Towards an Adaptive RED Algorithm for Achieving Delay-Loss Performance

Rahul Verma, Aravind Iyer and Abhay Karandikar

Information Networks Laboratory,
Department of Electrical Engineering,
Indian Institute of Technology, Mumbai, India-400076
E-mail: {rahulv, aiyer, karandi}@ee.iitb.ac.in

Abstract

Random Early Detection (RED) [1] is an active queue management strategy that has been deployed extensively in routers. However, its effectiveness has been limited owing to the difficulty in tuning its parameters under different network conditions. In this paper, we define the notion of an operating point that an Active Queue Management (AQM) policy should target. We suggest that the RED parameters should be set or varied with the aim of reaching this operating point. We have performed extensive simulations to understand the effect of max_{th} and max_p on the network dynamics. Based on these simulations, we give guidelines to adapt the RED parameters.

Key words: AQM, TCP, RED, Congestion Control

Preprint submitted to IEE

1 Introduction

Active Queue Management (AQM) policies are mechanisms for congestion avoidance, which drop packets pro-actively in order to provide an early congestion notification to sources. The sources can then reduce their window sizes (or rates), thereby preventing further congestion and packet loss. Random Early Detection (RED) [1] is one such AQM strategy that has been widely deployed in routers.

The RED algorithm works by dropping packets with a certain probability which is computed based on an exponentially weighted moving average (EWMA), q_{avg}, of the instantaneous queue length. The averaging process smoothens out temporary traffic fluctuations and allows small bursts to pass through unharmed, dropping packets only during sustained overloads. RED maintains two queue thresholds: min_{th} and max_{th}. If q_{avg} exceeds max_{th}, all incoming packets are dropped, whereas if q_{avg} is less than min_{th}, no packet is dropped. If q_{avg} lies between the two thresholds, packets are dropped with a probability p which increases linearly from 0 (at min_{th}) to max_p (at max_{th}).

An AQM policy, such as RED, can give more flexibility with regard to a choice between low average queuing delay and high link utilization as compared to a Drop-Tail gateway in which the packets are dropped only when the buffer is full. One of the major limitations of RED is that its performance depends severely on the setting of its parameters. As pointed out in [2], "tuning of RED parameters has been an inexact science for sometime now". Various authors have given guidelines and proposals to set RED parameters [2–6] or to adaptively vary these parameters [7,8]. Although congestion avoidance remains at the heart of all these schemes, at the same time it is not very clear how a particular setting of RED parameters helps in achieving the additional goals of low loss rate, low average queuing delay and high link utilization.

In this paper, we elaborate on the notion of an operating point [9] that an AQM policy should target. We argue that the optimal RED parameter settings would depend on the targeted operating point and the network load. Further, any setting or tuning of RED parameters, in order to meet certain performance measures, should be based on a study of how the parameters impact the particular performance measures. In this paper, we study the effect of max_{th} and max_p on the packet loss rate, link utilization and average queuing delay. Based on our extensive simulation studies, we give certain guidelines for the design of an adaptive RED algorithm.

The rest of the paper is organized as follows. In Section 2, we discuss some of the related work. In Section 3, we define the notion of an *operating point* and briefly discuss its relation to the RED parameters. We present our simulation studies in Section 4. The section also contains guidelines to adaptively vary the RED parameters. Finally, Section 5 concludes the paper.

2 Related Work

The performance of the RED algorithm can be evaluated using a number of performance metrics including packet loss rate, average queuing delay, link utilization and fairness to incoming flows. The original RED algorithm [1] was proposed with the main goal of providing "congestion avoidance by controlling the average queue size" where the performance of the RED algorithm has been compared to that of Drop-Tail and Random Drop algorithms in terms of average queue size, average link utilization and source throughput (with varying buffer sizes).

Feng et al. [7] propose a mechanism for adaptively varying one of the RED parameters, max_p, with the aim of reducing the packet loss rates across congested links. The performance has also been compared in terms of the throughput (defined as the rate at which packets are forwarded through the congested interface).

In [2], Hollot et al. have studied the problem of tuning RED parameters from a control theoretic stand point. Their objective is to improve the throughput by controlling oscillations in the instantaneous queue and they suggest mechanisms to set RED parameters in order to achieve this objective.

Floyd et al. in [8] discuss algorithmic modifications to the self-configuring RED algorithm [7] for tuning max_p adaptively. Their objective is to control the average queue length around a pre-decided target. The choice of the target queue size, which determines the trade-off between delay and link utilization, is left to the network operator. However, it may be noted that controlling the average queue size has a limited impact on regulating the packet loss rate.

Though the above papers have made significant progress towards improving the understanding of tuning RED parameters, the complexity of the problem of tuning RED parameters and their impact on various performance metrics leaves scope for throwing further light in this direction. In particular, Feng et al. [7] have not investigated the effect of adaptively varying max_p on the *average queuing delay*. The notion of leaving the target queue size to the network operator has been discussed in [8]. But the effect of adaptively varying max_{th} in attaining a targeted queue size has not been studied.

3 Operating Point

An AQM policy can be used to control both the packet loss rate as well as the average queuing delay. For a given buffer size, the average queuing delay depends directly on the average queue size which in turn is controlled by the AQM policy and the scheduling policy. Moreover, for equal sized packets, scheduling has a limited role to play in controlling the average queue size and it is the AQM policy that is the main factor affecting the average queue size. The packet loss rate is also directly controlled by the AQM policy. A particular setting of RED parameters, therefore, has a definite bearing on the average queuing delay and the packet loss rate.

A *correct* choice of RED parameters makes sense only when the final objectives of the AQM policy (in this case, RED) have been clearly specified. In other words, one should specify the *operating point* that one wants to achieve given a particular network scenario. The operating point can, for instance, be specified in terms of the average queuing delay and packet loss rate that one wants to achieve. We denote such an operating point by (d^*, l^*) where d^* is the target average queuing delay and l^* denotes the target packet loss rate. One might also specify the operating point in terms of other performance metrics such as link utilization.

The suitable setting of RED parameters could be different for different operating points. Consider, for example, the case where our objective is to minimize the packet loss rate. In such a scenario, keeping a high value of max_{th} results in a lower packet loss rate though at the expense of a higher average queuing delay. Therefore, it is important to specify the point where one wants to operate. It is possible that such an operating point might not be achievable. In that case, the aim of the AQM policy (RED) should be to approach this operating point as closely as possible.

3.1 Need for Adaptive RED

The operating point (d_1^*, l_1^*) for a particular router may change due to different delay, loss rate requirements or due to a change in the network conditions. Therefore, the same set of RED parameters may not be suitable for a different operating point, say (d_2^*, l_2^*). Given an operating point, the RED parameters should be varied in such a way that the network performance converges to the given operating point. Further, even with the same operating point, due to a change in the network conditions, the same set of RED parameters may no longer be suitable. This provides the motivation for tuning the RED parameters adaptively based on the new network conditions. For setting the RED parameters as well as adaptively varying these parameters, an understanding of the dependence of network dynamics on various RED parameters is important. We deal with this issue in the next section.

4 Role of RED Parameters

In this section, we study the effect of the RED parameters, max_{th} and max_p, on packet loss rate, average queuing delay and link utilization, by means of simulations.

4.1 Experimental Network

The simulations have been performed using the **ns v2.1b7a** network simulator. The network topology shown in Figure 1 is used for simulating the various scenarios. It may be noted that the simulation topology used is similar to the topology used in other papers, for consistent comparison. The issue of multiple congested gateways has not been considered here. The number of TCP sources (N) is varied to generate different loads. The buffer size for the RED router is kept as 50 packets and the simulations are performed with equal sized packets. The results are obtained for the RED router and the bottleneck link shown in the figure.

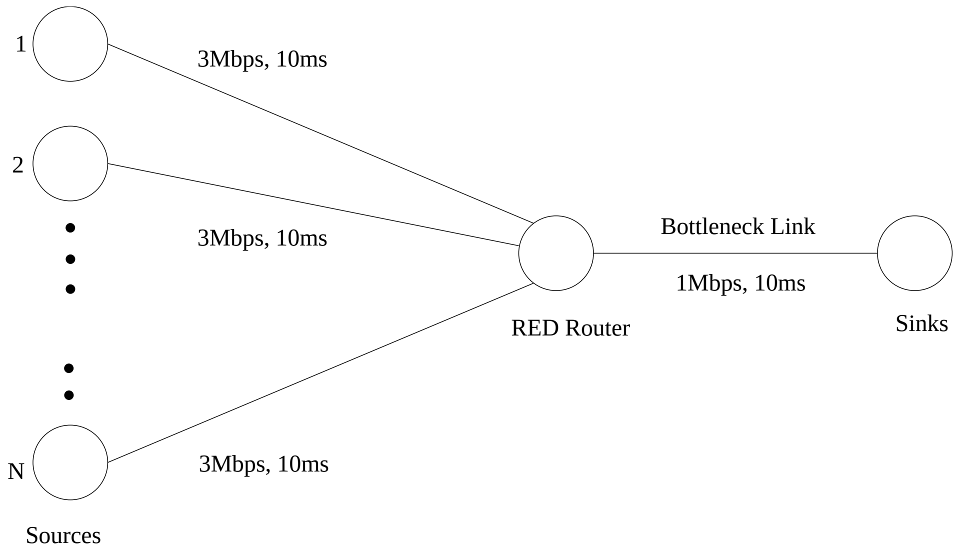

Fig. 1. Network Topology

4.2 Effect of max_{th}

Figure 2 shows the variation of the packet loss rate with increasing maximum threshold, max_{th}. The different plots, as indicated, correspond to different number of sources (different network loads). The value of max_p has been set to 0.1 and w_q to 0.002. The plots clearly show that the packet loss rate decreases with increasing max_{th} and increases with increasing network load. Figure 3 shows the variation of the average queuing delay. The plots are all

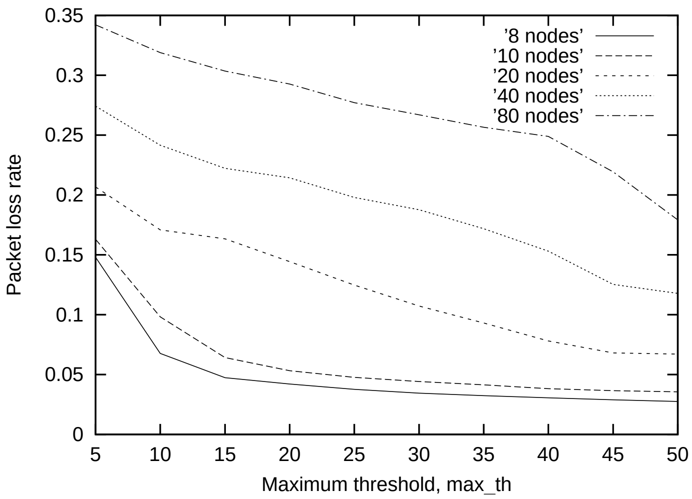

Fig. 2. Packet loss rate vs. max_{th} (different loads)

monotonically increasing, indicating that to achieve a lower average queuing delay, max_{th} should be set as low as possible. However, there is not much variation in the average queuing delay with the number of sources (especially for values of $max_{th} \leq 0.5$ * (Buffer size)).

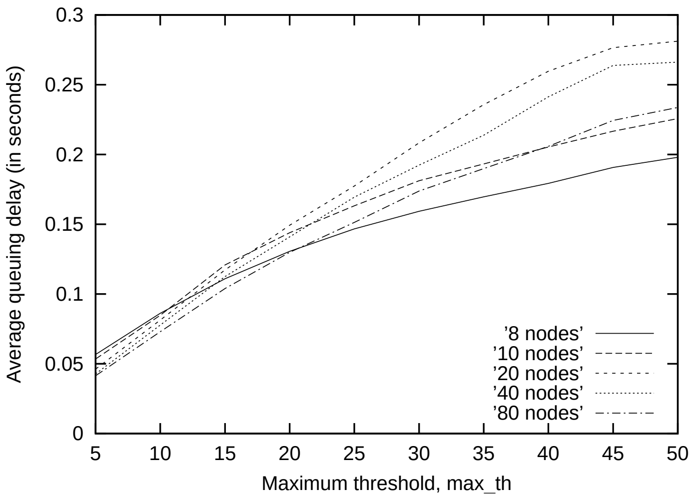

Fig. 3. Average queuing delay vs. max_{th} (different loads)

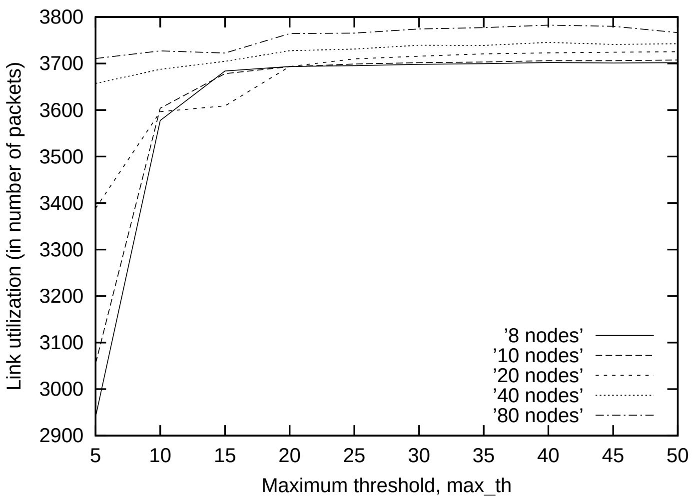

Fig. 4. Packets transmitted vs. max_{th} (different loads)

Figure 4 depicts the variation of the link utilization against max_{th}. The plots indicate that for heavy network loads, the link utilization is more or less independent of max_{th}. On the other hand, for low network loads, max_{th} cannot be reduced below a certain value (≈ 15), without affecting the link utilization adversely. Also, for low network loads, the packet loss rate becomes independent of max_{th} beyond a certain value (≈ 15) (see Figure 2). The same conclusions also hold true for different values of w_q. The packet loss rates and the link utilization with increasing max_{th} for $w_q = 0.005$ are illustrated in Figures 5 and 6. We shall be reporting results only for $w_q = 0.002$ henceforth for lack of space, but we have verified that the value of w_q does not affect the conclusions we draw in this paper.

In scenarios where the network load is low, we should aim to keep the link utilization high. Based on the above discussion, this can be done by *setting max_{th} to a relatively high value (say $max_{th} \approx 0.6$ * (Buffer size))* without adversely affecting the average queuing delay to a considerable extent. Once max_{th} is set, max_p can be adaptively varied to achieve a trade-off between packet loss rate and average queuing delay. This is explained further in the next subsection.

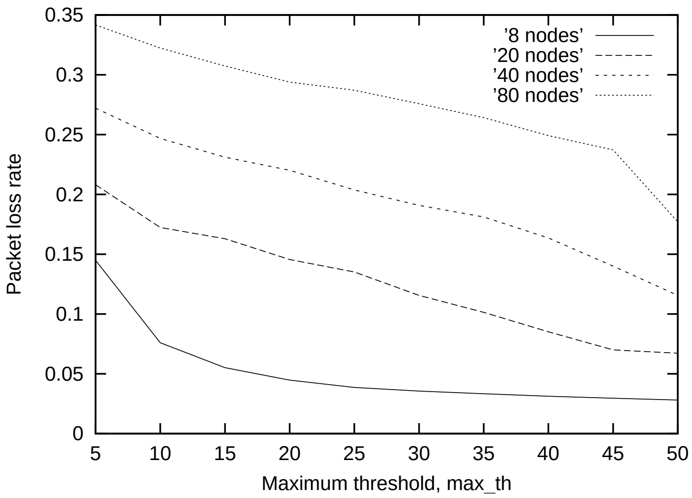

Fig. 5. Packet loss rate vs. max_{th} (different loads) for $w_q = 0.005$

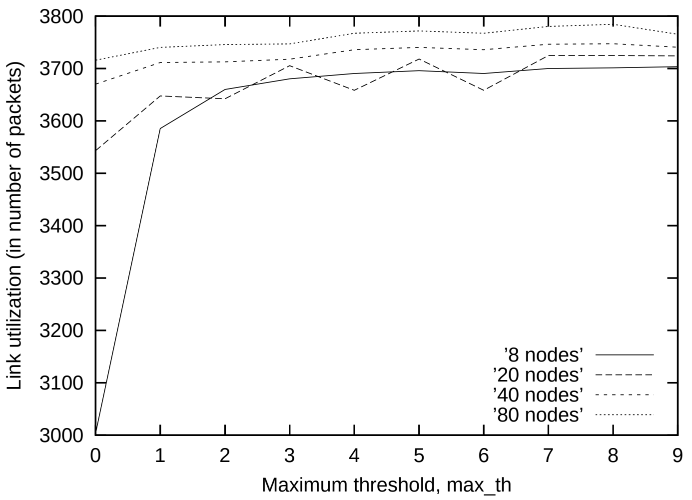

Fig. 6. Packets transmitted vs. max_{th} (different loads) for $w_q = 0.005$

The previous research efforts on tuning RED parameters have not considered the effect of varying max_{th} on the performance of RED. Intuitively, it would seem that small buffers in routers may be desirable to keep the queuing delay low. But on the other hand, large buffers may be required to keep the packet loss rate low. We illustrate that a trade-off between packet loss rate and queuing delay could be achieved by adjusting max_{th} appropriately in RED.

In Figure 7, the trade-off between packet loss rate and average queuing delay is illustrated. The extreme right point on each curve corresponds to $max_{th} = 50$ and as we move along the curve towards left, we get points corresponding to lower values of max_{th}. As seen earlier, a high value of max_{th} results in a lower loss rate, but it is at the cost of a higher average queuing delay. Ideally, one would like to operate the network as close to the origin as possible, as that would mean a low delay and a low packet loss. But as the number of sources increases, the delay-loss trade-off curve shifts away from the origin and it becomes more important to specify the operating point clearly. To achieve a lower average queuing delay, one might have to incur a greater packet loss rate. An operating point above the trade-off curve is an *achievable* operating point, while one below it is *unachievable*. If the operating point is unachievable, one would like to operate at such a point on the curve which is closest to the specified operating point.

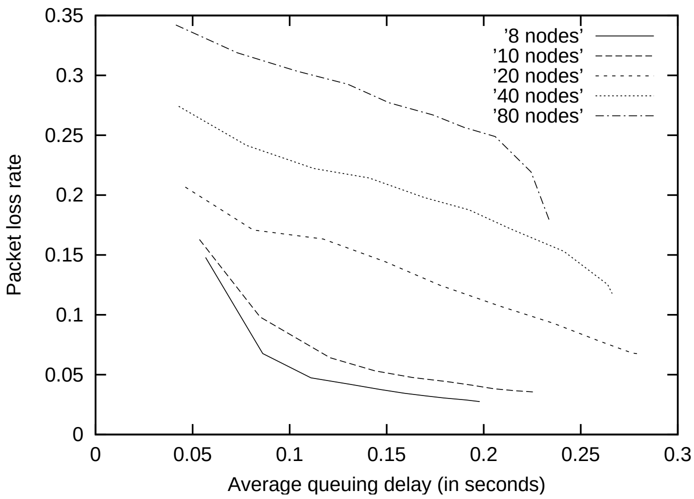

Fig. 7. Delay-loss trade-off (different loads)

4.3 Effect of max_p

The simulations for studying the effect of max_p have been done using 80 nodes. Figure 8 shows the packet loss rate as a function of the maximum dropping probability, max_p. The different plots correspond to different values of max_{th}. As can be seen from the figure the packet loss rate curve resembles a convex non-monotonous function and achieves a minimum for a certain value of max_p. This is explained as follows. If max_p is too large, this results in RED being too aggressive (when q_{avg} lies between min_{th} and max_{th}) and hence results in a high packet loss rate. If max_p is too low, RED becomes conservative and results in a delay in the congestion notification. This leads to more packet drops before the source(s) can reduce its (their) window-size(s). This also results in the average queue length exceeding max_{th} too often leading to further packet drops of all incoming packets. Therefore the packet loss rate attains a minimum at a certain value of max_p between the two extremes ($max_p = 0$ and $max_p = 1$).

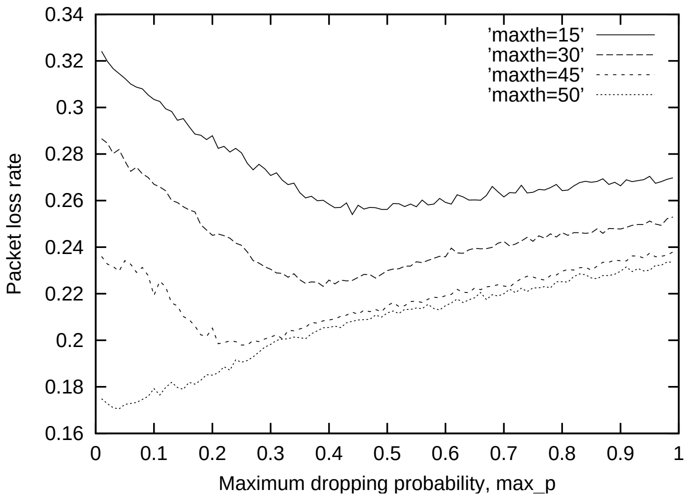

Fig. 8. Packet loss rate vs. max_p (different max_{th})

Further, this minimum shifts to lower values of max_p as max_{th} is increased. For a small value of max_p, when max_{th} is large, the component of packet loss due to the average queue length exceeding max_{th} is substantially reduced compared to the case when max_{th} is small. Also even though a small value of max_p results in a delay in the congestion notification, a higher value of max_{th} ensures that more packets are admitted. As a result, the packet loss rate is less for smaller values of max_p when max_{th} is large. The packet loss rate for any given value of max_p also reduces as max_{th} is increased. The above observation is true for low network loads (≤ 20 nodes) as well. In such a case, as stated earlier, max_{th} is set to a relatively high value. Therefore, *a low value of max_p (≈ 0.1)* can give a better performance in terms of packet loss rate (for low network loads).

Figure 9 shows the average queuing delay as a function of max_p. As max_p increases, the

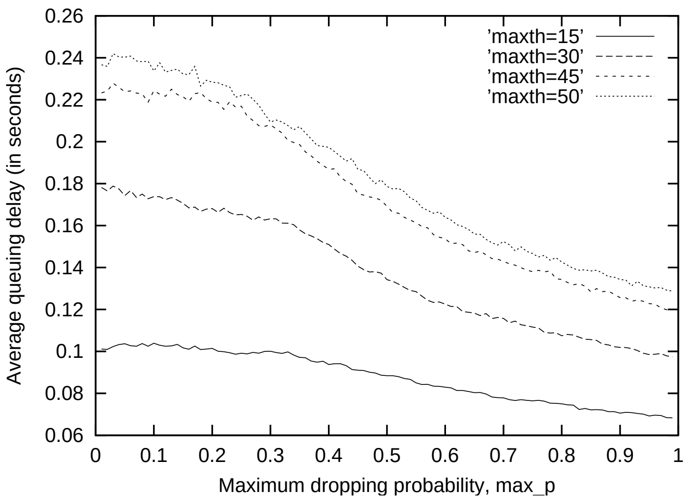

Fig. 9. Average queuing delay vs. max_p (different max_{th})

average queue size and hence, the average queuing delay reduces. Also, for any given value of max_p, the average queuing delay increases as max_{th} increases.

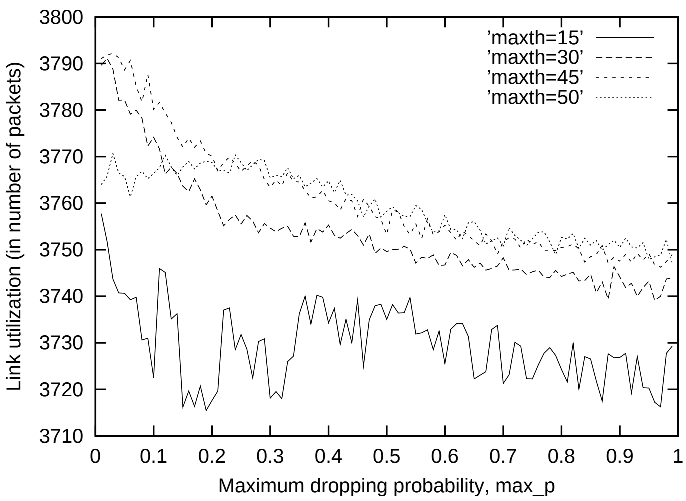

Fig. 10. Packets transmitted vs. max_p (different max_{th})

Figure 10 shows the number of packets transmitted (link utilization) as a function of max_p. The link utilization increases as max_{th} is increased. It does not have a direct relationship with max_p though higher values of max_p result in a lower link utilization. The variations in link utilization with respect to max_p are particularly high for low values of max_{th}.

The trade-off between packet loss rate and average queuing delay is illustrated in Figure 11 for different values of max_{th}. The extreme left point on the delay-loss trade-off curve for each max_{th} corresponds to $max_p = 1$ and as we move along the curve from that point, we get

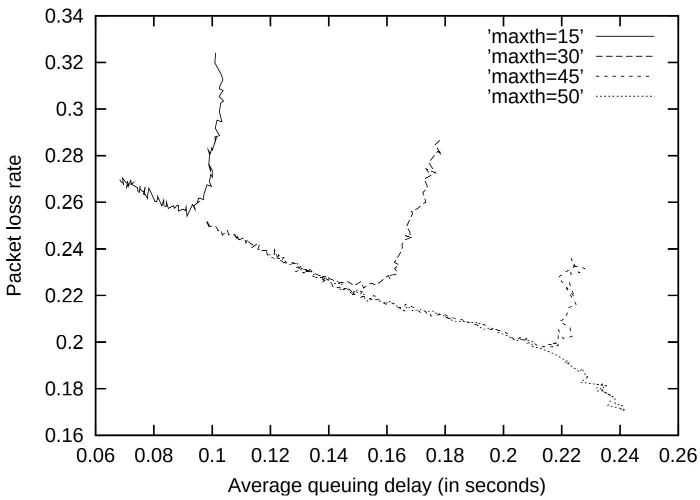

Fig. 11. Delay-loss trade-off (different max_{th})

points corresponding to lower values of max_p. As can be seen from the figure, the delay-loss trade-off curves can be broken into two parts. In the first part (which lies towards the left), the packet loss rate is low for higher values of the average queuing delay and the loss rate increases as average queuing delay is reduced. This part corresponds to higher values of max_p. In the second part (which corresponds to lower values of max_p), both the packet loss rate and the average queuing delay are high. One would ideally want to operate only in the first part of the curve. Therefore, if one is operating in the second part of the curve, max_p should be increased so as to reach the first part. It may be noted that the second part of the curve becomes less prominent as max_{th} is increased. The point at which the nature of the curve changes corresponds to the minimum in the packet loss rate curve (Figure 8).

The curve further illustrates that for high network loads, a high value of max_p (≈ 0.3) might actually give a better performance than a relatively low value of max_p (say $max_p = 0.1$), both in terms of the average queuing delay and in terms of the packet loss rate. Therefore, a simple guideline in the presence of heavy network loads is to *set* max_p *to a relatively high value (≈0.3) and adaptively vary* max_{th} to achieve a trade-off between packet loss rate and average queuing delay. For a given packet loss rate (average queuing delay) and a fixed max_p, the average queuing delay (packet loss rate) can be minimized by *reducing* (*increasing*) max_{th} to a point where the packet loss (delay) requirement is just met.

Figures 12 and 13 show the variation of average queuing delay and packet loss rate with max_p for different number of sources (different loads). The value of max_{th} is set to 30. For

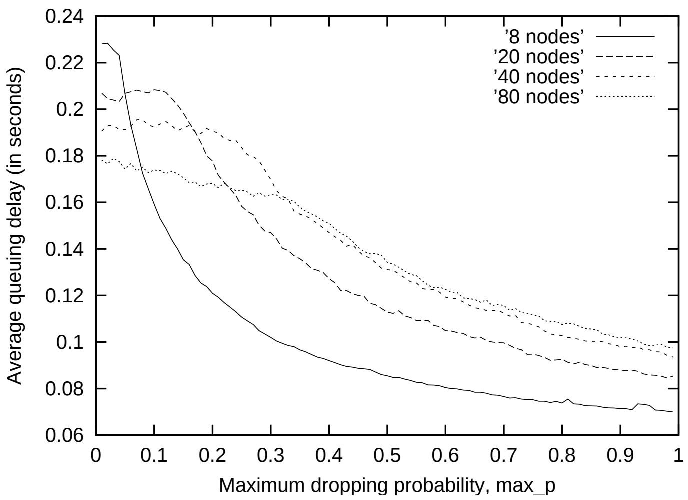

Fig. 12. Average queuing delay vs. max_p (different loads)

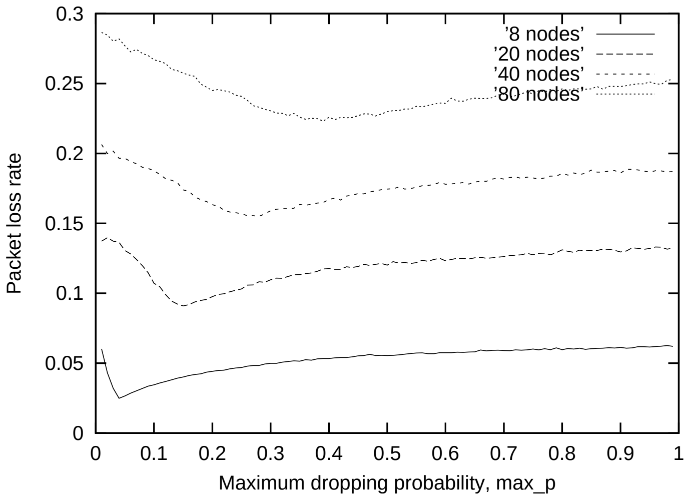

Fig. 13. Packet loss rate vs. max_p (different loads)

low network loads, the average queuing delay reduces as max_p is increased. The decrease in the average queuing delay is less drastic for higher network loads. As explained earlier, the packet loss rate curves as a function of max_p are non-monotonous and achieve a minimum for a certain value of max_p. Further, as the network load is reduced, the minimum shift towards lower values of max_p. The trend is similar to the one observed when max_{th} is increased for a given load.

5 Discussion and Conclusions

We have introduced the notion of an operating point that an AQM scheme should target. The operating point can be defined in terms of the performance metrics that a network operator wishes to achieve. We suggest that any setting or tuning of the RED parameters should be done based on such an operating point. We have performed extensive simulations to investigate the effect of max_{th} and max_p on packet loss rate, average queuing delay and link utilization. Based on these, we give the following guidelines to adaptively vary max_{th} and max_p:

(1) For low network loads, max_{th} should be set to a relatively high value (≈ 0.6 * (Buffer size)) to achieve high link utilization and max_p should be adaptively varied to achieve a trade-off between packet loss and queuing delay.

(2) For high network loads, max_p should be set to a relatively high value (≈ 0.3) and max_{th} should be adaptively varied to achieve a trade-off between packet loss and queuing delay. The link utilization, in such cases, is independent of max_{th}.

(3) The point of minimum in the packet loss rate versus max_p curve shifts towards lower values of max_p as the load decreases (Figure 13) and also as max_{th} increases (Figure 8). Therefore, a low value of max_p (≈ 0.1) helps in achieving a low packet loss rate, for low network loads (especially with high max_{th}).

The above guidelines provide a foundation for designing an adaptive algorithm to vary max_{th} and max_p in conjunction. We are currently working in this direction. The paper also illustrates the effect of max_{th} and max_p on the trade-off that exists between the average queuing delay and the packet loss rate.

Acknowledgements

The authors would like to acknowledge Mr Abhishek Jain for help in preparing the final manuscript.

References

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

[2] C. Hollot, Vishal Misra, D. Towsley, and Wei-Bo Gong, "A Control Theoretic Analysis of RED", *Proceedings of IEEE INFOCOM*, pp. 1510-1519, April 2001.

[3] S. Floyd, "RED: Discussions of Setting Parameters", `http://www.aciri.org/floyd/REDparameters.txt`.

[4] T. Ziegler, S. Fdida, and C. Brandeur, "Stability Criteria for RED with Bulk-data TCP Traffic", *Technical Report*, `http://www-rp.lip6.fr/publications/production.html`, August 1999.

[5] V. Jacobson, K. Nichols, and K. Poduri, "RED in a Different Light", `www.cnaf.infn.it/~ferrari/papers/ispn/red_light_9_30.pdf`, September 1999.

[6] H. Ohsaki and M. Murata, "Steady State Analysis of the RED Gateway: Stability, Transient Behavior, and Parameter Setting", *IEICE Transactions on Communication*, Vol. E85-B, No. 1, January 2002.

[7] Wu-chang Feng, Dilip D. Kandlur, Debanjan Saha and Kang G. Shin, "A Self-Configuring RED Gateway", *Proceedings of IEEE INFOCOM*, pp. 1320-1328, 1999.

[8] Sally Floyd, Ramakrishna Gummadi, and Scott Shenker, "Adaptive RED: An Algorithm for Increasing the Robustness of RED's Active Queue Management", `www.icir.org/floyd/papers/adaptiveRed.pdf`, August 2001.

[9] Wu-chang Feng, Dilip D. Kandlur, Debanjan Saha, Kang G. Shin, "Stochastic Fair Blue: A Queue Management Algorithm for Enforcing Fairness" *Proceedings of IEEE INFOCOM*, pp. 1520-1529, April 2001.