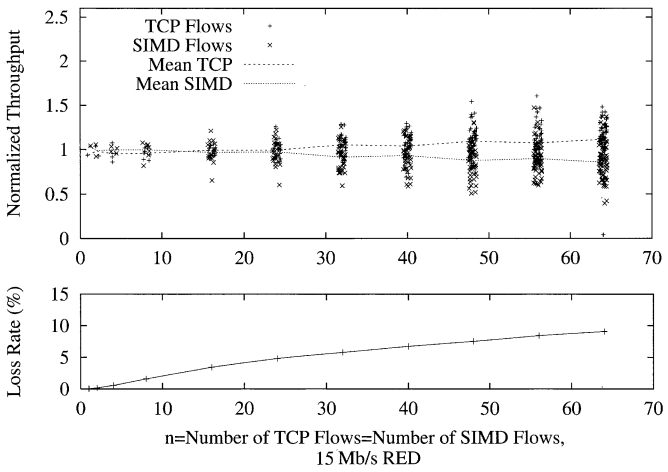
2) TCP-Compatibility: We use the method described in [1] to test TCP-compatibility. n SIMD flows and n standard TCP SACK flows compete for bandwidth over a shared bottleneck link. There are also four background TCP flows transmitting packets in the opposite direction to introduce random ACK delays. We consider both RED and DropTail queues. Fig. 6 shows the simulation results for RED queues without ECN bit set.[9] Results are shown for a bottleneck link bandwidth of 15 and 60 Mb/s. The measured average round-trip delay is around 0.1 s. Each point in the graph represents the throughput of an individual flow in the last 60 s, and the dashed lines represent the average throughput of SIMD and standard TCP flows. In the lower graphs, we also plot the packet loss rate.

As can be observed from the graphs, when the loss rate is low, SIMD achieves very close throughput as standard TCP. When the loss rate exceeds a certain level, SIMD achieves a slightly lower average throughput. This is partly due to the reason we illustrate in Fig. 5. Another possible explanation is that when severe congestion happens, SIMD cannot compete well against standard TCP since compared to TCP, SIMD opens its congestion window more conservatively at the beginning of

[8]The weakness of $\mathrm{AIMD}(\alpha, \beta)$ with small β under intermediate loss conditions is also reported in [1], [3]. The authors try to compensate for the bandwidth loss by increasing the value of α. However, when loss rate is small (e.g., less than 3%), AIMD with large α could achieve significantly higher bandwidth than standard TCP and become less TCP-friendly. Therefore, we maintain the theoretical α values throughout our simulations.

[9]Similar results were obtained for RED queues with ECN bit set [14].

$$\lambda(p,\alpha,\beta) \approx \min\left(\frac{W_{\max}}{R}, \frac{1}{R\sqrt{\frac{2\beta}{\alpha(2-\beta)}p} + T_0 \min\left(1, 3\sqrt{\frac{\beta(2-\beta)}{2\alpha}p}\right) p(1+32p^2)}\right)$$



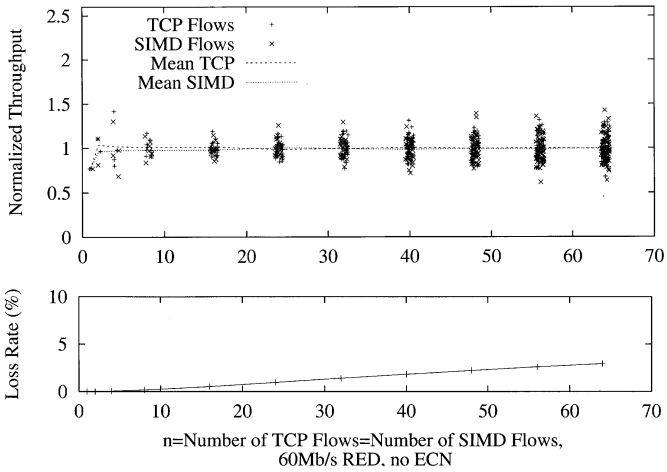
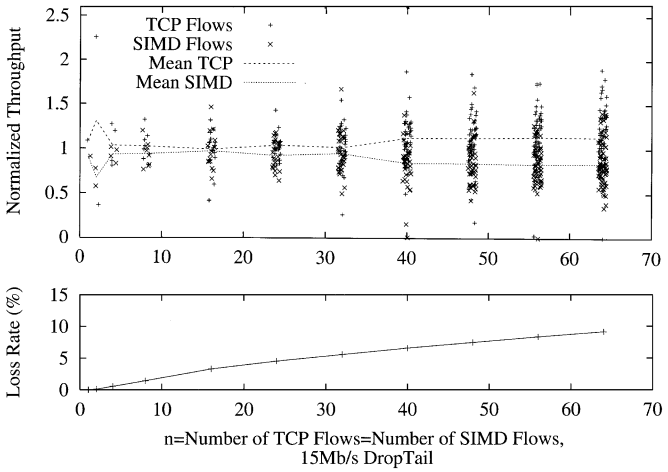


Fig. 6. TCP competing with $\mathrm{SIMD}(1/16)$, RED without ECN.



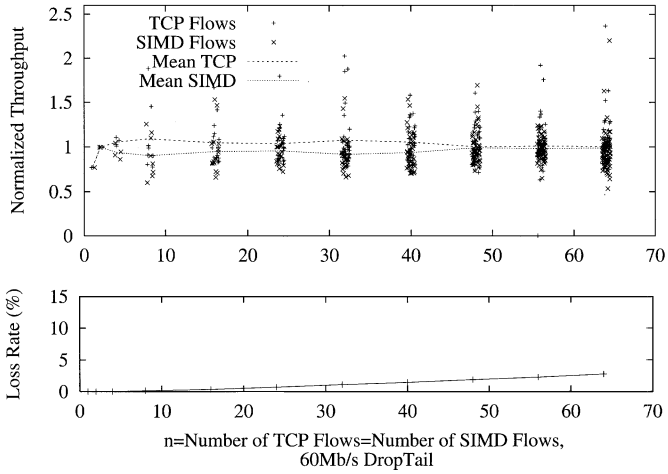


Fig. 7. TCP competing with $\mathrm{SIMD}(1/16)$, with DropTail.

each congestion epoch. Therefore, when the time between two consecutive packet losses is short, the more aggressive TCP tends to gain more throughput. However, in a reasonable loss regime with loss rate below 10%, SIMD shows very impressive TCP-compatibility.[10]

We also found that with DropTail queue management, as shown in Fig. 7, SIMD can still be TCP-friendly and TCP-compatible. The difference, compared to the RED queue experiment, is that the variance becomes larger and SIMD now gets slightly less share of bandwidth. Note that the assumption of randomized packet losses made in our analysis does not apply to DropTail. Under DropTail, packet losses are more correlated. We conjecture that because the RTTs of connections are randomized in the simulation, the chance of having synchronized packet arrivals is small, and the side effect of a DropTail queue (correlated drops for each flow) is thus not so significant.[11]

[10]Note that in case of 60-Mb/s link and less than four flows, the length of the measurement period (60 s) is too short compared to the length of each congestion epoch (more than 40 s), thus, the variance of the results appears to be large.

[11]Similar results were obtained for AIAD competing for bandwidth with TCP [14].

B. Smoothness, Responsiveness, Aggressiveness

1) Smoothness: As revealed by the study in [1], the long-term smoothness of traffic is mainly determined by packet loss patterns and it tends to follow the same distribution at large time-scales (more than 100 RTTs), regardless of which congestion control is used. We thus focus our simulation on short-term smoothness and use the simulation code contributed by [1] to study the traffic generated by the congestion controls under investigation. To this end, we let n such flows compete for a bottleneck link (with capacity C) with another n standard TCP flows. There are also some TCP flows traversing in the opposite direction to introduce random ACK delays. In Fig. 8, we show the case for $n = 16$ and $C = 60$ Mb/s, which corresponds to roughly 0.3% packet drop rate. The bottleneck queue strategy is RED with ECN enabled.[12] Each graph shows one flow's throughput on the congested link during the time interval between 250–270 s of a 500-s simulation. The throughput is averaged over 0.2-s intervals, which correspond to twice a typical RTT for this simulation. As in [5], we also plot the time at which a packet is marked at the bottom of each curve.

We can observe from the graphs that all four controls, AIMD, IIAD, SIMD, and AIAD, have roughly the same scale

[12]Similar results were obtained with ECN turned off [14].

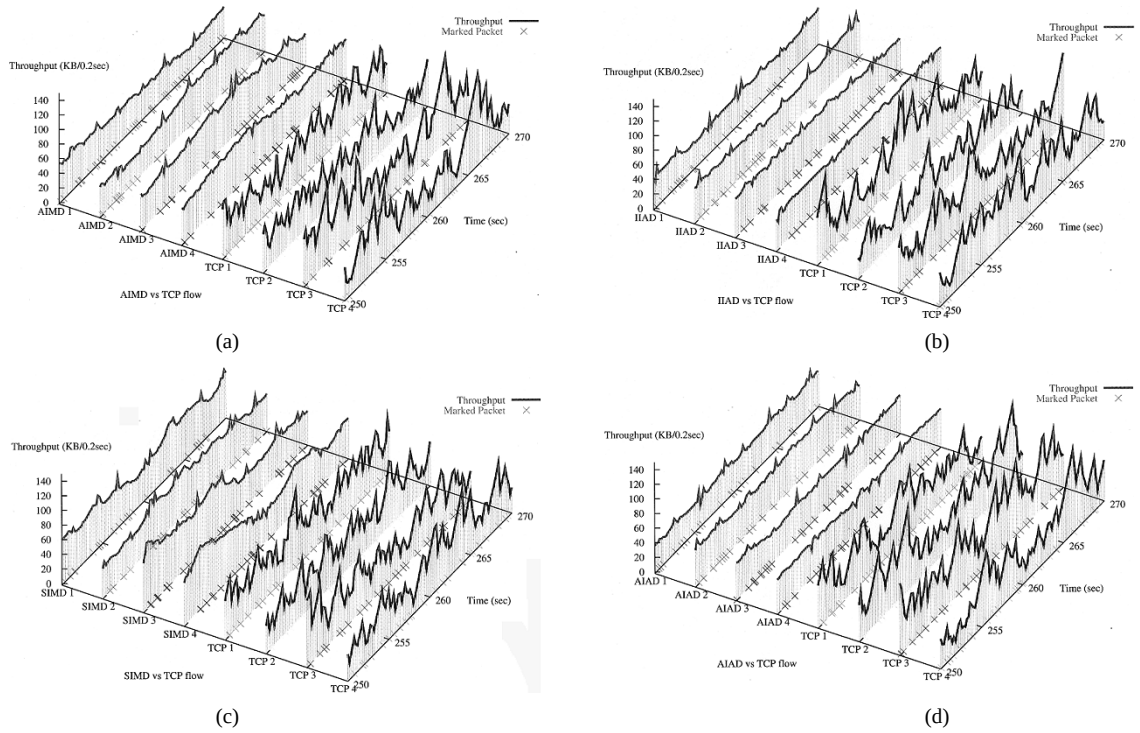


Fig. 8. Traffic smoothness, $16 + 16$ flows, 60 Mb/s link, RED with ECN. (a) $\mathrm{AIMD}(1/10, 1/16)$ with TCP. (b) IIAD with TCP. (c) $\mathrm{SIMD}(1/16)$ with TCP. (d) $\mathrm{AIAD}(2/3)$ with TCP.

of short-term burstiness, with SIMD having a little larger variation. This agrees with our analysis (cf. Section IV). In particular, by plugging in equations of Table II the values we choose in our simulation of $\beta = 1/16$ for AIMD and SIMD, and $\beta = 2/3$ for IIAD and AIAD, and since the average window size in this simulation is about 23 packets, or $W \approx 23$, we find that the order of the coefficients of variation of these controls (from low to high) is IIAD (and AIAD), AIMD, and SIMD. Our experiment results show that this is indeed the case.

Results for a decreased C of 15 Mb/s (thus, increased congestion level to nearly 5% loss rate) can be found in [14]. We observe that the smoothness of all four controls becomes worse when the network becomes more congested. This is again due to the self-clocking mechanism of window-based congestion control. With a smaller average congestion window, the chance that a retransmission timeout happens becomes higher, so does the chance that the congestion window reduces to one. We thus can observe abrupt reductions of the sending rate more frequently. Although, in general, AIMD, IIAD, AIAD, and SIMD still exhibit smoother transmission than TCP, it appears not easy for window-based schemes to achieve high smoothness.[13] This is probably a common weakness of window-based schemes. On the contrary, equation-based schemes [5] can achieve high smoothness even when the loss rate is high.

We also observe that the throughput of AIMD degrades significantly. IIAD and AIAD also get less than their fair share. This is in part due to the reason mentioned in Section VI-A1, that is, AIMD becomes less competitive than standard TCP in this loss regime. The other reason, we conjecture, is that AIMD control does not give any preference to the sender with smaller congestion window (cf. Section VI-C). Thus, when no loss happens, TCP increases its congestion window more aggressively and gets higher throughput than AIMD, which eventually gives up the fair share it deserves. SIMD overcomes this problem and can achieve throughput close to TCP in this scenario.

2) Impulse Response: To better illustrate the aggressiveness and responsiveness properties of different controls, we now study the behavior of different controls responding to impulse disturbance from a periodical ON/OFF constant-bit-rate (CBR) flow.[14] The model is similar to the square-wave model used in the simulation study of [23]. In the experiment, we let the CBR flow alternate between ON and OFF state, each of which lasts for t_{on} and t_{off}, respectively. The sending rate of the CBR flow during the active period is set to γ times C, the capacity of the bottleneck link. We intend to see the effect of such bandwidth oscillation on the transmission of a long TCP flow using the control under study. The results reported here are for $C = 1.5$ Mb/s, average end-to-end RTT (including queueing delay) $= 100$ ms, $t_{\mathrm{on}} = 30$ s, $t_{\mathrm{off}} = 30$ s, and $\gamma = 0.5$. Both flows start around time 0 with some random disturbance. Fig. 9(a)–(c) plots the congestion window value of different controls over time period [480:600].

We also prolong our simulation to repeat this impulse disturbance pattern and measure the average aggressiveness and responsiveness according to our definitions in Section IV and report these data in Table V. We choose the steady-state error to be one packet within the target window size, and the simulation results are shown in the form of 95% confidence intervals.

As expected, standard TCP is highly variable, IIAD and AIMD are the smoothest since the average window size is

[13]The use of the Limited Transmit algorithm can avoid some of the retransmit timeouts to get slightly smoother rate.

[14]To make the graphs more readable, we use error detection mechanisms of TCP NewReno, instead of SACK, so that different controls detect and react to loss at about the same time, in response to duplicate acknowledgments. Using TCP SACK does not qualitatively change the conclusion.

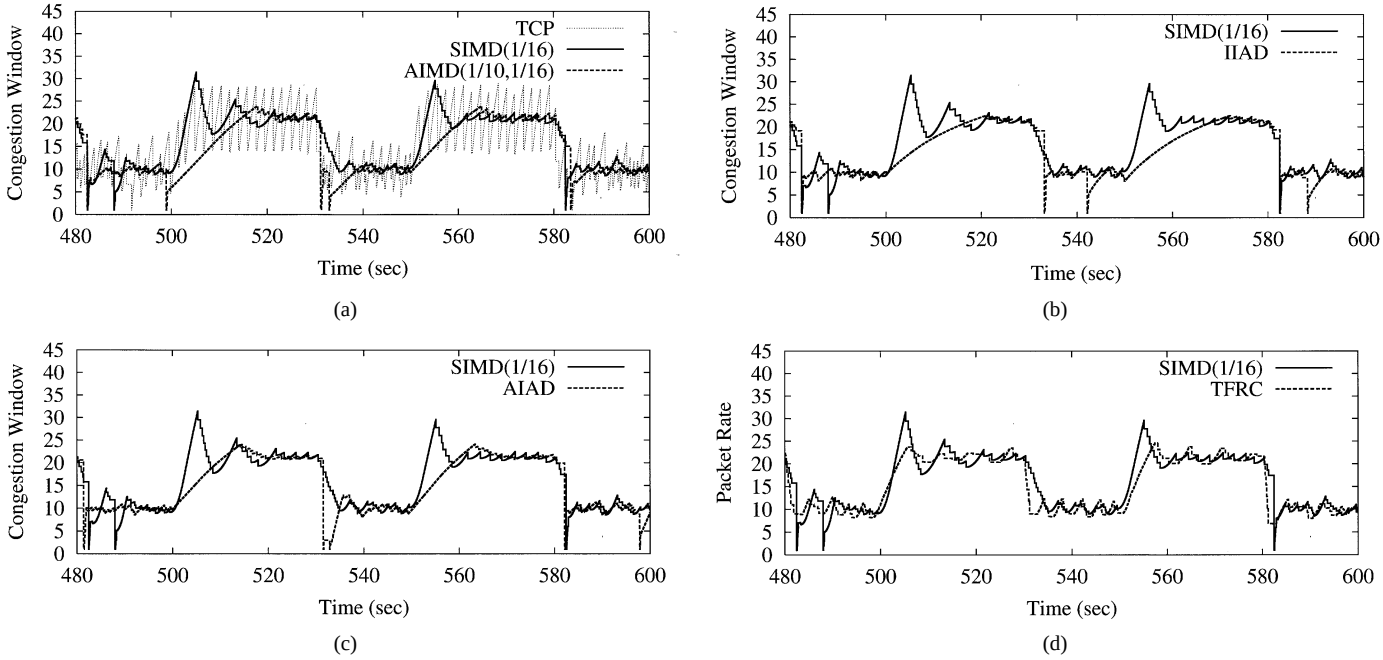


Fig. 9. Impulse response to square-wave CBR flow.

TABLE V
QUANTITATIVE MEASURES

Algorithm	*1/Aggressiveness (RTT)*		*1/Responsiveness (losses)*	
	simulation	analysis	simulation	analysis
TCP	(12.8,14.1)	14.7	(1.54,1.63)	1
AIMD	(108.1,110.7)	117.6	(3.85,5.19)	10.7
IIAD	(172.8,176.0)	181.5	(4.20,5.82)	16.5
SIMD	(31.6,34.6)	41.5	(4.21,5.47)	10.7
AIAD	(103.3,107.9)	121.0	(3.73,4.81)	16.5

larger than 10, at the expense of slow response to bandwidth increases. With similar smoothness, SIMD is much more aggressive than AIMD, IIAD, and AIAD. In addition, AIAD is more aggressive than IIAD. Notice the close match between the simulated measure of aggressiveness and the analytical results.

Aggressiveness of a congestion control is directly related to how much bandwidth a flow can get when it is competing with other flows. It has been shown in [23] that the set of slowly responsive congestion controls proposed so far all tend to receive significantly less bandwidth than competing standard TCP flows in a highly dynamic network environment. However, since SIMD maintains good aggressiveness property, the loss of bandwidth is relatively minor (cf. Fig. 6).

Notice that the responsiveness of a control is hard to measure due to the extreme way TCP responds to a burst of losses, which will occur when it sees a sudden decrease of bandwidth. In this case, all TCP flows reduce their congestion window to one regardless of which congestion avoidance strategy is used. However, we still show the measured responsiveness in Table V to provide a qualitative comparison. Generally speaking, the smooth transmission of a slower responsive flow comes at the cost of more packet losses when available bandwidth is suddenly decreased.

For completeness, we compare the impulse response of SIMD with the equation-based TFRC scheme [5], which also uses history information but is rate-based and requires modification at both sender and receiver sides. Fig. 9(d) shows the result of SIMD versus TFRC with default settings. It is evident that $\mathrm{SIMD}(1/16)$ and the default TFRC have similar smoothness at steady state, and SIMD is more aggressive in probing bandwidth but less responsive to bandwidth decrease.

C. Convergence to Fairness and Efficiency

In this section, we assume a homogeneous protocol environment, i.e., all flows use the same algorithm for congestion control. We then vary the network configuration to study the convergence time of different algorithms.

We use the topology shown in Fig. 10 to perform this experiment. In the beginning of the simulation, there are $c_1 + 1$ connections sharing link $(b1, b2)$, two connections sharing link $(b2, b3)$, and $c_2 + 1$ connections between $b3$ and $b4$. Link bandwidths and delays are shown in the figure. At time 400, all background flows terminate and only two flows ($s1$-$r1$) and ($s2$-$r2$) stay to compete for the bottleneck link $(b2, b3)$. We use packet size of 500 bytes in these experiments.

1) Convergence to Fairness $(W_1+W_2=W, W_1<W_2)$*:* We create this scenario to study the convergence time to fairness given that the initial point (W_1, W_2) is on the efficiency line $(w_1 + w_2 = W)$. To create this setup, we let $c_1 = 15$, $c_2 = 0$, $x = 6$ Mb/s, and $y = 6$ Mb/s. So the bottleneck link for flow $(s2, r2)$ remains link $(b2, b3)$, but for flow $(s1, r1)$, the bottleneck changes from link $(b3, b4)$ to $(b2, b3)$ at time 400. We can also compute that $W \approx 110$, $W_1 \approx 7$, and $W_2 \approx 100$. Fig. 11 plots the transient behavior of the congestion window of different protocols.

We observe that standard TCP has the highest convergence speed, and IIAD generates the smoothest but least responsive traffic. It is worth noticing that in this scenario, where significant bandwidth change happens, our proposed algorithm converges much faster than AIMD to the fair share of the bandwidth.

Table VI gives the convergence time to fairness (T_2). Here we use $\epsilon = 10$ packets (cf. Section V). The theoretical value is also

TABLE VI
QUANTITATIVE MEASURES ON CONVERGENCE TIME

Algorithm	Experiment 1				Experiment 2					
	W_1	W_2	T_2 (RTT)		W_1	W_2	T_1 (RTT)		Δ (pkts)	
			simu	anal			simu	anal	simu	anal
TCP	6.1	99.6	68.0	88.7	8.8	13.8	55	43.7	5.8	6.0
AIMD	7.9	99.2	776	1217	12.7	31.0	349	342	18.6	18.3
IIAD	7.7	99.8	4232	6684	11.8	31.2	1284	1242	8.1	7.6
SIMD	6.6	96.3	**218**	**852**	10.2	33.2	90	85.1	13.6	12.3

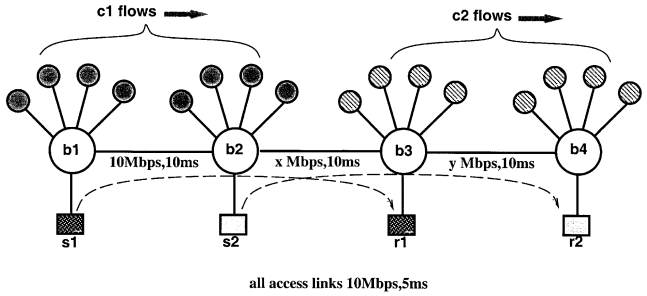


Fig. 10. Simulation topology for convergence test.

given in the table for comparison. The following observations can be made from the table.

First, the simulation results agree with the theoretical analysis in the ranking of various protocols except that all measured convergence times are smaller than the corresponding theoretical values. This is expected since our analysis is based on synchronized feedback assumption, and routers that do not differentiate among flows when dropping packets. In contrast, in the simulation, we use RED, so flows with larger window sizes would see more packet drops. In other words, RED helps the convergence speed to fairness.

Second, SIMD benefits from RED much more than other schemes. The T_2 value from simulations is much smaller than the value obtained from analysis (shown in boldface). This is because RED allows SIMD flows with smaller windows to experience fewer packet losses, which gives them a better chance to become more aggressive.[15] On the contrary, AIMD does not fully capitalize on the random loss property of RED since its window increase rate does not change. As a result, SIMD converges to fairness much faster.

2) Convergence to Efficiency $(W_1 < W_2 < W/2)$*:* To create such scenario, we let $c_1 = 11, c_2 = 3, x = 6$ Mb/s, and $y = 10$ Mb/s. So initially the bottleneck link for flow $(s1, r1)$ is $(b1, b2)$, and for flow $(s2, r2)$ the bottleneck is $(b3, b4)$. But at time 400, both of them switch to link $(b2, b3)$. Roughly, we have $W \approx 110$, $W_1 \approx 10$, and $W_2 \approx 30$. We can then study T_1, the convergence time to efficiency of different control schemes. Fig. 12 plots the transient behavior of different protocols.

The advantage of our SIMD algorithm is more pronounced in this scenario. TCP is still the fastest responding protocol, but still at the expense of high variability in steady state. In addition, general AIMD suffers from the problem of convergence efficiency, i.e., all flows have the same window increments, so before packet loss happens, they increase their congestion windows at the same rate and, thus, do not efficiently converge to the fair share. On the contrary, our SIMD algorithm allows the two competing flows to quickly transit to the fair steady state, since the flow with smaller window grows more aggressive than the one with larger window. IIAD takes a much longer time to converge due to its inherent weak aggressiveness (sublinear increase).

We also give convergence time to efficiency T_1 in Table VI. Analytical results closely match the simulation results.

[15]Recall that the congestion window size of a SIMD connection increases in proportion to $1/w_{\max}$.

VII. RELATED WORK

The earliest congestion controls known are Jacobson's TCP Tahoe [17] and Ramakrishnan and Jain's DECbit scheme [24]. To provide smoother transmission rate than that given by TCP, several TCP-like window-based congestion control mechanisms have been proposed, including the general AIMD [1], [3] and TEAR [2]. These mechanisms use a moderate window decrease parameter to reduce rate variability, meanwhile using a matching window increase parameter to satisfy TCP-friendliness.

Nonlinear control was initially considered not robust and not suitable for practical purposes [15]. On the contrary, Bansal and Balakrishnan [4] proposed binomial controls that interact well with TCP. Binomial controls are memoryless in that they use only the current window size in their control rules. Our controls are fundamentally different from memoryless binomial controls. To our knowledge, not much work has focused on using history information in control rules (an exception is [12] which uses history to adapt its backoff strategy). We proposed and evaluated the first set of window-based TCP-friendly congestion controls that use history information to improve transient behavior without sacrificing smoothness in steady state.

Another approach to provide smoother transmission rate is equation-based congestion controls [5]–[7], first proposed in [25]. In these schemes, the end-systems measure the packet loss rate and RTT, and use the TCP-friendly equation [21] to compute the transmission rate. Two comparisons [1], [9] of equation-based and window-based congestion controls have shown that equation-based schemes and window-based AIMD share similar transient behavior but equation-based schemes provide higher smoothness. However, the aggressiveness of equation-based schemes is limited by the nature of rate-based control, which lacks a self-clocking mechanism for overload protection as in window-based control. In [23], Bansal *et al.* add a parameter to control the degree of self-clocking in the equation-based control to enhance its safety in deployment. They also compared such enhanced control with other slowly responsive but smooth congestion control schemes such as binomial controls. Their simulation results show that all schemes become less competitive to standard TCP in a highly dynamic environment. They also have the problem of converging slowly to fairness in case

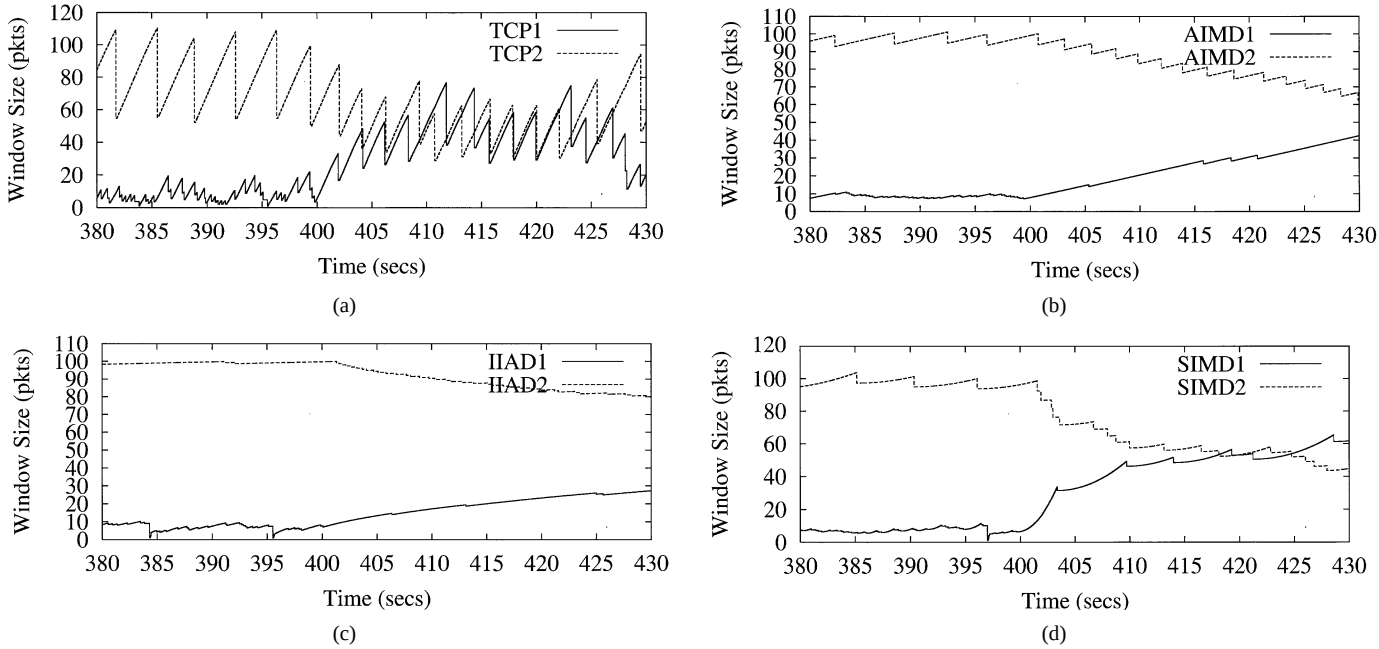


Fig. 11. Two flows converge to fair share of bandwidth. (a) TCP. (b) $\mathrm{AIMD}(1/10, 1/16)$. (c) IIAD. (d) $\mathrm{SIMD}(1/16)$.

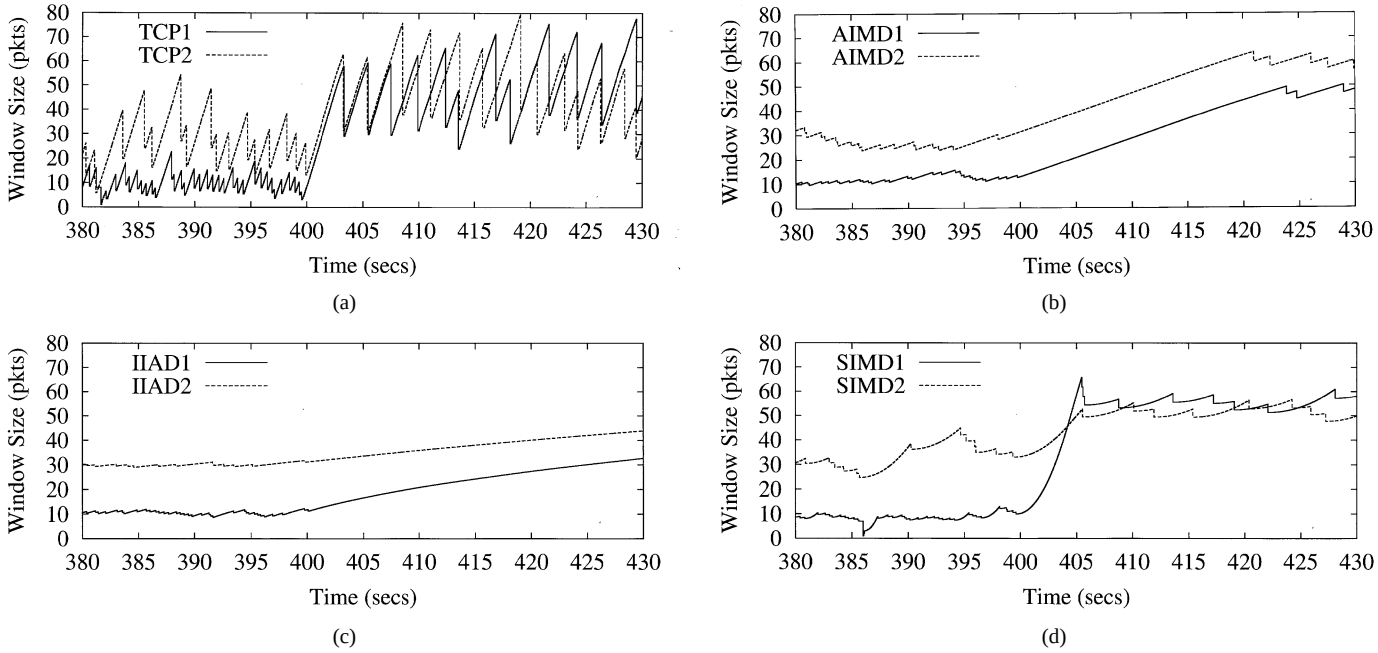


Fig. 12. Two flows converge to fair share of bandwidth. (a) TCP. (b) $\mathrm{AIMD}(1/10, 1/16)$. (c) IIAD. (d) $\mathrm{SIMD}(1/16)$.

of sudden increase/decrease of available bandwidth. Notably, equation-based schemes use more history information up to eight congestion epochs [5]. Therefore, our work is a step toward enhancing transient measures like aggressiveness by exploring the design space between window-based memoryless control schemes and equation-based schemes that make use of longer history.

Much of the literature has focused on the modeling of TCP congestion control [21], [26]–[31]. Ott *et al.* showed that if packet losses are independent with small probability p, the average window size and long-term throughput are of the order of $1/\sqrt{p}$. Lakshman *et al.* [28] studied the properties of TCP in a regime where the bandwidth-delay product is high and losses are random. In [29], Mathis *et al.* studied the relationship between TCP throughput and packet loss rate when TCP is in congestion avoidance mode and came up with the well-known TCP-friendly equation. Padhye *et al.* [21] extended this method and used a stochastic model that also captures the effect of TCP's timeout mechanism on throughput. Altman *et al.* [26] analyze TCP throughput under a more general loss process which is assumed to be stationary. The model thus can account for any correlation and inter-loss time distributions. Recently, Low *et al.* [31] presented a duality model of TCP Vegas congestion control mechanism [32].

VIII. Conclusion

We proposed a spectrum of TCP-like window-based congestion controls. Unlike memoryless controls such as AIMD and binomial controls, our controls utilize history information. They are TCP-friendly and TCP-compatible under RED queue management. They possess different smoothness, aggressiveness, and responsiveness tradeoffs. Thus, instances from our spectrum can be chosen as the transport schemes of various applications, for example, streaming applications on the Internet which are required to be TCP-friendly and need smoothness of transmission rates. We conducted extensive simulations using the *ns* simulator. In particular, we presented simulation results of SIMD, AIMD, and AIAD as special instances. Analysis and simulation were used to demonstrate the TCP-friendliness and TCP-compatibility of our controls, the possible tradeoffs among smoothness, aggressiveness, and responsiveness, as well as better convergence behavior of our SIMD instance. The code for our *ns* implementations and the simulation scripts used for this paper are available on line [33].

To summarize, most encouragingly, in a new design space where control rules use history information, window-based congestion control mechanisms can be TCP-friendly, and still provide smoothness as well as better transient behavior. They can solve the problem raised by slowly responsive congestion controls. Given that equation-based congestion control schemes use longer history, we believe comparisons between equation-based schemes and our scheme remain an interesting future work.

Appendix A
TCP-Friendliness of Our Control

This Appendix explains our choice of α in (4), or equivalently, the choice of c in (5) to make our control scheme TCP-friendly. We assume packet losses occur randomly with a fixed probability p, and the window size variation is small. We do not consider the effect of TCP's timeout mechanisms.

We first derive the value of c under the periodic loss model. Then using approximation, we derive it under the random loss model. We show that the two values from the two models differ only by a small constant.

Consider many congestion epochs where the window increases and decreases alternately in steady state, as shown in Fig. 13. Let W_i be the window size in the beginning of the ith epoch. In this epoch, the window size is decreased to $W_i - \beta W_i^l$, then increased by, say, I_i packets, to W_{i+1} before the first packet loss happens. Assume X_i packets are sent successfully in this epoch.

1) Periodic Losses: Under a periodic loss model, the window size increase and decrease are deterministic. Both W_i and X_i are constants, denoted as W and X, respectively. I_i is a constant equal to βW^l.

Given the window increase function (1) in Section II, we can compute the duration (in RTTs) of each congestion epoch as

$$T = \left(\frac{\beta W^l}{c}\right)^{1/u}$$

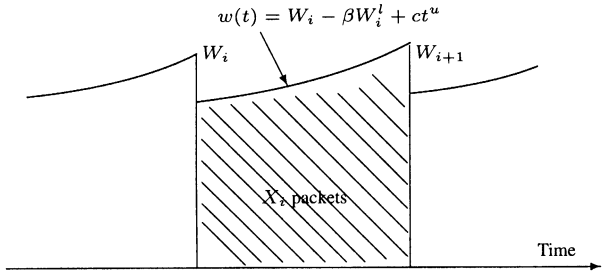


Fig. 13. Window increases with time, and decreases on packet losses.

and the number of packets in each epoch is given by

$$\begin{aligned} X &= \int_0^T (W - \beta W^l + ct^u)\, dt \\ &= (W - \beta W^l)T + \frac{c}{u+1}T^{u+1}. \end{aligned}$$

For the congestion control to be TCP-friendly, the throughput and loss rate relationship must hold. Without considering the effect of TCP's timeout mechanisms, the relationship is $\lambda = \sqrt{3/2}/(R\sqrt{p})$, where λ is the average throughput and R is the RTT. We have $\lambda = X/(TR)$, i.e., average throughput is the number of packets between two consecutive losses divided by the time (in seconds) between the two losses. We also have $p = 1/X$. Plugging them into the (λ, p) relationship, we get

$$c = \left(\frac{3}{2\left(1 - \frac{1}{k+2}\beta W^{l-1}\right)}\right)^{\frac{1}{k+1}} \beta W^{l-\frac{1}{k+1}}. \qquad (9)$$

Notice that here $w_{\max}$ is equal to W, by definition. Therefore, under the periodic loss model, this definition satisfies TCP-friendliness.

2) Random Losses: Now we consider a random loss model where the losses are Bernoulli trails: packets are dropped uniformly with a fixed probability p. Consider the random process $\{X_i\}$ where X_i is the number of packets sent in the ith epoch up to but not including the first packet lost. Given the random loss model, the probability that j packets are acknowledged successfully before the first loss is

$$\begin{aligned} P[X_i = j] &= (1-p)^j p, \quad j = 0, 1, 2, \ldots \\ &\approx pe^{-pj}, \quad p \ll 1. \end{aligned}$$

Let T_i denote the number of rounds between two consecutive loss events. T_i can be computed by X_i divided by the average window size in the ith epoch $\overline{w_i}$, i.e., $T_i = X_i/\overline{w_i}$. Using (1), this results in a window increase of size

$$I_i \approx c\left(\frac{X_i}{\overline{w_i}}\right)^u.$$

Computing $E[I_i]$ is difficult since X_i and $\overline{w_i}$ are correlated. However, when the window size variation is small enough, we ignore such correlation and use the time-average window size $\bar{w}$ to approximate $\overline{w_i}$. Therefore

$$I_i \approx c\left(\frac{X_i}{\bar{w}}\right)^u.$$

Then the expected window increase is

$$\begin{aligned} E[I_i] &= \sum_{j=0}^{\infty} I_i P[X_i = j] \\ &\approx \sum_{j=0}^{\infty} c\left(\frac{j}{\bar{w}}\right)^u (1-p)^j p \\ &\approx \int_0^{\infty} c\left(\frac{x}{\bar{w}}\right)^u p e^{-px}\, dx \\ &= \frac{c\Gamma(u+1)}{(p\bar{w})^u}. \end{aligned} \tag{10}$$

Note that, under the periodic loss model, $X_i = 1/p$, and $T_i = X_i/\bar{w} = 1/(p\bar{w})$. Therefore

$$E[I_i] = \frac{c}{(p\bar{w})^u}. \tag{11}$$

For TCP-friendliness, we need to equalize the expected window increases $E[I_i]$ under both loss models. In steady state, the expected increase of the window size is equal to the expected decrease of the window size. Under both loss models, the expected decreases of the window size are roughly equal, given the same loss rate and roughly the same average window size. Therefore, we need only to equalize the expected increases under both loss models. Noticing the only difference between (10) and (11) is a factor of $\Gamma(u+1)$, we only adjust the definition in (9). Thus, we get (5), and equivalently, (4).

Considering that the random loss model is obviously more realistic, we use the definition in (4) and (5) in this paper. In Section VI, we use simulations to validate the TCP-friendliness of SIMD under a wide range of loss rate.

REFERENCES

[1] S. Floyd, M. Handley, and J. Padhye. (2000, May) A comparison of equation-based and AIMD congestion control. [Online]. Available: http://www.aciri.org/floyd/papers.html

[2] I. Rhee, V. Ozdemir, and Y. Yi, "TEAR: TCP emulation at receivers—Flow control for multimedia streaming," Dept. Comput. Sci., North Carolina State Univ., Raleigh, NC, Tech. Rep., Apr. 2000.

[3] Y. R. Yang and S. S. Lam, "General AIMD congestion control," in *Proc. Int. Conf. Network Protocols (ICNP)*, Nov. 2000, pp. 187–198.

[4] D. Bansal and H. Balakrishnan, "Binomial congestion control algorithms," in *Proc. IEEE INFOCOM*, Apr. 2001, pp. 631–640.

[5] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-based congestion control for unicast applications," in *Proc. ACM SIGCOMM*, Aug. 2000, pp. 43–56.

[6] J. Padhye, J. Kurose, D. Towsley, and R. Koodli, "A model-based TCP-friendly rate control protocol," in *Proc. NOSSDAV*, June 1999, pp. 137–151.

[7] W.-T. Tan and A. Zakhor, "Real-time internet video using error resilient scalable compression and TCP-friendly transport protocol," *IEEE Trans. Multimedia*, vol. 1, pp. 172–186, June 1999.

[8] S. Floyd and K. Fall, "Promoting the use of end-to-end congestion control in the internet," *IEEE/ACM Trans. Networking*, vol. 7, pp. 458–472, Aug. 1999.

[9] Y. R. Yang, M. S. Kim, and S. S. Lam, "Transient behavior of TCP-friendly congestion control protocols," in *Proc. IEEE INFOCOM*, Apr. 2001, pp. 1716–1725.

[10] S. Jin, L. Guo, I. Matta, and A. Bestavros, "TCP-friendly SIMD congestion control and its convergence behavior," in *Proc. Int. Conf. Network Protocols (ICNP)*, Nov. 2001, pp. 156–164.

[11] E. Amir *et al.*. UCB/LBNL/VINT Network Simulator—ns (v.2). Information Sciences Inst., Los Angeles, CA. [Online]. Available: http://www.isi.edu/nsnam/ns/

[12] R. Puri, K.-W. Lee, K. Ramchandran, and V. Bharghavan, "An integrated source transcoding and congestion control paradigm for video streaming in the internet," *IEEE Trans. Multimedia*, vol. 3, pp. 18–32, Mar. 2001.

[13] S. Jin, L. Guo, I. Matta, and A. Bestavros. (2001, July) A spectrum of TCP-friendly window-based congestion control algorithms. Tech. Rep. BU-CS-2001-015. Dept. Comput. Sci., Boston Univ., Boston, MA. [Online]. Available: http://www.cs.bu.edu/techreports/

[14] ——, (2002, Oct.) Addendum to: A spectrum of TCP-friendly window-based congestion control algorithms. Tech. Rep. BU-CS-2002-027. Dept. Comput. Sci., Boston Univ., Boston, MA. [Online]. Available: http://www.cs.bu.edu/techreports/

[15] D.-M. Chiu and R. Jain, "Analysis of the increase and decrease algorithms for congestion avoidance in computer networks," *Comput. Networks ISDN Syst.*, vol. 17, pp. 1–14, 1989.

[16] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Networking*, vol. 1, pp. 393–417, Aug. 1993.

[17] V. Jacobson, "Congestion avoidance and control," in *Proc. ACM SIGCOMM*, Aug. 1988, pp. 314–329.

[18] M. Christiansen, K. Jeffay, D. Ott, and F. D. Smith, "Tuning RED for web traffic," in *Proc. ACM SIGCOMM*, 2000, pp. 249–264.

[19] S. Floyd. (2000, Mar.) Recommendation on using the "Gentle_" variant of RED. [Online]. Available: http://www.aciri.org/floyd/red/gentle.html

[20] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow, "TCP selective acknowledgment options," Network Working Group, RFC 2018, Apr. 1996.

[21] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP throughput: A simple model and its empirical validation," in *Proc. ACM SIGCOMM*, 1998, pp. 303–314.

[22] CAIDA Website. Cooperative Assoc. Internet Data Analysis. [Online]. Available: http://www.caida.org

[23] D. Bansal, H. Balakrishnan, S. Floyd, and S. Shenker, "Dynamic behavior of slowly responsive congestion control algorithms," in *Proc. ACM SIGCOMM*, Aug. 2001, pp. 263–274.

[24] K. Ramakrishnan and R. Jain, "Congestion avoidance in computer networks with a connectionless network layer—Part IV: A selective binary feedback scheme for general topologies," Digital Equipment Corp., Tech. Rep. TR510, Aug. 1987.

[25] J. Mahdavi and S. Floyd. (1997, Jan.) TCP-friendly unicast rate-based flow control. Note sent to end2end-interest mailing list. [Online]. Available: http://www.psc.edu/networking/papers/tcp_friendly.html

[26] E. Altman, K. Avrachenkov, and C. Barakat, "A stochastic model of TCP/IP with stationary random losses," in *Proc. ACM SIGCOMM*, Aug. 2000, pp. 231–242.

[27] S. Floyd, "Connections with multiple congested gateways in packet-switched networks—Part 1: One-way traffic," *Comput. Commun. Rev.*, vol. 21, no. 5, pp. 30–47, Aug. 1991.

[28] T. V. Lakshman and U. Madhow, "The performance of TCP/IP for networks with high bandwidth-delay products and random loss," *IEEE/ACM Trans. Networking*, vol. 5, pp. 336–350, June 1997.

[29] M. Mathis, J. Semske, J. Mahdavi, and T. Ott, "The macroscopic behavior of the TCP congestion avoidance algorithms," *Comput. Commun. Rev.*, vol. 27, no. 3, pp. 67–82, July 1997.

[30] T. J. Ott, J. H. B. Kemperman, and M. Mathis. (1996) The stationary behavior of ideal TCP congestion avoidance. [Online]. Available: http://www.argreenhouse.com/papers/tjo

[31] S. H. Low, L. Peterson, and L. Wang, "Understanding TCP Vegas: A duality model," in *Proc. ACM SIGMETRICS*, June 2001, pp. 226–235.

[32] L. S. Brakmo and L. L. Peterson, "TCP Vegas: End-to-end congestion avoidance on a global Internet," *IEEE J. Select. Areas Commun.*, vol. 13, pp. 1465–1480, Oct. 1995.

[33] S. Jin, L. Guo, I. Matta, and A. Bestavros. (2001, Nov.) Simulations for stateful TCP congestion control. [Online]. Available: http://csr.bu.edu/simd/sims.html

Shudong Jin received the B.S. and M.S. degrees in computer science from Huazhong University of Science and Technology, China, in 1991 and 1994, respectively. He is currently working toward the Ph.D. degree in computer science at Boston University, Boston, MA.

His research concerns Web and streaming access workload characterization, caching algorithms, performance evaluation of large-scale content delivery techniques, network protocols, and Internet measurement and modeling.

Liang Guo (S'98) received the B.S. and M.Eng. from Beijing University of Aeronautics and Astronautics, Beijing, China, in 1994 and 1997, respectively. He is currently working toward the Ph.D. degree in computer science at Boston University, Boston, MA.

His research interests include scalable Internet QoS mechanisms, performance evaluation, dynamic system modeling, and high-speed communication and protocols.

Mr. Guo is a Member of the Association for Computing Machinery.

Ibrahim Matta (M'93) received the Ph.D. in computer science from the University of Maryland, College Park, in 1995.

He is currently an Assistant Professor in the Department of Computer Science, Boston University, Boston, MA. His research involves the design and analysis of QoS and wireless architectures and protocols, and Internet topology and traffic analysis. He was a Guest Co-Editor of two special issues on Transport Protocols for Mobile Computing in the *Journal of Wireless Communications and Mobile Computing*, February 2002, and on Quality of Service Routing in the *IEEE Communications Magazine*, June 2002. He was Technical Program Co-Chair of the first Workshop on Wired/Wireless Internet Communications (WWIC 2002), Publication Chair of the IEEE INFOCOM 2003, and Tutorial and Panel Chair of the Hot Interconnects Conference 2001. He was the representative of the IEEE Technical Committee on Computer Communications for the IEEE GLOBECOM 1999.

Dr. Matta received the National Science Foundation CAREER Award in 1997. He is a Member of the Association for Computing Machinery.

Azer Bestavros (M'87) received the M.S. and Ph.D. degrees from Harvard University, Cambridge, MA.

He is currently Associate Professor and Chairman of the Department of Computer Science at Boston University, Boston, MA, where he conducts research in the general areas of networking and of real-time systems.

Dr. Bestavros has received the ACM and the IEEE Excellence Awards for services rendered to the Computer Science community, including his organization and PC chairmanship of a number of IEEE and ACM technical meetings, and his maintenance of the archives of the IEEE Computer Science Technical Committee on Real-Time Systems. He is a Member of the Association for Computing Machinery.