A Theory of Window-based Unicast Congestion Control *

Nishanth R. Sastry† Simon S. Lam‡

Department of Computer Sciences,
The University of Texas at Austin

Abstract

This work presents a comprehensive theoretical framework for window-based congestion control protocols that are designed to converge to fairness and efficiency. We first derive a sufficient condition for convergence to fairness. Using this, we show how fair window increase/decrease policies can be constructed from suitable pairs of monotonically non-decreasing functions. We show that well-studied protocols such as TCP, GAIMD and Binomial congestion control can be constructed using this method. Thus we provide a common framework for the analysis of such window-based protocols. To validate our approach, we present experimental results for a new TCP-friendly protocol, LOG, designed using this framework with the objective of reconciling the smoothness requirement of streaming media-like applications with the need for a fast dynamic response to congestion.

1 Introduction

Van Jacobson's congestion control and avoidance mechanisms for TCP [9] have been instrumental for the success and stability of the Internet. Chiu and Jain [4] proved that, assuming synchronous feedback, any additive-increase multiplicative-decrease (AIMD) mechanism such as TCP converges to fairness and efficiency. Thus these mechanisms allow flows to equitably share a bottleneck link's bandwidth by adjusting their sending rates, while at the same time efficiently utilizing all of the available bandwidth. In abstract terms, TCP can be viewed as a distributed algorithm to fairly and efficiently apportion a common resource *without having to explicitly exchange any information between the users of the resource*. This is arguably the main reason for the success of TCP.

Unfortunately, recent experience indicates that TCP may not be suitable for all applications. For example, TCP's abrupt decrease-by-half response to losses or congestion indications cannot be tolerated by streaming media applications. Consequently, there is a pressing need to design custom congestion control protocols adapted to the needs of different applications. To simplify the task of designing a protocol that not only suits a given situation, but also satisfies the nice property of convergence to fairness and efficiency, we propose a window-based congestion control framework called CYRF (for *C*hoose *Y*our *R*esponse *F*unction). By choosing specific window increase and decrease policies (or response functions), CYRF can be easily adapted to suit different application and network needs.

Our main theoretical result is that given two monotonically non-decreasing functions $f(x)$ and $g(x)$, $f(x) > 0$ and $0 < g(x) \leq 1$ for all $x \geq 1$, a set of flows using the following increase/decrease policy converges to fairness and efficiency:

$$\begin{aligned} \mathcal{I} : x(t+R) &\leftarrow x(t) + x(t)/f(x(t)) \\ \mathcal{D} : x(t+R) &\leftarrow x(t) - x(t)g(x(t)) \end{aligned} \tag{1}$$

Throughout this paper, we use $x(t)$ to represent the current window size and $x(t+R)$, the next window, after a round-trip time R. We also use $\mathcal{I}$ for the increase policy and $\mathcal{D}$ for the decrease policy.

We term these protocols as *step-wise convergent* because each application of the above policy moves the system closer to fairness. We also obtain a more general class of smooth *epoch-wise convergent* protocols called 1-CYRF that allow unfair decrease steps but still converge to fairness over each congestion epoch if we drop the requirement that $g(x)$ be monotonic, and instead only ask that the product $f(x)g(x)$ be monotonically non-decreasing, and always greater than 1 for $x > 1$.

An interesting aspect of this work is that all commonly known window-based protocols, namely TCP, GAIMD and Binomial Congestion Control are special cases of CYRF (some binomial congestion control protocols are only a special case of 1-CYRF), thus providing a powerful unified framework for the analysis of these protocols. We obtain

*This paper is a slightly revised version of one in the hard-copy IEEE ICNP 2002 Proceedings. Research sponsored in part by Texas Advanced Research Program grant no. 003658-0439-2001 and by NSF grant no. ANI-9977267.

†Currently with IBM. E-mail: nishanth_sastry@us.ibm.com

‡E-mail: lam@cs.utexas.edu

new proofs for the fairness of these protocols as special cases of the results for CYRF. As a second application of CYRF, we describe a new classification of the space of window-based protocols.

The main application envisaged for CYRF is the design of new window-based protocols to suit different application and network needs. We briefly describe a new 1-CYRF TCP-friendly protocol called LOG that balances the need of streaming media applications for smooth changes in sending rate with the network requirement of a fast response to congestion indications. We will also experimentally demonstrate the validity of the main theorems for CYRF using LOG.

This is the first of a series of related papers on CYRF. In a future paper, we will investigate two separate approaches that use CYRF congestion control but also allow co-existence with legacy TCP flows that dominate the Internet. In the first method, we will describe simple rules to make smooth CYRF flows TCP-friendly and mathematically derive the LOG protocol as a case in point. The second approach relaxes the TCP-friendly requirement to apply only when the flow is sufficiently large, and obtains a much better dynamic response and compatibility with drop-tail queues.

The rest of this paper is organized as follows. In Section 2, we describe related proposals. Section 3 explains some of the simplifying assumptions used in our analysis and derives sufficient conditions for 2 and $n > 2$ flows to converge to fairness and efficiency. This is used in section 4 as the basis for CYRF. Section 5 experimentally validates the main theorem and presents our simulations of LOG. Section 6 concludes the work.

2 Related work

TCP is the most widely used congestion control mechanism today. Most modern TCP implementations [25] incorporate algorithms introduced by Van Jacobson [1, 9] into 4.3BSD to fix the original 1988 congestion collapse. In congestion avoidance mode, TCP increases its window size by 1 when a window is acknowledged, and decreases the window to half its previous size when a loss is detected. Thus its increase and decrease policies are given by:

$$\begin{aligned} \mathcal{I} : x(t+R) &\leftarrow x(t) + 1 \\ \mathcal{D} : x(t+R) &\leftarrow x(t) - x(t)/2 \end{aligned} \qquad (2)$$

Congestion avoidance was simultaneously investigated in a series of papers by Jain, Ramakrishnan and Chiu which is summarized in [11]. Prominent congestion control protocols in classical networking literature include Clark *et al.* 's NETBLT [5] and Ramakrishnan and Jain's DECBit [18].

The notion of TCP-friendliness [14, 15] has given rise to a number of new proposals [2, 8, 12, 20, 21, 24, 27] for the transport of streaming multimedia.

The closest in approach to CYRF are GAIMD and Binomial congestion control, which are both shown to be special cases of CYRF. GAIMD [11, 27] generalizes TCP to an Additive Increase Multiplicative Decrease (AIMD) policy with different increase and decrease parameters. Its increase and decrease policies are given by

$$\begin{aligned} \mathcal{I} : x(t+R) &\leftarrow x(t) + \alpha\,; & \alpha > 0 \\ \mathcal{D} : x(t+R) &\leftarrow x(t) - \beta x(t)\,; & 0 < \beta < 1 \end{aligned} \qquad (3)$$

with $\alpha = 3\beta/(2-\beta)$ for TCP-friendliness[7].[1] Binomial congestion control [2] proposes the following non-linear increase/decrease policies:

$$\begin{aligned} \mathcal{I} : x(t+R) &\leftarrow x(t) + \alpha/x(t)^k\,; & \alpha > 0 \\ \mathcal{D} : x(t+R) &\leftarrow x(t) - \beta x(t)^l\,; & 0 < \beta < 1 \end{aligned} \qquad (4)$$

with the further condition that $k + l = 1$ to ensure TCP-friendliness. We can use $l > 1$ only if we know that the maximum window size is $x < (1/\beta)^{l-1}$. Otherwise, we will need to separately deal with the possibility of negative window sizes after an application of $\mathcal{D}$. A similar policy is briefly considered in section 4 of the Chiu-Jain paper [4].

Notice that plugging $\alpha = 1$ and $\beta = 1/2$ for GAIMD (Equation 3) gives us TCP. GAIMD itself is a special case of the binomial algorithm(Equation 4) with $k = 0$ and $l = 1$. SQRT ($k = l = 0.5$) and IIAD ($k = 1$, $l = 0$) are two non-linear binomial controls in [2] that we will use in later sections.

TFRC [8] is a rate-based scheme which directly uses the TCP throughput equation [16] to estimate its sending rate. TEAR [21] estimates the TCP-friendly rate by emulating the entire TCP state machine at the receiver. RAP [20] modifies the inter-packet gap to provide fine-grained delay-based congestion avoidance. The Loss-Delay Adjustment Algorithm [24] uses feedback from RTP [23] for rate adjustment.

In this work, we do not look at multicast congestion control. We also do not consider application-specific adaptive approaches such as [19] that try to make the best use of the available network support. We consider all flows equal so that schemes like MulTCP [6] fall outside our framework. Similarly, we allow only binary feedback, either through packet loss or an ECN-like indication. Thus proposals such as Explicit Window Adaptation [13] which requires per-flow network feedback are not considered. Finally, we confine ourselves to the classical memoryless model of congestion control. Thus recent schemes such as SIMD [12] fall outside our scope.

[1] This does not consider the effect of timeouts. [27] gives a slightly different condition for TCP-friendliness with timeouts.

3 Convergence requirements

In this section, we first outline the simplifying assumptions made in the rest of the paper and then formalize the notions of convergence to fairness and efficiency. It is important to note that while these assumptions simplify our proofs, they do not restrict the applicability of the results. Section 5 shows that the theorems apply even in experimental simulations where the ideal-case assumptions do not hold.

3.1 Notation

Following Chiu and Jain [4], in the rest of this work we adopt the following conventions for notation. We use x_i to represent the current window size of the i^{th} flow. Δx_i denotes a change to it due to the application of an increase policy $\mathcal{I}$ or a decrease policy $\mathcal{D}$. In general, the numerical subscript i will be used to denote a quantity on flow i, and the number of flows is represented by n.

3.2 The model

The following analysis uses Chiu and Jain's *synchronous feedback* assumption [4] that all the flows in the network get the same feedback and get this feedback simultaneously. Furthermore, the feedback is *binary* and limited to a single bit indicating whether the network is overloaded(1) or if there is additional available bandwidth (0). This feedback can be implicit, for example, through packet losses, or through an explicit mechanism such as a "congestion experienced" bit in an ECN aware network [17, 18].

If the network feedback is 1, then the next window size is determined by the decrease policy $\mathcal{D}$, otherwise, the increase policy $\mathcal{I}$ is applied. Usually, this adjustment occurs upon receiving an ACK. Here we use the *continuous fluid model* which assumes that this happens as a continuous process.

We also assume a *saturated sender* whose window size is limited only by the network, and not by the receiver's window or the amount of outstanding data at the sender. Finally, we ignore mechanisms such as slow-start by assuming that *steady state* has been reached.

3.3 1-responsiveness

Steady state can be characterized by a sequence of *congestion epochs* which we define as the largest period of time which contains (and ends with) exactly one application of the decrease policy. Most analyses (for example, [2, 7]) implicitly assume that each congestion epoch has at least one application of an increase policy, or equivalently, that each decrease is preceded by at least one increase. With the synchronous feedback assumption, this means that for each flow, the decrease in window size from a single application of $\mathcal{D}$ must at least wipe out the previous increase resulting from the last application of $\mathcal{I}$, so that the next feedback from the network does not indicate overload. In other words, the following criterion must be satisfied:

$$|\Delta(w_{\mathcal{I}})| \leq |\Delta(w_{\mathcal{D}})| \tag{5}$$

where $\Delta(w_{\mathcal{I}})$ is the increase resulting from a single application of $\mathcal{I}$ and $\Delta(w_{\mathcal{D}})$ the decrease in window size because of $\mathcal{D}$. We term protocols which satisfy Equation 5 for *sufficiently large window sizes* as 1-*responsive* to distinguish them from k-*responsive* protocols which require $k > 1$ applications of $\mathcal{D}$ to offset an increase.

Unless otherwise stated, the protocols are assumed to be 1-responsive. As can be seen from Equations 2, 3 and 4, many interesting protocols like TCP, GAIMD and the TCP-friendly version of Binomial Congestion Control are 1-responsive in general [2]. Thus, this is not a very restrictive assumption.

3.4 Convergence to efficency

We require the system to react in such a way as to move the total bottleneck link utilization closer to the link capacity. This can be achieved if the total utilization across all flows (i.e., sum of window sizes) increases when the bottleneck link is underutilized and decreases when the bottleneck link is overloaded. This is just the principle of negative feedback [4]. An easy way to achieve this is to have each flow increase its window size when the bottleneck link is under-utilized and decrease its window size when the bottleneck link is overloaded.

3.5 Convergence to fairness

Fairness is the most important criterion for the feasibility of any end-to-end congestion control protocol. Intuitively, this means that regardless of the initial window size values, all flows sharing a single bottleneck link must eventually end up with identical window sizes at each instant (in steady state).

When the eventual goal of equal window sizes is not satisfied, the flows share the link unfairly. To quantify this, we use the Jain-Chiu-Hawe Fairness index [10] F:

$$\mathrm{F} = \frac{(\sum x_i)^2}{n(\sum x_i^2)} \tag{6}$$

Observe that F is a continuous differentiable function upperbounded by 1, and this upperbound is reached when the allocation is totally fair ($x_1 = x_2 = \ldots = x_n$).

[2]Note that the minimum possible window size is 1 and for this window, Equation 5 fails. But we assume a congestion epoch with a sufficiently large minimum window size.

In this section, we derive a simple sufficient condition that guarantees convergence to fairness. The following numerical example will motivate and provide an intuitive feel for the result:

Example 1: Consider two flows with windows of size $x_1 = 8$ and $x_2 = 10$. If an application of the increase policy must result in a fair window size of 11 for both, the smaller flow must change (increase) by a larger amount: $\Delta x_1 = 3$, as compared to $\Delta x_2 = 1$. Similarly, if a decrease must result in a fair window size of 7, the smaller flow must decrease by a smaller amount, or equivalently, change by a larger amount: $\Delta x_1 = -1$ which is greater than $\Delta x_2 = -3$. ■

Thus the *signed* change in the window size Δx, must be greater for the flow with the smaller window. The theorem below shows that it is sufficient for the *signed proportional* change $\Delta x_1 / x_1$ to be greater.

Theorem 1 (2-Flow Fairness Condition) *Two flows with window sizes x_1 and x_2, $x_1 < x_2$, sharing a bottleneck link will eventually converge to and maintain a totally fair allocation of bottleneck link bandwidth if the following condition is satisfied (after each application of an increase policy $\mathcal{I}$ and decrease policy $\mathcal{D}$ or over any reasonably small period of time):*

$$\frac{\Delta x_1}{x_1} \geq \frac{\Delta x_2}{x_2} \tag{7}$$

At least one of the two policies must ensure a strict inequality.

Proof: The proof proceeds as follows: Suppose two flows with windows x_1 and x_2 share a bottleneck link. Let $\Delta \mathrm{F}$ be the change in F corresponding to a small change Δx_1 in x_1 and Δx_2 in x_2. If $\Delta \mathrm{F}$ is positive at each application of an increase/decrease policy, then eventually F reaches its maximum value of 1 regardless of its initial value, and the system moves to a totally fair allocation. Thus we only need to ensure $\Delta \mathrm{F} \geq 0$ always.

For $n = 2$, Equation 6 becomes

$$\mathrm{F} = \frac{(x_1 + x_2)^2}{2(x_1^2 + x_2^2)}.$$

Using

$$d\mathrm{F} = \frac{\partial \mathrm{F}}{\partial x_1} dx_1 + \frac{\partial \mathrm{F}}{\partial x_2} dx_2$$

and making the continuous fluid approximation that the changes Δx_1 and Δx_2 represent infinitesimal changes to x_1 and x_2:

$$dx_1 \approx \Delta x_1,\ dx_2 \approx \Delta x_2 \text{ and } d\mathrm{F} \approx \Delta \mathrm{F} \tag{8}$$

we get

$$\Delta \mathrm{F} = \frac{\left\{ \left(x_1^2 + x_2^2\right)(x_1 + x_2) - (x_1 + x_2)^2 x_1 \right\} \Delta x_1}{(x_1^2 + x_2^2)^2} + \frac{\left\{ \left(x_2^2 + x_1^2\right)(x_2 + x_1) - (x_2 + x_1)^2 x_2 \right\} \Delta x_2}{(x_1^2 + x_2^2)^2}$$

Imposing the condition $\Delta \mathrm{F} \geq 0$, we get

$$(x_1^2 + x_2^2)(\Delta x_1 + \Delta x_2) \geq (x_1 + x_2)(x_1 \Delta x_1 + x_2 \Delta x_2) \tag{9}$$

Simplifying, and using $x_2 - x_1 > 0$, we can write

$$\frac{\Delta x_1}{x_1} \geq \frac{\Delta x_2}{x_2}$$

Note that we need at least one of $\mathcal{I}$ or $\mathcal{D}$ to ensure $\Delta \mathrm{F} > 0$ so that F increases over each congestion epoch and eventually becomes 1. Thus at least one of them must have a strict inequality in Equation 7. Also, once $x_1 = x_2$, this equality is maintained under synchronous feedback and the values of the window sizes will increase or decrease in lockstep with each other. ■

We can use a linear interpolation of the window size between two applications of $\mathcal{I}$ for GAIMD and TCP, so that $dx_1 = \Delta x_1$, $dx_2 = \Delta x_2$ and $d\mathrm{F} = \Delta \mathrm{F}$ which is stronger than Equation 8. Thus the above proof applies to these protocols even though the changes Δx_1 and Δx_2 are not infinitesimal.

This is used in the following corollary which gives a new algebraic proof of convergence to fairness for two GAIMD (or TCP) flows. Chiu and Jain [4] give a different proof for the convergence of GAIMD under the same conditions. This validates the correctness of our results in a way. We also give the first algebraic proof of convergence for binomial congestion control. (The original proof in [2] is a geometric proof based on the Chiu-Jain phase plot.)

Corollary 2 *Two TCP or GAIMD flows converge to fairness under the assumption of synchronized feedback*

Proof: For TCP, $\Delta x = 1$ for the increase policy and Equation 7 becomes: $1/x_1 > 1/x_2$ if $x_1 < x_2$. Similarly $\Delta x = -x/2$ for the decrease policy and Equation 7 reduces to $-(1/2) = -(1/2)$.

For GAIMD, $\Delta x = \alpha$ for the increase policy and Equation 7 becomes: $\alpha/x_1 > \alpha/x_2$ if $x_1 < x_2$. Similarly $\Delta x = -\beta x$ for the decrease policy and Equation 7 reduces to $-\beta = -\beta$. ■

For Binomial congestion control, $\Delta x = \alpha/x^k$ for the increase policy and Equation 7 becomes: $\alpha/x_1^{k+1} \geq \alpha/x_2^{k+1}$ if $x_1 < x_2$. Similarly $\Delta x = -\beta x^l$ for the decrease policy and Equation 7 reduces to $-\beta x_1^{l-1} \geq$

$-\beta x_2^{l-1}$ if $x_1 < x_2$. Thus, the increase and decrease policy separately ensure convergence to fairness only if $k > -1$ and $l > 1$.

However, SQRT and IIAD, the two instances of binomial congestion control experimentally evaluated in [2] have values of $l < 1$. Also, as discussed in section 2, we can use $l > 1$ only if we know the maximum window size. The following corollary shows that binomial congestion control converges to fairness if $k, l \geq 0$, which is satisfied by both SQRT ($k = 1/2,\ l = 1/2$) and IIAD ($k = 1,\ l = 0$).

The proof proceeds as follows: We have shown above that each application of an increase policy increases fairness if $k > -1$. Thus any sequence of window size updates using only the increase policy increases fairness. The proof shows that even though the decrease policy will worsen fairness when $0 \leq l < 1$, the increase in fairness from the previous application of the increase policy more than offsets this decrease in fairness. Also, for sufficiently large window sizes, binomial congestion control is 1-responsive. Thus each application of a decrease policy is always preceded by an increase policy, so that the value of F increases over each congestion epoch.

Corollary 3 *Binomial congestion control converges to fairness if $k, l \geq 0$.*

Proof: Clearly, binomial congestion control with $k, l \geq 0$ satisfies the 1-responsiveness criterion (Equation 5) for sufficiently large window sizes. Thus, each application of a decrease policy is preceded by an application of the increase policy.

Suppose the window sizes of the two flows are x_1, x_2 ($x_1 < x_2$), just before the application of the increase policy that is followed by an application of the decrease policy. It is sufficient to show that the fairness index increases over this subsequence of window size adjustments (an increase followed by a decrease) since we already know that sequences consisting only of applications of increase policy improve the fairness index if $k > -1$. (Because of 1-responsiveness we need not consider a subsequence with two or more window decreases.)

We need to show that if $x_1 \leq x_2$,

$$\frac{\frac{\alpha}{x_1^k} - \beta(x_1 + \frac{\alpha}{x_1^k})^l}{x_1} \geq \frac{\frac{\alpha}{x_2^k} - \beta(x_2 + \frac{\alpha}{x_2^k})^l}{x_2}$$

Approximating $\beta(x + \alpha/x^k)^l \approx \beta x^l$, we need to prove

$$\frac{\frac{\alpha}{x_1^k} - \beta(x_1)^l}{x_1} \geq \frac{\frac{\alpha}{x_2^k} - \beta(x_2)^l}{x_2}$$

Or,

$$\frac{\alpha - \beta x_1^{k+l}}{x_1^{k+1}} \geq \frac{\alpha - \beta x_2^{k+l}}{x_2^{k+1}}$$

But when $x_1 \leq x_2$, we have $1/x_1^{k+1} \geq 1/x_2^{k+1}$ and $\alpha - \beta x_1^{k+l} > \alpha - \beta x_2^{k+l}$ because $k + l > 0$.

Thus the decrease in fairness due to the application of the decrease policy is offset by the increase in fairness resulting from the previous increase in window size. Thus F always improves over a congestion epoch, and binomial congestion control converges to fairness even if $0 \leq l < 1$. ■

Theorem 1 can be extended for n flows:

Theorem 4 (n-Flow Fairness Condition) *n-flows with windows x_1,x_2,… x_n converge to fairness if the following condition is satisfied (after each application of an increase policy $\mathcal{I}$ and decrease policy $\mathcal{D}$ or over any reasonably small period of time):*

$$\sum_{i=1}^{i=n} x_i^2 \sum_{i=1}^{i=n} \Delta x_i \geq \sum_{i=1}^{i=n} x_i \sum_{i=1}^{i=n} x_i \Delta x_i \tag{10}$$

Again, at least one of the two policies should ensure a strict inequality.

The proof is very similar to the 2-flow case. Notice that the n-flow result reduces to Equation 9 for $n = 2$.

Corollary 5 *N GAIMD or TCP flows converge to fairness*

Proof: We derive the results for GAIMD. We can show that n TCP flows converge to fairness in exactly the same way. In fact, the result for GAIMD implies the result for TCP because TCP is a special case of GAIMD.

- Case 1: The increase policy satisfies Equation 10.

  In this case $\Delta x_i = \alpha$. Since F is upperbounded by 1, we get (from Equation 6)

  $$1 \geq \frac{(\sum x_i)^2}{n(\sum x_i^2)}$$

  Rewriting this, we get,

  $$\sum_{i=1}^{i=n} x_i^2 \sum_{i=1}^{i=n} 1 \geq \sum_{i=1}^{i=n} x_i \sum_{i=1}^{i=n} x_i \cdot 1$$

  Multiplying both sides by α,

  $$\sum_{i=1}^{i=n} x_i^2 \sum_{i=1}^{i=n} \alpha \geq \sum_{i=1}^{i=n} x_i \sum_{i=1}^{i=n} x_i \cdot \alpha$$

  which is Equation 10 with $\Delta x_i = \alpha$.

- Case 2: The decrease policy satisfies Equation 10

  In this case $\Delta x_i = \beta x_i$. Equation 10 becomes

  $$\sum_{i=1}^{i=n} x_i^2 \sum_{i=1}^{i=n} \beta x_i = \sum_{i=1}^{i=n} x_i \sum_{i=1}^{i=n} x_i \cdot \beta x_i$$

Thus, GAIMD ensures that each application of the increase policy leads to an increase in fairness, but maintains the fairness index when the decrease policy is applied.

■

It can be shown that n binomial flows also satisfy Equation 10 and hence converge to fairness. The proof sketch is very similar to the proof for Theorem 9. However, this result is easily obtained in section 4.2 as a special case of Theorem 10. Thus we have:

Corollary 6 $n > 2$ *binomial flows converge to fairness.*

4 CYRF

In this section, we adopt a novel approach to protocol design. Since the primary motivation behind this work is the wide range of requirments of different applications, we would like to know what latitude an application can have in choosing a response function. We ask the question: "What is the class of increase/decrease policies that satisfy Equation 7?". This yields a new family of congestion control protocols that are *designed* to converge to fairness and efficiency.

4.1 $f(\cdot), g(\cdot)$ congestion control

Suppose two flows share a bottleneck. Assuming that Δx is some function of x, we can see that Equation 7 (and Example 1) implies some kind of monotonicity for Δx. Also, this function must be differentiable for the proof of Theorem 1 to apply. Furthermore, for convergence to efficiency, the principle of negative feedback discussed in section 3.4 must be satisfied. These requirements are expressed in the following theorem.

Theorem 7 (2-Flow Fairness for CYRF) *Let $f(x)$ and $g(x)$ be any differentiable monotonically non-decreasing functions (at least one of them strictly increasing) with $f(x) > 0$ and $0 < g(x) \leq 1$ for all $x > 1$. Then the increase and decrease policies in Equation 1 ensure converge to fairness and efficiency for two flows sharing a bottleneck link.*

Proof: $\Delta x = x(t)/f(x(t))$ for the increase policy and $\Delta x = -x(t)g(x(t))$ for the decrease policy.

Convergence to fairness: It is easy to see that, if x_1, x_2 ($x_1 < x_2$), are the two window sizes, then because of the monotonicity of $f(\cdot)$ and $g(\cdot)$, the increase policy satisfies Equation 7: $1/f(x_1) \geq 1/f(x_2)$ and similarly for the decrease policy,$-g(x_1) \geq -g(x_2)$. Note that since at least one of the two functions is strictly increasing, we have a strict inequality for at least one of the two policies as required by Theorem 1.

Convergence to efficiency: For CYRF to be efficient, the principle of negative feedback must apply and $\mathcal{I}$ must increase the window and $\mathcal{D}$ must decrease the window size. This is clearly satisfied for all window sizes $x(t) \geq 1$, because Δx is positive for the increase policy and negative for the decrease policy (due to the constraints $f(x(t)) > 0$ and $g(x(t)) > 0$).

The upperbound $g(x) \leq 1$ ensures that $\mathcal{D}$ does not lead to negative window sizes. ■

Because each application of an increase or decrease policy moves the system towards fairness, we call this class of protocols as *step-wise convergent*.

For protocols with a *smooth* increase policy[3], we can drop the requirement that $g(x)$ be a monotonically non-decreasing function; instead, we only require that $f(x)g(x)$ be monotonically non-decreasing and greater than 1 for $x > c$, for some small constant c. We then obtain *epoch-wise convergent* protocols that only converge over each congestion epoch. Note that we need either $f(x)$ or $f(x)g(x)$ to be strictly increasing to meet the "strict inequality" requirement of Theorem 1. We call this class of protocols 1-CYRF because if $f(x)g(x) > 1$, the protocol is 1-responsive.

Theorem 8 *1-CYRF converges to fairness for $n = 2$ flows.*

Proof: In 1-CYRF, each application of the increase policy will still increase the fairness index. However, the decrease policy can now *worsen* fairness if $g(x)$ is not monotonic. But if the increase in F can offset the decrease, the system will still converge to fairness over each *congestion epoch*. To accomplish this, we impose a stronger constraint that the increase in F from a single application of $\mathcal{I}$ must be more than the decrease in F from a single application of $\mathcal{D}$ and use the 1-responsiveness condition to ensure that each application of a decrease policy is preceded by at least one increase. Thus, in deterministic steady-state, F will still increase over each congestion epoch.

A single application of $\mathcal{I}$ followed by an application of $\mathcal{D}$ increases fairness if Equation 7 is satisfied. Here $\Delta x = x/f(x) - (x + x/f(x))g(x + x/f(x)) \approx x(1/f(x) - g(x))$ because of smoothness. Thus we require $1/f(x_1) - g(x_1) \geq 1/f(x_2) - g(x_2)$ if $x_1 \leq x_2$. Notice that if $f(x)g(x)$ is monotonically *non-decreasing*, then

$$\frac{1}{f(x)} - g(x) = \frac{1 - f(x)g(x)}{f(x)}$$

is monotonically *non-increasing*. Thus the inequality required by Equation 7 will be automatically satisfied. ■

CYRF was *designed* to converge to fairness for the two-flow case. The following theorem proves that CYRF converges to fairness for $n > 2$ flows also.

[3]Formally, a policy is smooth if the window size increase from a single application of the policy is at least an order of magnitude smaller than the current window size, for large enough windows.

Theorem 9 *CYRF converges to fairness for $n > 2$ flows.*

Proof: We will prove a stronger result, viz., that CYRF flows satisfy the sufficiency condition given by Equation 10. The proof proceeds as follows: Without loss of generality, we will order the flows by increasing window size. We then use mathematical induction to show that if Equation 10 is satisfied with k flows, adding a $(k+1)th$ flow with a larger window size preserves the fairness condition. The key fact used is that $1/f(x)$ and $-g(x)$ are both monotonically non-increasing, so that $1/f(x_{k+1}) \leq 1/f(x_i)$ and $-g(x_{k+1}) \leq -g(x_{k+1})$ for all $1 \leq i \leq k$.

- Case 1: The increase policy increases fairness.

  Theorem 7 forms the base case.

  Without loss of generality let $x_1 \leq x_2 \leq \ldots \leq x_k \leq x_{k+1} = X$ be the window sizes of $k{+}1$ flows. Suppose Equation 10 is satisfied with $n \leq k$ flows. Here, $\Delta x = x/f(x)$. So we get:

$$\sum_{i=1}^{k} x_i^2 \sum_{i=1}^{k} \frac{x_i}{f(x_i)} \geq \sum_{i=1}^{k} x_i \sum_{i=1}^{k} \frac{x_i^2}{f(x_i)} \tag{11}$$

  Consider $n = k+1$. Because $x_{k+1} = X \geq x_i$, for all $1 \leq i \leq k$, and $f(\cdot)$ is monotonically non-decreasing, we can write

$$X \sum_{i=1}^{k} \frac{X x_i - x_i^2}{f(x_i)} \geq \frac{X}{f(X)} \sum_{i=0}^{k} (X x_i - x_i^2) \tag{12}$$

  Rewriting, we get

$$X^2 \sum_{i=1}^{k} \frac{x_i}{f(x_i)} + \frac{X}{f(X)} \sum_{i=1}^{k} x_i^2 \geq X \sum_{i=1}^{k} \frac{x_i^2}{f(x_i)} + \frac{X^2}{f(X)} \sum_{i=1}^{k} x_i \tag{13}$$

  Adding Eqn. 11 and Eqn. 13, adding $X^3/f(X)$ to both sides and factoring, we get:

$$\left(\sum_{i=1}^{k} x_i^2 + X^2\right)\left(\sum_{i=1}^{k} \frac{x_i}{f(x_i)} + \frac{X}{f(X)}\right) \geq \left(\sum_{i=1}^{k} x_i + X\right)\left(\sum_{i=1}^{k} \frac{x_i^2}{f(x_i)} + \frac{X^2}{f(X)}\right)$$

  This is just Equation 10 for $n = k+1$. Hence, by the principle of mathematical induction, Equation 10 holds for any number of flows, n.

- Case 2: The decrease policy increases fairness. The algebra is exactly the same as above, except that $1/f(\cdot)$ is replaced by $-g(\cdot)$, which is also a non-increasing function.

Since one of $f(x)$ or $g(x)$ is strictly increasing, one of the two policies $\mathcal{I}$ or $\mathcal{D}$ will ensure a strict inequality as required. ■

Note that the previous argument for convergence to efficiency still holds for the n-flow case and need not be repeated again.

It can also be shown that 1-CYRF converges to fairness for n-flows. The proof is very similar to Theorem 9. The increase case is exactly the same. In the decrease case, instead of $-g(\cdot)$, we use $1/f(x) - g(x)$, which is a decreasing function as shown above. Thus we have:

Theorem 10 1-*CYRF converges to fairness for $n > 2$ flows.*

4.2 Applications of CYRF

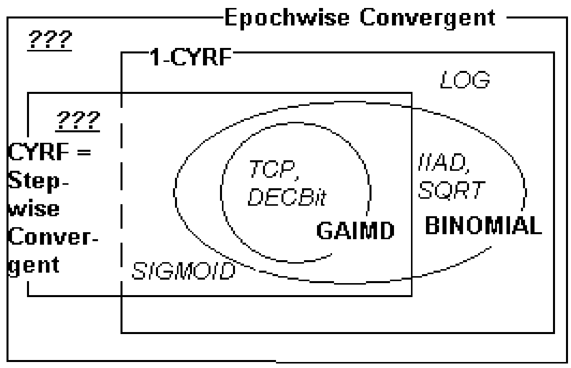


Figure 1. A classification of window-based protocols

The first application of CYRF is a "natural" classification of window based protocols. Observe that any window increase or decrease function can be written in the form of Equation 1. However, if there is a range $[x_1, x_2]$ where the f and g are not monotonic, the function is not CYRF. In such a case we can construct an increase/decrease step similar to Example 1 with window sizes in $[x_1, x_2]$ which *decreases* fairness. Thus CYRF implies stepwise convergence and vice-versa.

Notice that if $f(x)$, $g(x)$ are monotonically non-decreasing, so is their product $f(x)g(x)$. Thus if a CYRF protocol is smooth and 1-responsive (and all important protocols are), it is also 1-CYRF. Clearly the converse is not true – Binomial congestion control can be written in terms of Equation 1:

$$f(x) = x^{k+1}/\alpha \; ; \; g(x) = \beta x^{l-1} \tag{14}$$

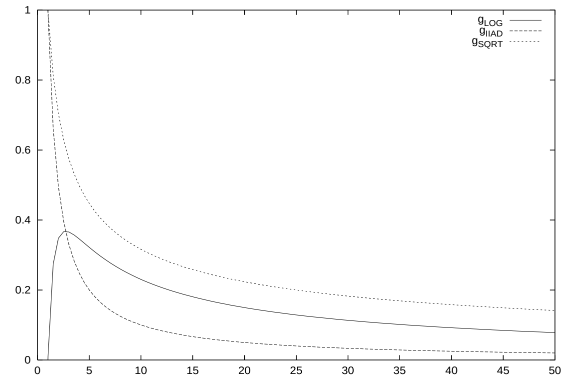


Figure 2. A Smoothness hierarchy through functions of different orders of magnitude: $g(x)$ for SQRT, IIAD and LOG

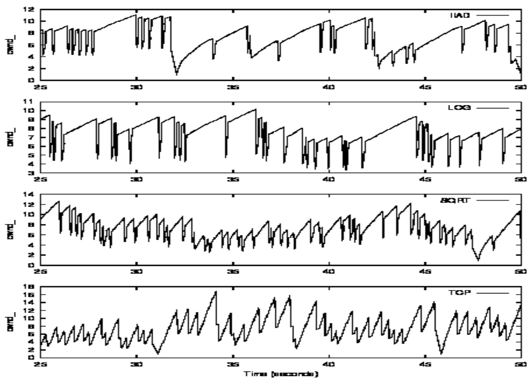


Figure 3. The Smoothness hierarchy: cwnd_ variations of (from the top) IIAD, LOG, SQRT and TCP

However this is not CYRF for $l < 1$ because $g(x)$ will then monotonically *decrease*. Fortunately, all TCP-friendly Binomial controls satisfy $k + l = 1$ [2] so that $f(x)g(x) = \beta x/\alpha > 1$ for $x > (\alpha/\beta)$. Thus TCP-friendly binomial congestion control is 1-CYRF but not CYRF.

Also, by substituting $f(x) = x$, and $g(x) = 1/2$ in Equation 1, we see that TCP is a special case of CYRF. Similarly, GAIMD is also a special case of CYRF ($f(x) = x/\alpha$, and $g(x) = \beta$).

Figure 1 summarizes the relationships. The rectangles represent new classes introduced in this work, and the ovals show known classes. Specific protocols in each class are given in italics. In particular, LOG and SIGMOID are two new protocols suitable for streaming media-like applications [22]. They were developed using the CYRF framework and their design will be the subject of a future paper. However, we briefly describe LOG in the next section to provide a concrete example for the use of the CYRF framework.

Observe that SIGMOID is in the class of stepwise convergent protocols wheras LOG is a 1-CYRF protocol and thus is only epochwise convergent. As the "???" marks indicate, we do not know of any protocols that are 1-CYRF but not CYRF or protocols that are epoch-convergent but not 1-CYRF. We believe that TCP-friendly 1-CYRF may represent the widest class of smooth window-based memoryless binary feedback TCP-friendly congestion control protocols that always converge to fairness.

CYRF can also be thought of as a framework for analyzing window-based protocols. For example, it follows from the above discussion that Binomial, GAIMD and TCP converge to fairness and efficiency because they fall within the CYRF framework.

The third (and main) application envisaged for the CYRF framework is the design of new protocols to suit different application and network needs, without having to worry about convergence to efficiency and fairness. The next section sketches an example protocol design.

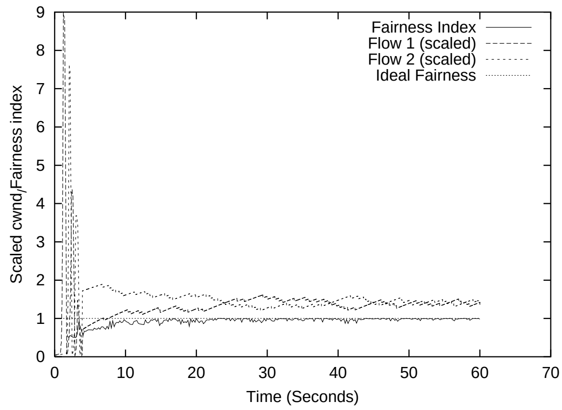
4.3 LOG

CYRF represents a wide class of window-based protocols. Applications can choose different $f(x)$ and $g(x)$ to get different window-based protocols. For example, from Equation 1 it is easy to see that an application that uses a slowly increasing function for $f(x)$ will be more aggressive and make full use of network bandwidth as soon as it becomes available. Similarly a slowly increasing $g(x)$ results in a smoother response to congestion indications. We cannot choose an arbitrarily aggressive increase policy together with a very smooth decrease policy because of the TCP-friendliness constraint [14, 15] which requires all flows to limit their sending rate to that of a comparable TCP flow. Thus there is a continuum of protocols with different *degrees of smoothness*.

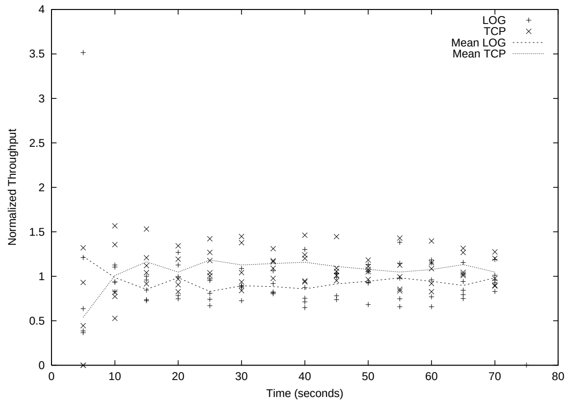
While smoother protocols are better (from the application point of view, for streaming media applications), this is true only in steady-state. Aggressive protocols are also more responsive protocols and usually have better transient and dynamic behaviors [3, 26].

Here we trade-off between smoothness and dynamic or transient behavior by designing a protocol whose properties are "between" SQRT ($k = l = 0.5$ in eqn. 4) and IIAD ($k = 1$, $l = 0$), the two non-linear binomial controls studied in [2]. From Equation 14, we see IIAD is smoother than SQRT because $g_{\text{IIAD}}(x) = 1/x$ is always less than $g_{\text{SQRT}}(x) = 1/\sqrt{x}$.

As shown in Figure 2, by choosing a function that lies



(a)



(b)

Figure 4. Single Bottleneck (a) Fairness index and scaled cwnd_ of 2 competing LOG flows, (b)Normalized throughput (Kb/500ms) of $n = 5$ LOG and $n = 5$ TCP flows sharing the bottleneck

between these two, such as $g(x) = log(x)/x$, we get a new protocol with intermediate properties. For this to be TCP-friendly, we need $f(x) \propto x^2/log(x)$[4]. Thus the increase and decrease policies in Equation 1 become:

$$\begin{aligned} \mathcal{I}: x(t+R) &\leftarrow x(t) + log(x)/x \\ \mathcal{D}: x(t+R) &\leftarrow x(t) - 0.5 * log(x) \end{aligned} \quad (15)$$

We call this protocol LOG. From fig. 2, we see that just like IIAD and SQRT, $g(x)$ for LOG is a decreasing function and hence LOG is only 1-CYRF. The decrease policy will worsen fairness. However, unlike IIAD or SQRT, $g(x)$ increases for very small window values; thus LOG is CYRF in this region. Due to this, it is likely to perform better in extremely congested situations when the window sizes are small.

5 Experimental validation

We have implemented CYRF in the *ns-2* network simulator[5] by changing TCP's congestion avoidance mechanism to use arbitrary increase or decrease functions. CYRF inherits other mechanisms such as slow-start and timeouts from TCP. We use LOG as an example CYRF protocol to validate the main theoretical results.

Our first setup uses the standard "dumb bell" topology: a single bottleneck link with a default bandwidth of 10Mb and a delay of 1ms. RED queues with a maximum queue size Q_{max} equal to the bandwidth-delay product and maximum and minimum drop thresholds at 20 and 80 percent of Q_{max} are used. n flows are started at random times in the first two seconds. All flows use 1Kb packets and saturated senders are simulated by using the FTP application in *ns-2*. A random number of Reno TCP flows in the opposite direction form background traffic.

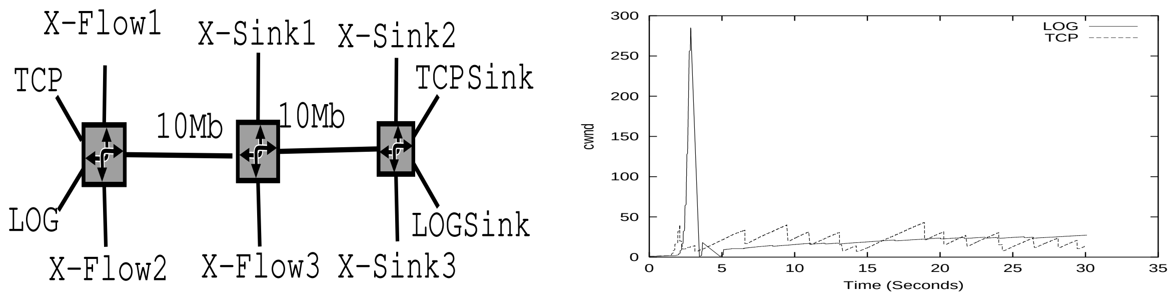
Fig. 4 (a) shows the scaled cwnd values of 2 competing LOG flows across the bottleneck and the corresponding Chiu-Jain-Hawe fairness index F. These results show that LOG rapidly converges to a value of F near 1 validating that Theorems 1 and 7 hold in practice too. The repeated instantaneous decreases in fairness are because of packet drops (mostly random RED drops). Note that LOG is a 1-CYRF protocol and drops invoke the decrease policy which worsens fairness.

Fig. 4 (b) shows the normalized throughputs of n TCP and n LOG flows sharing the bottleneck link. This shows that TCP/LOG protocols interact well and experimentally shows the TCP-friendliness of LOG. We repeated this experiment with LOG/IIAD and LOG/SQRT flows to validate that different CYRF protocols interact well together. The results look very similar to the TCP/LOG case above and are omitted for space reasons.

In the next experiment, we validate the predicted smoothness for LOG. We use the multiple-bottleneck topology shown in Fig. 5 (a) to examine the effect of cross-traffic. The inter-router links are 10Mb with a delay of 1ms, and the others are 100Mb with 24ms delay, designed to saturate the routers. All queues are RED. X-Flow$i$ is the source of a TCP/Reno cross-flow to X-Sink$i$ that starts at a random time in the first two seconds. A TCP/Reno flow from TCP to TCPSink and a LOG flow from LOG to LOGSink are examined. Fig 5 (b) shows that the congestion window of the LOG flow varies much more smoothly than TCP and shares

[4] It can be shown that for TCP-friendliness we need $f(x)g(x) \propto x$ [22].

[5] Available from `http://www.isi.edu/nsnam/ns/`



(a) (b)

Figure 5. Multiple-Bottlenecks: (a) Topology, (b) Window size variation of TCP and LOG

the bandwidth effectively. Similar results were obtained for the dumb-bell topology.

In Section 4.3, it was predicted that LOG would be smoother than SQRT and less smoother than IIAD because $g_{\mathrm{LOG}}()$ was between $g_{\mathrm{IIAD}}()$ and $g_{\mathrm{SQRT}}()$ (c.f. Fig. 2) SQRT is expected to be smoother than TCP. To validate this, we plotted the cwnd_ values of TCP, SQRT, LOG and IIAD flows in the single bottleneck experiments described above after giving sufficient time for startup transients to stabilize. Fig. 3 shows the four curves between $t = 25s$ and $t = 50s$. As expected, we see a *smoothness hierarchy* with the topmost curve (IIAD) being the smoothest, and the bottom curve (TCP) the least smooth. It is interesting to note that LOG falls outside the binomial class of protocols, and yet it falls "between" IIAD and SQRT.

6 Conclusion

In this paper, we presented CYRF, a novel approach to protocol design that is *designed* to converge to fairness and efficiency. This allows protocol designers to choose the appropriate response function given the application and network issues at hand, without having to worry about convergence issues. We validated the CYRF results on a sample protocol called LOG designed using the CYRF framework to have smoothness properties intermediate to those of two well-studied binomial congestion control protocols: IIAD and SQRT.

On the theoretical front, we gave a new intuitive sufficient condition for convergence to fairness which can easily be made use of outside the CYRF framework. CYRF itself includes most well-known window-based protocols as special cases and can be used to understand these protocols better. Finally, we gave a new classification of window-based protocols based upon the results of this paper.

In a future paper, we will discuss two new protocols for streaming media-like applications and show how to make smooth CYRF protocols TCP-friendly.

References

[1] Mark Allman, Vern Paxson, and W. Stevens. *TCP Congestion Control*. Internet Engineering Task Force, April 1999. RFC 2581 (Standards Track).

[2] D. Bansal and H. Balakrishnan. Binomial congestion control algorithms. In *Proceedings of IEEE INFOCOM 2001*, April 2001.

[3] D. Bansal, H. Balakrishnan, S. Floyd, and S. Shenker. Dynamic behavior of slowly-responsive congestion control algorithms. In *Proceedings of ACM SIGCOMM 2001*, San Diego, CA, August 2001.

[4] Dah-Ming Chiu and Raj Jain. Analysis of the increase and decrease algorithms for congestion avoidance in computer networks. *Computer Networks and ISDN Systems*, 17:1–14, 1989.

[5] D. Clark, M. Lambert, and L. Zhang. NETBLT: A high throughput transport protocol. In *Proceedings of ACM SIGCOMM '88*, August 1988.

[6] J. Crowcroft and P. Oechslin. Differentiated end-to-end internet services using a weighted proportional fair sharing TCP. *ACM Computer Communication Review*, 28(3), July 1998.

[7] S. Floyd, M. Handley, and J. Padhye. A comparison of equation-based and AIMD congestion control. Available from `http://www.aciri.org/tfrc`, May 2000.

[8] S. Floyd, M. Handley, J. Padhye, and J. Widmer. Equation-based congestion control for unicast applications. In *Proceedings of ACM SIGCOMM 2000*, August 1988. Extended version available as International Computer Science Institute tech report TR-00-03, March 2000.

[9] Van Jacobson and Mike Karels. Congestion avoidance and control. *ACM Computer Communication Review*, 18(4):314–329, August 1990. Revised version of Sigcomm '88 paper.

[10] R. Jain, D-M. Chiu, and W. Hawe. A quantitative measure of fairness and discrimination for resource allocation in shared computer systems. Technical Report TR-301, DEC Research Report, September 1984.

[11] R. Jain, K. K. Ramakrishnan, and D-M. Chiu. Congestion avoidance in computer networks with a connectionless network layer. Technical Report DEC-TR-506, Digital Equipment Corporation, 1988. Reprinted in C. Partridge, Ed., *Innovations in Internetworking*, published by Artech House, October 1988.

[12] Shudong Jin, Liang Guo, Ibrahim Matta, and Azer Bestavros. Tcp-friendly simd congestion control and its convergence behavior. In *Proceedings of International Conference on Network Protocols 2001*.

[13] Lampros Kalampoukas, Anujan Varma, and K. K. Ramakrishnan. Explicit window adaptation: A method to enhance TCP performance. In *Proceedings of IEEE INFOCOM '98*, San Francisco, California, March/April 1998.

[14] J. Mahdavi and S. Floyd. TCP-friendly unicast rate-based flow control. Technical Note sent to the end2end-interest mailing list, January 1997. Available from `http://www.psc.edu/networking/papers/tcp_friendly.html`.

[15] J. Mahdavi and S. Floyd. The TCP-friendly website. `http://www.psc.edu/networking/tcp_friendly.html`, June 1999.

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

[17] K. K. Ramakrishnan, S. Floyd, and D. Black. *The Addition of Explicit Congestion Notification (ECN) to IP*. Internet Engineering Task Force, September 2001. RFC 3168 (Standards Track).

[18] K. K. Ramakrishnan and Raj Jain. A binary feedback scheme for congestion avoidance in computer networks. *ACM Transactions on Computer Systems*, 8(2):158–181, May 1990.

[19] R. Rejaie, M. Handley, and D. Estrin. Quality adaptation for unicast audio and video. In *Proceedings of ACM SIGCOMM '99*, September 1999.

[20] R. Rejaie, M. Handley, and D. Estrin. RAP: An end-to-end rate-based congestion control mechanism for realtime streams in the internet. In *Proceedings of IEEE INFOCOM '99*, March 1999.

[21] I. Rhee, V. Ozdemir, and Y. Yi. TEAR: TCP emulation at receivers – flow control for multimedia streaming. Technical report, North Carolina State University, April 2000.

[22] Nishanth R. Sastry and Simon S. Lam. CYRF: A Framework for Window-based Unicast Congestion Control. Technical Report TR-2002-09, Department of Computer Sciences, The University of Texas at Austin, January 2002. Revised September 23, 2002.

[23] H. Schulzrinne, S. Cassner, R. Frederick, and V. Jacobson. *RTP: A Transport Protocol for Real-Time Applications*. Internet Engineering Task Force, January 1996. RFC 1889 (Standards Track).

[24] D. Sisalem and H. Schulzrinne. The loss-delay based adjustment algorithm: A TCP-friendly adaptation scheme. In *Proc. 8th International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, 1998.

[25] G. Wright and W. R. Stevens. *TCP/IP Illustrated, Volume 2: The Implementation*. Addison Wesley, Reading, MA, 1995.

[26] Y. R. Yang, M. S. Kim, and S. S. Lam. Transient behaviors of TCP-friendly congestion control protocols. In *Proceedings of IEEE INFOCOM 2001*, Anchorage, Alaska, April 2001.

[27] Y. R. Yang and S. S. Lam. General AIMD congestion control. In *Proceedings of the 8th IEEE International Conference on Network Protocols*, Osaka, Japan, November 2000.