Available as a Los Alamos Unclassified Report LA-UR 03-2372.

GREEN: A Practical Solution for Ensuring Fairness in a Best-Effort Network*

Apu Kapadia†
Dept. of Computer Science
University of Illinois at Urbana-Champaign
1304 W. Springfield Ave.
Urbana, IL 61801
akapadia@uiuc.edu

Wu-chun Feng
Computer & Computational Sciences Division
Los Alamos National Laboratory
Los Alamos, NM 87545
feng@lanl.gov

Roy H. Campbell
Dept. of Computer Science
University of Illinois at Urbana-Champaign
1304 W. Springfield Ave.
Urbana, IL 61801
rhc@uiuc.edu

ABSTRACT

GREEN is a proactive queue-management algorithm that removes TCP's bias against connections with longer round-trip times resulting in a high degree of fairness, while maintaining high link utilization, low packet-loss, and low average queue sizes. GREEN applies the knowledge of the steady-state behavior of TCP connections to drop packets proactively, thus preventing TCP flows from inducing congestion. This prevents shorter round-trip time flows, which are more aggressive, from grabbing more than their fair share of bandwidth. As a result, GREEN ensures much higher fairness between flows than other recent active queue management schemes.

GREEN's performance relies on its ability to gauge a flow's round-trip time. Preliminary work on GREEN has focused on the performance of an ideal GREEN router, which is assumed to have knowledge of a flow's round-trip time and the number of active flows at the router. In this paper, we present a practical estimator for GREEN, and compare its performance against other estimators of round-trip time, including IDMaps, an Internet host distance-estimation service. We present two other round-trip time estimation techniques, *forced estimation* and *passive estimation*. We show that our solution is practical and maintains high fairness and link utilization, and low packet-loss rates and queue sizes.

*This work was supported by the U.S. Dept. of Energy's Laboratory-Directed Research & Development Program through Los Alamos National Laboratory contract W-7405-ENG-36

†Apu Kapadia was funded in part by the U.S. Dept. of Energy's High-Performance Computer Science Fellowship through Los Alamos National Laboratory, Lawrence Livermore National Laboratory, and Sandia National Laboratory.

1. INTRODUCTION

Because network congestion induced by TCP traffic leads to lost packets, thus wasting all the resources that the packet consumed on its way from source to destination, active queue-management (AQM) schemes such as RED [8], Blue [4], AVQ [12], REM [1], and PI [10] have been proposed to actively detect congestion early and appropriately react to the impending congestion that would otherwise fill the queue and cause a burst of packet drops. GREEN [6] is a *proactive queue management (PQM)* scheme which attempts to regulate TCP flows over the same link to a fair sending rate to prevent even well-behaved TCP flows from inducing congestion.

REM, PI, and AVQ are designed to increase the link utilization at a router while maintaining small queue sizes. While RED and Blue were designed to stabilize queue sizes at low levels, Flow Random Early Drop (FRED) [14] and Stochastic Fair Blue (SFB) [5] improve on their performance by operating at the flow level. FRED and SFB attempt to enhance throughput-fairness between flows by penalizing flows of higher bandwidth. Simulations of FRED and SFB in this paper show that they do not perform well on TCP flows of varying round-trip times (RTT). GREEN's congestion-prevention proactive queue management scheme is based on a mathematical model of the steady-state behavior of TCP's [15] congestion avoidance algorithm, so that well-behaved TCP flows can be regulated to receive their fair share of the bottleneck link bandwidth while simultaneously maintaining high link utilization and low packet-loss. However, this mathematical model relies on the ability of the router to infer the RTT of a flow. In [6] the authors presented preliminary results for an *ideal* GREEN router (referred to as *GREEN-Ideal*) that was assumed to have access to RTT information and the number of active flows. In this paper we present a practical solution for GREEN based on several RTT estimation techniques, including IDMaps [9], an Internet host distance estimation service, passive RTT estimation similar to that described in [11], and a novel technique that we call *forced estimation*. We also include an estimator for the number of flows. While IDMaps does not provide exact RTT estimates, we examine the impact of an IDMaps-based solution for GREEN and compare it with GREEN-Ideal. We show that even in the face of slightly inaccurate RTT estimates, GREEN still outperforms other

flow-based AQM schemes like FRED and SFB, while still maintaining high link utilization and low packet-loss. We then present results based on our passive and forced RTT estimation techniques, which offer a deployable solution for GREEN in today's Internet. We also highlight some of GREEN's limitations with regard to short-lived and low-bandwidth flows, and suggest future directions for GREEN in this regard. All our results are based on simulations using the *ns-2* [16] simulator.

The rest of the paper is organized as follows. In Section 2 we present the basic algorithm for GREEN. Section 3 briefly discusses the placement of GREEN routers. In Section 4 we discuss various AQM techniques and motivate the comparison of GREEN with FRED and SFB. Section 5 describes GREEN based on IDMaps. In Section 6 we compare the performance of GREEN-Ideal with GREEN using IDMaps, FRED, SFB, and Droptail. In Section 7 we introduce our passive and forced RTT estimation techniques. In Section 8 we evaluate a practical implementation of GREEN based on our RTT and flow estimators with FRED, SFB, and Droptail using three sets of experiments. Section 9 briefly discusses GREEN's state and computational requirements, and we finally conclude in Section 10.

2. ALGORITHM

GREEN applies knowledge of the steady-state behavior of TCP connections at the router to drop (or mark) packets intelligently for congestion notification. By using such a mechanism, a router can give each connection its fair share of bandwidth. The throughput of a TCP connection depends, among other factors, on its round-trip time (RTT) and the probability that its packets are dropped in the network. Specifically, Mathis et al. [15] show that a connection's throughput satisfies the following equation under certain simplifying assumptions:

$$BW = \frac{MSS \times c}{RTT \times \sqrt{p}} \quad (1)$$

where BW is the bandwidth/throughput of the connection, MSS is the maximum segment size, RTT is its round-trip time, and p is the packet-loss probability for that connection. c is a constant that depends on the acknowledgment strategy that is used (i.e., delayed or every packet) and whether packets are assumed to be lost periodically or at random.

In general, this model may not be applicable in environments where there are sustained multiple packet-losses for a flow within a single RTT (causing repeated timeouts). This model may also not apply to very short connections that never reach steady state, or to connections whose window sizes are artificially limited by the receiver's flow control window. For our analysis, we assume that all connections satisfy the assumptions required for this model. Specifically, our simulations focus on long-lived FTP connections that are able to attain their steady state bandwidths.

Now, let us consider a scenario where there are N active flows at a router on a particular outgoing link of capacity L. In GREEN, an active flow is a TCP source that has outstanding data to be sent. If a flow has had at least 1 packet go through the router within a certain *window* of time GREEN assumes that the flow is still active. The fair-share throughput of each flow is L/N (assuming each source attempts to transmit at least at that rate). Substituting L/N for BW in Equation (1), we derive the following expression for loss probability p:

$$p = \left(\frac{N \times MSS \times c}{L \times RTT} \right)^2 \quad (2)$$

By using this value of p as the dropping probability for congestion notification, GREEN "coerces" flows into sending at their fair rate. Note that GREEN applies this marking probability to *all* arriving packets, where the value of p depends on the flow. Because p depends on the number of flows and the round-trip time of each flow, congestion notification is more aggressive for large N and small RTT. And by including RTT as an inverse parameter in the equation, GREEN eliminates the bias of favoring TCP connections with smaller $RTTs$ with respect to throughput [13]. (Recall that TCP connections with smaller $RTTs$ can increase their window size faster due to the smaller RTT, and are more aggressive. These flows are able to grab more than their fair share of bandwidth, which leads to this bias.)

3. ISSUES AND PLACEMENT OF GREEN ROUTERS

End-to-end schemes have been proposed to correct for this bias by requiring TCP senders to increase their congestion windows by a constant proportional to the square of the RTT [7][17]. However, these schemes rely on a *window constant* that is hard to calculate and varies with the topology of the network. In contrast, not only can GREEN accurately calculate the drop probabilities irrespective of network topology, it also does not require any end-to-end modifications. In this paper we focus on developing developing a practical and deployable solution. Hence we examine GREEN's performance under a packet-dropping model (as opposed to a marking model, which relies on ECN-capable end points), and use TCP-Reno at the end-points of our simulations. Such a solution would only require a GREEN-capable router without any modification to the end-points.

Since GREEN calculates p, the drop probability for a flow, GREEN routers cannot be "composed," since this would alter the overall drop probability for a flow. Hence GREEN is mainly suited as an edge router, where organizations can enforce fairness between flows leaving the organization through a bottleneck link. This ensures that the GREEN drop probability is applied to a flow only once. Another ideal application of GREEN would be the outgoing link of an FTP server, where there are several competing TCP connections, which are long-lived and have varying RTT's. Another justification for keeping GREEN routers at the edge is that shorter-RTT flows, which cause unfairness at a link, are effectively limited closer to the source. Routers in the core of the network usually carry longer-haul flows, where GREEN's algorithms may not be too effective. Having GREEN routers limit shorter-RTT flows closer to the source is more desirable since this affects flows closer to the source. We hope to validate this claim more rigorously in future work.

Perhaps the main limitation of GREEN in its current form is that it assumes that it is the bottleneck router for all

flows. Hence a flow that is limited at some other bottleneck router will cause underutilization of the outgoing link at the GREEN router. This can be avoided by monitoring flows, and compensating for unused bandwidth. We leave this for future work.

4. RELATED WORK

Since RED [8] was not designed to discriminate between flows, RED applies the same loss rate to all flows irrespective of their bandwidths. Flow-RED (FRED) [14] attempts to remedy this "unfairness" by applying a loss-rate to a flow that is based on its buffer occupancy statistics. Hence, higher bandwidth flows that consume more resources at the router, receive a higher loss-rate. Since RED relies only on buffer-occupancy statistics, maintaining a stabilized queue is difficult without correct parameterization. Blue [4] proposes a different approach, which uses link utilization and packet-loss history to manage congestion. In particular, Blue increases the marking probability when packet-loss increases, and decreases the marking probability when link-utilization decreases. Blue does not discriminate between flows, and applies this marking probability to all flows irrespective of their bandwidths. Stochastic Fair Blue (SFB) [5] tries to correct this problem by applying a Bloom filter [2] to hash flows into L levels of N bins. Each bin maintains queue occupancy statistics for flows that map into that bin and a corresponding drop probability p_m. For a given flow, SFB calculates the marking probability based on queue occupancy statistics of the various bins. This approach is effective in applying a more aggressive marking probability to flows of higher bandwidth, and a low marking probability to flows of lower bandwidth. Hence, FRED and SFB are two recent AQM approaches that attempt to ensure bandwidth-fairness between flows.

In contrast to the approaches described above, more recent AQM schemes apply a more control-theoretic view to increase the link-utilization at a router. Random Exponential Marking (REM) [1] relies on REM-capable routers that support explicit congestion notification (ECN). Congestion measures are inserted into packets by these routers, and the overall marking probability is calculated based on this. This results in high utilization and low delay at the queues. The Proportional Integral (PI) [10] controller attempts to regulate the steady-state value of queue-size. The authors argue that PI has better theoretical properties and performance than RED and can stabilize queue sizes at low levels. Adaptive Virtual Queue (AVQ) [12] uses a *virtual queue* with a capacity less than the capacity of the link. Packets in the actual queue are marked when the packet induces an overflow in the virtual buffer. Following this, the link capacities are recalculated. Such an approach can result in the desired utilization at a link. While AVQ's performance stability relies on a delay bound (d) explicit RTT information for flows is not exploited to provide throughput fairness.

While these approaches are important in that they increase the link-utilization at a router and provide queue stability, they do not discriminate between flows and hence cannot ensure bandwidth fairness. In terms of bandwidth fairness, we expect the behavior of such schemes to be similar to that of Droptail (*FIFO* queueing with finite buffer) or RED, and hence do not compare GREEN to PI, REM, and AVQ. Instead we focus on FRED and SFB since they were designed to provide a higher degree of fairness between flows. Future work could include applying the GREEN algorithm to AQM schemes designed for higher utilization. For example, when such an AQM decides to mark a packet, a packet from the queue could be picked based on probabilities calculated by GREEN. This would enhance both the utilization *and* fairness at a router.

5. ESTIMATING RTT USING IDMAPS

In [6], the authors presented preliminary results for GREEN-Ideal, where the RTT was assumed to be known at the router. In this paper, we relax this constraint by making use of IDMaps. IDMaps [9] is a scalable Internet-wide service that aims to provide Internet distance estimates. For example, the authors have suggested that IDMaps can be used by hosts for nearest mirror selection. Such a service is also well suited to GREEN, which can obtain RTT estimates for flows using IDMaps. We propose an architecture where GREEN routers are part of the IDMaps framework, and therefore, can perform fast lookups in a local IDMaps database.

5.1 IDMaps - Architecture

Jamin et al. [9] argue that providing highly accurate delay estimates (within 5% for example) is not feasible. Instead they aim to provide a scalable solution with existing technology to provide delay estimates that are accurate to within a factor of two. Jamin et al. propose the deployment of *tracers* in the Internet. Tracers maintain raw distances amongst themselves and address prefixes (AP). The use of AP's, as opposed to actual IP addresses, makes this solution feasible, trading off accuracy for scalability. The delay between two IP addresses is estimated by calculating the sum of the delays between the two tracers closest to the two address prefixes, and the tracer-AP delays.

5.2 GREEN using IDMaps

We propose a solution in which GREEN routers also perform the duties of tracers and exchange distance information with other tracers. We do not expect this to add much overhead to existing traffic from routing updates. Furthermore, since GREEN is an edge router, the delays from sources within the organization to the GREEN router will be fairly low. GREEN can perform fast lookups in the local IDMaps database to obtain RTT estimates for a flow, based on the destination IP addresses (since the source IP address is assumed to be within the organization). GREEN calculates the drop probability based on the *estimated RTT*. The accuracy of IDMaps estimates is sensitive to the number of tracers and their placement on the Internet. Jamin et al. have evaluated several graph-theoretic approaches as well as simple heuristics. In general, the accuracy of estimates increases when tracers are closer to the AP's (Address Prefixes). As mentioned earlier, GREEN routers will be co-located with the AP's of that organization, and hence, will result in more accurate estimates.

6. GREEN PERFORMANCE

Here, we evaluate the performance of GREEN using IDMaps (GREEN-IDMaps) with GREEN-Ideal. We also compare the performance with respect to FRED and SFB. We make

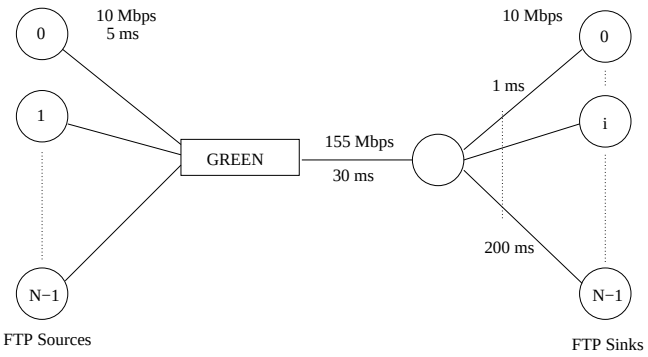


Figure 1: Network Topology

comparisons with FRED and SFB since these active queue management schemes are also flow-based. We also include results for Droptail queueing to provide a baseline for assessing performance.

We assume that a router knows the bandwidth (L) of the attached outgoing link. N is the number of active flows, i.e., flows that have had at least 1 packet go through the router within a certain *window* of time. Since active flows not experiencing repeated timeouts send several packets per RTT, we use $window = 1sec$, which results in near-perfect estimates in our simulations. We leave more fine tuning of the window parameter for future work. In our experiments, we chose MSS to be 1 KB. The value of c, in our "random dropping, acknowledgment per packet" model was fixed at 1.31 [15].

Since IDMaps aims to provide delay estimates within a factor of two, we simulate the effect of IDMaps by estimating each RTT to be a uniformly random number between one and two times the actual RTT (as inferred from the topology). Because of this, our model of IDMaps always results in over-estimates of the RTT. In the absence of real-world data for IDMaps RTT estimates, we submit that this model will give us an idea of the worst-case performance of GREEN-IDMaps, assuming that IDMaps will usually provide better estimates than this.

We used *ns* [16] to evaluate the performance of GREEN over a network with the topology shown in Fig. 1. We try to simulate an organizational topology with low latencies to the "left" of the bottleneck edge router (GREEN). We simulate connections of varying RTTs for the links on the "right" and vary their latencies linearly from $1ms$ to $200ms$. This results in RTT's varying linearly from $72ms$ to $470ms$. N Sources and N sinks are connected to the routers over $10Mbps$ links. We varied the number of flows N, from 50 to 400. The bottleneck link has a bandwidth of 155 Mbps and a delay of 30 ms.

We started FTP connections from the leftmost nodes to the rightmost nodes within the first $40sec$ of simulation and ran it for $180sec$. GREEN-Ideal and GREEN-IDMaps were implemented at the gateway, which is the bottleneck router in our simulation. All of the metrics presented in this section — link utilization, fairness, packet-loss, queue size — are measured at this gateway. We present results for link utilization, fairness, and queue size, after the first $50sec$ to remove the startup transient effects and to study the steady-state behavior.

6.1 Fairness

As mentioned in Section 2, GREEN attempts to regulate all the TCP flows to their fair share of the outgoing link bandwidth. We use Jain's Fairness Index [3] to assess GREEN's ability to maintain equal bandwidths between TCP flows. We briefly describe how the fairness index is calculated, and then present our results.

6.1.1 Jain's Fairness Index

Given the set of throughputs $(x_1, x_2, \ldots, x_n)$, the fairness index is calculated as follows:

$$f(x_1, x_2, \ldots, x_n) = \frac{(\sum_{i=1}^{n} x_i)^2}{n \sum_{i=1}^{n} x_i^2}$$

The fairness index always lies between 0 and 1. Hence, a higher fairness index indicates better fairness between flows. The fairness index is 1 when all the throughputs are equal. When all the throughputs are not equal, the fairness index drops below 1.

6.1.2 Results

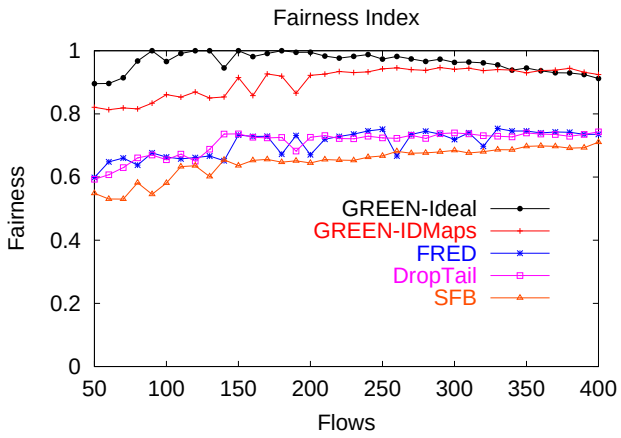
As shown in Fig. 2(a), GREEN-Ideal provides significantly higher bandwidth fairness than the other queue management schemes. The curve for Droptail shows us the fairness we would expect at most gateways in the Internet today. FRED is able to outperform Droptail and SFB because it queues at least two packets[1] of a flow before marking a packet from that flow. This provides much better fairness as long as each flow maintains one to two outstanding packets at the gateway. SFB exhibits poor fairness because it is sensitive to varying RTTs between flows, and breaks down under a large number of connections with varying RTTs [5].

Figure 2(b) best summarizes the benefit of GREEN's approach. At the end of our simulation with 200 FTP flows, we plot the number of packets sent by each flow. We can see TCP's inherent bias against larger-RTT flows in the curve for FRED (Droptail and SFB exhibited similar behavior, and we leave this out for clarity). However, we observe that GREEN has corrected TCP's bias, and all flows achieve the same average bandwidths at the end of the simulation.

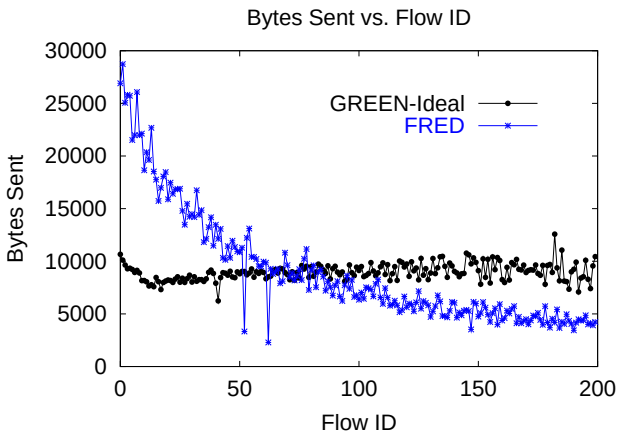
GREEN-IDMaps is able to achieve bandwidth fairness significantly better than FRED, SFB, and Droptail despite the rough RTT estimates. While the fairness provided by GREEN-IDMaps is not as good as that provided by GREEN-Ideal, we can see that a solution based on IDMaps is indeed practical and can be deployed in the Internet. The reason for this is that despite its inaccuracy in RTT estimation, IDMaps is able to distinguish between longer and shorter RTT flows.

6.2 Link Utilization

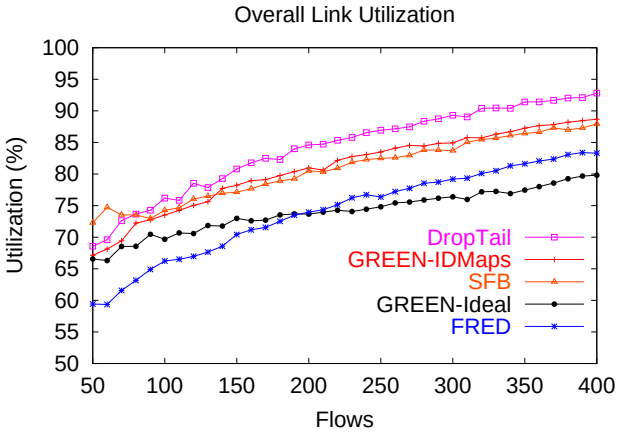
[1] In our experiments, we operate FRED under the *many-flow* mode.



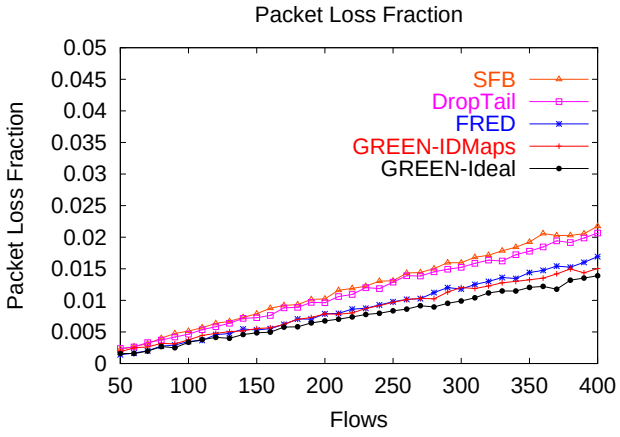
(a) Jain's Fairness Index vs. Number of Flows



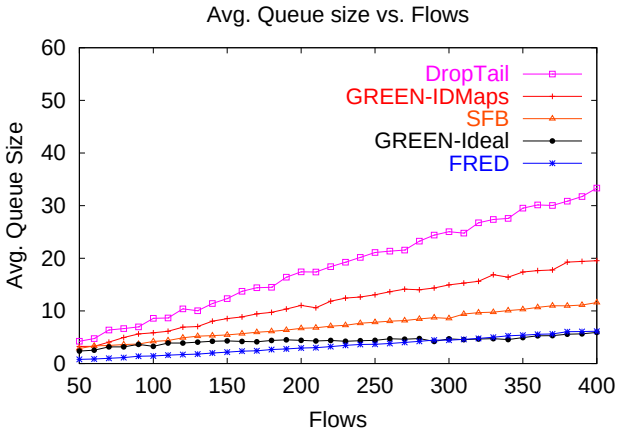
(b) Bytes Sent vs. Flow ID, RTT increases as Flow ID increases



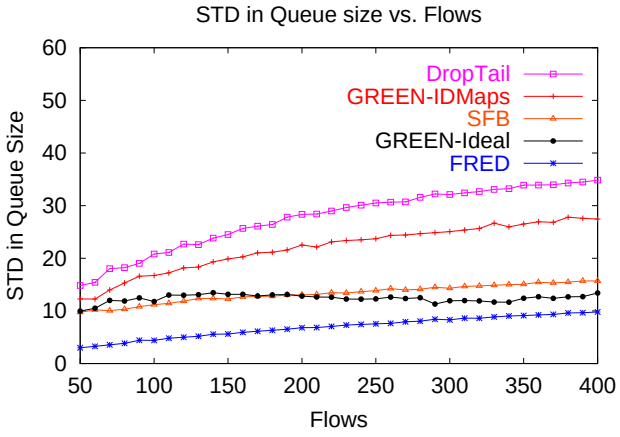
(c) Overall Link Utilization vs. Number of Flows



(d) Overall Packet-Loss vs. Number of Flows



(e) Avg. Queue Size vs. Number of Flows



(f) STD in Queue Size vs. Number of Flows

Figure 2: Comparison of GREEN's behavior with other approaches

Here we compare GREEN's performance with other schemes in terms of overall link utilization. At the end of each simulation, the overall link utilization is calculated as follows:

$$utilization = \frac{byte\_departures_t}{bandwidth \times t}$$

The numerator equals the total number of bytes delivered by the link during the interval of t *sec*, and the denominator equals the total possible bytes that could have left the link in the same interval.

Fig. 2(c) shows that GREEN-Ideal achieves higher link utilization than SFB and FRED because GREEN can maintain the average bandwidths of all flows, while FRED simply relies on queue-occupancy statistics to regulate queue size, and SFB relies on link utilization and packet-loss statistics to regulate queue size. Droptail achieves higher utilization because the flows with shorter RTTs are allowed to be aggressive. While this yields better link utilizations, it sacrifices fairness heavily, as seen in Fig. 2(a).

In all cases, GREEN-IDMaps achieves better utilization than GREEN-Ideal because GREEN-IDMaps overestimates the RTTs, which results in lower dropping probabilities. This in turn results in over-subscribing of the available bandwidth, which results in higher sending rates for all the flows and more queueing at the GREEN-IDMaps router[2]. However, as noted in Section 6.1, GREEN-IDMaps provides superior fairness compared to Droptail, FRED, and SFB. This makes the version of GREEN based on IDMaps attractive since it has high link utilization as well as a high fairness index. Hence, fairness is traded off for better utilization when the outgoing link is over-subscribed. Future work can look at tuning the amount of bandwidth over-subscribed based on the desired utilization.

6.3 Packet-Loss

As shown in Fig. 2(d), the packet-loss percentage is roughly the same for all flows and stays below 2%. Equation (1) provides good estimates when p is the order of a few percent [15]. Since the overall packet-loss stays below a few percent in our simulations, both GREEN-Ideal and GREEN-IDMaps are able to limit the rates of flows to their fair share of bandwidth. As mentioned in Section 6.2, GREEN-IDMaps underestimates the drop probabilities because it overestimates the RTTs. This results in Droptail like behavior when the queues get full, and hence we see a higher packet-loss rate than for GREEN-Ideal.

6.4 Queue Size

Queue sizes were sampled at $20ms$ intervals. The average and standard deviation were calculated at the end of the simulation. As we can see in Figures 2(e) and 2(f), the average queue size for Droptail increases dramatically as the number of flows increases. In contrast, GREEN-Ideal, FRED, and SFB are able to keep the average queue sizes low. GREEN-IDMaps' performance lies between that of SFB and Droptail. Even though the increase in queue lengths is not as dramatic as in Droptail, we can see how the rough RTT estimation affects GREEN-IDMaps. In our simulations, GREEN-IDMaps overestimates the RTT for a flow, which results in lower drop probabilities than GREEN-Ideal. In effect, GREEN-IDMaps over-subscribes the available bandwidth, which results in queues building up and exhibiting Droptail-like behavior when the link capacity is reached.

FRED keeps queue sizes low by marking packets beyond a certain threshold and limiting the amount of buffer-space for each flow. SFB does so by increasing drop rates when there is packet-loss and reducing drop rates when the link is underutilized. Hence, FRED and SFB attempt to dynamically converge to the correct "operating point" for low queue sizes. GREEN-Ideal achieves its operating point by calculating drop probabilities for each flow, based on their fair share of bandwidth. By ensuring that the aggregate bandwidth of the flows is equal to the available bandwidth at the link, there is no sustained buildup in queue length.

In summary, we see that GREEN performs well even when highly accurate RTT information is not available. This is because slightly inaccurate estimates are still effective in differentiating between flows of longer RTT's and shorter RTT's. The use of IDMaps, which is still a proposed service that has not yet been deployed, motivates the study of other practical RTT estimation techniques. We discuss two such techniques in the following section.

7. PASSIVE AND FORCED RTT ESTIMATION

In this section we introduce two techniques for estimating the RTT of a flow: *passive* and *forced* estimation, and show how a combined approach is practical and achieves high performance. We present results on the performance of these techniques in Section 8.

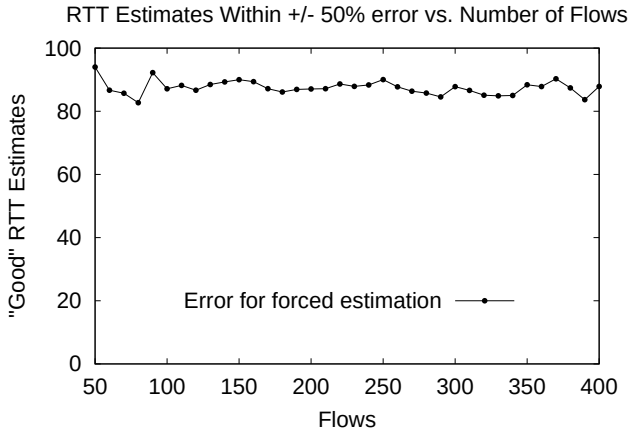


Figure 3: Performance of forced estimation with varying number of flows

7.1 Forced Estimation

We use a novel technique to estimate the RTT of a flow. Again, observing from Mathis' formula, we can derive an

[2]As mentioned earlier, this behavior is a result of our assumption that IDMaps overestimates the RTT. We expect that an actual deployment of an IDMaps service will perform better than this.

expression for the RTT of a flow:

$$RTT = \frac{MSS \times c}{BW \times \sqrt{p}} \quad (3)$$

In our simulations the GREEN router maintains per-flow statistics on the number of packets seen for a flow, the number of packets dropped at the GREEN router, and the time period over which these statistics are collected. For each flow that is being monitored, GREEN waits until enough data is available to apply Equation 3. This formula cannot be applied if no packets have been dropped for that flow. In our simulations, GREEN checks to see if there were any packet-drops at the end of $5sec$. If not, then the window is extended by another $5sec$ and so on. Once an RTT estimate is obtained, the flow is not monitored any more. Note that this technique requires the assumption that the GREEN router is the bottleneck router for that flow. To this effect, we induce a drop probability of 2% on flows during the observation period. We picked a value of 2% because this is at the higher end for which the model is still valid, and will ensure that enough data is available at the end of an observation period. This value also increases the chances that the GREEN router will act as the bottleneck during the observation period. It may be more desirable for a GREEN router to maintain average packet-loss statistics, and adaptively apply the average packet-loss rate to such a flow. We leave the fine-tuning of such parameters for future work.

We found that the margin of error for this technique is very similar to that of IDMaps. Fig. 3 shows that approximately 90% of RTT estimates are within ±50% of the actual RTT value. Recall that we assume IDMaps estimates to be a uniformly random number between one and two times the actual RTT. We refer to GREEN using forced estimation as *GREEN-Forced*. We expect GREEN-Forced to result in smaller average queue sizes than GREEN-IDMaps because GREEN-IDMaps always overestimates the RTT for a flow, whereas GREEN-Forced both underestimates and overestimates RTT's.

7.2 Passive Estimation

In [11], the authors present various techniques for passively estimating the RTT of a flow at a router. In our simulations, we use a technique similar to the Slow Start (SS) RTT estimation technique in [11]. Essentially, GREEN applies the knowledge of TCP's slow-start behavior to estimate the RTT of a flow. Recall that in slow-start, TCP doubles its congestion-window every RTT. GREEN logs the first four packets, where the first packet is assumed to correspond to the "first burst" of packets, the second (and third) packet to be from the next burst, and the fourth packet from the third burst. Hence the difference in times when the first and fourth packets are observed should be equal to $2 \times RTT$ for that flow. While we were able to accurately infer the RTT for most flows using this technique, the authors in [11] point out that in practice such passive estimation works in 55% to 85% of the cases. Passive estimation usually fails if a flow observes a packet loss during slow-start. In such a case, passive estimation results in extremely high RTT estimates. Hence, we apply forced monitoring to all flows with $RTT > 1sec$. As future work, GREEN could adaptively apply forced monitoring to flows of higher bandwidths, and refine the RTT estimates for those flows. From here on, we refer to this combined technique as *GREEN-Passive*. In our simulations we assume that the RTT for 70% of the flows can be inferred passively, and the rest of the flows are monitored using the forced technique described in the previous section. 70% of the flows are randomly identified for passive estimation at the beginning of the simulation and the rest are identified for forced estimation.

8. EXPERIMENTS AND EVALUATION

In this section we present three sets of experiments. The first set of experiments compares the various RTT-estimation techniques and their impact on GREEN. We show how GREEN-Passive is a practical solution which enhances the performance of GREEN-Forced. In the second set of experiments we introduce a more dynamic environment by introducing background FTP traffic at two intervals. Here we compare GREEN with other AQM approaches. The third set of experiments is similar to the second set of experiments, but with background *Pareto* traffic. This simulates GREEN's behavior with low-bandwidth background traffic, and brings to light some of GREEN's limitations in its current form.

8.1 Experiment 1

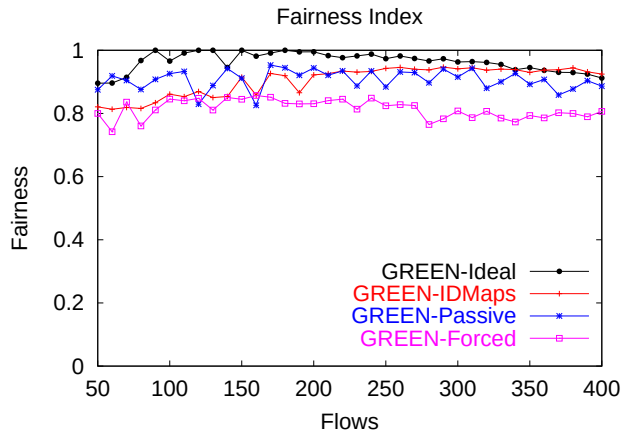
In Section 6, we compared GREEN-Ideal and GREEN-IDMaps with other AQM approaches. This motivated the need for better or alternative RTT-estimation techniques, which we described in Section 7. We use the same experimental setup as in Section 6, and compare GREEN-Ideal, GREEN-IDMaps, GREEN-Forced, and GREEN-Passive with each other.

8.1.1 Fairness

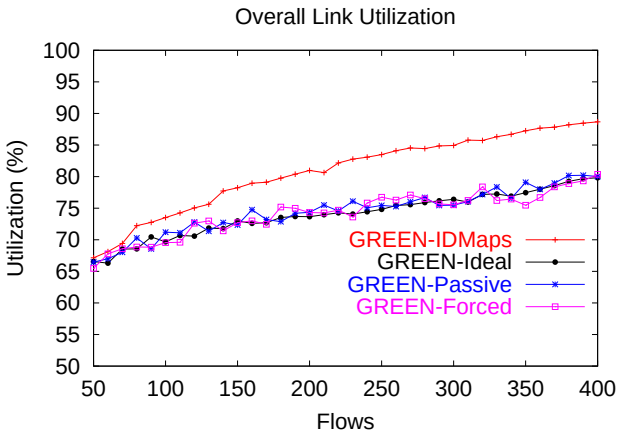
As we can see in Fig. 4(a), GREEN-Forced results in a high degree of fairness. While GREEN-Forced outperforms other AQM schemes (refer to curves in Fig. 2(a)), as expected it does not perform as well as GREEN-Ideal. We also note that GREEN-IDMaps also performs better. This is because we assume GREEN-IDMaps is able to infer *all* RTT's within 1 to 2 times the actual value. While GREEN-Forced has a similar margin of error, in Section 7.1 we showed that GREEN-Forced is not able to produce acceptable estimates for about 10% of the flows. This results in a slightly degraded performance as compared to GREEN-IDMaps. However, when we use GREEN-Passive, which augments GREEN-Forced with highly accurate estimates for about 70% of the flows, we see that GREEN-Passive performs very well, and is comparable to GREEN-IDMaps in fairness.

8.1.2 Link Utilization and Queue Size

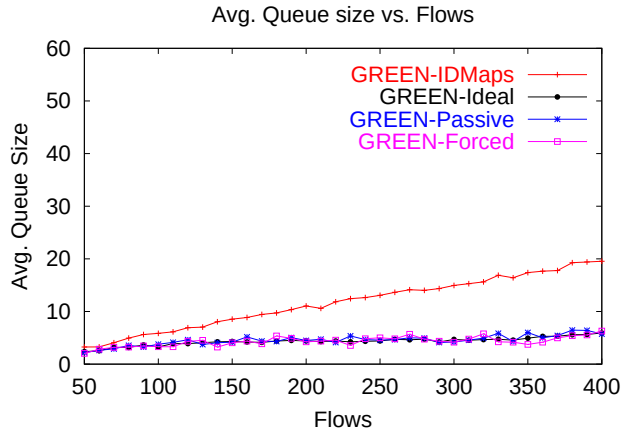
In Section 6.2 we saw that GREEN-IDMaps had higher utilization because it over-subscribed the available bandwidth, which resulted in higher average queue sizes. In Figures 4(b), 4(c), and 4(d), we see that GREEN-Forced, GREEN-Passive, and GREEN-Ideal all have comparable link utilizations, average queue sizes and variance in queue size. This is because while IDMaps over-subscribes the link bandwidth, GREEN-Forced both under and over-subscribes the link bandwidth. These effects act to cancel each other out, and the overall utilization is similar to that of GREEN-Ideal.



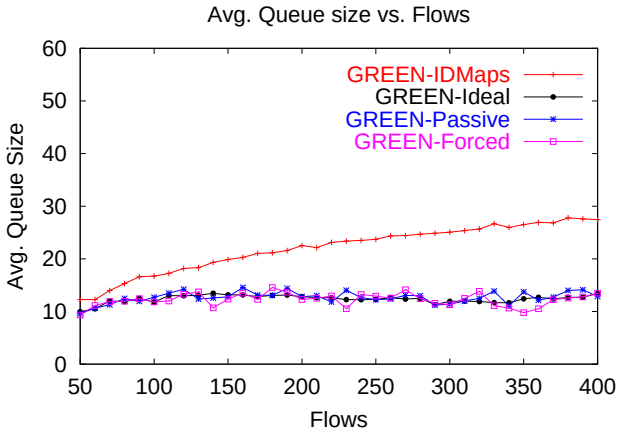
(a) Jain's Fairness Index vs. Number of Flows



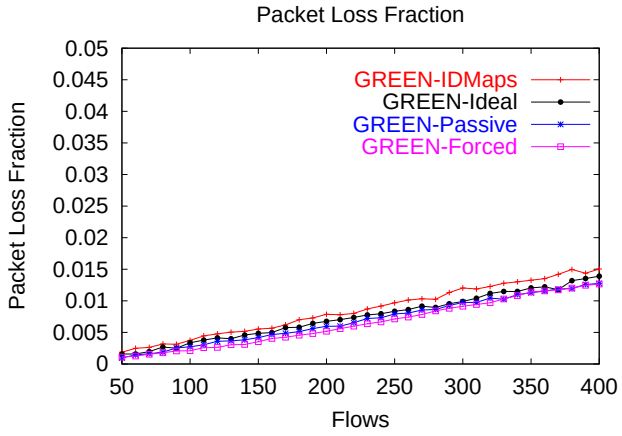
(b) Overall Link Utilization vs. Number of Flows



(c) Avg. Queue Size vs. Number of Flows

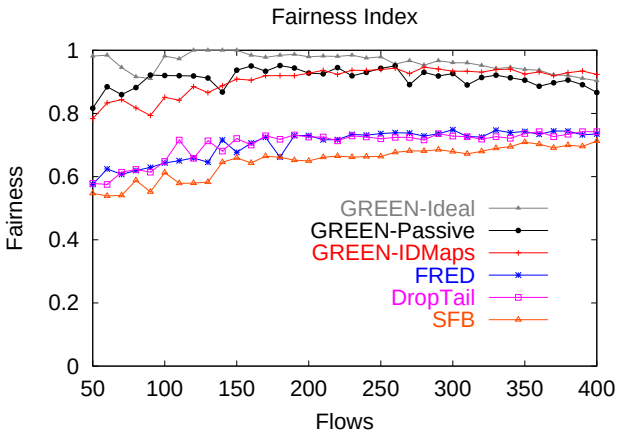


(d) STD in Queue Size vs. Number of Flows

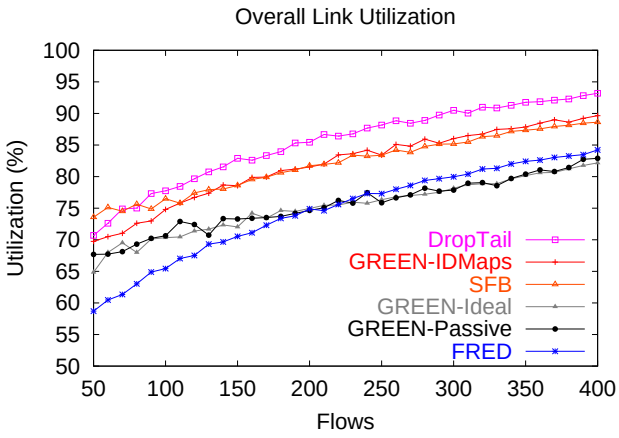


(e) Overall Packet-Loss vs. Number of Flows

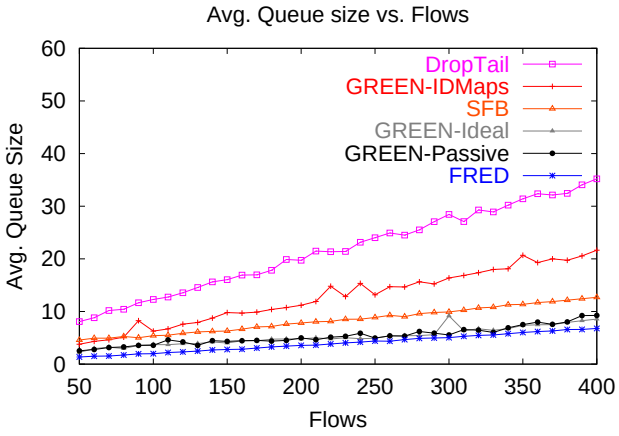
Figure 4: Experiment 1: GREEN's Performance with various RTT Estimators



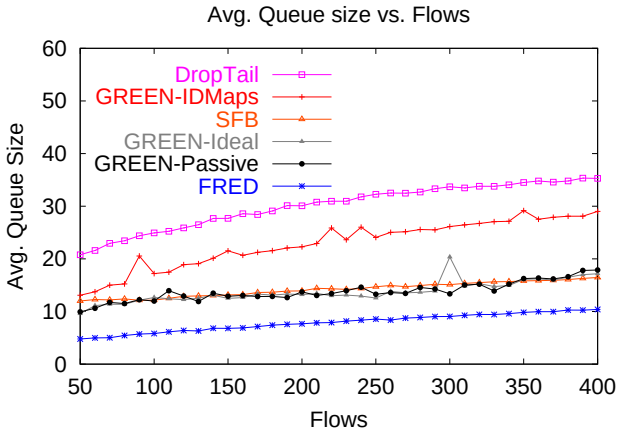
(a) Jain's Fairness Index vs. Number of Flows



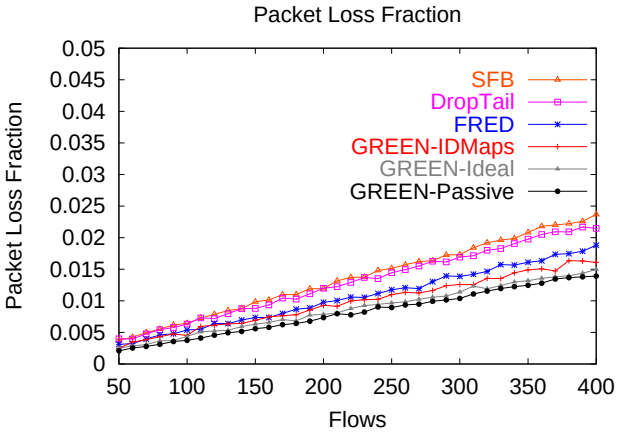
(b) Overall Link Utilization vs. Number of Flows



(c) Avg. Queue Size vs. Number of Flows

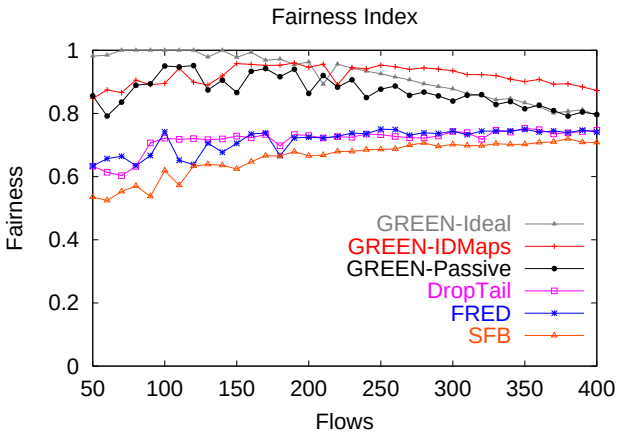


(d) STD in Queue Size vs. Number of Flows

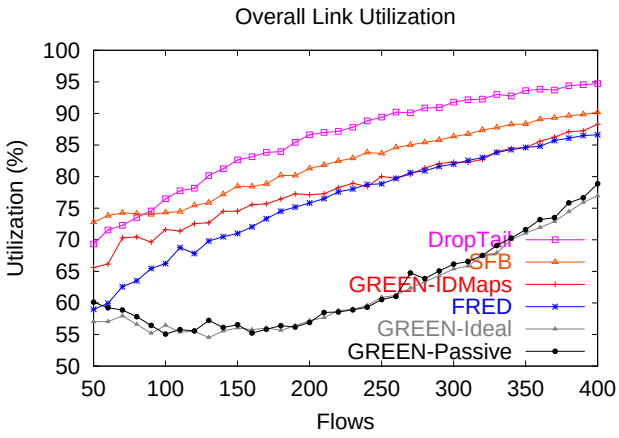


(e) Overall Packet-Loss vs. Number of Flows

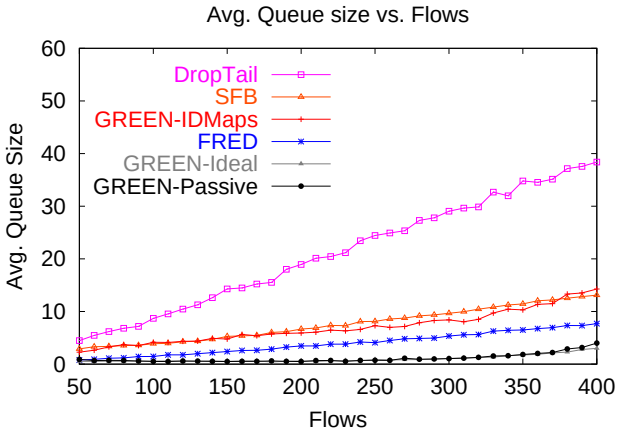
Figure 5: Experiment 2: Comparison of GREEN with other AQM schemes, background FTP traffic



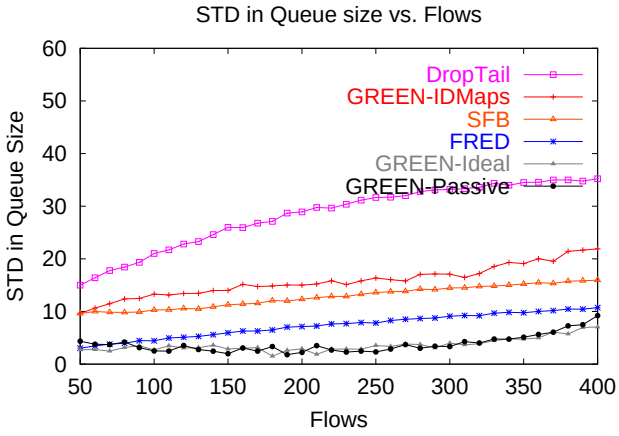
(a) Jain's Fairness Index vs. Number of Flows



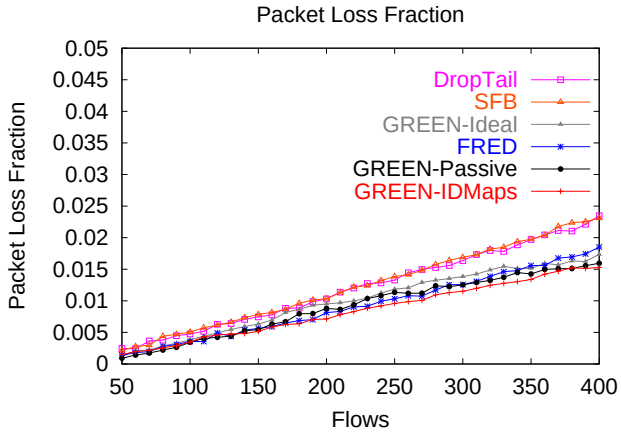
(b) Overall Link Utilization vs. Number of Flows



(c) Avg. Queue Size vs. Number of Flows



(d) STD in Queue Size vs. Number of Flows



(e) Overall Packet-Loss vs. Number of Flows

Figure 6: Experiment 3: Comparison of GREEN with other AQM schemes, background pareto traffic

8.1.3 Packet-loss

We note from Fig. 4(e) that the packet-loss is below 2% for all schemes. We omit further discussion on packet-loss in subsequent experiments as the packet-loss trends are similar and offer little new information.

From these experiments, it is clear that GREEN-Passive is the technique of choice. It behaves most like GREEN-Ideal, with only slightly lesser fairness. GREEN-IDMaps results in slightly better fairness than GREEN-Passive, but in much higher average queue sizes and variance. In subsequent experiments, we do not simulate GREEN-Forced because GREEN-Passive is an enhancement to GREEN-Forced.

8.2 Experiment 2

In this experiment, we introduce 50 FTP background traffic connections which are active from $80s$ to $100s$ and $140s$ to $160s$. This attempts to simulate different loads at the GREEN router twice during the simulation, and show how GREEN is able to maintain performance under such conditions. These FTP connections are composed of sources connected to the GREEN router through $5ms$, $10Mbps$ links, and receivers connected through similar links on the other side of the bottleneck link.

8.2.1 Results

In Fig. 5(a) we can see that GREEN-Passive produces high fairness and is comparable to GREEN-IDMaps. FRED, Droptail, SFB all have low fairness. The reasons are as explained in Section 6.1. The remaining results are similar to that of Section 6. This shows us that GREEN-Passive performs well even under large changes in traffic load. However, we note that all the traffic in this experiment is FTP traffic, and we examine the effects of short lived connections in the next experiment.

8.3 Experiment 3

In this experiment we pair each FTP flow, with a corresponding Pareto flow, which shares the same link characteristics of the FTP flow. This is essentially two copies of the topology shown in Fig. 1, sharing the same bottleneck link and GREEN router. One copy runs the FTP flows discussed in the previous experiments, and the other copy runs Pareto flows with the following parameters: $packetsize = 1000bytes$, $bursttime = 2sec$, $idletime = 4sec$, and $rate = 160Kbps$. This simulates short-lived, low-bandwidth TCP connections, which do not attain their steady-state bandwidths. Since GREEN assumes that all active flows will make use of their fair share of bandwidth, in Fig. 6(b) we observe that the utilization is lower than that of other schemes. This is because GREEN allocates the Pareto traffic with the same bandwidth as the FTP flows. However, the Pareto traffic does not utilize this bandwidth, which results in under-utilization at the link. One remedy to this situation would be to have GREEN recognize short-lived or low-bandwidth flows like HTTP and telnet, and ignore or compensate for these flows in its computation. We leave this as future work. Nevertheless, we see the usual characteristics of GREEN, with GREEN-Passive exhibiting high fairness in Fig. 6(a), and low average queue sizes and variance in Figures 6(c) and 6(d). These low queue sizes are expected as a consequence of GREEN's underutilization.

9. STATE REQUIREMENTS

The basic operation of GREEN-IDMaps does not require per-flow state information. N and MSS can be easily estimated. Since we propose that GREEN-IDMaps routers operate as IDMaps tracers, GREEN-IDMaps will maintain state proportional to the number of tracers deployed in the Internet. The amount of state used depends on how the tracers are connected through virtual links. This is discussed in more detail in [9]. GREEN-Passive maintains a small amount (less than 100 bytes) of per-flow state to estimate the RTT of a flow (using passive and forced estimation). FRED keeps per-flow state information for flows that have packets buffered at the link. SFB does not maintain per-flow state information, but instead, employs a Bloom filter [2] to hash flows into L levels of N bins. Each bin maintains queue occupancy statistics for flows that map into that bin and a corresponding drop probability p_m. Hence, SFB's state requirement is $O(N * L)$. A discussion on the selection of L and N is discussed in [5].

Note that the computational requirements for GREEN are not that demanding. For each packet, GREEN updates flow information and calculates the corresponding drop probability for that flow. We compare this with SFB, which computes L hashes for each packet, updates statistics for that flow, and then calculates the drop probability for that flow.

10. CONCLUSION

In this paper, we explore a practical implementation of GREEN based on IDMaps. Since IDMaps is still a proposed service that has not yet been deployed, we developed another method, which we called GREEN-Passive. GREEN-Passive uses a combination of *passive* and *forced* RTT-estimation. We simulated an organizational topology and show how a practical implementation of GREEN would perform as an edge router, providing high fairness and link utilization, and low average queue sizes and packet-loss for an organizational network.

We also showed that GREEN in its present form results in low link-utilization under the presence of short-lived or low-bandwidth connections. This is because GREEN assumes that each flow will actually use its fair share of steady-state bandwidth. While this is true for long-lived flows like file transfers, this is not true for short-lived or low-bandwidth connections. This behavior can be remedied by enhancing GREEN to recognize short-lived or low-bandwidth flows (e.g., HTTP and telnet traffic), and factor this into its fair bandwidth calculations.

11. ACKNOWLEDGMENTS

The authors wish to thank Prof. Wu-chang Feng from the OGI School of Science & Engineering at the Oregon Health and Science University for his insightful comments and feedback on his Blue and Stochastic Fair Blue (SFB) active-queue management schemes as well as his contributions on the framework and viability of the GREEN approach.

We also would like to thank the following people: Sunil Thulasidasan for his initial implementation of GREEN and Prasad Naldurg and Geetanjali Sampemane for their helpful comments.

12. REFERENCES

[1] S. Athuraliya, V. H. Li, S. H. Low, and Q. Yin. REM: Active Queue Management. In *IEEE Network*, June 2001.

[2] B. Bloom. Space/Time Trade-offs in Hash Coding with Allowable Errors. *Communications of the ACM*, 13(7), July 1970.

[3] D. Chiu and R. Jain. Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks. *Computer Networks and ISDN Systems*, 17:1–14, 1989.

[4] W. Feng, D. Kandlur, D. Saha, and K. Shin. BLUE: A New Class of Active Queue Management Algorithms. Technical report, April 1999.

[5] W. Feng, D. Kandlur, D. Saha, and K. Shin. Stochastic Fair Blue: A Queue Management Algorithm for Enforcing Fairness. In *Proc. of IEEE INFOCOM 2001*, April 2001.

[6] W. Feng, A. Kapadia, and S. Thulasidasan. GREEN: Proactive Queue Management over a Best-Effort Network. In *Proc. of IEEE Globecom 2002*, November 2002.

[7] S. Floyd. Connections with multiple congested gateways in packet-switched networks part1: One-way traffic. *ACM Computer Communication Review*, 21(5):30–47, Oct. 1991.

[8] S. Floyd and V. Jacobson. Random Early Detection Gateways for Congestion Avoidance. *IEEE/ACM Transactions on Networking*, 1(4):397–413, August 1993.

[9] P. Francis, S. Jamin, C. Jin, Y. Jin, D. Raz, Y. Shavitt, and L. Zhang. IDMaps: A Global Internet Host Distance Estimation Service. *IEEE/ACM Transactions on Networking*, 2001 October.

[10] C. V. Hollot, V. Misra, D. Towsley, and W. Gong. On Designing Improved Controllers for AQM Routers Supporting TCP Flows. In *IEEE INFOCOM*, 2001.

[11] H. Jiang and C. Dovrolis. Passive estimation of TCP round-trip times . In *ACM SIGCOMM Computer Communication Review*, volume 32(2). July 2002.

[12] S. Kunniyur and R. Srikant. Analysis and Design of an Adaptive Virtual Queue (AVQ) Algorithm for Active Queue Management. In *ACM SIGCOMM*, 2001.

[13] T. Lakshman and U. Madhow. The Performance of TCP/IP for Networks with High Bandwidth-Delay Products and Random Loss. *IEEE/ACM Transactions on Networking*, 5(3):336–350, June 1997.

[14] D. Lin and R. Morris. Dynamics of Random Early Detection. In *Proc. of ACM SIGCOMM 1997*, September 1997.

[15] M. Mathis, J. Semke, J. Mahdavi, and T. Ott. The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm. *Computer Communication Review*, 27(3), July 1997.

[16] ns-2. Network Simulator. http://www.isi.edu/nsnam/ns.

[17] T.Henderson, E. Sahouria, S. McCanne, and R. Katz. On Improving the Fairness of TCP Congestion Avoidance. In *Proc. of IEEE Globecom*, November 1998.