# Application Specific Unicast Congestion Control

by

**Nishanth R. Sastry, B.E.**

**Thesis**

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

**Master of Arts**

**The University of Texas at Austin**

December 2001

# Application Specific Unicast Congestion Control

To Amma and Nana, for being there. . .

# Acknowledgments

Several people have been instrumental in helping me finish this work, either directly or indirectly.

I owe the greatest thanks to Prof. Simon Lam, who has been an ideal advisor in every respect. He took me on as a student soon after I came to Austin, even though he knew very little about me at the time and has been extremely supportive ever since. His keen judgement and appraisals of research ideas lead me to strive towards greater heights and towards this thesis. I hope that I can attain and continue to meet his exacting standards.

I would also like to thank Prof. Mohamed Gouda for agreeing to be on my supervising committee. His class on Computer Networks was one of the most enjoyable classes I took at UT. I will remember his reassuring words when I was still struggling to prove a few of the results.

I owe my current position to my parents, Gayathri and Ramakrishna and my sister Nikhila. I would not have made it without you and I am doing this because of you. I am deeply indebted to many in my family. Over the last year, SHMM in particular played a big role in helping me make the transition. Lastly, life would have been impossible without my friends in Austin. Thanks to y'all.

NISHANTH R. SASTRY

*The University of Texas at Austin*
*December 2001*

# Application Specific Unicast Congestion Control

Nishanth R. Sastry, M.A.

The University of Texas at Austin, 2001

Supervisor: Simon S. Lam

Some new Internet-based applications such as streaming media cannot function well with TCP's decrease-by-half response to congestion indications from the network. However, to prevent congestion collapse, it is imperative that all applications use some form of congestion control. Furthermore, for such a mechanism to be feasible, it must converge to fairness and efficiency, just as TCP does.

This thesis presents a framework that allows an application to choose a suitably aggressive or smooth congestion response from a wide family of window-based protocols called CYRF (for *C*hoose *Y*our *R*esponse *F*unction) that are designed to converge to fairness. We also give a simple rule for smooth CYRF flows to be TCP-friendly.

We first derive a sufficient condition that ensures convergence to fairness. We then present the surprising result that an application can satisfy this fairness condition and construct a congestion response tailored to its needs by choosing from almost any pair of monotonically non-decreasing functions. Constructing a window increase policy using a slowly increasing function results in an aggressive protocol. Similarly, a slowly increasing function in the decrease policy gives rise to smooth protocols. We characterize TCP-friendliness in steady-state and show that any smooth CYRF protocol can be TCP-friendly if the product of its window increase and decrease functions is proportional to the window size.

An interesting aspect of this work is that all commonly used window-based protocols such as TCP, GAIMD, and binomial congestion control are shown to be special cases of a single family of protocols, thus providing a powerful unified framework for analyzing them. We derive most of the important results about these protocols as special cases of the results for CYRF.

# Contents

**Acknowledgments** **v**

**Abstract** **vi**

**Chapter 1 Introduction** **1**

1.1 The Problem . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2

1.2 The Solution . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3

1.3 Organization of the Thesis . . . . . . . . . . . . . . . . . . . . . . . . 4

**Chapter 2 Related Work** **6**

2.1 Historical Approaches . . . . . . . . . . . . . . . . . . . . . . . . . . . 6

2.2 Recent Work on Congestion Control . . . . . . . . . . . . . . . . . . . 7

2.2.1 Comparison . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8

**Chapter 3 The Model** **10**

3.1 Notation Conventions . . . . . . . . . . . . . . . . . . . . . . . . . . . 10

3.2 The WBF Model . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 10

3.2.1 Example . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 11

3.3 Assumptions . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 11

3.3.1 Binary feedback model . . . . . . . . . . . . . . . . . . . . . . . 12

3.3.2 Synchronous feedback . . . . . . . . . . . . . . . . . . . . . . . 12

3.3.3 Saturated senders . . . . . . . . . . . . . . . . . . . . . . . . . . 12

3.3.4 Steady-state and 1-responsiveness . . . . . . . . . . . . . . . . . 12

3.3.5 Continuous fluid approximation . . . . . . . . . . . . . . . . . . 13

3.3.6 Congestion avoidance mode . . . . . . . . . . . . . . . . . . . . . . . . . . 14

**Chapter 4 Constraints on Feasible WBF Protocols** **15**

4.1 Convergence to Fairness . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15

4.2 Convergence to Efficiency . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 16

**Chapter 5 Conditions for Convergence to Fairness** **17**

5.1 The Two-Flow Case . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17

5.2 The N-Flow Case . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 21

**Chapter 6 CYRF: A Generalized WBF Protocol** **23**

6.1 $f(\cdot), g(\cdot)$ Congestion Control . . . . . . . . . . . . . . . . . . . . . . . . . . . 24

6.2 $N$-flow Convergence to Fairness . . . . . . . . . . . . . . . . . . . . . . . . . 25

6.3 1-CYRF: Generalizing CYRF . . . . . . . . . . . . . . . . . . . . . . . . . . . 27

6.4 1-CYRF as a Unified Framework for WBF . . . . . . . . . . . . . . . . . . . . 28

**Chapter 7 TCP-Friendly CYRF** **30**

7.1 TCP-Friendliness . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 30

7.2 Smoothness . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 33

7.3 CYRF with the TCP-Friendliness Constraint . . . . . . . . . . . . . . . . . . . 34

**Chapter 8 Conclusion and Future Work** **37**

**Bibliography** **39**

**Vita** **44**

*In my beginning is my end. –T.S. Eliot,*

*Opening Lines of Four Quartets. "East Coker," pt. 1.*

# Chapter 1

# Introduction

In an unregulated network, if the users simultaneously demand more resources from the network than it has a capacity to offer, the network gets overloaded. This is termed as *congestion* and in extreme cases, can lead to a state of *congestion collapse* as reported by Nagle [24]. Informally, the classical congestion collapse is a stable condition in which an increase in network load results in a decrease in the useful work done by the network [13]. Floyd et al [13, 14] perceive a greater danger from a new form of congestion collapse due to bandwidth wasted by delivering packets through the network that are dropped before reaching their ultimate destination.

To prevent such a disastrous situation from occuring, most applications in the Internet implement some form of end-to-end congestion control, typically by using TCP. This requires each flow to adapt its sending rate based upon the available network bandwidth. The "available bandwidth" is inferred from some form of feedback given by the network indicating whether it is overloaded or has available capacity.

TCP-based end-to-end congestion control works because flows equitably share the bottleneck link bandwidth by adjusting their sending rates, while at the same time efficiently utilizing all of the available bandwidth. We say an end-to-end congestion control protocol is *feasible* if and only if it ensures convergence to fairness and efficiency, atleast in "ideal" conditions.

In this work, we look at one important subclass of end-to-end congestion control protocols called *Window based Binary Feedback Congestion Control Protocols* (WBF Protocols, from here on). TCP [28, 37], GAIMD [41], and Binomial congestion control [2] are some of the well known WBF protocols. These protocols allow the application to simultaneously send a bunch of packets, called a *window*. The size of the window represents the protocol's estimate of its share of the bottleneck link. At the heart of every WBF protocol is a *response function* $S_{\mathcal{P}}$, that periodically adjusts the window size after receiving feedback from the network. Typically, $S_{\mathcal{P}}$ depends both on the current window size, $w$, and the nature of the feedback. Equivalently, we can consider two different response functions, depending only on $w$: an *increase* response function $S_{\mathcal{I}}$ that is used when the feedback indicates available bandwidth and a *decrease* response function $S_{\mathcal{D}}$ corresponding to an overload or congestion indication. Different response functions give rise to different WBF protocols with different increase/decrease *policies*.

## 1.1 The Problem

Most of the classical Internet applications such as file transfer, electronic mail and the Web have *elastic* bandwidth requirements [9]: they perform better when more bandwidth is made available, but they can function with much lesser bandwidth. Hence, the ideal response function for these applications should grab available bandwidth quickly and also back off quickly to alleviate congestion. TCP is well suited for this kind of task.

However, several new applications have different needs and may need to respond to congestion indications differently. For example, multimedia applications require smoothness and cannot tolerate TCP's abrupt decrease-by-half response to a triple-duplicate-ACK congestion indication. New classes of protocols such as GAIMD and binomial congestion control with smoother increase/decrease policies have been invented for streaming media applications.

In this work, we seek a unifying framework that will help us choose (or invent) an appropriate *feasible* WBF unicast congestion control protocol to suit the needs of different applications.

## 1.2 The Solution

Given that a WBF protocol's properties are governed by its response functions, it is natural to ask "What response functions are feasible?". As our first contribution, we derive a sufficient condition that guarantees convergence to fairness and efficiency in a WBF protocol (under the assumption of synchronous feedback).

We then find that a surprisingly general class of WBF protocols satisfy this condition. In particular, we show that given a pair of real, differentiable, non-negative, monotonically non-decreasing functions $f(x)$ and $g(x)$[1], $g(x)$ upperbounded by 1, the following increase/decrease policy converges to fairness:

$$\begin{aligned} \mathcal{I}: x(t+R) &\leftarrow x(t) + x(t)/f(x(t)) \; ; \quad f(\cdot) > 0 \\ \mathcal{D}: x(t+R) &\leftarrow x(t) - x(t)g(x(t)) \; ; \quad 0 < g(\cdot) < 1 \end{aligned}$$

Here $x(t)$ represents the current window size and $x(t+R)$, the next window size.

An application can choose a slowly increasing function for $f(x)$ if it needs to 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. Because of this, we call these protocols CYRF[2], for *C*hoose *Y*our *R*esponse *F*unction.

However, because of the large install-base of TCP based applications, new congestion control protocols must also interact well with the large number of TCP flows that dominate the Internet [14]. Thus, we cannot (should not) choose a highly aggressive increase response function $S_{\mathcal{I}}$ and a very smooth decrease response function $S_{\mathcal{D}}$ at the same time. Most newly proposed protocols achieve this by adhering to the convention known as "TCP-Compatibility" [4] or "TCP Friendliness" [20, 21]. Roughly, it means that all flows must send at the same mean rate as a TCP flow. We derive a general condition that couples the increase and decrease response functions together so that a CYRF flow is TCP-friendly. Specifically, we show that for CYRF to be TCP-friendly in deterministic steady-state, we need $f(x)g(x) \propto x$.

From a protocol-design viewpoint, CYRF represents a novel approach in that it has been *designed* to be feasible, and its correctness need not be proved separately. CYRF is

[1] Atleast one must be monotonically increasing.

[2] To be pronounced as *sirf*, the motivation coming from the hinglish *"sirf ek protocol"*, which can mean *"CYRF: a protocol"* or *"Just a protocol"*.

shown to be feasible because each application of the increase or decrease policy moves the system closer towards total fairness.

We will also briefly consider a variant called 1-CYRF which relaxes this strict fairness criterion. Specifically, we drop the requirement that $g(x)$ be a monotonically non-decreasing function, and instead only ask $f(x)g(x)$ to be monotonically non-decreasing, and always greater than 1 for $x > 1$. Most CYRF protocols are also 1-CYRF. But if $g(x)$ is not monotonic, the application of a decrease policy will actually worsen fairness. This decrease in fairness is shown to be offset by the increase in fairness from the previous application of an increase policy. The name 1-CYRF comes from the fact that the condition $f(x)g(x) > 1$ corresponds to a 1-responsive protocol (section 3.3.4), which ensures that each decrease is preceded by an increase in window size (in deterministic steady-state). Thus the fairness of 1-CYRF increases over each application of an increase policy and we eventually obtain a totally fair allocation. The TCP-friendliness conditions apply to 1-CYRF as well as CYRF.

An interesting aspect of this work is that all commonly known WBF protocols, namely TCP, GAIMD and Binomial Congestion Control are shown to be 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 easily derive most of the important results about known WBF protocols as corollaries of the results for CYRF. For example, it directly follows that TCP, GAIMD and Binomial congestion control protocols converge to fairness and efficiency. It is well-known (due to a result of Chiu and Jain [6]) that GAIMD (and hence TCP) converges to fairness and efficiency. (Notice that this validates the correctness of our results in a way.) However, the proof of convergence to fairness of $n$-binomial flows is an important by-product of this work.

## 1.3 Organization of the Thesis

The rest of this thesis is organized as follows: In chapter 2, we briefly summarize some of the other approaches to tackling the problem of congestion and in particular other recent developments in TCP-friendly congestion control. Chapter 3 formalizes the WBF model and explains some of the simplifying assumptions used in our analysis. We then describe the constraints of fairness, efficiency and TCP-friendliness in chapter 4. Chapter 5 derives sufficient conditions for 2 and $n > 2$ WBF flows to converge to fairness which is used in Chapter 6 as the basis for CYRF. In chapter 7 we obtain simplified conditions for any WBF protocol to be TCP-friendly and then describe how CYRF can be made TCP-friendly. Chapter 8 concludes the work.

*If I have seen further, it is by standing upon the shoulders of giants.*

*–Sir Isaac Newton. Letter to Robert Hooke, Feb. 5, 1675.*

# Chapter 2

# Related Work

In this work, we do not look at the important issue of multicast flows and multicast congestion control, which has also benefited from the TCP-friendly idea. We also do not consider application-specific adaptive approaches such as [31] that try to make the best use of the available network support. Instead, our goal is to prescribe the best WBF congestion control that suits the needs of a particular application.

## 2.1 Historical Approaches

The main goal of a congestion control mechanism is preventing congestion collapse. There have been two main approaches to this problem. The first approach, used in ATM, switched telephone networks, and the ISPN architecture [5, 8], defines the notion of a flow [27] and calls for the deployment of packet scheduling disciplines to isolate each flow from the effects of other flows. Each flow must also reserve resources using a protocol such as RSVP [42] before using the resource, thus preventing overload.

The other major approach is end-to-end congestion control, which is based on the end-to-end argument [34] and supports one of the fundamental design goals of the DARPA Internet – survivability in the face of failure [7]. End-to-end policies can be divided into open loop admission control based schemes such as the leaky bucket algorithm [38] and closed loop feedback-based adaptive mechanisms such as TCP. TCP is the most widely used congestion control mechanism today. Most modern TCP implementations [39] incorporate algorithms introduced by Van Jacobson [1, 15] into 4.3BSD to fix the original 1988 congestion collapse.

Congestion avoidance via end-to-end congestion control was simultaneously investigated in a series of papers by Jain, Ramakrishnan and Chiu. Their work is summarized in [18]. In particular, Chiu and Jain characterize increase/decrease policies of a wide class of linear WBF protocols under the metrics of fairness, efficiency, etc. in [6]. This thesis uses several ideas from this famous paper including the metrics of fairness and efficiency, and the Chiu-Jain fairness index (section 4.1).

## 2.2 Recent Work on Congestion Control

The notion of TCP-friendliness [20, 21] has given rise to a number of new proposals for unicast end-to-end congestion control [2, 12, 32, 33, 36, 41]. The primary motivation for most of these new protocols is the transport of streaming multimedia which requires smooth changes in sending rate.

RAP [32] essentially proposes Additive Increase-Multiplicative Decrease (AIMD) where each data packet is acknowledged. Redundant information in the ACKs allows some ACKs to be lost. The ratio between short-term and long-term mean RTT is used to modify the inter-packet gap to provide fine-grained delay-based congestion avoidance. The Loss-Delay Adjustment Algorithm [36] uses feedback from RTP [35] for rate adjustment. The additive increase rate depends upon the ratio of the current sending rate and the bottlneck bandwidth, wheras the sending rate is decreased depending upon the number of packets lost.

GAIMD [18, 41] generalizes TCP to an AIMD policy with different increase and decrease parameters. It proposes that the current window $x(t)$ be changed to $x(t+R)$ using the increase/decrease policies:

$$\begin{aligned} \mathcal{I} : x(t+R) &\leftarrow x(t) + \alpha \; ; \qquad \alpha > 0 \\ \mathcal{D} : x(t+R) &\leftarrow x(t) - \beta x(t) \; ; \quad 0 < \beta < 1 \end{aligned} \tag{2.1}$$

with $\alpha = 3\beta/(2-\beta)$ for TCP-friendliness[11].[1] Binomial congestion control [2] proposes the

[1]This does not consider the effect of timeouts. [41] gives a slightly different condition for TCP-friendliness with timeouts.

following non-linear increase/decrease policies:

$$\begin{aligned}\mathcal{I} : x(t+R) &\leftarrow x(t) + \alpha/x(t)^k \ ; \quad \alpha > 0 \\ \mathcal{D} : x(t+R) &\leftarrow x(t) - \beta x(t)^l \ ; \quad 0 < \beta < 1 \end{aligned} \tag{2.2}$$

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 considered in section 4 of the Chiu-Jain paper [6].

Notice that plugging $\alpha = 1$ and $\beta = 1/2$ in the GAIMD equation (Equation 2.1) gives us TCP. GAIMD itself is a special case of the binomial algorithm with $k = 0$ and $l = 1$.

TFRC [12] is a rate-based scheme which directly uses the TCP throughput equation [26] to estimate its sending rate. The receiver sends periodic acknowledgements and estimates the loss rate. For greater stability, it ignores spurious packet losses by using a history discounting policy and by reporting a weighted average of the recent loss-event rate rather than the individual packets lost. The sender estimates the round-trip time and the appropriate value for the retransmit timeout-timer. Using these quantities, the expected sending rate is calculated using the TCP throughput equation.

TEAR [33] emulates the entire TCP state machine at the receiver. Using this, it maintains an exponentially weighted moving average of the congestion window size and calculates a TCP-friendly sending rate by dividing this by the estimated round trip time.

### 2.2.1 Comparison

Conceptually, it may be easy for some applications to specify the rate. For other applications it does not matter and may actually find it easier to implement application-specific quality adaptation using known window sizes - Suppose the application generates a data packet $P$ when the sender's buffer size is $B$ bytes and the window size is $W$. Assuming that the variations in $W$ are small (because of smoothness), a ballpark figure for the time it takes $P$ to reach the receiver is $BR/W$ where R is the RTT. An application with a maximum delay bound can use this information to adapt the quality of the data stream depending on the bandwidth available [23, 31].

From a congestion control viewpoint, the current window size is the instantaneous load on the network and is thus the quantity which must be regulated. Also, it is may be natural and easier to implement self-clocking [15] in window based approaches [16]. However, [3] which also shows the importance of self-clocking in dynamic situations, gives an ad-hoc mechanism to limit the TFRC sending rate following a loss-event, that performs well in the test-scenarios considered.

Floyd et al. [11], Yang et al. [40] and Bansal et al. [3] compare the performance of several window-based and rate-based protocol under various dynamic or transient conditions. This issue is not explored in our work.

An important distinction between CYRF and other approaches (including other window-based approaches) is the wide flexibility it offers to suit the needs of different applications. Some recent protocols, such as binomial and GAIMD offer a limited amount of flexibility. For example, the binomial algorithm allows the application to choose its increase response to be $S_{\mathcal{I}}(w) = \alpha/w^k$ for any $k$ and $\alpha > 0$. However, with CYRF, we have a richer class of increase response functions to choose from, namely, all differentiable, monotonically non-decreasing functions, thus providing a greater degree of control for the aggressiveness and responsiveness of the application.

*"Then you should say what you mean," the March Hare went on. "I do," Alice hastily replied; at least-at least I mean what I say-thats the same thing, you know." "Not the same thing a bit!" said the Hatter. "Why you might just as well say that 'I see what I eat' is the same thing as 'I eat what I see'!"*
*–Lewis Carroll. Alice's Adventures in Wonderland.*

# Chapter 3

# The Model

The rest of this work uses the WBF model and makes some assumptions to simplify analysis. These are explained below.

## 3.1 Notation Conventions

Following Chiu and Jain [6], 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. $\Delta x_i$ 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 $i$.

## 3.2 The WBF Model

WBF protocols allow a flow to send a bunch of packets, termed as a *window*, in each *round trip time* $R$. The network provides a *binary* feedback about whether this window overloaded the network or not. If the network was overloaded, the window for the next round trip is estimated according to some *decrease policy* $\mathcal{D}$. Otherwise, the window size is increased using an *increase policy* $\mathcal{I}$. (In this way, the WBF protocol continuously tries to "efficiently" make use of its "fair" share of the currently available network bandwidth.)

Thus, the size of the window is determined by the feedback from the network. The magnitude of the change in window size characterizes an increase/decrease policy $\mathcal{P}$. Typically, this change $S_{\mathcal{P}}$ is a function of the current window size and we call this the *response function* of the policy. Without loss of generality, $\mathcal{P}$ can be mathematically denoted as:

$$\mathcal{P} : x(t+R) \leftarrow x(t) + S_{\mathcal{P}}(x(t))$$

Here $x(t)$ represents the current window size and $x(t+R)$ is the next window size, after a round-trip time $R$. Note that the pair of increase and decrease response functions $S_{\mathcal{I}}(x)$ and $S_{\mathcal{D}}(x)$ completely characterize a WBF protocol.

### 3.2.1 Example

In congestion avoidance mode, each TCP flow increases its window size by 1 packet if the network is not overloaded (i.e. the whole window is acknowledged). Otherwise, it reduces its window size to half the previous value (if it gets a triple duplicate ACK). Without timeouts, the increase and decrease policies are:

$$\begin{aligned} \mathcal{I} : x(t+R) &\leftarrow x(t) + 1 \\ \mathcal{D} : x(t+R) &\leftarrow x(t) - \frac{x(t)}{2} \end{aligned} \tag{3.1}$$

where x(t) is the window size of a flow at time t and R the round trip time of the flow. Thus TCP is a WBF protocol with the response functions: $S_{\mathcal{I}}(x) = 1$ and $S_{\mathcal{D}}(x) = -x/2$. Note that actual implementations increase the window by $1/x$ on the receipt of each acknowledgement instead of increasing by 1 upon getting a window's worth of ACKs [37].

■

## 3.3 Assumptions

In the analysis that follows, we make some simplifying assumptions. Specifically, we assume *saturated senders* in *steady state* with a *continuous fluid approximation* of the *binary feedback model* with *synchronous feedback*. Each of these terms is explained below.

### 3.3.1 Binary feedback model

This model assumes that each flow gets a binary feedback from the network indicating whether it is overloaded or if there is additional available bandwidth (1 = overloaded, 0 = underloaded). This feedback can be implicit, for example, through losses detected by timeouts, or explicit, for example, through a "congestion experienced" bit in an ECN aware network [29, 30]. 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.

This model is natural for a truly distributed system like the current Internet. Without explicit co-operation by the users or information from the network about the fair share of each flow, this model is the only practical solution. Thus our model does not consider schemes such as Explicit Window Adaptation [19] which requires per-flow network feedback.

### 3.3.2 Synchronous feedback

The model follows Chiu and Jain's assumption [6] that all the flows in the network get the *same feedback* and get this feedback *simultaneously*.

While this does not model the real Internet by any stretch of imagination, our analysis becomes simpler, and the proof of convergence to fairness under these "ideal" conditions provides a good intuitive understanding about the fairness properties of WBF protocols in more general situations.

### 3.3.3 Saturated senders

This assumes that the sender's window size is limited by the network, and not by the receiver's window or the amount of outstanding sender's data. Thus the equations for $\mathcal{I}$ and $\mathcal{D}$ completely determine the window size and the instantaneous load on the network.

### 3.3.4 Steady-state and 1-responsiveness

Most analyses implicitly assume that each flow applies the increase policy $\mathcal{I}$ several times until a network feedback indicates overload, at which time the decrease policy $\mathcal{D}$ is applied and the cycle starts all over again. We call this the deterministic steady-state, or in short, steady-state, and each such cycle is called a *"congestion epoch"*. If an increase/decrease policy had to be applied $n$ times, the epoch is said to be of size $n$.

Typically (for example, in [2, 11]), it is assumed that the decrease policy needs to be applied exactly once by all flows, at the end of the epoch. With the synchronous feedback assumption, this means that for each flow, the decrease in window size from a single application of $\mathcal{D}$ must atleast 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:

$$|\Delta(w_{\mathcal{I}})| \leq |\Delta(w_{\mathcal{D}})| \tag{3.2}$$

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 3.2 for sufficiently large window sizes as 1-*responsive* to distinguish them from $k$-*responsive* protocols which require $k > 1$ applications of $\mathcal{D}$ in a congestion epoch. An interesting consequence of 1-responsiveness is that each application of a decrease policy is preceded by atleast one application of an increase policy. This is used in section 6.3 and Corollary 5.2. In the following, unless otherwise stated, the protocols considered are 1-responsive.

Most interesting WBF protocols are 1-responsive. From Equations 3.1, 2.1 and 2.2, we can easily see that TCP, GAIMD and the TCP-friendly version of Binomial Congestion Control follow this weak criterion in general.[1] Thus, this is not a very restrictive assumption, and it makes modeling and mathematical analyses simple. However, it is not essential for protocol correctness or performance reasons.

### 3.3.5 Continuous fluid approximation

In the real world, the increase and decrease policies are applied once per round trip time. Because of the self-clocking property of TCP [15], this means that the window size is updated upon receiving an ACK in most implementations. Here we assume that this happens as a continuous process. Formally, the discrete time process $(W_n)_{n=0}^{\infty}$ denoting the consecutive window values is approximated by the continuous time process $W(t)_{t=0}^{\infty}$ [25].

[1]The minimum possible window size is 1 and for this window, the assumption fails. But we assume a congestion epoch with a sufficiently large minimum window size.

### 3.3.6 Congestion avoidance mode

Many WBF protocols such as TCP operate in different modes at different times. The increase/decrease policy used varies depending on the current window size, recent losses experienced etc. In our analysis, we assume that such protocols are operating in a "congestion avoidance" mode (similar to TCP's congestion avoidance mode), with a fixed increase/decrease policy. TCP SACK [22] approaches this ideal condition.

*The shallow consider liberty a release from all law, from every constraint. The wise man sees in it, on the contrary, the potent Law of Laws. – Walt Whitman, "Freedom," Notes Left Over (1881)*

# Chapter 4

# Constraints on Feasible WBF Protocols

We restrict the possible choices for WBF protocols in the above model by imposing two conditions that must hold eventually in steady state, namely fairness and efficient utilization of bottleneck link capacity. We call protocols that satisfy these conditions as *feasible* [6]. Only feasible protocols can be safely deployed in any arbitrary network.

## 4.1 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 [17] F given by

$$\mathrm{F} = \frac{(\sum x_i)^2}{n(\sum x_i^2)} \tag{4.1}$$

This index has the following interesting properties some of which we will use in the proofs that follow:

- F is upperbounded by 1, and this upperbound is reached when the allocation is totally fair ($x_1 = x_2 = \ldots = x_n$).
- F is lowerbounded by 0, which is the index for a totally unfair allocation (one flow gets the whole link bandwidth).
- If only k of n users equally share the resource, then F = $k/n$.
- F is a continuous differentiable function for all $x_i$.
- F is independant of scale — the unit of measurement chosen for $x_i$'s does not matter.

Since F = 1 only when $x_1 = x_2 = \ldots = x_n$, the fairness requirement that eventually a totally fair allocation is reached can be restated as the following mathematical condition:

$$\mathrm{F} \rightarrow 1 \tag{4.2}$$

Our model considers all flows to be equal. Thus schemes such as MulTCP [10] which simulate priority by giving a preferred flow $n$ times the bandwidth of an ordinary TCP flow do not fall into this framework.

## 4.2 Convergence to Efficiency

Ideally, the bottleneck link must never be underutilized. However, overloading the link is just as unacceptable. In feedback based protocols, in steady state, the system oscillates between overload and underload, relying on the feedback mechanism to get back in place. Thus 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 [6].

An easy way to achieve this is to have each flow increase its window size when the bottleneck link is under-utilized (i.e., network feedback is 0), and decrease its window size when the bottleneck link is overloaded (when the feedback is 1). We call WBF protocols which behave in this manner as *efficient*.

*"Would you tell me, please, which way I ought to go from here?" "That depends a good deal on where you want to get to," said the Cat. "I don't much care where-" said Alice. "Then it doesn't matter which way you go," said the Cat. "As long as I get somewhere," Alice added as an explanation. "Oh, you're sure to do that," said the Cat, "if you only walk long enough." – Lewis Carroll, Alice's Adventures in Wonderland.*

# Chapter 5

# Conditions for Convergence to Fairness

As mentioned in Section 4.1, the requirement for convergence to fairness essentially means that assuming synchronous feedback from the network, the steady state window sizes of all flows sharing a bottleneck link must eventually move in lockstep with each other. In this chapter we derive a sufficient condition for this to happen for both $n = 2$ and $n > 2$ flows. We find that TCP, GAIMD and binomial congestion control satisfy these conditions which again confirms that these protocols converge to fairness.

## 5.1 The Two-Flow Case

In this section we obtain the intuitive result that if the smaller of two windows increases more than the larger one (relative to their window sizes), then the window sizes move closer to equality. Similarly, when the window sizes decrease, the smaller window must decrease less than the larger one.

To do this, we use the fact that F is upperbounded by 1 and that F = 1 represents a totally fair allocation: Supposing two flows $x_1$ and $x_2$ share a bottleneck link. Let $\Delta\mathrm{F}$ be the change in F corresponding to a small change $\Delta x_1$ in $x_1$ and $\Delta x_2$ in $x_2$. If each $\Delta\mathrm{F}$ is positive at each application of an increase/decrease policy, then eventually $\mathrm{F} \to 1$ regardless of its initial value, satisfying Equation 4.2. Notice that this technique works with any function of $x_1$, $x_2$ which has a known limiting value, and the property that $x_1 = x_2$ when the limit is reached. Thus the proof is independant of the fairness index F that is chosen, as long as the index has the properties we require.

We make use of the continuous fluid approximation in this and the next section. Specifically, we assume that the changes $\Delta x_1$ and $\Delta x_2$ represent infinitesimal changes to $x_1$ and $x_2$ so that

$$dx_1 \approx \Delta x_1,\ dx_2 \approx \Delta x_2 \text{ and } d\mathrm{F} \approx \Delta\mathrm{F} \tag{5.1}$$

**Theorem 5.1 (Inverse Proportional Change to Fairness)** *Two flows sharing a bottleneck link will eventually converge to and maintain a totally fair allocation of bottleneck link bandwidth if at each application of an increase or decrease policy, the proportional change in the larger quantity is lesser than that in the smaller quantity.*

*Mathematically, for two flows with window sizes $x_1$ and $x_2$, $x_1 < x_2$, satisfying the following condition at each application of $\mathcal{I}$ or $\mathcal{D}$ ensures convergence to fairness:*

$$\frac{\Delta x_1}{x_1} \geq \frac{\Delta x_2}{x_2} \tag{5.2}$$

*Atleast one of $\mathcal{I}$ or $\mathcal{D}$ must ensure a strict inequality.*

**Proof:** For $i = 2$, Equation 4.1 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 use of Equation 5.1 we get

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

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

$$(x_1^2 + x_2^2)(\Delta x_1 + \Delta x_2) \geq (x_1 + x_2)(x_1\Delta x_1 + x_2\Delta x_2) \tag{5.3}$$

$$\Rightarrow x_2(x_2 - x_1)\Delta x_1 \geq x_1(x_2 - x_1)\Delta x_2 \tag{5.4}$$

By hypothesis $x_2 - x_1 > 0$, so we can write

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

Note that we need atleast one of $\mathcal{I}$ or $\mathcal{D}$ to ensure $\Delta\mathrm{F} > 0$ so that F increases over each congestion epoch (section 3.3.4) and eventually becomes 1. Thus atleast one of them must have a strict inequality in Equation 5.2.

Also, once $x_1 = x_2$, this equality is maintained under synchronous feedback: Assuming that $S_{\mathcal{I}}$ depends only on the current window size, at each application of $\mathcal{I}$, $S_{\mathcal{I}}(x_1) = S_{\mathcal{I}}(x_2)$ since $x_1 = x_2$ so that the window sizes after the application of $\mathcal{I}$, given by $x_1 + S_{\mathcal{I}}(x_1)$ and $x_2 + S_{\mathcal{I}}(x_2)$, are also equal. Similarly, $S_{\mathcal{D}}(x_1) = S_{\mathcal{D}}(x_2)$ so that the values of the window sizes will move in lockstep with each other after an application of $\mathcal{D}$ also. ■

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 5.1. Thus the above proof applies to these protocols even though the changes $\Delta x_1$ and $\Delta x_2$ 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 [6] 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 5.1** *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 5.2 becomes: $1/x_1 > 1/x_2$ if $x_1 < x_2$. Similarly $\Delta x = -x/2$ for the decrease policy and Equation 5.2 reduces to $-(1/2) = -(1/2)$.

For GAIMD, $\Delta x = \alpha$ for the increase policy and Equation 5.2 becomes: $\alpha/x_1 > \alpha/x_2$ if $x_1 < x_2$. Similarly $\Delta x = -\beta x$ for the decrease policy and Equation 5.2 reduces to $-\beta = -\beta$. ■

For Binomial congestion control, $\Delta x = \alpha/x^k$ for the increase policy and Equation 5.2 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 5.2 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 2.2, 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 5.2** *Binomial congestion control converges to fairness if $k, l \geq 0$.*

**Proof:** Clearly, binomial congestion control satisfies the 1-responsiveness criterion (Equation 3.2) 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 to be followed by an application of the decrease policy. It is sufficient to show that the fairness index increases over this sequence of window size adjustments since we already know that sequences consisting only of applications of increase policy improve the fairness index if $k > -1$.

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 $0 \leq l < 1$. ■

## 5.2 The N-Flow Case

In the previous section, we placed a restriction on the number of flows. In this section, we will derive a straightforward generalization of Theorem 5.1 for $n > 2$ flows also.

**Theorem 5.2 ($n$-flow convergence to fairness)** *$n$-flows converge to fairness if at each application of an increase/decrease policy we have,*

$$\sum_{i=1}^{i=n} x_i^2 \sum_{i=1}^{i=n} \Delta x_i \geq \sum_{i=1}^{i=n} x_i \sum_{i=1}^{i=n} x_i \Delta x_i \qquad (5.5)$$

*Atleast one of the two policies should ensure a strict inequality.*

**Proof:** Using

$$d\mathrm{F} = \sum_{i=1}^{i=n} \frac{\partial \mathrm{F}}{\partial x_i} dx_i$$

we get, from Equation 4.1 and the approximation 5.1,

$$\Delta \mathrm{F} = \frac{2 \sum_{i=1}^{i=n} x_i}{n \left( \sum_{i=1}^{i=n} x_i^2 \right)^2} \left( \sum_{i=1}^{i=n} x_i^2 \sum_{i=1}^{i=n} \Delta x_i - \sum_{i=1}^{i=n} x_i \sum_{i=1}^{i=n} x_i \Delta x_i \right)$$

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

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

Notice that this reduces to Equation 5.3 for $n = 2$. ■

**Corollary 5.3** *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 5.5.

  In this case $\Delta x_i = \alpha$. Since F is upperbounded by 1, we get (from Equation 4.1)

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

  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 $\alpha$,

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

  which is Equation 5.5 with $\Delta x_i = \alpha$.

- Case 2: The decrease policy satisfies Equation 5.5

  In this case $\Delta x_i = \beta x_i$. Equation 5.5 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 5.5 and hence converge to fairness. The proof sketch is very similar to the proof for Theorem 6.2. However, this result is easily obtained in section 6.4 as a special case of Theorem 6.2. Thus we have:

**Corollary 5.4** $n > 2$ *binomial flows converge to fairness.*

*You perceive I generalize with intrepidity from single instances. It is the tourist's custom. –Mark Twain*

*"When I use a word," Humpty Dumpty said, in a rather scornful tone, "it means just what I* choose *it to mean, neither more nor less." – Lewis Carroll, Alice's Adventures in Wonderland.*

# Chapter 6

# CYRF: A Generalized WBF Protocol

Equation 5.2 gives a *sufficient* condition for two flows sharing a bottleneck link to converge to fairness. In other words, two flows of *any* protocol with increase/decrease policies that satisfy Equation 5.2 are guaranteed to converge to fairness.

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 5.2?". This yields a new family of congestion control protocols that are *designed* to converge to fairness and efficiency when there are 2 flows sharing a bottleneck link. We then show that even for $n > 2$ flows, these protocols converge to fairness.

We show that this class of protocols can be used as transport for applications with a wide range of aggressiveness and smoothness needs. Because the application can *choose* from a wide range of response functions to congestion indications, we call this class of protocols CYRF, which stands for $C$hoose $Y$our $R$esponse $F$unction.

Finally, we will briefly consider a variant of CYRF and show how many WBF protocols are actually part of a single family of protocols, thus providing a powerful framework for their analysis.

## 6.1 $f(\cdot), g(\cdot)$ Congestion Control

In this section, we derive a general class of functions for which Equation 5.2 applies. Assuming that $\Delta x$ is some function of x, we can see that Equation 5.2 implies some kind of monotonicity for $\Delta x$. Also, this function must be continuous and differentiable for Equation 5.2 to apply. Furthermore, for convergence to efficiency, the principle of negative feedback discussed in section 4.2 must be satisfied. Thus $\Delta x$ must be positive for the increase policy and negative for the decrease policy. These requirements are expressed in the following theorem.

**Theorem 6.1** *Let $f(x)$ and $g(x)$ be any (continuous) differentiable non-negative monotonically non-decreasing functions (atleast one of them must be strictly increasing) for $x \geq 1$. Also, let $g(x)$ be upperbounded by $1$*[1]*. Then the following increase and decrease policies ensure converge to fairness and efficiency for two flows sharing a bottlneck link (under assumption of synchronized feedback). We call this CYRF:*

$$\begin{aligned} \mathcal{I} : x(t+R) &\leftarrow x(t) + x(t)/f(x(t)) \ ; \quad f(x(t)) > 0 \\ \mathcal{D} : x(t+R) &\leftarrow x(t) - x(t)g(x(t)) \ ; \quad 0 < g(x(t)) < 1 \end{aligned} \tag{6.1}$$

**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 5.2:

$$\frac{1}{f(x_1)} \geq \frac{1}{f(x_2)}$$

and similarly for the decrease policy,

$$-g(x_1) \geq -g(x_2)$$

[1] *The upperbound ensures that the decrease policy $\mathcal{D}$ does not lead to negative window sizes*

Note that since atleast one of the two functions is strictly increasing, we have a strict inequality in one of the above two cases as required by Theorem 5.1.

**Convergence to efficiency**: For CYRF to be efficient (Section 4) each flow should increase its window if the network feedback is 0 and decrease its window if the feedback is 1. Clearly, for all window sizes, $x(t) \geq 1$ and $\Delta x$ is positive for the increase policy:

$$x(t)/f(x(t)) \geq 0 \; ; \qquad \text{if } f(x(t)) > 0$$

and negative for the decrease policy:

$$-x(t)g(x(t)) \leq 0 \; ; \qquad \text{if } 0 < g(x(t)) < 1$$

■

CYRF represents a wide class of WBF protocols. Applications can choose different $f(x)$ and $g(x)$ to get different WBF protocols. For example, an application can choose a slowly increasing function for $f(x)$ if it needs to 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. It is this flexibility that distinguishes CYRF from other approaches.

## 6.2 $N$-flow Convergence to Fairness

CYRF was *designed* to converge to fairness for the two-flow case. For $n > 2$ flows, we can consider flows pairwise and apply the argument of Theorem 6.1 to show convergence to fairness. Alternatively, we can consider all the $n$-flows together, and show that the fairness index of the whole system increases and thus provide a straightforward proof of convergence to fairness. Thus we get the following theorem:

**Theorem 6.2** *With synchronized feedback, 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 5.5. 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 5.5 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 6.1 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 5.5 is satisfied with $n \leq k$ flows. Here, $\Delta x = x/f(x)$. So we get:

$$\sum_{i=1}^{k} x_i^2 \sum_{i=1}^{k} \frac{x_i}{f(x_i)} \geq \sum_{i=1}^{k} x_i \sum_{i=1}^{k} \frac{x_i^2}{f(x_i)} \tag{6.2}$$

  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

$$X \sum_{i=1}^{k} \frac{X x_i - x_i^2}{f(x_i)} \geq \frac{X}{f(X)} \sum_{i=0}^{k} (X x_i - x_i^2) \tag{6.3}$$

  Rewriting, we get

$$X^2 \sum_{i=1}^{k} \frac{x_i}{f(x_i)} + \frac{X}{f(X)} \sum_{i=1}^{k} x_i^2 \geq X \sum_{i=1}^{k} \frac{x_i^2}{f(x_i)} + \frac{X^2}{f(X)} \sum_{i=1}^{k} x_i \tag{6.4}$$

  Adding Equation 6.2 and Equation 6.4,

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

  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 5.5 for $n = k+1$. Hence, by the principle of mathematical induction, Equation 5.5 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 by Theorem 5.2. ■

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

## 6.3 1-CYRF: Generalizing CYRF

In section 5.1, it was shown that the decrease policy of binomial congestion control may actually worsen the fairness index for $0 \leq l < 1$. However, Corollary 5.2 showed that it can still converge to fairness under suitable conditions. In this section, we generalize this corollary for CYRF and obtain another class of protocols, called 1-CYRF because they are required to be 1-responsive (section 3.3.4).

Specifically, suppose we drop the constraint that $g(x)$ should be monotonically non-decreasing without changing the requirements on $f(x)$. Now, each application of the increase policy will still increase the fairness index. However, the decrease policy can now *worsen* fairness.

However, 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 $\mathcal{D}$. Further, we enforce the 1-responsiveness condition. This will ensure that each application of a decrease policy is preceded by atleast one increase so that in deterministic steady-state, F will still increase over each congestion epoch.

Clearly, the 1-responsiveness condition (Equation 3.2) is satisfied if $f(x)g(x) \geq 1$ for sufficiently large window sizes. An appliation of $\mathcal{I}$ followed by an application of $\mathcal{D}$ increases fairness if Equation 5.2 is satisfied. Here $\Delta x = x/f(x) - xg(x + x/f(x)) \approx x(1/f(x) - g(x))$. To satisfy Equation 5.2, we must have

$$1/f(x_1) - g(x_1) \geq 1/f(x_2) - g(x_2)$$

if $x_1 \leq x_2$. Notice that if $f(x)g(x)$ is a monotonically non-decreasing function, then

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

is a monotonically non-increasing function and thus the inequality required above will be automatically satisfied. Again, atleast one of $f(x)$ or $f(x)g(x)$ must be strictly increasing according to the requirements of Theorem 5.1. Thus we are led to the following result:

**Theorem 6.3** *Let $f(x)$ and $g(x)$ be any (continuous) differentiable non-negative functions for $x \geq 1$. Also, let $f(x)$ and $f(x)g(x)$ be monotonically non-decreasing and let $g(x) \leq 1$ and $f(x)g(x) \geq 1$. Then the following increase and decrease policies ensure converge to fairness and efficiency for two flows sharing a bottleneck link:*

$$\begin{aligned} \mathcal{I} : x(t+R) &\leftarrow x(t) + x(t)/f(x(t)) \;; \quad f(x(t)) > 0 \\ \mathcal{D} : x(t+R) &\leftarrow x(t) - x(t)g(x(t)) \;; \quad 0 < g(x(t)) < 1 \end{aligned} \tag{6.5}$$

It can also be shown that 1-CYRF converges to fairness for $n$-flows. The proof is very similar to Theorem 6.2. 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 6.4** *With synchronized feedback, 1-CYRF converges to fairness for $n > 2$ flows.*

## 6.4 1-CYRF as a Unified Framework for WBF

Notice that most CYRF protocols are also 1-CYRF protocols – if $f(x)$, $g(x)$ are monotonically non-decreasing, so is their product $f(x)g(x)$. We only require 1-responsiveness, $f(x)g(x) \geq 1$, which may not be satisfied by some CYRF protocols.

Also, by substituting $f(x) = x$, and $g(x) = 1/2$ in Equation 6.1, we get Equation 3.1. Thus TCP is a special case of CYRF. Observe that $f(x) = x$ and $g(x) = 1/2$ are both differentiable monotonically non-decreasing functions and $g(x)$ is upperbounded by 1 as required. Similarly, we can show that GAIMD is a special case of CYRF ($f(x) = x/\alpha$, and $g(x) = \beta$). Binomial congestion control ($f(x) = x^{k+1}/\alpha$, and $g(x) = \beta x^{l-1}$) is an example of a 1-CYRF protocol that is not CYRF.

This provides us with a powerful framework for analyzing WBF protocols. For example, Corollaries 5.1, 5.3 follow directly as special cases of Theorems 6.1, 6.2 respectively. In fact, Theorem 6.4 also implicitly proved that $n$ binomial flows converge to fairness although Corollary 5.4 did not explicitly attempt to prove this.

*However feeble the sufferer and however great the oppressor, it is in the nature of things that the blow should recoil upon the aggressor.*
*– Ralph Waldo Emerson*

# Chapter 7

# TCP-Friendly CYRF

While Chapter 6 showed that any number of competing CYRF flows sharing a bottleneck link will converge to fairness, this is not sufficient in a heterogeneous Internet where the other competing flows may be non-CYRF flows, or even CYRF with the different $f(x)$ and $g(x)$. In this chapter, we derive a simple condition for a smooth CYRF flow to be TCP-friendly. As discussed below, TCP-friendliness can be regarded as a weaker fairness constraint, and also as an essential condition for deployment in the heterogeneous Internet. In particular, smooth protocols are by definition slowly responsive to congestion indications. TCP-friendliness is a way of limiting the aggressiveness of such flows so that the other flows are not hurt by an over-aggressive and non-responsive flow.

## 7.1 TCP-Friendliness

Several new congestion control protocols have been motivated by the TCP-friendliness argument. Previously, the fairness requirement dictated that all flows in a network should use the same WBF protocol, which by default, was TCP. TCP-friendliness is a relaxation of this constraint. It allows non-TCP applications as long as they use the "TCP-friendly equation," (that is, they maintain the arrival rate to at most some constant $c$, over the square root of the packet loss rate $p$) [21]. Since TCP's throughput is proportional to the $1/\sqrt{p}$ [26], TCP-friendly flows with a suitable constant $c$ send roughly the same amount of data as a conformant TCP flow under comparable conditions (same RTT, MTU etc). In this chapter, we will occasionally leave the constant unspecified.

This criterion that no flow should send more than a comparable conformant TCP flow is called as TCP-compatibility [4]. Thus TCP-friendliness can be regarded as a way of enforcing the TCP-compatibility requirement. TCP-friendliness is seen as being essential for several reasons:

- TCP-friendlines ensures a weak fairness bound for same-protocol flows —If a protocol is TCP-friendly, then all flows send at some constant times the rate of TCP, and thus roughly send the same amount of data in a sufficiently long time-frame.
- By the same argument, different-protocol flows sharing a bottleneck link can be fair to each other. This distinguishes TCP-friendliness from the previous fairness constraint.
- It ensures that the currently dominant transport protocol over the Internet, TCP, does not suffer as a result of newer protocols being deployed.
- This does not require any change to the existing network architecture, and can be easily deployed since it is an end-to-end mechanism.

We now obtain an alternative characterization of TCP-friendliness in steady-state.

**Lemma 7.1 (TCP-Friendliness Criterion)** *A 1-responsive protocol with a congestion epoch of size $n$ during which $S_n$ packets are sent is TCP-friendly in deterministic steady-state if:*

$$\frac{n^2}{S_n} = \frac{2}{3} \tag{7.1}$$

**Proof:** First, we will show that if $n^2/S_n = c$ for some constant $c$, then the throughput $T$ is inversely proportional to $1/\sqrt{p}$. We will then use the fact that TCP itself is TCP-friendly and deduce that $c = 2/3$.

Suppose the packet size is B and the steady-state (or average) round-trip time is R. Then the (long-term) throughput is given by $T = S_nB/nR$. Since $B$ and $R$ are constant for a given flow, if $n \propto \sqrt{S_n}$ we have that

$$T \propto \sqrt{S_n}$$

But by definition, the loss rate $p$ is given by $p = 1/S_n$. Rewriting the above we get

$$T \propto 1/\sqrt{p}$$

Thus if

$$\frac{n^2}{S_n} = c \tag{7.2}$$

then the flow is TCP-friendly.

To get the proportionality constant, we just need to plug in the values of $n$ and $s_n$ for some TCP-friendly protocol. In particular, we know that TCP itself is TCP-friendly. Suppose the maximum window size in a TCP congestion epoch is $W_{TCP}$. From Equation 3.1 we can see that the successive window sizes during a sequence of applications of the increase policy form an arithmetic series with a term difference of 1. Thus we have

$$W_{TCP} = W_{TCP}/2 + (n - 1)$$

This gives us

$$n \approx W_{TCP}/2$$

We also get the number of packets sent as the sum of the series:

$$S_n = \frac{n}{2}\left(2\frac{W_{TCP}}{2} + (n-1)\cdot 1\right) \approx \frac{3W_{TCP}^2}{8}$$

Plugging these values into Equation 7.2, we get

$$\frac{n^2}{S_n} = \frac{2}{3}$$

for TCP-compatibility. ■

By a very similar argument, we can show that a $k$-responsive protocol is TCP-friendly if

$$\frac{kn^2}{S_n} = \frac{2}{3} \tag{7.3}$$

In this proof, we made extensive use of the fact that that within a single congestion epoch, the successive window sizes of a TCP flow form an arithmetic series. Next we show that we can say the same for any smooth flow. This result will be useful in showing that an arbitrary smooth protocol confirms to the TCP-friendly criterion.

## 7.2 Smoothness

TCP-friendliness was originally motivated by the requirements of multimedia flows which cannot tolerate TCP's abrupt decrease-by-half of the sending rate. A TCP-friendly flow can decrease its sending rate by a lesser amount and also increase its sending rate more slowly in such a way that its mean throughput over a sufficiently large time-scale is roughly the same as that of TCP.

Thus a TCP-friendly flow usually has *smoothness* as an additional criterion. Intuitively, the smoother a flow is, the lesser will be its decrease in response to a congestion indication from the network. Smoothness has been variously measured by the largest reduction of the sending rate in one round-trip time in the deterministic steady-state scenario (Floyd et al. [11]) and the coefficient of variation of the time-series representing successive window sizes (Yang et al. [40]). In this work, we look at smoothness as an (optional) characteristic of a WBF protocol rather than as a metric.

We say a WBF protocol is *smooth* if its increase and decrease policies are smooth. An window increase or decrease policy $\mathcal{P} : x_{t+R} \leftarrow x_t + \Delta x$, where $x$ is the window size, is said to be smooth if

$$|\Delta x| \ll x \tag{7.4}$$

As we have noted before, typically, $\Delta x$ is a function of $x$.

An interesting aspect of smooth WBF protocols is that their successive window sizes can be approximated by an arithmetic series.

**Lemma 7.2 (Smoothness Lemma)** *The window sizes corresponding to successive applications of a smooth increase/decrease policy $\mathcal{P}$ form an arithmetic series (approximately).*

**Proof:** Suppose the policy $\mathcal{P}$ is successively applied several times as follows:

$$\begin{aligned} x_{t+R} &\leftarrow x_t + \Delta x_t \\ x_{t+2R} &\leftarrow x_{t+R} + \Delta(x_{t+R}). \\ &\vdots \end{aligned}$$

Because $|\Delta x| \ll x$ and $|\Delta(x_{t+R})| \ll x_{t+R}$, we can write

$$x_{t+2R} \approx x_t + 2\Delta(x_t)$$

or in general,

$$x_{t+nR} \approx x_t + n\Delta(x_t) \tag{7.5}$$

for $n$ successive applications of $\mathcal{P}$ ($n$ being of atmost the order of $x$ so that the errors dont add up significantly). ■

Note that GAIMD's decrease policy does not satisfy $|\Delta x| \ll x$. However, the relation (7.5) can still be applied to the increase policy. In fact, because $\Delta x$ is constant, we have the following for both TCP and GAIMD's increase policies:

$$x_{t+nR} = x_t + n\Delta(x_t) \quad \text{(GAIMD, TCP)} \tag{7.6}$$

## 7.3 CYRF with the TCP-Friendliness Constraint

In this section, we will use the above results to derive the conditions for a CYRF flow to be TCP-friendly. We will also obtain, as special cases of this, the previously known conditions for GAIMD and Binomial Congestion Control protocols to be TCP friendly.

The following theorem gives a relation between $f(x)$ and $g(x)$ so that the TCP-friendliness condition (Equation 7.1) is satisfied. We use the smoothness criterion (Equation 7.4) and the Smoothness Lemma (Lemma 7.2) to simplify the analysis.

**Theorem 7.1** *In steady-state, smooth 1-responsive CYRF protocols are TCP-friendly if:*

$$f(x)g(x) \propto x \tag{7.7}$$

*and strictly TCP-compatible if:*

$$f(X) = \frac{X(2 - g(X))}{3g(X)} \tag{7.8}$$

**Proof:** The proof proceeds as follows: We invoke the Smoothness Lemma and approximate the successive window sizes during the application of the increase policy of Equation 6.1 by an arithmetic series. Then we find $n$, the size of the congestion epoch, as the number of terms in this series, and $S_n$, the number of packets sent during this epoch, as the sum of the series. Using this in Equation 7.1, we get the desired result. Along the way, we make use of the approximation $g(x) \ll x$, which holds because of the smoothness of the protocol.

Suppose $X$ is the maximum window size, just before the application of $\mathcal{D}$. The minimum window size, just after the application of $\mathcal{D}$ is also the initial window size, $x$, because of the steady-state condition.

$$x = X - Xg(X) \tag{7.9}$$

To find n, we write

$$X = x + (n-1)x/f(x) \tag{7.10}$$

Because of smoothness, we can approximate

$$\frac{x}{f(x)} = \frac{X(1-g(X))}{f(X(1-g(X)))} \approx X/f(X) \tag{7.11}$$

Plugging into Equation 7.10 we get

$$n = f(X)g(X) + 1 \approx f(X)g(X) \tag{7.12}$$

Similarly we find the number of packets sent as:

$$\begin{aligned} S_n &= \frac{n}{2}\left(2(X - Xg(X)) + f(X)g(X)\cdot\frac{X}{f(X)}\right) && (7.13)\\ &= \frac{n}{2}X(2-g(X)) && (7.14)\end{aligned}$$

Substituting from Equations 7.12, 7.13 in Equation 7.1 we get the TCP-compatibility condition as

$$\frac{2(1+f(X)g(X))}{X(2-g(X))} = \frac{2}{3}$$

Simplifying,

$$f(X) = \frac{X(2-g(X))}{3g(X)}$$

which is the condition for strict TCP-compatibility.

If $g(X) \ll 2$ we can write

$$f(X)g(X) = X/3 \tag{7.15}$$

Hence we can get the result

$$f(X)g(X) \propto X$$

which ensures TCP-friendliness.

We have implicitly assumed in Equation 7.9 that this protocol is 1-responsive. Observe that if the window size x is greater than 3, Equation 7.15 does indeed ensure that the following 1-responsiveness condition for CYRF holds:

$$f(x)g(x) \geq 1 \tag{7.16}$$

■

Although GAIMD is not a smooth protocol, we have noted previously that the smoothness lemma applies to GAIMD also. Similarly observe that the approximation in Equation 7.11 also holds for GAIMD. Thus Equation 7.8 holds for GAIMD (and hence TCP) also. Substituting $f(x) = x/\alpha$, and $g(x) = \beta$, we get the following result:

**Corollary 7.1** *GAIMD is TCP-compatible if*

$$\alpha = \frac{3\beta}{(2-\beta)} \tag{7.17}$$

This is a simplified condition for GAIMD to be TCP-compatible ignoring the effect of timeouts. A slightly different proof of this result is given by [11]. Again, this verifies Equation 7.8 in a way. [41] gives a different condition taking timeouts into consideration.

Similarly, substituting $f(x) = x^{k+1}/\alpha$, and $g(x) = \beta x^{l-1}$ in Equation 7.7, we see that binomial congestion control will be TCP-friendly if $x^{k+l} \propto x$. Thus we have:

**Corollary 7.2** *Binomial Congestion Control is TCP-friendly if* $k + l = 1$.

This is called the $k + l$-rule in the original binomial congestion control paper [2].

*What we call the beginning is often the end*
*And to make an end is to make a beginning.*
*The end is where we start from.*
*– T.S. Eliot, Little Gidding (Four Quartets)*

# Chapter 8

# Conclusion and Future Work

In this thesis, we address the needs of several new Internet applications such as streaming media flows that cannot function well over TCP. We presented a framework that allows an application to choose from a huge palette of window based congestion control protocols called CYRF that have been *designed* to be feasible, i.e., they converge to fairness and efficiency.

In particular we showed that a window based protocol can safely grab bandwidth using the policy $x(t+R) \leftarrow x(t)+x(t)/f(x(t))$. An application can choose a slowly increasing function for $f(x)$ if it needs to to be aggressive and obtain bandwidth as soon as possible. Similarly, it can respond to congestion indications using the decrease policy $x(t+R) \leftarrow x(t) - x(t)g(x(t))$. Slowly increasing $g(x)$ give smoother flows and sublinear functions result in flows that are smoother than TCP. We also gave simple relations between $f(x)$ and $g(x)$ for smooth CYRF flows to satisfy the TCP-friendliness constraint.

We also briefly looked at a more general class of protocols called 1-CYRF and showed that commonly used window based congestion control protocols such as TCP, GAIMD, and binomial congestion control are special cases of this. We derived several known important results about these protocols as special cases of the results for CYRF and 1-CYRF. Thus, 1-CYRF and CYRF can also be thought of as a unified framework for the analysis of window based protocols.

We believe that some of the results obtained here can have applications outside the CYRF model. For example, the conditions for convergence to fairness derived in Chapter 5 and the TCP-friendly equation in Lemma 7.1 can easily be applied to other situations. On the other hand, the results for CYRF, specifically the proofs of convergence to fairness and efficiency (Chapter 6), and the conditions for TCP-friendliness and TCP-compatibility (Theorem 7.1) shed some insight into the notions of fairness, efficiency and TCP-friendliness.

Possible directions for future work include experimental evaluation of representative CYRF protocols under various static, transient and dynamic situations and in different network topologies. Recent work on binomial congestion control and GAIMD has shown using simulations that several CYRF protocols such as SQRT (binomial with $k = l = 1/2$), IIAD (binomial with $k = 1$, $l = 0$) and GAIMD with $\alpha = 0.31$ and $\beta = 7/8$ work well in several canonical network topologies and conditions. However, there are clearly other CYRF protocols that would satisfy the needs of different applications better and perhaps other CYRF protocols that work better for multimedia flows than IIAD, SQRT or GAIMD under certain conditions. The analysis and discovery of different WBF protocols for different kinds of applications is an exciting topic for further study.

Finally, to simplify our analysis, we assumed a rather restrictive model in Chapter 3 that does not truly represent the current Internet. It remains to be seen whether the results presented here hold under more general assumptions.

# Bibliography

[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] B. Braden, D. Clark, J. Crowcroft, B. Davie, S. Deering, D. Estrin, and S. Floyd. *Recommendations on Queue Management and Congestion Avoidance in the Internet*. Internet Engineering Task Force, April 1998. RFC 2309 (Informational).

[5] R. Braden, D. Clark, and S. Shenker. *Integrated Services in the Internet Architecture: an Overview*. Internet Engineering Task Force, June 1994. RFC 1633 (Informational).

[6] 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.

[7] D. Clark. The design philosophy of the DARPA Internet Protocols. In *Proceedings of ACM SIGCOMM '88*, pages 109–114, August 1988.

[8] D. Clark, S. Shenker, and L. Zhang. Supporting real-time applications in an Integrated Services Packet Network: Architecture and mechanisms. In *Proceedings of ACM SIG-COMM '92*, August 1992.

[9] D. E. Comer and D. L. Stevens. *Internetworking with TCP/IP, Volume II.* Prentice Hall, Engelwood Cliffs, NJ, 1994.

[10] 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.

[11] 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.

[12] 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.

[13] Sally Floyd. *Congestion Control Principles.* Internet Engineering Task Force, September 2000. RFC 2914 (Best Current Practice).

[14] Sally Floyd and Kevin Fall. Promoting the use of end-to-end congestion control in the Internet. *IEEE/ACM Transactions on Networking*, 7(4):458–472, August 1999.

[15] 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.

[16] R. Jain. Myths about congestion management in high speed networks. *Internetworking: Research and Experience*, Volume 3:101–113, 1992.

[17] 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.

[18] 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.

[19] 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.

[20] 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`.

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

[22] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow. *TCP Selective Acknowledgment Options.* Internet Engineering Task Force, October 1996. RFC 2018 (Standards Track).

[23] Sue Moon, Jim Kurose, and Don Towsley. Packet audio playout delay adjustment: Performance bounds and algorithms. *ACM/Springer Multimedia Systems*, 6:17–28, Jan 1998.

[24] J. Nagle. *Congestion Control in IP/TCP Internetworks.* Internet Engineering Task Force, January 1984. RFC 896.

[25] T. Ott, J. Kemperman, and M. Mathis. Window size behavior in TCP/IP with constant loss probability. In DIMACS Workshop on Performance of Realtime Applications on the Internet, November 1996. Another version available as *The Stationary Distribution of Ideal TCP Congestion Avoidance.*

[26] 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.

[27] C. Partridge. *A Proposed Flow Specification.* Internet Engineering Task Force, September 1992. RFC 1363.

[28] J. B. Postel. *Transmission Control Protocol.* Defense Advanced Research Projects Agency, September 1981. RFC 793.

[29] 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).

[30] 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.

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

[32] 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.

[33] 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.

[34] J. H. Saltzer, D. P. Reed, and D. D. Clark. End-to-end arguments in system design. *ACM Transactions on Computer Systems*, 2(4):277–288, 1984.

[35] 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).

[36] 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.

[37] W. R. Stevens. *TCP/IP Illustrated, Volume 1.* Addison Wesley, Reading, MA, 1994.

[38] J. S. Turner. New directions in communications (or which way to the information age?). *IEEE Communications Magazine*, 24(10):8–15, October 1986.

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

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

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

[42] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala. RSVP: A new Resource ReSerVation Protocol. *IEEE Network*, September 1993.

# Vita

Nishanth Sastry received the B.E. degree in Computer Science and Engineering from Bangalore University, India. He studied at the R.V. College of Engineering and graduated in First Class with Distinction. His final year undergraduate project entitled "*A Multipurpose Webserver and API for Web-enabling Compute Intensive Applications*" was co-recipient of the Best Undergraduate Project Award, 1999-2000 at the R.V. College of Engineering and won the Best Paper Award at the 34th Annual National Convention of the Computer Society of India. His current interests include all aspects of computer networks ranging from theoretical results to implementation details. He is a member of the UPE CS Honor society.

Permanent Address: "Krithika", No. 27,
$2^{nd}$ Main Road, Domlur $2^{nd}$ Stage
Bangalore, INDIA
PIN - 560 071
`nishanth@cs.utexas.edu`
`http://www.cs.utexas.edu/users/nishanth/`

This thesis was typeset with $\LaTeX\,2_\varepsilon$[1] by the author.

[1]$\LaTeX\,2_\varepsilon$ is an extension of $\LaTeX$. $\LaTeX$ is a collection of macros for $\TeX$. $\TeX$ is a trademark of the American Mathematical Society. The macros used in formatting this thesis were written by Dinesh Das, Department of Computer Sciences, The University of Texas at Austin.