WPI-CS-TR-03-28 July 2003

Enhancing Active Queue Management with Round-trip Time Awareness

by

Choong-Soo Lee
Mark Claypool
and Robert Kinicki

Computer Science Technical Report Series

WORCESTER POLYTECHNIC INSTITUTE

Computer Science Department
100 Institute Road, Worcester, Massachusetts 01609-2280

Enhancing Active Queue Management with Round-trip Time Awareness

Choong-Soo Lee, Mark Claypool, and Robert Kinicki
{clee01|claypool|rek}@cs.wpi.edu
Worcester Polytechnic Institute
Computer Science Department
100 Institute Road
Worcester, MA 01609

July 25, 2003

Abstract

Current congestion control approaches that attempt to provide fair bandwidth allocation among competing flows primarily consider only data rate when making decisions on which packets to drop. However, responsive flows with high round-trip times (RTTs) can still receive significantly less bandwidth than responsive flows with low round-trip times. Better bandwidth fairness control can avoid expensive Content Distribution Network deployment cost because CDNs provide improved, uniform performance to all clients regardless of proximity and can improve utilization in same circumstances. This paper proposes an enhancement to AQM schemes called *FIFA*[1] that addresses router unfairness in handling flows with significantly different RTTs. Using a best-case estimate of a flow's RTT provided in each packet by the flow source or by an edge router, FIFA computes a stabilized average RTT. The average RTT is then compared with the RTT of each incoming packet, dynamically adjusting the drop probability so as to protect the bandwidth of flows with high RTTs while curtailing the bandwidth of flows with low RTTs. We present simulation results and analysis that demonstrate that FIFA on top of any AQM schemes improves fairness without sacrificing throughput.

1 Introduction

The Internet relies upon cooperation between TCP hosts and subnet routers to adjust source data rates in the presence of network congestion along the path of the TCP flow. Drop-tail router queue management is the primary queue mechanism used in Internet routers to indicate congestion to edge hosts. While drop-tail routers are easy to implement and fit within the best-effort nature of the Internet, these routers distribute packet drops arbitrarily among competing flows.

For TCP-friendly flows, a flow's round-trip time (as well as packet size and drop rate) is directly responsible for determining a flow's data rate [8] [19]. With TCP's congestion control algorithms, a flow's throughput varies inversely with the round-trip time (RTT). Thus, a router that applies a uniform drop rate for all flows will result in flows with high RTTs getting less throughput than flows with low RTTs.

Current Internet Service Provider (ISP) user access fees are strictly connection-based and not distance-based, meaning the cost to access information on a Web server is not associated with the

[1]FIFA stands for FIFA Improves Fairness of AQMs

RTT from the client to that server. This creates the perception that all clients should receive the same quality of service from a Web server regardless of their proximity to that server. In fact, servers often use Content Delivery Networks (CDNs) to move Web server information closer to clients in order to provide improved, uniform performance, regardless of proximity. Thus, it can be argued that one of the goals of CDNs is to reduce the inherent bandwidth unfairness due to a client's large round-trip time. The approach of the FIFA algorithm presented here is to use RTT information at a congested router to provide better fairness among TCP clients and reduce the need to use CDNs to achieve bandwidth equity.

Long-lived flows with a high RTTs can also cause under-utilization of an outgoing link. During the congestion avoidance phase of TCP, flows with high RTTs take much longer to reach full utilization than flows with low RTTs, a problem that is exacerbated when packet drops are applied independently of the RTT. The approach of the FIFA algorithm presented here is to protect flows with high RTTs, increasing link-utilization under network conditions with long-lived, high RTT flows and shorter-lived, low RTT flows.

Most of the recent approaches for achieving bandwidth fairness have focused on mechanisms that throttle unresponsive flows to provide fairness relative to a small number of competing TCP flows ([21, 3, 4]). However these techniques do not necessarily provide fair bandwidth among responsive flows, as the results in Section 5 of this paper will show. The focus here is on fairness between responsive TCP flows with the assumption that a variation of one of the techniques that handle unresponsive flows can be integrated with our technique.

This paper presents FIFA, an AQM enhancement approach to fairness that takes into account a flow's round-trip time in determining a router's responsiveness to congestion avoidance. The primary goal of our work is to achieve fairness[2] among responsive flows with heterogeneous RTTs without reducing overall router performance (i.e., utilization, queuing delay and drop rate). FIFA can enhance any AQM that computes an aggregate drop probability, such as RED [10], PI [11], AVQ [14], Blue [6] and REM [2].

FIFA is designed to operate under the same *edge* and *core* architecture presented in Core Stateless Fair Queuing (CSFQ) [21] wherein core routers drop packets based on *hints* sent in packet labels by edge routers. In practice, the edge can be an ingress router or an end host and the hint can consist of the estimated data rate, as in CSFQ, the estimated window size, a delay tolerance, or any other flow attribute. In FIFA, the hint is an estimate of the best-case round-trip time.

Using round-trip time hints, the FIFA-enhanced core router computes an average round-trip time for all packets arriving at the router. When congestion is indicated, packets are dropped or marked based on their round-trip times in relation to the overall average round-trip time. In other words, FIFA alters the drop/mark probability the original AQM computes on a per-packet basis while keeping the benefits of the underlying AQM. This mechanism preferentially drops or marks packets with lower than average round-trip times and favors packets with higher than average round-trip times. This protects fragile flows with high round-trip times while inhibiting robust flows with low round-trip times from grabbing an unfair share of link bandwidth.

Through NS-2 [18] simulations, FIFA's effectiveness is demonstrated under a wide range of scenarios in combination with several popular AQMs and against several popular alternate fairness approaches. Test scenarios include: bottleneck capacity varying from 30 Mbps to 150 Mbps; number of flows varying from 30 to 120; mixes of flows with round-trip times varying from 40 ms to 1000 ms; and number of Web-like flows varying from 30 to 150.

Our simulation results show that FIFA provides fairness among flows that is far superior to the

[2]We focus on *min-max fairness*, since it is easy to interpret locally and makes no assumptions about behaviors elsewhere in the network. For completeness, *Jain's fairness index* [12] is also reported for all experiments.

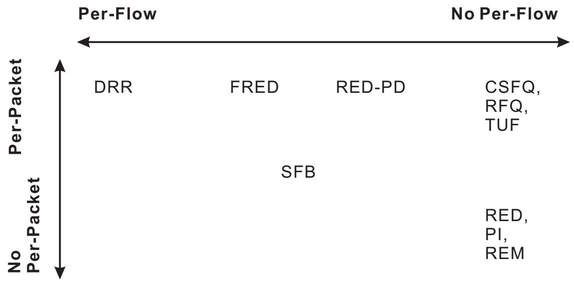


Figure 1: Classification of AQMs

underlying AQM in all scenarios tested. These FIFA improvements in fairness are accomplished while providing performance similar to the regular AQM with respect to drop rate, goodput and throughput. Moreover, AQMs enhanced by FIFA provide superior performance to alternate fairness approaches.

The rest of this paper is organized as follows: Section 2 reviews related work; Section 3 demonstrates possible benefits to users and networks from achieving fairness among heterogeneous flows; Section 4 focuses on the details and derivation of the FIFA mechanisms; Section 5 describes the set of simulation experiments to evaluate FIFA under a wide-range of conditions and includes analysis of the simulation results and detailed comparisons of the performance of FIFA-enhanced AQMs with regular AQM techniques; and Section 6 summarizes our findings and considers further extensions and future work.

2 Related Work

There have been numerous approaches to achieving per flow fair bandwidth allocation at congested routers. As noted in [17], there is a continuum of possible per-flow treatments, from complete per-flow treatment such as in Fair Queuing, to a complete absence of per-flow treatment such as in drop-tail and RED. Additionally, for queuing mechanisms without per-flow treatment, there is a continuum of possible per-packet treatments, from no per-packet treatments such as in RED to complete per packet treatments from CSFQ core routers. Figure 1 depicts the space of possible flow and packet treatment policies, with the approximate placements of policies discussed in this paper.

RED [10], probably the best known Active Queue Management (AQM) technique, keeps no per-flow state information. Packets are dropped probabilistically based on the long-term average queue size and fixed indicators of congestion (thresholds). RED uses randomization to drop arriving packets to avoid biases against bursty traffic and roughly drops packets in proportion to a flow's data rate at the router. However, flows with high RTTs and small window sizes are bursty, and this burstiness causes high variability in the perceived data rate of these flows as seen by RED routers. ECN is a variant of RED that marks packets to indicate congestion instead of dropping them [9].

e PI (Proportional Integral) [11] is developed through classical control system techniques and can be implemented by modifying the RED averaging algorithm. Not only does PI keep track of the current queue size and drop probability, it also saves the previous queue size and drop probability. PI uses the previous drop probability and differences in current and previous queue size to the reference queue size to compute the new drop probability.

REM (Random Exponential Marking) [2] uses an exponential drop probability curve in contrast to the linear one of RED. Instead of the average queue size that RED uses for congestion measure, REM tries to match the rate and to clear buffer at the same time trying to keep the queue size low at a target queue size.

RED, PI and REM all have the same weakness in unfair treatment of fragile flows. However, RED, PI and REM all compute a single drop probability and applies it to all incoming flows. FIFA can take advantage of any AQM that does not differentiate flows and help treat long term responsive flows fairly.

At the other extreme, Deficit Round Robin (DRR) [20], a variant of Fair Queuing (FQ) [5], keeps extensive information on every flow. DRR routers send packets approximately in the order a router would send them if packets could be sent one bit at a time. While DRR and other FQ variants achieve fairness among flows, the per-flow state information required and overhead needed to manage priority queues is expensive. Moreover, these schemes do not scale with an increased number of flows. This paper uses DRR as the best case scenario for achieving fairness among heterogeneous flows; namely the goal being to seek fairness comparable to DRR without DRR per-flow costs.

Flow Random Early Drop (FRED) [15] uses per-flow preferential dropping to achieve fairer allocation of bandwidth among flows. FRED builds per-flow state at the router by examining those packets that are currently in the queue. The packet drop rate for a flow is determined by the number of packets the flow has in the queue and is not directly influenced by the flow's data rate or round-trip time. We evaluate the effectiveness of FRED as a less expensive means than FQ of attempting per-flow fairness. [15] introduces the formal definitions of fragile and robust flows. *Fragile* flows are congestion aware but are either sensitive to packet losses or slower to adapt to more available bandwidth. *Robust* flows are also congestion aware, but are able to increase bandwidth usage quickly in presence of spare capacity.

In Core Stateless Fair Queuing (CSFQ) [21], edge routers classify flows based on their current sending rates and forward these rates as labels to core routers. Using these labels, core routers keep a running estimate of the fair share capacity of a flow on an outgoing link. The core router drops packets in a manner aimed at giving each flow its fair share of the link throughput. However, such preferential dropping based on data rate alone is not sufficient to achieve fairness. Since the response function for TCP-friendly flows is based on the RTT, dropping packets equally between two flows with the same short-term data rate but different RTTs will result in a higher long-term data rate for the flow with the lower RTT. While it has been shown that CSFQ achieves fairness for flows with the same RTT, we demonstrate that it is ineffective in achieving fairness among flows with heterogeneous RTTs.

The Rainbow Fair Queueing (RFQ) algorithm[3] is an extension of the source-edge-core architecture of CSFQ. Like CSFQ, RFQ uses an exponential averaging formula at the edge router to estimate flow arrival rate. RFQ proposes a color assignment scheme that partitions packets into color layers designed to prioritize packets for dropping at the core router. The core router algorithm dynamically adjusts the color dropping threshold when the queue length exceeds a queue threshold. The RFQ algorithm relies on a probabilistic color assignment scheme that is effective in controlling unresponsive UDP flows by assigning layer rates at the edge router. As with CSFQ, none of the reported simulations address the unfairness caused by differing RTT values for TCP flows.

The Stochastic Fair Blue (SFB) algorithm [7] combines the Blue algorithm [6] with the concept of Bloom filters. The primary goal of SFB, similar to CSFQ, is protecting TCP flows from unresponsive flows. However, unlike CSFQ, SFB can operate at core routers. Using the arriving flow id of a packet, SFB does L independent hashes into one of N bins for each level L. The objective is to provide fairness with only a small amount of state and buffer space at the core router. Each bin has a threshold (queue size) and marking/dropping probability controlled by the Blue algorithm.

SFB works well when there are relatively few unresponsive flows that are easily identified because relatively high bandwidth flows cause the thresholds to be exceeded in all N of their bins. However, SFB fairness suffers when there are more than a few unresponsive flows and has no ability to adjust the number of bins when there are a large number of flows.

Recognizing RED's inability to deal with unresponsive flows, RED with Preferential Dropping (RED-PD) [16] has been introduced as a RED modification that builds on CSFQ and FRED concepts. Relying on a target bandwidth characterized by a reference RTT, R, RED-PD uses partial flow state at the core-router. Based on the expected drop rate of the targeted bandwidth, RED-PD identifies high-bandwidth flows such that they become monitored flows that are preferentially dropped by a pre-filter at the core router. This technique deals both with unresponsive flows and aims at fairness among TCP flows with varying RTTs. If the target RTT is chosen well, RED-PD provides fairness among heterogeneous TCP flows. However, this scheme is not adaptive to large variations in RTTs. FIFA addresses this issue by adjusting its drop/mark probability based on a dynamic average RTT at the router.

The Tag-based Unified Fairness (TUF) technique [4] is similar to CSFQ and RFQ in that it relies on information sent as tags by a tagger to the TUF core routers, where the tagger can be a source host or an edge router. The tagger keeping track of several pieces of flow state, including a flow's RTT, produces an estimate of the flow's data rate to insert a tag into packets that indicates the flow's relative drop precedence. When a packet arrives to a full queue, the core router drops the packet with the highest tag value. Like FIFO, TUF drops only when the queue overflows and can cause bursts of losses. The emphasis in the TUF results is on maintaining fairness for a TCP flow when mixed with unresponsive UDP flows, but provides no mechanism to separate out flows with the same data rate and different RTTs.

3 Benefits from Fairness

In this section, we mathematically analyze potential benefits from enforcing bandwidth fairness at the router. We start with a common TCP throughput formula $T = b\frac{s}{R\sqrt{P}}$ [19] to work through the analysis (where T : Throughput, s : packet size, R : round-trip time, P : overall drop rate and b : constant). We compare the fair case and the unfair case. The fair case assumes bandwidth fairness while the unfair case lets TCP flows alone. Figure 2 shows the variables used in the analysis.

C : bottleneck link capacity
n : total number of flows
F : file transfer size
R_r : round-trip time of robust flows (low RTT)
R_f : round-trip time of fragile flows (high RTT)
a : R_r is a fraction a of R_f (ie. $R_r = aR_f$)
T_i : throughput of ith flow
s : packet size
b : a constant

Figure 2: Description of Variables used in Analysis

3.1 Response Time

Response time is the time it takes from the user request of an object to the end of the object transfer. Response time is particularly important for Web performance measurement because of user sensitivity to high round-trip times. Zona research estimates that e-commerce loses over $4 billion each year in the US because of high response times and introduces an "8 second rule" which states that a user is only willing to wait up to 8 seconds in response to a Web page request.

We consider a case where $(n-1)$ long lived robust flows are sharing the bottleneck bandwidth at full utilization. Then a fragile flow comes in to transfer an object. We calculate how long it takes for the fragile flow to transfer the file in a fair bandwidth case and an unfair bandwidth case.

In the fair case, each flow should get $T_i = \frac{C}{n}$. Therefore, the time to transfer the file is

$$t_{fair} = \frac{F}{T_i} = \frac{F}{C/n} = \frac{nF}{C}$$

In the unfair case, the robust flows get more throughput compared to the fragile flow. The throughput of the fragile flow is $T_f = \frac{C}{\left(\frac{n-1}{a}+1\right)}$. Therefore, the time to transfer the file is

$$t_{unfair} = \frac{F}{T_f} = \frac{F}{C}\left(\frac{n-1}{a}+1\right)$$

Finally, we take the ratio of the two in order to see the magnitude of difference:

$$\frac{t_{fair}}{t_{unfair}} = \frac{\frac{nF}{C}}{\frac{F}{C}\left(\frac{n-1}{a}+1\right)} = \frac{an}{a+n-1}$$

We can see the ratio only depends on a and n. In other words, it only depends on how robust the flows are compared to the fragile flow and the total number of flows. As n gets large, the ratio converges to a.

Example :

Assume $n = 4$, $a = 0.1$, $C = 10$ Mbps, $s = 1000$ bytes, then the ratio is 0.129. If every user can load the page in 6 seconds in the fair case, a fragile user takes 46.5 seconds to load the page in the unfair case.

3.2 Utilization

Utilization of the outgoing link is another important metric, especially to companies like ISPs and backbone providers that like to keep utilization high. However, fragile flows can result in low utilization because it takes a long time to increase their throughput due to their high round-trip time.

We consider a case where there is one fragile flow using up the congested link and $(n-1)$ robust flows arrive to transfer a small file and then leave. In Figure 3, the robust flows and the fragile flow fully utilizes the outgoing link before time X. Then the robust flows leave at time X. We analyze the reduction in throughput due to the robust flows arriving and how long it takes the fragile flow to go back up to the throughput that uses up the congested link. The rate at which the fragile flow increases the throughput is at $\frac{s}{R_f}$.

In the fair case, the throughput of the fragile flow goes down to $\frac{C}{n}$ and reaches full utilization at time Y. The time to regain the full congested link utilization is:

$$t_{fair} = \frac{C-\frac{C}{n}}{\frac{s}{R_f}} = \frac{R_f}{s}\left(C - \frac{C}{n}\right)$$

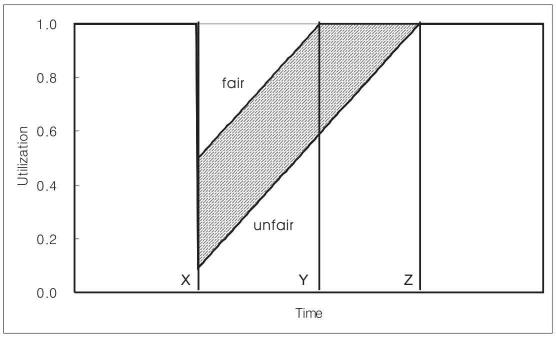


Figure 3: Utilization: In the unfair case depicted by the lower line, TCP takes longer to ramp back up to full utilization tan in the fair case. The gray area represents data that could have been sent in the fair case.

In the unfair case, the throughput of the fragile flow goes down to $\frac{C}{(\frac{n-1}{a}+1)}$ and reaches full utilization at time Z. The time to regain the full congested link utilization is:

$$t_{unfair} = \frac{R_f}{s}\left(C - C\frac{1}{(\frac{n-1}{a}+1)}\right)$$

Finally, we take the ratio of t_{fair} to t_{unfair}:

$$\frac{t_{fair}}{t_{unfair}} = \frac{a+n-1}{n}$$

The shaded area in Figure 3 is the data that could have been transfered by time Z if the fragile flow was treated fairly. This area is:

$$Area = \frac{R_f C^2 (n-1)^2}{2s}\left(\left(\frac{1}{a+n-1}\right)^2 - \left(\frac{1}{n}\right)^2\right)$$

Example : Assume $n = 4$, $a = 0.1$, $C = 10$ Mbps, $s = 1000$ bytes, then the ratio is 0.775. In other words, it takes the fragile flow in the fair case to reach full utilization only 77.5% as long compared to the unfair case. Due to unfairness, 701.3 Mb of data (the shaded area) is lost.

4 Mechanism

In this section, we present the FIFA algorithm, summarized in Figure 4. *prob* is the drop/mark probability calculated by the AQM that FIFA is implemented upon, *mark* is if the AQM is set to drop (0) or mark (1) the packets and p is the packet that arrived at the router. Each packet contains a round-trip time (RTT) label, as described in Section 4.1. For each incoming packet, the RTT label is used to update the RTT average kept by the router, as described in Section 4.2. The drop/mark probability for the packet is computed based on the RTT label and RTT average, as described in Section 4.3.

```
on receiving packet p
if (p.RTT > 0) then
   updateAvg(R_average, R_formula, p.RTT)
   d = calcDropMarkProbability(prob, p.RTT, R_formula))
   if (d > 1.0)
      d = 1.0
   if (mark)
      markPacket(p, d)
      enQueuePacket(p)
   else
      dropPacket(p, d)
```

Figure 4: The FIFA Algorithm

4.1 Round-Trip Time at the Edge

As in [21], we assume an architecture where edge routers on the ingress of a network label a packet with additional information, called an *edge hint* so that core routers on the interior of the network can make packet dropping or marking decisions efficiently. For our evaluation, the edge hint is given by the sending host using TCP-Reno and it is the lowest RTT recorded, as computed using the baseRTT computation from TCP Vegas code.

Based on discussion in [22], there are from 4 to 17 bits available in the IP header that can be used to carry edge hint information. We store the RTT in the IP packet using a 16-bit integer, but in practice typical ranges of RTTs would only require about 9 bits providing coverage of up to 80% of RTTs observed [1]. At a granularity of 10 ms, 9 bits would be sufficient to cover RTT ranges of up to 5 seconds.

Without host or edge support, it may be possible for the router to infer RTT as in [23]. This can be done by measuring the interval between first few packets of a TCP flow for RTT estimation. The drawback, of course, is the router would have to identify flows and keep track of per-flow RTT information.

4.2 Average Round-Trip Time at the Router

The average RTT at the router is calculated using an exponential weighted moving average, called $R_{average}$. To adjust quickly to changes in the network, the weight w_{RTT} is set to 0.1, which is necessary to quickly detect changes in the RTT of incoming packets, caused by the the addition of new flows, the termination of old flows or a change in route of some flows. To prevent excessive variation in average RTT computation under steady state, the algorithm uses a stabilized measure of the RTT, called $R_{formula}$, to compute drop/mark probabilities (see Section 4.3). $R_{formula}$ is only changed when $R_{average}$ has significantly moved, meaning $R_{average}$ has been out of a range of +/- 12.5ms over a period of 100 ms, an approximate RTT. If $R_{average}$ has moved from one side of the range to the other, called *crossing over* in Figure 5, we reset the time interval. When it is determined that $R_{average}$ has moved, $R_{forumla}$ is updated to move towards the new $R_{average}$.

We experimented with $R_{average}$ originally but it yielded slightly better fairness with the stabilized RTT, $R_{formula}$. $R_{average}$ is only an exponentially weighted average of all flows. Therefore, it oscillates around the actual average of round-trip times. $R_{formula}$ helps to find the value close to the true average of round-trip times and keeps it constant until it notices a significant change in the round-trip times.

now = getCurrentTime()
$R_{average} = (1 - w_{RTT})\ R_{average} + (w_{RTT})\ p.RTT$
if $(((R_{formula}$-12.5) $< R_{average}$ AND $(R_{formula}$+12.5) $> R_{average})$ OR $R_{average}$ crossed over) then
    $lasttime = now$
else
  if ((now - lasttime) > 100ms) then
    $lasttime = now$
    update $R_{forumula}$ towards new $R_{average}$

Figure 5: Algorithm for Computing Round-Trip Time at the Router

4.3 Drop/Mark Probability Based on Round-Trip Time

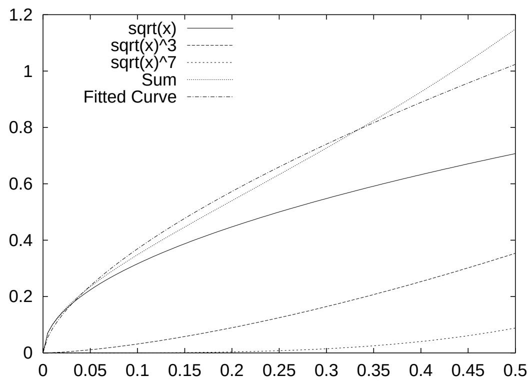


Figure 6: Contribution of p-Terms vs. Drop/Mark Probability.

In order to compute a drop/mark probability based on a packet's RTT relative to the average RTT ($R_{formula}$), we start with the TCP response function [19]:

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

Equation 1 provides the upper bound on the sending rate of T as a function of the packet size s, steady state loss rate p, round-trip time R and retransmission time out t_{RTO}. Although the drop/mark probability in AQMs is not exactly equivalent to the steady state loss rate p, p will be used to estimate the relationship between the drop/mark probability and the round-trip time. Combining the three terms involving p above using a constant and exponent, we get the simplified equation:

$$T = \frac{s}{Rcp^a} \tag{2}$$

Consider two flows with throughputs T_1 and T_2 and round-trip times R_1 and R_2, respectively. In order to achieve fair bandwidth allocation, T_1 and T_2 should be equal. This leads to the following derivation:

$$T_1 = T_2$$
$$\frac{s}{R_1 c p_1^a} = \frac{s}{R_2 c p_2^a}$$
$$p_2 = p_1 \left(\frac{R_1}{R_2}\right)^{\frac{1}{a}} \tag{3}$$

The exponent a needs to summarize the behavior of the denominator of the original equation. The three terms involving p in Equation (1) have exponents of 1/2, 3/2 and 7/2. Figure 6 depicts each p-term's contribution to the sum, and the best-fit curve to the sum. When p approaches 0, $p^{1/2}$ dominates, but when p approaches 0.5, each p-term contributes nearly equally to the sum. Figure 6 shows that the sum of the p-terms follows closely to square root of p as expected, with the estimated exponent a in the fitted curve function being about 0.63. Using $a = 0.63$, FIFA calculates the per-packet drop/mark probability from a base drop/mark probability p:

$$p = p_{base} \left(\frac{R_{formula}}{R}\right)^{\frac{1}{0.63}}$$
$$p = p_{base} \left(\frac{R_{formula}}{R}\right)^{1.58} \tag{4}$$

For the rest of the paper, the exponent used for the drop/mark probability for robust flows will be called α and the exponent used for the drop/mark probability for fragile flows will be called β.

For overhead, FIFA adds four additional variables (α, β, $R_{average}$, and $R_{formula}$). For each packet that arrives, FIFA must read the round-trip time hint in each packet and compute $R_{average}$, adjusting $R_{formula}$ as necessary. No per-flow state information is required.

5 Performance

In this section, we present the simulation setup and analyze the results. We first test performance of FIFA on top of RED, PI and REM on various conditions. Then, we validate FIFA on top of RED against RED, FRED, CSFQ and DRR.

5.1 Setup

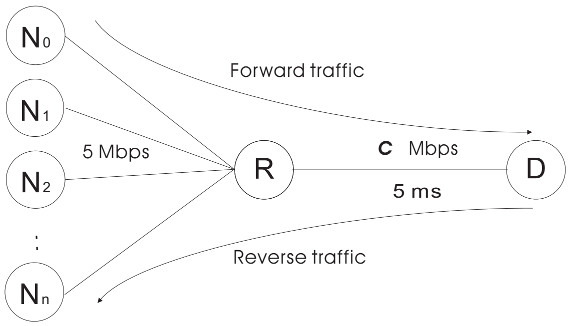


Figure 7: Network Topology

To show the practicality of FIFA, we implemented FIFA on top of several different popular AQMs by extending the existing NS codes with the algorithm described in Section 4. Then, we set up the network topology shown in Figure 7. There are n sources, N_0 *through* N_{n-1} going through a bottleneck router R to destination D. The bottleneck bandwidth is C Mbps and the delay 5 ms. Any setting not specified is set to the default NS-2 configuration. All flows use TCP-Reno with the RTT modification described in Section 4.1 and a maximum window size of 64 packets.

FIFA uses α and β values of 0.65 and 1.40 respectively for all simulations for all dropping AQM schemes while 1.60 and 1.40 respectively for all marking AQM schemes. These values were tuned during pilot studies through the experiment described in Section 5.3.

To numerically compute fairness, we computed Jain's fairness index. [12] Jain's fairness index takes the average goodputs of the flows and computes a normalized number between 0 and 1, where 0 denotes the maximum unfairness and 1 denotes the maximum fairness.

We use + to denote the AQMs with FIFA enhancement, e.g. RED+, to differentiate them.

5.2 Performance Analysis with RED, PI and REM

In the next four experiments, each source latency is calculated using Equation 5, which introduces a linear increase in latency from one source to the next.

$$latency\,(N_i) = 2\,[(i+1)\,15ms + 5ms] \tag{5}$$

Therefore, the 30 sources have round trip latencies ranging from 40ms to 330ms. And the bottleneck capacity C is set at 30 Mbps. The queue size is 1000 packets. RED's min_{th} and max_{th} are set at 100 and 300 packets respectively. The default settings are used for PI and REM as specified in NS-2.

The experiment simulations were run five times, each for 120 seconds and the goodput was averaged over a 90 second period starting 30 seconds after the beginning of each simulation.

5.2.1 Impact of Capacity

This experiment varies the bottleneck capacity from 30 Mbps to 150 Mbps in steps of 30 Mbps. The queue size, RED's min_{th} and max_{th} are increased linearly in proportion to the bottleneck capacity.

In Figure 8, the original AQMs are depicted by dotted lines while FIFA enhanced AQMs are depicted by solid lines. Each dot from left to right represents the bottleneck capacity: 30 Mbps, 60 Mbps, 90 Mbps, 120 Mbps and 150 Mbps respectively. We can see that RED+, PI+ and REM+ show improvements in fairness over their underlying AQMs. RED+ improves fairness up to 16.0%, PI+ 24.6% and REM+ 32.7% while keeping utilization essentially the same.

5.2.2 Impact of Number of Flows

This experiment varies the number of flows from 30 to 120 in steps of 15. In Figure 9, the original AQMs are depicted by dotted lines while FIFA enhanced AQMs are depicted by solid lines. Each dot from left to right represents the number of flows: 30, 45, 60, 75, 90, 105 and 120 respectively. We can see that RED+, PI+ and REM+ show improvements in fairness over their underlying AQMs. RED+ improves fairness up to 17.5%, PI+ 20.5% and REM+ 30.4% while keeping utilization essentially the same.

AQM	Bottleneck Capacity (Mbps) 30	60	90	120	150
RED	0.772	0.774	0.778	0.780	0.810
RED+	0.895	0.871	0.842	0.904	0.854
PI	0.735	0.683	0.657	0.660	0.654
PI+	0.916	0.835	0.780	0.769	0.808
REM	0.696	0.676	0.663	0.664	0.654
REM+	0.907	0.831	0.833	0.855	0.868

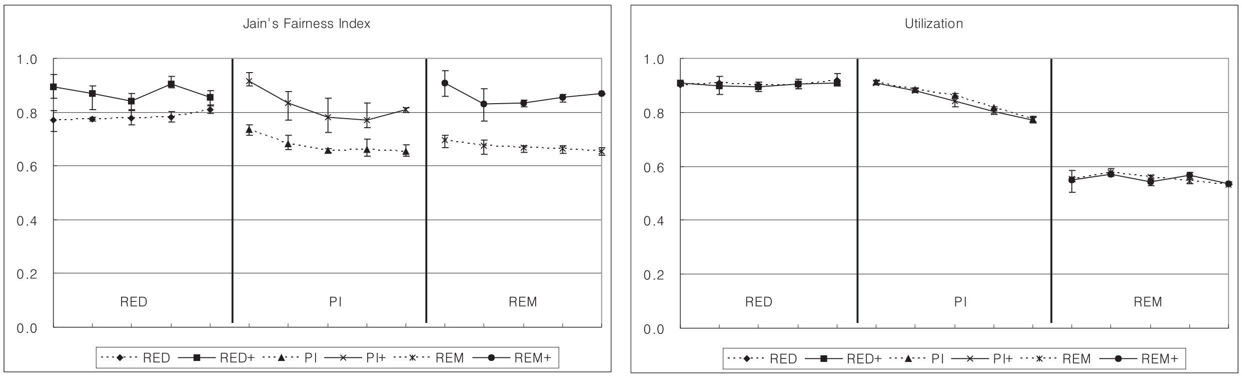


Figure 8: Impact of Capacity (30 Mbps, 60 Mbps, 90 Mbps, 120 Mbps, 150 Mbps)

AQM	Number of Flows 30	45	60	75	90	105	120
RED	0.772	0.818	0.798	0.828	0.832	0.849	0.849
RED+	0.903	0.937	0.938	0.946	0.941	0.950	0.941
PI	0.743	0.782	0.762	0.798	0.783	0.805	0.801
PI+	0.895	0.922	0.903	0.903	0.936	0.916	0.924
REM	0.709	0.758	0.776	0.803	0.798	0.846	0.840
REM+	0.924	0.953	0.782	0.903	0.900	0.865	0.870

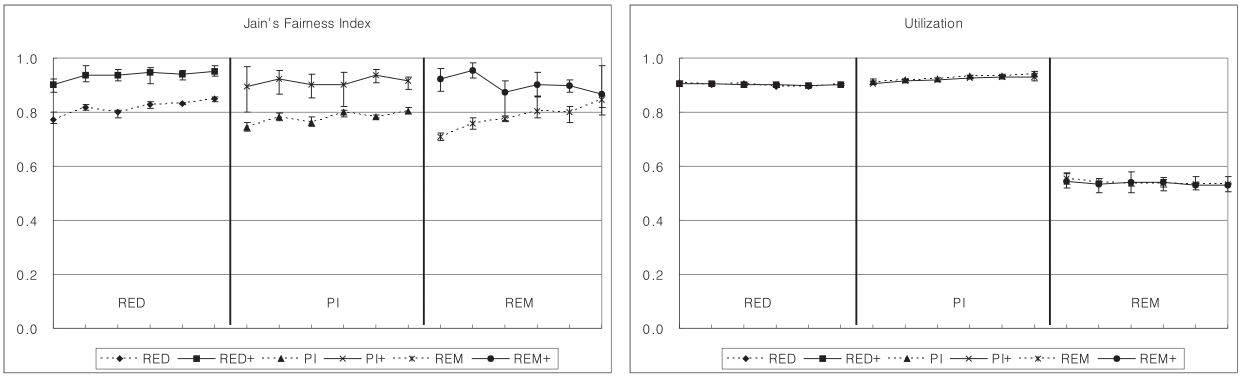


Figure 9: Impact of Number of Flows (30, 45, 60, 75, 90, 105, 120 flows)

AQM	Round-trip Time Spread (ms) 40-330	50-485	60-640	70-795	80-950
RED	0.782	0.708	0.618	0.584	0.521
RED+	0.938	0.816	0.763	0.677	0.686
PI	0.739	0.680	0.646	0.590	0.583
PI+	0.891	0.844	0.784	0.815	0.744
REM	0.700	0.629	0.610	0.573	0.567
REM+	0.921	0.868	0.803	0.822	0.775

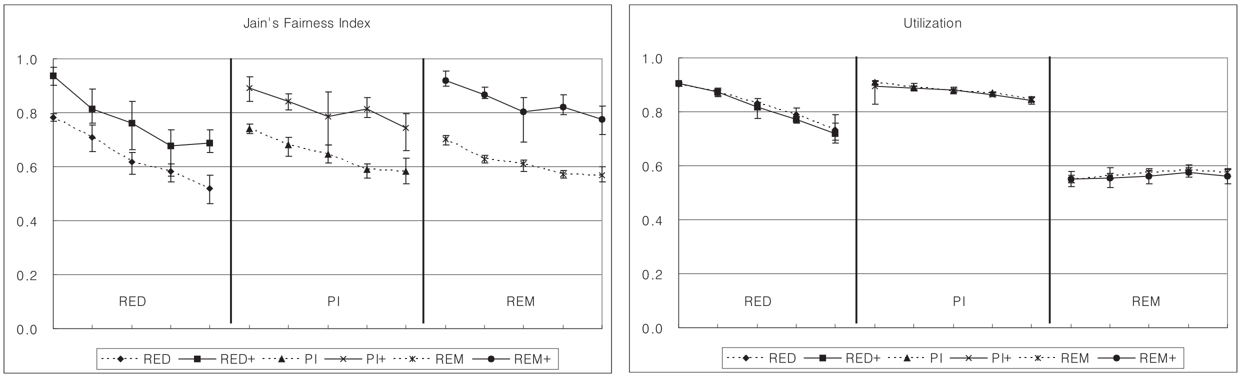


Figure 10: Impact of Round-trip Time (40-330 ms, 50-485 ms, 60-640 ms, 70-795 ms, 80-950 ms)

5.2.3 Impact of Round-trip Times

This experiment varies the round-trip time of all flows by multiplying the latencies to the sources by factor from 1.0 to 3.0 in steps of 0.5. For example, with factor of 1.0, the round-trip time range stays the same from 40 ms to 330 ms, while with factor of 2.0, the range increases from 60 ms to 640 ms with increments of 20 ms. In Figure 10, the original AQMs are depicted by dotted lines while FIFA enhanced AQMs are depicted by solid lines. Each dot from left to right represents the scale factor with which round-trip times are multiplied: 1.0, 1.5, 2.0, 2.5 and 3.0 respectively. We can see that RED+, PI+ and REM+ show improvements in fairness over their underlying AQMs. RED+ improves fairness up to 31.8%, PI+ 38.2% and REM+ 43.4% while keeping utilization essentially the same.

5.2.4 Impact of Web Traffic

This experiment introduces Web traffic which was set up using pareto traffic (shape_ = 1.35 [13]). The number of Web-like flows (mice) were varied from 30 to 150 in steps of 30 flows. In Figure 9, the original AQMs are depicted by dotted lines while FIFA enhanced AQMs are depicted by solid lines. Each dot from left to right represents the number of Web-like flows: 30, 60, 90, 120 and 150 respectively. We can see that RED+, PI+ and REM+ show improvements in fairness over their underlying AQMs. RED+ improves fairness up to 21.2%, PI+ 25.7% and REM+ 27.6% while keeping utilization essentially the same.

AQM	Number of Web-like Flows 30	60	90	120	150
RED	0.768	0.770	0.758	0.800	0.854
RED+	0.919	0.933	0.904	0.933	0.950
PI	0.731	0.749	0.747	0.784	0.822
PI+	0.898	0.915	0.940	0.908	0.870
REM	0.712	0.724	0.777	0.816	0.812
REM+	0.908	0.904	0.917	0.892	0.900

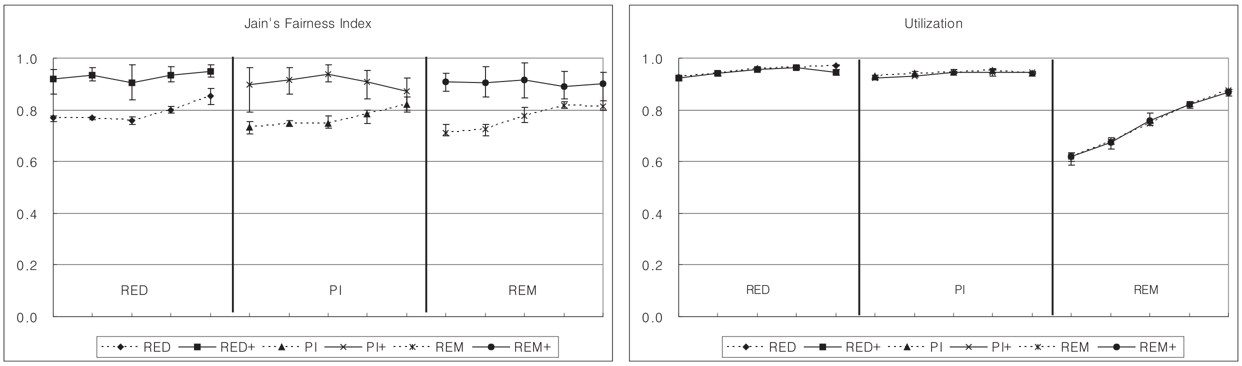
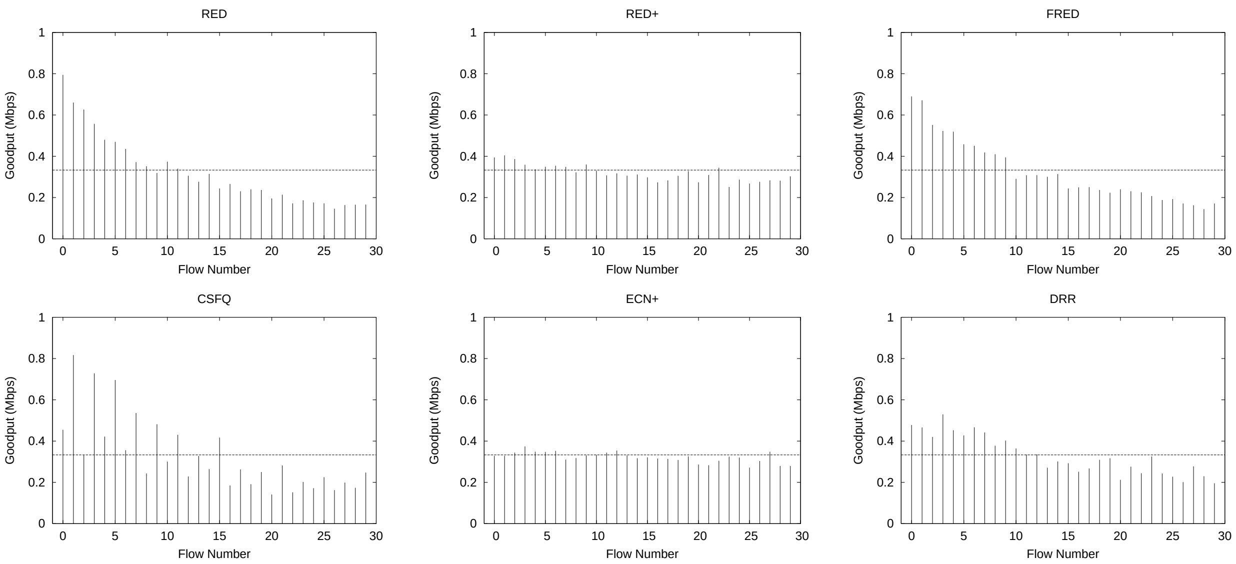


Figure 11: Impact of Web Traffic (30, 60, 90, 120, 150 mice)



AQM	CSFQ	RED	FRED	DRR	RED+	ECN+
Jain's Fairness Index	0.781	0.785	0.828	0.927	0.982	0.994

Figure 12: Fairness Comparison: The 30 flows have round-trip latencies ranging from 20 ms (left) to 310 ms (right).

5.3 Comparison with other AQM Fairness Approaches

We wished to compare the fairness of AQMs enhanced with FIFA with other fairness approaches, some of which use per-flow state information. We implemented FIFA on top of RED using drops

(RED+) and marks (ECN+).

In this experiment, each source latency is calculated using Equation 6, which introduces a linear increase in latency from one source to the next.

$$latency\,(N_i) = 2\,[(i+1)\,5ms + 5ms] \tag{6}$$

Therefore, the 30 sources have round trip latencies ranging from 20 ms to 310 ms. Pilot studies using this scenario were used to tune the α and β parameters for FIFA. Although the theoretical value derived in Section 4 was 1.578, the actual values used based on the best results from this experiment, as shown in Figure 7, are 0.65 and 1.40 for RED+ and 1.60 and 1.40 for ECN+. The reason for 0.65 for α is that the higher p gets, each p-term in Equation (1) contributes equally to the sum and results in a lower α.

The bottleneck capacity is 10 Mbps. min_{th} is set at 10 packets, max_{th} at 30 packets, w_q at 0.0008 and max_p at 0.1 for RED, FRED, RED+ and ECN+. For CSFQ, the averaging constants K (used in estimating the flow rate), K_α (used in estimating the fair rate), and K_c (used in making the decision on whether the link is congested or not) are set as recommended in [21] at 100ms. The queue is limited at 120 packets. Simulations were run for 150 seconds and the goodput was averaged over a 90 second period starting 30 seconds after the beginning of each simulation.

Figure 12 depicts the goodput for each flow. With 30 different flows, the fair share of each flow is $\frac{1}{3}$ Mbps, depicted by the horizontal line in each graph. Without any fair treatment, RED provides the most bandwidth (0.79 Mbps) to the most robust source and the least bandwidth (0.15 Mbps) to a fragile source. FRED does not improve the bandwidth for the most fragile flow at all (0.14 Mbps) but reduces the bandwidth of the most robust flow (0.69 Mbps). CSFQ provides high bandwidth to every other robust flow (the largest getting 0.82 Mbps), while the least bandwidth is comparable to that of RED (0.14 Mbps). Visually, DRR and RED+ perform much better (more sources are near the horizontal line), with DRR providing reasonable fairness between the most robust flow (0.53 Mbps) to the most fragile flow (0.20 Mbps), and RED+ providing the best fairness between the most robust flow (0.40 Mbps) and the most fragile flow (0.15 Mbps). Note, however, DRR requires per-flow state information while RED+ and ECN+ do not.

6 Summary

A TCP-friendly flow's round-trip time (RTT) is directly responsible for determining a flow's data rate [8, 19]. Current Internet routers and router congestion control approaches ignore round-trip time (RTT) in making packet dropping or marking decisions, providing unfair bandwidth allocation for flows with different RTTs. Consequently, Web servers often use Content Delivery Networks (CDNs) to reduce client RTTs in order to provide improved, uniform performance, regardless of proximity. Instead, we propose using RTT information in a congested router to provide fairness among TCP clients regardless of RTT and thus reduce the need to use CDNs for bandwidth equity.

This paper presents FIFA, an approach to enhance active queue management to achieve fairness among flows with heterogeneous RTTs. Using the distributed architecture presented in [22], packets are labeled with their minimum observed RTT, allowing the FIFA-enhanced router to make dropping or marking decisions based upon the RTT of each flow relative to the average RTT observed at the router. This provides the potential to protect fragile flows with a high RTT from receiving unnecessarily low bandwidth, while curtailing robust flows with a low RTT to their fair bandwidth share. Moreover, the use of the RTT label at the network edge allows FIFA to avoid keeping per-flow information. At the same time, FIFA inherits all the benefits of the underlying AQM scheme. It also supports marking AQM schemes as well by adjusting α and β parameters.

We evaluate FIFA over a range of traffic conditions including flows, RTTs, capacity and traffic mix. We find FIFA provides significant improvement to fairness under all conditions. We also compare FIFA-enhancement to RED, PI and REM and find FIFA provides much higher fairness than RED, PI and REM. We are not aware of any other router queue management techniques that can achieve better fairness without using per-flow information or FIFA.

Currently, FIFA does not curtail unresponsive flows receiving more than their fair share of bandwidth. In fact, under FIFA, unresponsive flows with a high RTT will be favored over responsive flows with a low RTT. The natural extension to FIFA is to combine it with a rate-based active queue management technique, such as CSFQ [21] or RED-PD [17]. High-bandwidth flows could be detected and monitored as in [17], or per packet drop probabilities could be computed based on both the bandwidth used by the flow as well as the RTT.

References

[1] M. Allman. A Web Server's View of the Transport Layer. *ACM Computer Communication Review*, Oct 2000.

[2] S. Athuraliya, S. Low, V. Li, and Q. Yin. REM Active Queue Management. *IEEE Network Magazine*, 15:48 – 53, May 2001.

[3] Z. Cao, Z. Wang, and E. Zegura. Rainbow Fair Queueing: Fair Bandwidth Sharing without Per-flow State. *Proceedings of IEEE Infocom 2000 Conference*, Mar 2000.

[4] A. Clerget and W. Dabbous. Tag-based Unified Fairness. *Proceedings of IEEE Infocom 2001 Conference*, Apr 2001.

[5] A. Demers, S. Keshav, and S. Shenker. Analysis and Simulation of a Fair Queuing Algorithm. *In Journal of Internetworking Research and Experience*, pages 3 – 26, Oct 1990.

[6] W. Feng, D. Kandlur, D. Saha, and K. Shin. Blue: An Alternative Approach To Active Queue Management. *Proceedings of NOSSDAV*, Jun 2001.

[7] W. Feng, D. Kandlur, D. Saha, and K. Shin. Stochastic Fair Blue: A Queue Management Algorithm for Enforcing Fairness. *Proceedings of IEEE Infocom*, Apr 2001.

[8] S. Floyd. Connections with Multiple Congested Gateways in Packet-Switched Networks Part 1: One-way Traffic. *ACM Computer Communication Reviews*, 21(5):30 – 47, Oct 1991.

[9] S. Floyd. TCP and Explicit Congestion Notification. *Computer Communication Review*, Oct 1994.

[10] S. Floyd and V. Jacobson. Random Early Detection Gateways for Congestion Avoidance. *IEEE/ACM Transactions on Networking*, Aug 1993.

[11] C. Hollot, V. Misra, D. Towsley, and W. Gong. On Designing Improved Controllers for AQM Routers Supporting TCP Flows. In *Proceedings of IEEE INFOCOMM Conference*, Apr 2001.

[12] R. Jain. *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling.* John Wiley and Sons, Inc., 1991.

[13] D. Katabi, M. Handley, and C. Rohrs. Congestion Control for High Bandwidth-Delay Product Networks. In *Proceedings of ACM SIGCOMM Conference*, Aug 2002.

[14] S. Kunniyur and R. Srikant. Analysis and Design of an Adaptive Virtual Queue. In *Proceedings of ACM SIGCOMM*, Aug. 2001.

[15] D. Lin and R. Morris. Dynamics of Random Early Detection. In *Proceedings of ACM SIGCOMM Conference*, Sep 1997.

[16] R. Mahajan and S. Floyd. Controlling High Bandwidth Flows at the Congested Router. *Proceedings of IEEE ICNP*, pages 1 – 12, Nov 2001.

[17] R. Mahajan, S. Floyd, and D. Wetherall. Controlling high-bandwidth flows at the congested routers. In *In Proceedings of the 9th International Conference on Network Protocols (ICNP)*, Nov 2001.

[18] U. of California Berkeley. The Network Simulator - ns-2. Interent site `http://www.isi.edu/nsnam/ns/`.

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

[20] M. Shreedhar and G. Varghese. Efficient Fair Queueing Using Deficit Round Robin. In *Proceedings of ACM SIGCOMM Conference*, pages 231 – 243, Sep 1995.

[21] I. Stoica, S. Shenker, and H. Zhang. Core-Stateless Fair Queueing: Achieving Approximately Fair Bandwidth Allocations in High Speed Networks. In *Proceedings of ACM SIGCOMM Conference*, Sep 1998.

[22] I. Stoica and H. Zhang. Providing Guaranteed Service Without Per Flow Management. In *ACM SIGCOMM Computer Communication Review , Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communication*, Aug 1999.

[23] X. Wu and I. Nikolaidis. Active Queue Management and Global Fairness Objectives. *International Performance, Computing, and Communications Conference (IPCCC '03)*, Apr 2003.