Analysis of RED-Family Active Queue Management Over a Wide-Range of TCP Loads

Jae Chung and Mark Claypool
Computer Science Department
Worcester Polytechnic Institute
Worcester, MA 01609, USA
1-508-831-5357
{goos|claypool}@cs.wpi.edu

Abstract

RED is an Active Queue Management (AQM) technique that is intended to achieve high link utilization with a low queuing delay. Recent studies show that RED is difficult to configure for some rapidly changing traffic mixes and loads [2]. Other studies show that under some conditions, the performance gains of RED and its variants over traditional drop-tail queue management is not significant given the additional complexity required for proper configuration [9, 3]. Recent variants of RED, such as Adaptive-RED [8], are designed to provide more robust RED performance under a wider-range of traffic conditions. This paper provides additional analysis of RED and newer variants of RED over a wider range of traffic mixes and loads than has previously been studied. Through extensive simulation results and analysis, this paper confirms that RED-like AQM techniques that employ packet dropping do not significantly improve performance over that of drop-tail queue management. However, when AQM techniques use Explicit Congestion Notification (ECN) as a method to notify TCP sources of congestion rather than packet drops, the performance gains of AQM in terms of goodput and delay can be significant over that of drop-tail queue management.

1 Introduction

To prevent congestion collapse, the current Internet uses end-to-end congestion control where responsive traffic sources, like TCP, adjust their transmission rate based on the network congestion information they detect while monitoring their own transmission. In the network, traditional drop-tail routers implicitly notify end-systems of network congestion by dropping incoming packets when the router buffer overflows. Unfortunately, under heavy load, drop-tail routers can result in consistently full router queues and bursty packet drops.

Active Queue Management (AQM) enhances network support for end-to-end congestion control by having routers actively detect impending congestion and notify end-systems, allowing responsive end-systems to adjust transmission rate earlier and avoid unwanted packet drops. AQM can also reduce queuing delays by keeping a lower average queue length. Moreover, since AQM routers are able to predict impending congestion before buffer overflows, they can explicitly notify end-systems of network congestion by using Explicit Congestion Notification (ECN) [5], rather than implicitly notify end-systems by dropping packets.

Random Early Detection (RED) [6] is a well-known lightweight AQM that uses the average queue size and minimum threshold (min_{th}) and maximum threshold (max_{th}) to detect impending congestion and determine congestion notification probability. When the average queue size is in between min_{th} and max_{th}, RED randomly drops (or marks in ECN mode) incoming packets with a probability that increases linearly from zero to the maximum probability (max_p) as the average queue size grows from min_{th} to max_{th}. However, when the average queue size is greater than max_{th}, RED drops all incoming packets until the average queue size drops below max_{th}.

Studies show that RED can improve throughput and fairness over drop-tail queue management while maintaining a low average queuing delay [6, 10]. However, this benefit is only achieved for "well-configured" RED under some traffic loads, specifically when the average queue length does not significantly oscillate and stays under max_{th}. Other researchers conclude that RED is too complicated to configure, and show that end-to-end performance of RED is no better than drop-tail queue management, and may even be worse than that of drop-tail queue management in some cases [2, 3, 9]. While RED does ensure a tightly bounded upper limit on the average queue length, it can result in many consecutive packet drops [3] that can significantly decrease end-to-end throughput and goodput by causing TCP timeouts. These reports raise the concern that using RED router queue management is not practical in a real Internet environment over a wide-range of TCP loads.

Some of the difficulties in RED configuration can be explained by a TCP-RED feedback control system theory [4] indicated in Figure 1. Assuming a single congested router that uniformly drops incoming packets, the packet drop rate at the router determines a stable state average

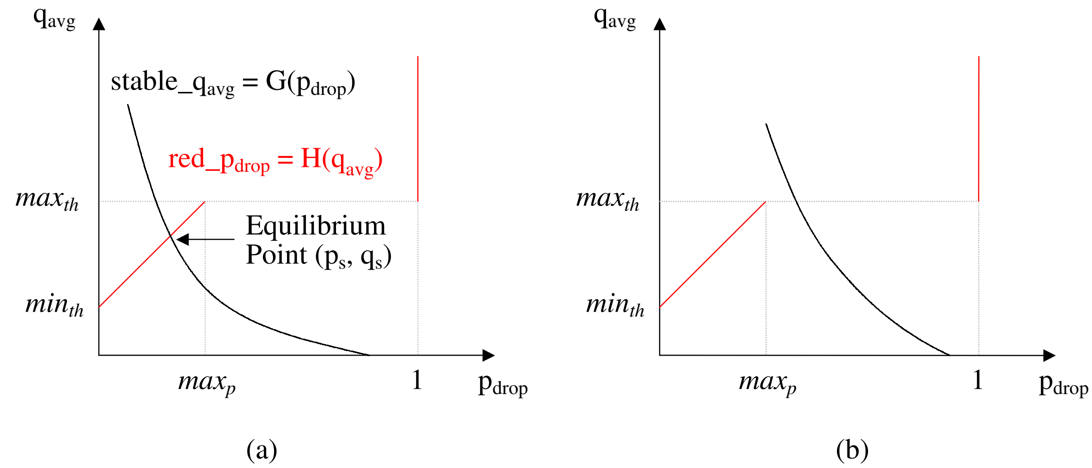

Figure 1. Well-configured RED (a) and poorly configured RED (b) [4]

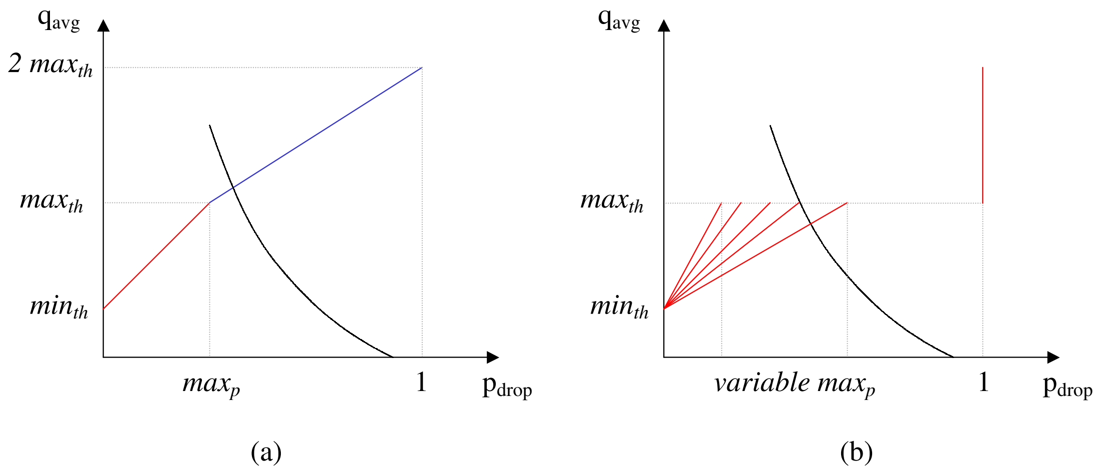

Figure 2. RED in Gentle Mode (a) and Adaptive RED (b)

queue length for a given TCP traffic load indicated by the curve labeled "stable avg_q." As shown in Figure 1 (a), if RED is configured such that there exists a drop probability within max_p whose RED function average queue length and the resulting stable state average queue length are the same, the queue can stabilize without many oscillations. In this case, we say RED is "well-configured" for the traffic load.

However, RED configurations that work well for one traffic mix and load may not work well for another traffic mix and load, especially as an increase in traffic load moves the stable state queue curve to the right. When the stable state average queue length at max_p is above max_{th}, as shown in Figure 1 (b), the RED average queue grows beyond max_{th} in order to bring the feedback control system to a stable state. Yet, once the average queue is above max_{th} RED drops all incoming packets until the average queue length goes down below the max_{th}. This periodic behavior of the average queue length growth and drop rate of 1 creates queue oscillations around max_{th} resulting in bursts of packet drops larger than would drop-tail bursts, often degrading end-to-end performances significantly. Although general RED configuration guidelines that work well for a large set of traffic load have been proposed [4], RED configuration difficulties will exist where Internet traffic varies.

As a fix to the above problem, the "gentle" modification to RED was proposed [10], which replaces the packet drop behavior when the average queue size is over max_{th} as shown in Figure 2 (a). Instead of setting the drop probability to 1 after the average queue size goes over max_{th}, gentle-RED linearly increases the drop probability from max_p to 1 as average queue size grows from max_{th} to 2 times max_{th}. This modification loosens the bound on the average queue length for a continuous probabilistic drop behavior. In other words, gentle-RED allows to find a stable state drop probability over max_p that results in a stabilized average queue length at some point greater than max_{th}.

As an effort to make RED well configured under a wider range of conditions, researchers recently proposed Adaptive RED (A-RED) [8] which tries to adapt to changing traffic load by slowly adjusting max_p as shown in Figure 2 (b). A-RED tries to dynamically configure itself to a well-configured state by defining a target region for the average queue within min_{th} and max_{th}. A-RED seeks the average queue target region by additively increasing max_p up to a limit (0.5 in default) if the average queue size goes above the region and multiplicatively decreasing max_p down to a limit (0.001 by default) in case the average queue size goes below the region. In short, A-RED tries to find a slope for the dropping probability that can intersect the queue law curve to make the feedback control system stable for current traffic load. However, A-RED still does not guarantee that it will find a slope within the range given by the limit for the max_p, in which case an unstable queue oscillation will take place as the case of original RED. For this reasons, it is also recommended to use the "gentle" setting with A-RED.

Unfortunately, there are reports even with the "gentle" modification, RED is not practical to use since the performance gain over drop-tail queue management is not significant given the complexity of implementation [9, 3]. These reports show that gentle-RED can achieve a lower average queuing delay than drop-tail, compensating for a lower goodput (or higher packet loss rate) under fairly heavy traffic load. Although it is attractive that RED and its variants give us control over average queuing delay, especially when considering Quality of Service (QoS) for interactive multimedia applications, improving goodput is critical, since it is a measure of how efficiently network resources are used without wasting bandwidth.

It is true that the overall network performance gain over drop-tail queue management may be relatively small when RED uses packet drop as means of congestion notification since packet drops directly degrade goodput. However, when using only ECN marks, the cost of congestion notification in terms of packet drop rate goes down to zero and has no effect on goodput. Thus, the real potential for RED and its variants (or AQM in general) lies in using ECN in a well-configured state, where packet drop rate at the router could be close to 0, queuing delay can be low, while still achieving a high link utilization. Then, the performance gains of AQM would be significant over that of drop-tail.

This paper seeks to demonstrate that Adaptive RED [8] using ECN can be "well-configured" for a wide range of traffic mixes, achieving significant performance gains over drop-tail queue management.

In Section 2, this paper develops a simple but effective model for load on the router from TCP traffic that captures a wide-range of key TCP flow characteristics, including long-lived and short-lived flows.

In Section 3, this paper applies the traffic model to ECN traffic to show key router configuration characteristics in order to be well-configured in the presence of ECN traffic. This paper verifies that the traffic model is effective for ECN traffic, and provides support for recommendation that RED (or AQM in general) should apply a much higher marking rate for ECN traffic than for TCP traffic.

In Section 4, using the model for AQM with drops and marks, this paper determines a set of RED configurations that illustrates the behavior of RED and its variants well, and measure the performance of RED, gentle-RED, A-RED, RED-ECN, gentle-RED-ECN and A-RED-ECN over a continuum of TCP traffic loads. We compare the performance of the above AQMs with the predictions made using the queue laws to verify the usefulness and correctness of the our model. At the same time, we compare the performance of the RED family AQMs with one another in terms of packet loss rate, delay and queue oscillation, also compared with that of drop-tail queue management.

In Section 5 this paper concludes that RED family AQMs, particularly Adaptive RED using ECN, can, indeed, be "well-configured" for variety of TCP traffic mixes, achieving both a very low network packet drop rate and a low queuing delay, which can never achieved with drop-tail queue management alone.

2 Router Load from TCP Traffic

In general, load at a router queue can be expressed as the ratio of the incoming data rate over the outgoing service rate. Increased load at a router is typically detected by an increase in average queue size, an increase in packet drop rate, or both. Firoiu and Borden analytically derive a "queue law" that governs the average size of a router queue given a fixed drop probability and a fixed set of TCP flows [4]. We extend the queue law to encompass more TCP characteristics so that we may better understand the impact of TCP traffic on the router queue.

We ran a series of NS [11] simulations using the network setup shown in Figure 3. For the traffic sources, we used bulk transfer FTP applications on top of TCP NewReno and set the *cwnd_limit* of all TCP agents to infinite. For the congested router queue, we implemented an infinite queue that randomly drops incoming packets with a uniform drop probability as in [4].

Figure 4[1] shows the queue law of the congested router for simulations that differ only by the number of TCP connections: 100, 200 or 300. At a given drop rate where all of the queue averages are greater than 0, the average queue size increases linearly with the number of flows.

Figure 5 shows the queue law of the congested router for simulations that differ only by the link bandwidth: 5, 10, 15 and 20 Mbps. At a given drop rate where all of the queue averages are greater than 0, the average queue size decreases linearly with the link bandwidth.

[1] "rtld" denotes round trip link delay in this and subsequent figures

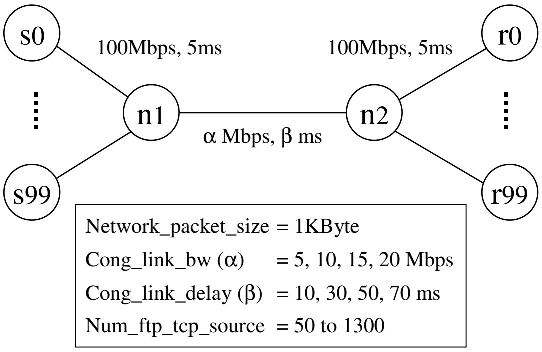

Figure 3. Simulation Network Setup.

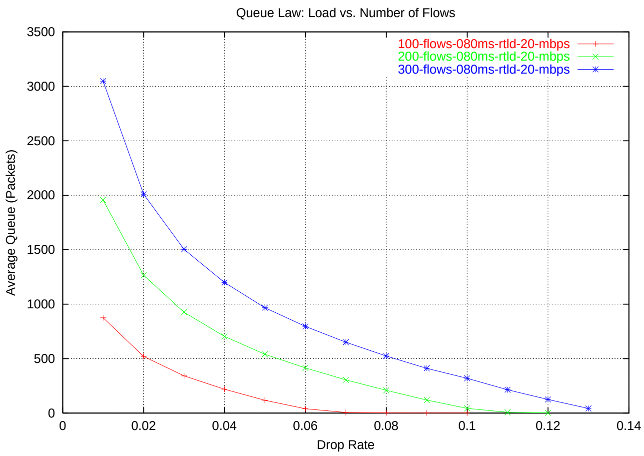

Figure 4. Queue Law: Load vs. Number of Flows

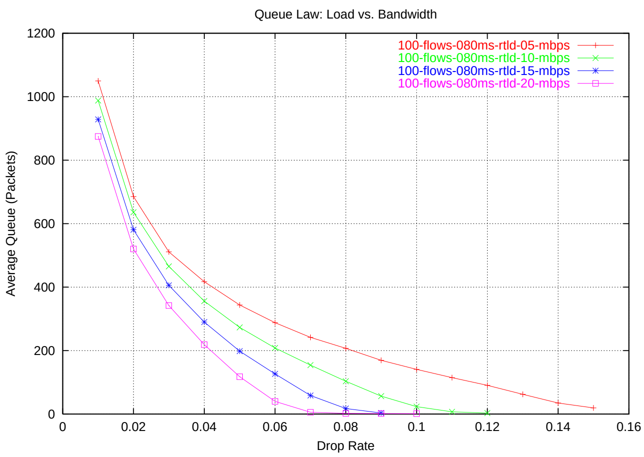

Figure 5. Queue Law: Load vs. Bandwidth

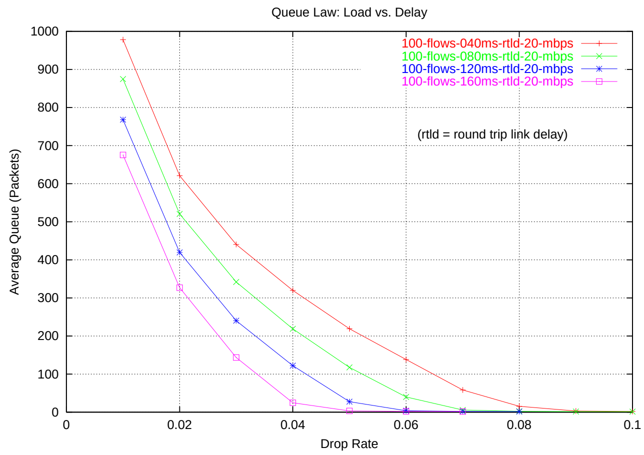

Figure 6. Queue Law: Load vs. Delay

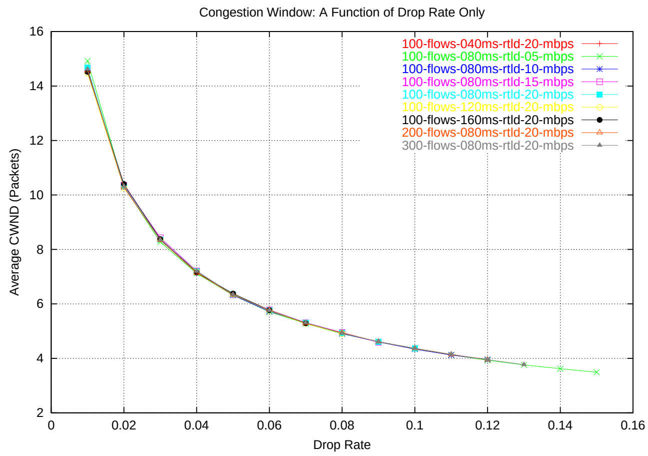

Figure 7. Congestion Window Size Average over All TCP Flows

Figure 6 shows the queue law of the congested router for simulations that differ only in the average round trip link delay of each source: 40, 80, 120 and 160 ms. At a given drop rate where all of the queue averages are greater than 0, the average queue size increases linearly with the round trip link delay.

The bandwidth used by a TCP flow per round trip time is directly proportional to the size of *cwnd*. Figure 7 shows the *avg_cwnd* vs. the uniform drop rate for various combinations of link bandwidth, number of TCP flows and round trip link delay settings. The *avg_cwnd* curves from various simulation configurations directly overlap one another, indicating that *avg_cwnd* is the function of the random packet drop rate only.

Thus, the equation for load (L) at the router is:

$$L = \frac{n \times avg\_cwnd(p)}{B \times avg\_rtt} \tag{1}$$

where n is the number of TCP flows, *avg_rtt* is the average round trip link delay, B is the link bandwidth capacity, p is the packet drop rate, and *avg_cwnd(p)* is the average *cwnd* value over all flows. Equation 1 implies that TCP traffic load is linearly proportional to the number of connections (or flows), and inversely linearly proportional to the link bandwidth and average round trip link delay.

In the above analysis, the congestion window size limit (*cwnd_limit*) of all TCP flows is infinite, and the TCP flows have an infinite amount of data to transmit. In real networks, TCP connections have congestion window size limits and finite amounts of data to transmit.

In Figure 7, the average congestion widow (*avg_cwnd*) was the function of packet drop rate only, which gives a smooth *avg_cwnd* curve for all of the network and traffic configurations. However, in real networking environments where factors such as the congestion window limit and data object size impose different window operation limits and alter the congestion response behavior of TCP flows, the *avg_cwnd* curve may not be smooth nor the same from one TCP traffic mix to another.

As a typical example, consider a TCP traffic mix that consist entirely of short-lived Web flows in which small Web objects limit the congestion window growth before the transmission ends. The congestion window size averaged over all connections in average will often be less than in the case of unlimited thresholds given the same drop rate, especially a low drop rates.

We illustrate this in simulations by setting the TCP congestion window limits to a low value, assuming this is similar to downloading a small network object, and drawing queue law graphs. The congestion window sizes for all TCP sources are set to first 12 packets, and then 6 packets. The number of TCP connections used in the simulations is 700 and 1300 correspondingly in order to have the same queue average at a drop rate of 0.01. For this set of simulations, the congested link bandwidth and the round trip link delay is set the same as in the previous unlimited congestion window simulation that had 300 TCP connections. Figure 8 shows the *avg_cwnd* graph, and Figure 9 shows the corresponding queue law graph.

Figure 8 shows that as the congestion window limit decreases, the average congestion window curve flattens. This means that small average window limits make TCP connections much less responsive, especially for changes in relatively low drop rates. For the simulations with the TCP sources limited to a *cwnd* of 6 packets, we had to approximately double the number of TCP flows to achieve an equivalent queue average at a drop rate of 0.01. Thus, a router that is congested with many of short Web flows will need to apply a relatively high drop rate to keep the average queue length within a certain range since the short TCP flows are less responsive. For example, the simulation with smaller *cwnd*s (6 packet limit) has to apply about twice as high a drop rate as the simulation with the larger *cwnd*s (12 packet limit) to maintain an average queue length of 3000 packets.

In this section, we developed a model for load (L) at the router based on TCP traffic, validated it using simula-

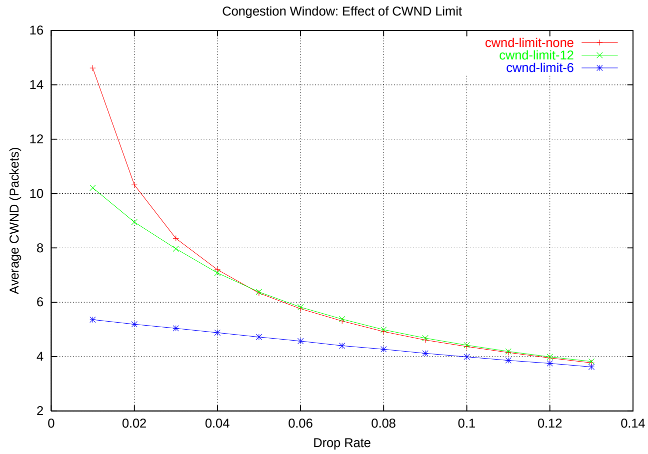

Figure 8. Average Congestion Window: Effect of Limiting TCP Congestion Window

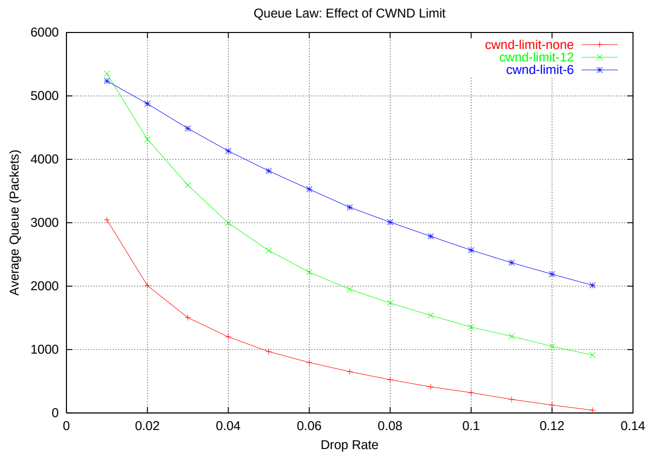

Figure 9. Queue Law: Effect of Limiting TCP Congestion Window

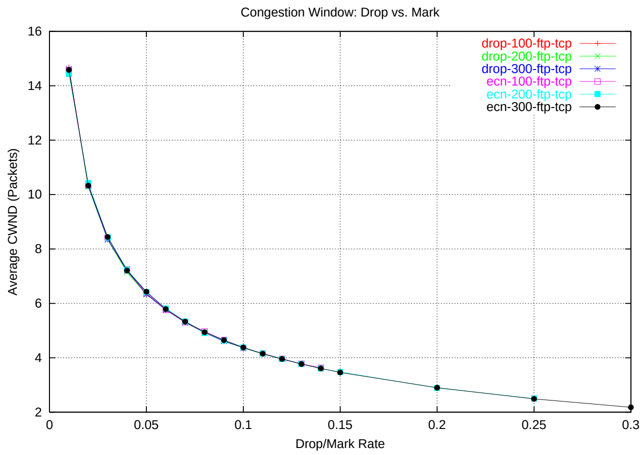

Figure 10. Congestion Window: Drop vs. Mark

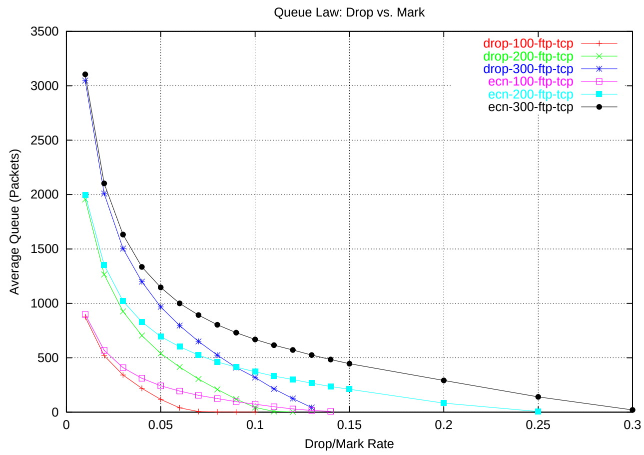

Figure 11. Queue Law: Drop vs. Mark

tion, and used it to explain some of the traffic load characteristics from different traffic mixes with different congestion window limits (representing different data object sizes). We believe that our TCP traffic model can benefit active queue management (AQM) in estimating the change in traffic load and making decisions to choose a proper drop rate for dynamically changing TCP traffic mixes. As we can infer from Figure 7 and Equation 1, the degree of benefit would be maximized when every flow is long-lived, and the average TCP congestion window can be determined as a function of the drop rate.

Even with short-lived flows, routers still use our model to estimate an average congestion window to adapt to the approximate current traffic mix based on average responses to changes in the drop rate. Labels, such as average RTT, provided by DiffServ edge routers [1] can help determine a proper packet drop rate that fits with the router's policy. For example, an Adaptive RED-like AQM may use the information to more quickly determine a proper max_p when traffic changes, without incrementally adjusting max_p. Of course there always is a price to pay to get the source hints.

3 Feedback Method: Drop vs. Mark

The objectives in this section are to measure and verify the existence of an ECN queue law, to compare and contrast the ECN queue law with the drop queue law, and to determine whether ECN is reasonable to deploy by analyzing the curves. We re-ran simulations with 100, 200 and 300 FTP-TCP flows with infinite congestion windows that were ECN enabled, on the network with an 80 ms round trip link delay and 20 Mbps congested link bandwidth.

Figure 10 compares the *avg_cwnd* vs. the uniform drop rate for the simulations with TCP and with TCP with ECN enabled. The TCP congestion window behavior is the same whether packet drops or ECN packet marks are used as the notification method. Figure 11 compares the queue laws for the simulations with TCP and with TCP with ECN enabled. For the same number of flows, the average queue lengths for TCP and TCP with ECN are almost the same when the drop/marking rate at the congested router is low. However, in increasing the drop/mark rate, the average queue length of the queue with TCP with ECN decreases noticeably slower and steadier than the average queue length with TCP with ECN.

It follows that several significant points can be made:

First, an ECN enabled AQM should be configured to apply a significantly higher marking rate than the same AQM using packet drops in order to operate with a reasonably low queuing delay. We believe that a common mistake that many researchers make is in using the same AQM settings for both packet drops and ECN marks, resulting in a mark rate that is too low.

Second, for a reasonable average queue length target (for example, 500 packets in Figure 11), as traffic load increases linearly, the difference between the stable state mark rate and the stable state drop rate to maintain the queue length at the same level increases exponentially, which indicates that ECN should increase its mark rate exponentially above any drop rate. However, the queue law for ECN converges towards an average queue size of 0 for a mark probability of 1, suggesting that there exists a mark rate that can keep the average queue length at a reasonable low target even for a highly loaded situation. Thus, the benefits of ECN should still be effective, even under a wide range of TCP traffic load.

Third, the slowly and steadily decreasing average queue length curve of ECN compared to that of packet drops as the random drop/mark rate increases indicates that the average queue length can be more easily stabilized for AQMs with ECN. We illustrate this further in Section 4.

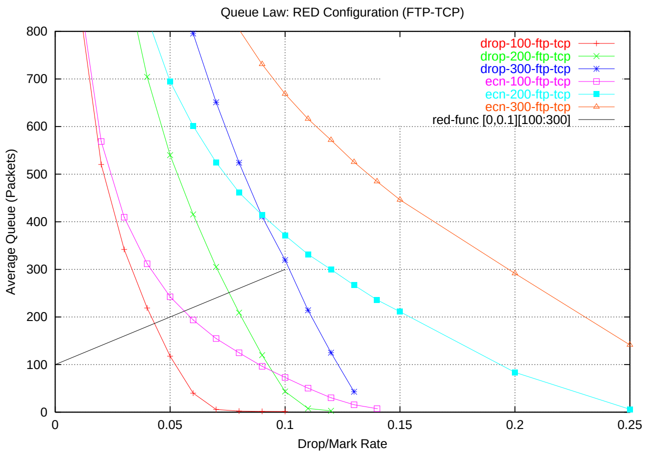

Figure 12. Queue Law: RED Configuration

Figure 13. Queue Statistics: RED

4 Analysis of RED Family AQM

This section evaluates RED, gentle-RED, Adaptive RED, RED-ECN, gentle-RED-ECN and Adaptive RED-ECN using the queue law curves for random packet dropping systems [4] and for ECN systems from the previous section, and verifies the effectiveness of the queue law in characterizing RED performance. At the same time this section compares the performance of RED family AQMs with one another and also with that of drop-tail queue management in terms of throughput and packet loss rate.

The first objective is to test how well the queue law can accurately predict whether a given RED configuration will make the router queue work in a stable condition for a given TCP traffic load. The second objects is to see how RED and its variants behave as they are pushed out of a well-configured state as the traffic load increases.

As in the previous sections, we use the network configuration shown in Figure 3 setting the congested link bandwidth to 20 Mbps and the round trip time link delay to 80 ms. Each simulation starts with 50 FTP-TCP flows, with 50 more FTP-TCP flows added every 50 seconds. The physical queue length is set to 500 packets, with the packet size set to 1 Kbyte. For RED parameter settings, max_p is set to 0.1, min_{th} is 100 packets, and max_{th} is 300 packets, based on recommendations [7]. Although not shown is Figure 12, the limit of max_p for Adaptive RED is set to 0.5 (the default value), which gives the router queue a chance to be well-configured for all the the given TCP traffic loads.

In general, comparing the queue behavior of each RED family AQM with the queue law shown in Figure 12, demonstrates that the queue law indeed works very well predicting RED behavior. For RED, the queue law indicates that RED will be stably manage TCP traffic up to about 200 flows. In Figure 13, RED's average queue was stable up to a traffic load of 150 flows, but at 200 flows it hit the maximum threshold and becomes increasingly unstable. Gentle-RED, shown in Figure 14, was able to manage load up to 200 flows since there no longer a sudden increase in drop probabilities from the max_p 0.1 to 1 at max_{th}. For RED-ECN, shown in Figure 16, the average queue becomes unstable at a load of 150 flows, as the queue law indicated. And as is the case of gentle-RED, gentle-RED-ECN, shown in Figure17, also gets the benefit of the gentle behavior for 200 flows.

Our results show that the gentle setting for RED is beneficial when the offered TCP traffic load is slightly greater than the stable target load for a given configuration. However, the benefit of the gentle setting is not as clear in terms of queue oscillations when a RED router is highly overloaded (250+ flows, in our simulations), although the gentle behavior does reduce the packet loss rate somewhat, as shown in Figure 20. Comparing the queue behavior of Adaptive RED, shown in Figure 15, and Adaptive-RED-ECN, shown in Figure 18, with non-adaptive versions of RED clearly shows the benefits of adjusting max_p. That is, by finding the proper drop/marking slope for changing traffic load conditions, Adaptive RED can stably handle a very wide range of TCP traffic.

We next analyze the delay-loss tradeoffs between drop-tail and RED. Starting with link utilization, Figure 19 shows that the bottleneck link was fully utilized for all TCP traffic loads and thus goodput is affected by packet loss rate only in our simulations. Figure 20 shows the packet loss rates at the routers, which suggests that all the RED family queue mechanisms that use drops for congestion notification have consistently higher packet loss rate than does drop-tail queue management. Drop-tail does not actively drop packets, so the drop distribution that results from buffer overflow at a drop-tail queue may be bursty. However, with many TCP sources, the drops are uniform across flows, resulting in a well-configured state matching the queue law near the drop-tail buffer size. Thus, the delay-loss tradeoff between drop-tail and RED is clear in that RED, using drops as congestion notification method, pays the price in terms of higher packet drop rates over that

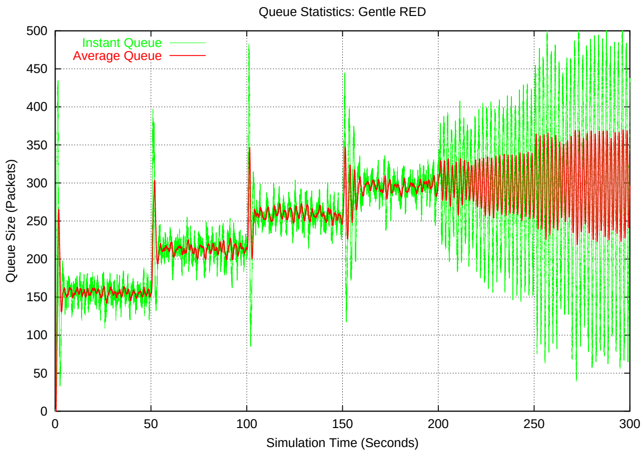

Figure 14. Queue Statistics: Gentle RED

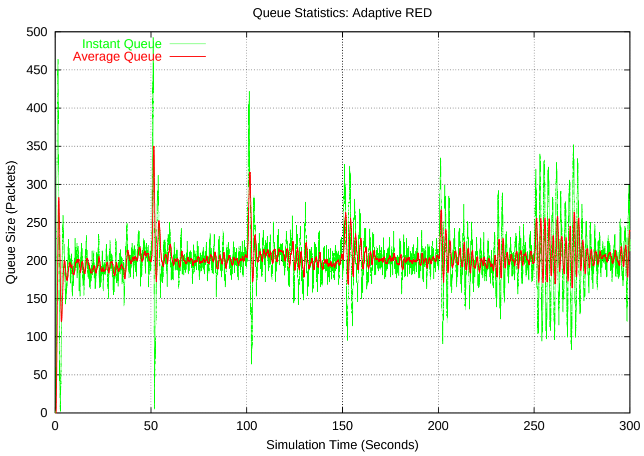

Figure 15. Queue Statistics: Adaptive RED

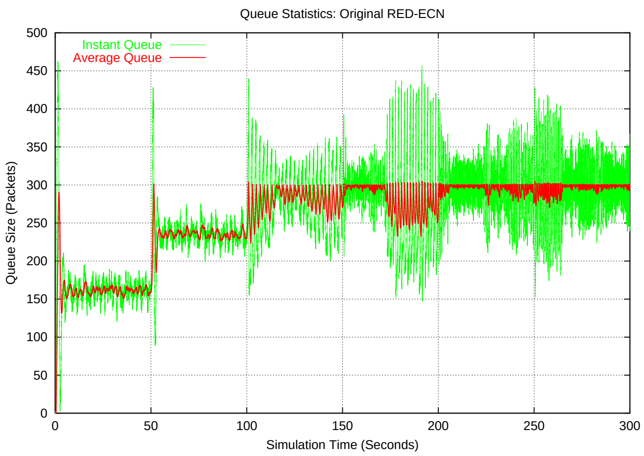

Figure 16. Queue Statistics: RED-ECN

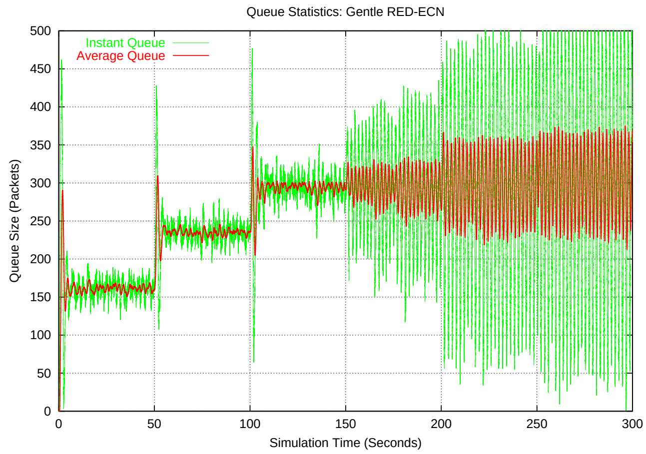

Figure 17. Queue Statistics: Gentle RED-ECN

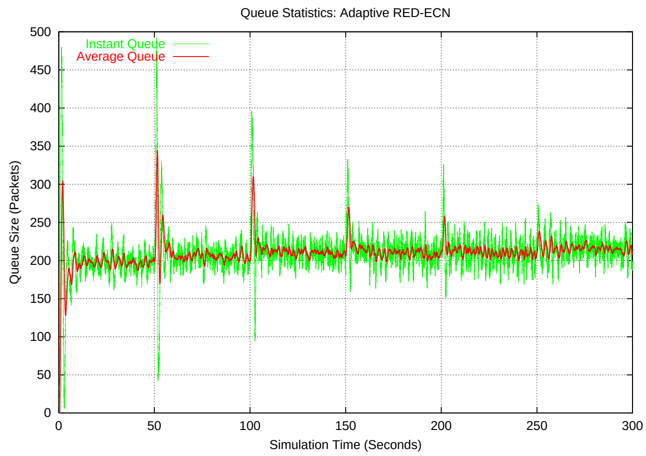

Figure 18. Queue Statistics: Adaptive RED-ECN

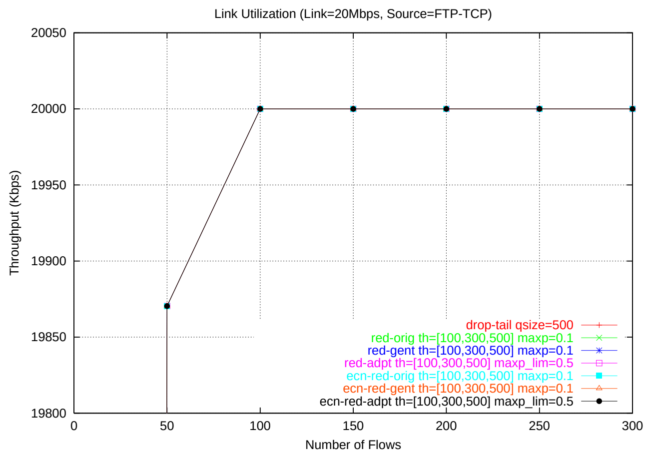

Figure 19. Link Utilization

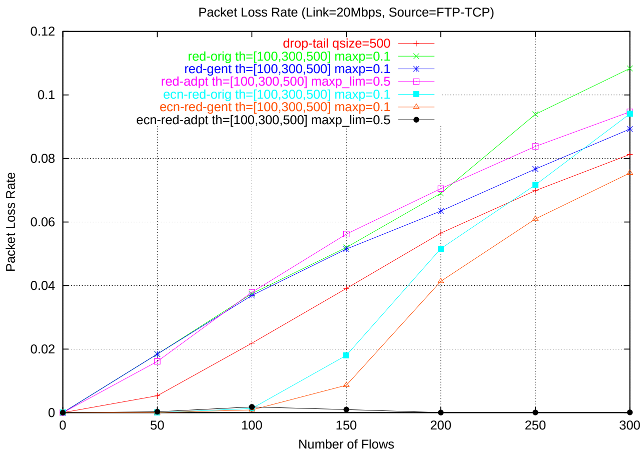

Figure 20. Packet Loss Rate

of drop-tail to maintain the lower average queue size.

We next consider the benefits of marking over dropping as an indicator of congestion. One of the main issues that discourages deployment of RED (or AQM in general) is that the complexity price for AQM design is too high compared with the potential gain of a lower average queue size [9, 3]. However, even with the required higher ECN congestion notification rate, the "price" of the notification in terms of packet loss rate or reduced goodput is zero compared to the price for dropping packets. Figure 20 shows this clearly. ECN enabled RED and its variants in a "well-configured" state can bring down the packet loss rate to zero. Furthermore, Adaptive RED-ECN is able to achieve a packet loss rate very close to zero for the entire range of traffic loads. In addition, as mentioned in Section 3, ECN enabled AQM can be more stable than AQMs without ECN as the queue law curve decreases far more slowly and steadily under high loads than when using drops. This is shown by by comparing the average queue of Adaptive RED and Adaptive RED-ECN, where the average queue oscillation of the ECN enabled one remains more stable even at a high traffic load compare to the one that does not not use ECN.

5 Summary

In this paper, we developed a model for load on the router from TCP traffic that captures a wide-range of key TCP flow characteristics, including long-lived and short-lived flows. We apply the traffic model to ECN flows to show key router configuration characteristics required to be well-configured. For AQM with both drops and marks, our model well-represents the behavior of RED and its variants under a variety of configurations. Our model is useful for predicting the the performance of RED, gentle-RED, A-RED, RED-ECN, gentle-RED-ECN and A-RED-ECN over a continuum of TCP traffic loads.

At the same time, we compare the performance of the RED family AQMs with one another in terms of packet loss rate, delay and queue oscillation, also compared with that of drop-tail queue management. Our model clearly demonstrates the trade-offs between drop-tail queue management and RED family AQMs. This paper concludes that RED family AQMs, particularly Adaptive RED using ECN, can, indeed, be well-configured for variety of TCP traffic mixes, achieving both a very low network packet drop rate and a low queuing delay, often far superior to that of drop-tail queue management.

Future work includes extending our load model to better support short-lived traffic as well as a mixture of ECN and non-ECN TCP flows. In addition, we intend to build an adaptive AQM technique that makes use of our model to more quickly adapt to a well-configured state in the presences of changing network load.

References

[1] Y. Bernet, J. Binder, S. Blake, M. Carlson, B. E. Carpenter, S. Keshav, E. Davies, B. Ohlman, D. Verma, Z. Wang, and W. Weiss. A Framework for Differentiated Services. Internet Draft draft-ietf-diffserv-framework-02.txt, IETF, February 1999.

[2] M. Christiansen, K. Jeffay, D. Ott, and F.D. Smith. Tuning RED for Web Traffic. In *Proceedings of ACM SIGCOMM Conference*, August 2000.

[3] Christophe Diot, Gianluca Iannaccone, and Martin May. Aggregate Traffic Performance with Active Queue Management and Drop from Tail. Sprint ATL Technical Report TR01-ATL-012501, 2001.

[4] Victor Firoiu and Marty Borden. A Study of Active Queue Management for Congestion Control. In *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, Mar 2000.

[5] S Floyd. TCP and Explicit Congestion Notification. *ACM Computer Communication Review*, 24.

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

[7] Sally Floyd. RED: Discussions of Setting Parameters. http://www.icir.org/floyd/REDparameters.txt.

[8] Sally Floyd, Ramakrishna Gummadi, and Scott Shenker. Adaptive RED: An Algorithm for Increasing the Robustness of RED's Active Queue Management. Under submission, http://www.icir.org/floyd/papers/adaptiveRed.pdf.

[9] Martin May, Jean Bolot, Christophe Diot, and Bryan Lyles. Reasons not to deploy RED. In *Proceedings of the IEEE/IFIP International Workshop on Quality of Service (IWQoS'99)*, June 1999.

[10] V. Rosolen, O. Bonaventure, and G. Leduc. A RED discard strategy for ATM networks and its performance evaluation with TCP/IP traffic. *ACM Computer Communication Review*, Jul 1999.

[11] VINT. Virtual InterNetwork Testbed, A Collaboration among USC/ISI, Xerox PARC, LBNL, and UCB. http://www.isi.edu/nsnam/vint/.