# ON TUNING OF RED PARAMETERS

*Rahul Verma, Aravind Iyer, Abhay Karandikar*

Department of Electrical Engineering
Indian Institute of Technology
Mumbai-400076
{rahulv,aiyer,karandi}@ee.iitb.ac.in

**ABSTRACT**

Random Early Detection (RED) [1] is a widely deployed active queue management scheme in packet-switched networks. Although RED can improve packet loss rates, its performance depends severely on the tuning of its operating parameters. To alleviate the problem of parameter dependence, we propose an algorithm to adaptively vary the queue weight $w_q$ in conjunction with the maximum packet dropping probability $max_p$.

## 1. INTRODUCTION

Future generation Internet routers will be expected to maintain low packet loss rates and to be fair to various flows, even under conditions of heavy congestion. Traditional queue management, where packets are dropped at a router only when the buffer of the router overflows (known as the Drop-Tail Gateway), has been found to be inadequate. The reasons for this include a lack of early congestion notification to sources, due to which more packets are lost by the time the source detects congestion. The solution to this problem lies in employing active queue management strategies which attempt to warn the sources of incipient congestion. One of the commonly deployed active queue management schemes is RED [1].

RED aims to *detect* congestion when it is building up and to forestall it by dropping packets *early*, before the buffer is full. The sources can then reduce their window sizes (or rates), thereby preventing further congestion and packet loss. Even though RED reduces the packet loss rate, it suffers from several shortcomings, including a high degree of dependence on the tuning of its parameters, unfairness to flows with different round-trip times and the problem of global synchronization.

Several studies have been performed to address the above shortcomings [2, 3, 4]. The idea of adaptively varying RED parameters has been proposed in [2]. In [2], one of the RED parameters $max_p$ is varied adaptively. Although this improves the packet loss rate and the instantaneous queue length stability, our simulations illustrate that it leads to lower link utilization in certain scenarios. We demonstrate that varying $w_q$, by the technique illustrated in the paper, along with varying $max_p$, in a manner similar to that of [2], leads to an improvement in terms of packet loss rate and queue stability without adversely affecting the link utilization.

The rest of the paper is organized as follows. Section II explains the problem of parameter sensitivity and discusses an algorithm to modify $max_p$ adaptively. In Section III, we propose an algorithm for adaptively varying $w_q$. Section IV studies the relative performance of the various algorithms and Section V concludes the paper.

## 2. BACKGROUND

### 2.1. RED scheme

RED gateways use an exponentially weighted moving average (EWMA) of the instantaneous queue length to detect incipient congestion. The average queue length is computed at every packet arrival as:

$$q_{avg} = (1 - w_q)q_{avg} + w_q.q \tag{1}$$

where $q$ is the instantaneous queue length. The decision to drop the incoming packet is based on where $q_{avg}$ lies with respect to two thresholds, viz., $min_{th}$ and $max_{th}$. If $q_{avg}$ exceeds $max_{th}$ the incoming packet is dropped, whereas if $q_{avg}$ is less than $min_{th}$ the packet is admitted. If $q_{avg}$ lies in between the two thresholds, the incoming packet is dropped with a certain probability $p$ which is computed as:

$$p = max_p \left( \frac{q_{avg} - min_{th}}{max_{th} - min_{th}} \right)$$

By dropping packets *early*, RED notifies TCP sources of the incipient congestion.

The various control parameters of RED are listed below (Table I). The performance of a gateway employing RED depends significantly upon its parameters (see [2, 5]). Certain guidelines have been given in [6] to decide the values of these parameters. However, experiments and simulations show that it is difficult to find appropriate values of parameters that will enable RED gateways to perform equally well under different congestion scenarios. In cases where the parameters are not suitable for the given network traffic load, the performance of the RED gateway can approach that of the traditional Drop-Tail gateway [2, 5].

Table 1: RED Parameters

| Parameter | Function |
|---|---|
| $w_q$ | Weight for calculating average queue |
| $max_p$ | Maximum dropping probability |
| $min_{th}$ | Lower threshold below which no packet is dropped |
| $max_{th}$ | Upper threshold above which all incoming packets are dropped |

### 2.2. Self-Configuring RED

In [2], an algorithm to adapt the parameter $max_p$ based on the changes in average queue length is described. The essence of the scheme is to control the aggressiveness of RED by examining the variations in average queue length $q_{avg}$. If $q_{avg}$ oscillates around $min_{th}$, then $max_p$ is reduced by a factor $\alpha$ (i.e. $max_p = max_p/\alpha$), in order to reduce the aggressiveness of RED. On the other hand, if $q_{avg}$ oscillates around $max_{th}$, then $max_p$ is increased by a factor $\beta$ (i.e. $max_p = max_p * \beta$), so that RED becomes more aggressive. In the rest of the paper, we will refer to this algorithm as the $max_p$ algorithm (Figure 1). The algorithm is successful in reducing oscillations in the instantaneous queue length under various traffic conditions.

---

On the $n$th $q_{avg}$ Update

- Calculate average queue size as $q_{avg_n} \leftarrow (1 - w_q) q_{avg_{n-1}} + w_q . q$
- If $(min_{th} < q_{avg_n} < max_{th})$
  status = Between
- If $(q_{avg_n} < min_{th}$ && status!=Below)
  status = Below
  $max_p = max_p/\alpha$
- If $(q_{avg_n} > max_{th}$ && status!=Above)
  status = Above
  $max_p = max_p * \beta$
- Rest of the RED algorithm remains the same.

---

Figure 1: *Self-Configuring RED algorithm ($max_p$)*

## 3. ADAPTIVE RED

The Self-Configuring RED ($max_p$) algorithm improves the performance in terms of the instantaneous queue stability by reducing oscillations. However, (as shown in the simulations below) it results in a lower link utilization[1] under conditions of congestion. Our simulation results show that adapting the parameter $w_q$ has a positive effect on link utilization under conditions of heavy load. This motivates us to look at algorithms which vary $w_q$ adaptively in conjunction with $max_p$, in order to get an overall improvement in performance.

[1] The total number of packets transmitted *successfully* on a particular link (for a given time interval) is taken as a measure of the link utilization for that link.

As shown by the simulations in this paper, varying both $w_q$ and $max_p$ results in an improvement in queue stability and loss rate without bringing down the link utilization.

We propose to vary the parameter $w_q$ adaptively based on the variations of the average queue length. If the change in average queue length is significant, then at the next $q_{avg}$ update, greater weight should be given to the instantaneous queue length and hence $w_q$ should be increased. On the other hand, if the change in average queue length is negligible, $w_q$ should be reduced and greater weight should be given to the previous value of the average queue size. In this paper, we suggest an algorithm ($w_q$-*thresh* algorithm, Figure 2) which can be used to vary $w_q$.

The $w_q$-*thresh* algorithm modulates the aggressiveness of RED scheme by varying the value of $w_q$ based on the changes in the average queue length. We denote the change in the average queue size[2] ($q_{avg_n} - q_{avg_{n-1}}$) by $|\Delta avg|$. The idea is to have two thresholds $\rho_1$ and $\rho_2$, such that $\rho_1 < \rho_2$ and vary the queue weight based on the comparison of $|\Delta avg|$ with the two thresholds. If $|\Delta avg|$ is less than the lower threshold $\rho_1$, $w_q$ is chosen to be $w_1$. If $|\Delta avg|$ is greater than the higher threshold $\rho_2$, $w_q$ is chosen to be $w_2$. If $|\Delta avg|$ lies in between the two thresholds, we chose the default value of $w_q$. Here, $w_1 < w_{default} < w_2$.

The performance of the $w_q$-*thresh* algorithm has been studied for various values of $w_1$, $w_2$, $\rho_1$ and $\rho_2$. The rationale behind choosing particular values (which are used in the simulations) for the above parameters is given below. The change in the average queue length, $|\Delta avg|$ is given by:

$$|\Delta avg| = |q_{avg_n} - q_{avg_{n-1}}| = w_q|q_n - q_{avg_{n-1}}| \quad (2)$$

where $q_n$ denotes the instantaneous queue length at the $n$th $q_{avg}$ update. When the instantaneous queue length is varying rapidly, the average queue length tends to deviate from the instantaneous. We determine that if the difference in the instantaneous queue length and the average length is

[2] $q_{avg_n}$ denotes the average queue size at the $n$th $q_{avg}$ update.

---

On the $n$th $q_{avg}$ Update

- Calculate average queue size as
  $q_{avg_n} \leftarrow (1 - w_q)q_{avg_{n-1}} + w_q.q$
- If $|\Delta avg| < \rho_1$
  $w_q = w_1$
  else if $\rho_1 \le |\Delta avg| < \rho_2$
  $w_q = w_{default}$
  else
  $w_q = w_2$
  where $\Delta avg = q_{avg_n} - q_{avg_{n-1}}$ and
  $w_1 < w_{default} < w_2$
  $\rho_1 < \rho_2$ are two threshold values
- Rest of the RED algorithm remains the same.

---

Figure 2: *Adaptive RED algorithm ($w_q$: Thresholding)*

around 20-30 packets[3], when the buffer size is 50 packets, then it indicates that the instantaneous queue is varying rapidly. In this case, $|\Delta avg| \approx 0.002 * 25 = 0.05$. On the other hand, if the instantaneous queue length is reasonably stable, then the average converges towards the same ($|q_n - q_{avg_{n-1}}| \approx 2-3$). In such a case, $|\Delta avg| \approx 0.004 - 0.006$. Based on the above reasoning, we choose $\rho_1 = 0.005$ and $\rho_2 = 0.05$. The values of $w_1$ and $w_2$ control the extent to which the average queue length follows the instantaneous queue length. $w_1$ and $w_2$ are varied close to 0.0020 which is the default value of $w_q$ in the non-adaptive RED scheme (as suggested in [1]). We have chosen $w_1$ and $w_2$ as 0.0018 and 0.0022 respectively. We would also like to mention that the performance varies marginally with respect to the parameters $w_1$ and $w_2$.

## 4. PERFORMANCE EVALUATION

### 4.1. Experimental network

The simulations have been performed using the **ns v2.1b7** network simulator ([7]). The network

[3] The simulations have been performed with equal length packets.

topology shown in Figure 3 is used for simulating the various scenarios. The value of $N$ is varied for different scenarios. The default RED parameters are kept as $w_q = 0.0020$, $max_p = 0.1$, $max_{th} = 15$ and $min_{th} = 5$. The results are obtained for the RED router and the bottleneck link shown in the figure.

In all the simulations, the values of $\alpha$ and $\beta$ for the $max_p$ algorithm are taken as 2 and 1.5 respectively. The reason behind choosing these values is that $max_p$ should be decreased more drastically compared to when it is increased. A large increase in $max_p$ can lead to high loss rates and hence it should be increased cautiously. In [2], $\alpha$ and $\beta$ are taken as 3 and 2 respectively. However, we have found that the values $\alpha = 2$ and $\beta = 1.5$ give a relatively better performance for the scenarios considered in the paper and these have been used in the simulations. We study the performance under conditions of high (*Scenario I*) and moderate (*Scenario II*) packet loss rates.

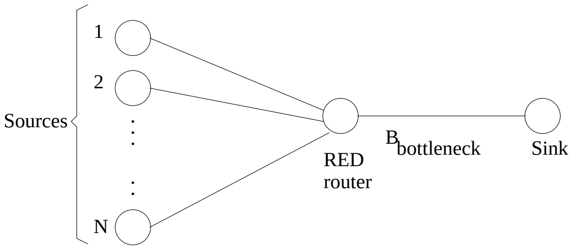


Figure 3: *Network Topology*

### 4.2. Simulation Results

For scenario I, the total number of nodes has been taken as 80. The bottleneck bandwidth is 1Mbps while the other link bandwidths are 3Mbps. Such a scenario results in a high packet loss rate ($\approx$ 25-30%) and high link utilization ($\approx$ 97-98%). For the scenario with moderate loss rate (scenario II), the number of nodes has been taken as 25. The packet loss rate is considerably lower ($\approx$ 10%) in this case.

From the simulation results (refer Tables[4] II and III), we can infer the following:

[4]By *packets*, we mean the number of *successfully* transmitted packets and *stability* refers to instantaneous queue stability.

Table 2: Performance Comparison (Scenario I): $w_q$-*thresh* and $max_p$ algorithms

| Algorithm | Drop% | Packets | Stability |
|---|---|---|---|
| Default | 30.39 | 3741.9 | No |
| Only $w_q$-*thresh* | 30.32 | 3755.6 | No |
| Only $max_p$ | 26.61 | 3724.9 | Yes |
| $w_q$-*thresh*+$max_p$ | 26.33 | 3740.2 | Yes |

Table 3: Performance Comparison (Scenario II): $w_q$-*thresh* and $max_p$ algorithms

| Algorithm | Drop% | Packets | Stability |
|---|---|---|---|
| Default | 12.69 | 2369.3 | No |
| Only $w_q$-*thresh* | 13.06 | 2375.6 | No |
| Only $max_p$ | 11.84 | 2345.7 | Yes |
| $w_q$-*thresh*+$max_p$ | 11.77 | 2366.7 | Yes |

The $w_q$-*thresh*+$max_p$ algorithm is much better than the default scheme in terms of queue length stability (see Figures 4, 5, 6 and 7). Using only the $max_p$ algorithm results in a drop in the number of packets transmitted successfully. Varying $w_q$ in conjunction with $max_p$ helps in achieving the benefits of varying $max_p$ (lower loss rate and better instantaneous queue stability) and at the same time counters the adverse effect of $max_p$ on

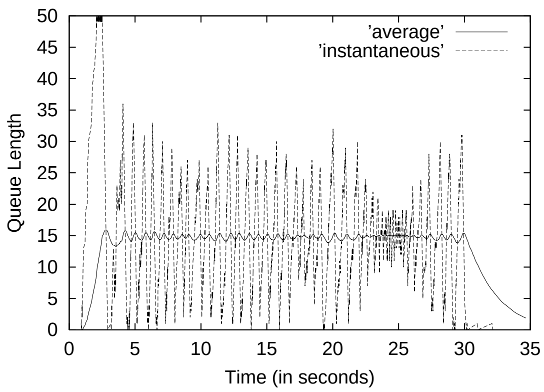


Figure 4: *Average and instantaneous queues with default RED (Scenario I)*

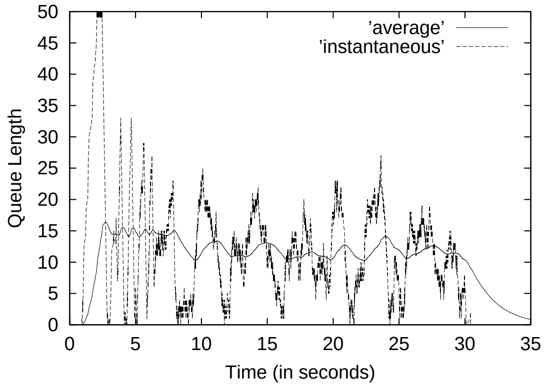


Figure 5: *Average and instantaneous queues with* $max_p$*+*$w_q$*-thresh algorithm (Scenario I)*

link utilization. The simulations also suggest that $max_p$ has a greater influence on the packet loss rate and queue stability while varying $w_q$ has a greater positive influence on link utilization compared to $max_p$. Further, varying only $w_q$ is not sufficient to lower the loss rate or to improve the instantaneous queue stability.

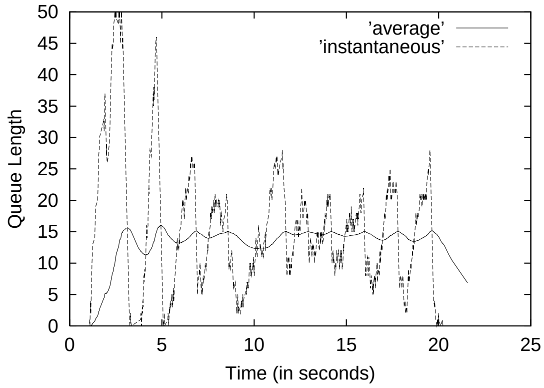


Figure 6: *Average and instantaneous queues with default RED (Scenario II)*

## 5. CONCLUSIONS

In this paper, we have looked at the performance of RED scheme in relation to its operating parameters. We have identified the impact of varying $max_p$ on link utilization and packet loss rate. We have studied the effect of varying $w_q$ in conjunction with $max_p$ and demonstrated how this can serve to counter the decrease in link utilization caused by varying $max_p$ alone. We conclude that

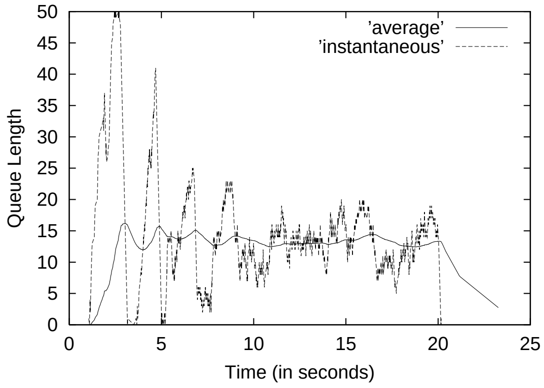


Figure 7: *Average and instantaneous queues with* $max_p$*+*$w_q$*-thresh algorithm (Scenario II)*

varying $max_p$ has a greater influence on packet loss rate and queue length stability, while varying $w_q$ improves the link utilization.

## 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] 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.

[3] Wu-chang Feng, Dilip D. Kandlur, Debanjan Saha and Kang G. Shin, "Blue: A New Class of Active Queue Management Algorithms", University of Michigan Technical Report CSE-TR-387-99, April 1999.

[4] D. Lin and R. Morris, "Dynamics of Random Early Detection", Proceedings of SIGCOMM 97, pp. 127-138, September 1997.

[5] Mikkel Christiansen, Kevin Jeffay, David Ott and F. Donelson Smith, "Tuning RED for Web Traffic", Proceedings of ACM SIGCOMM 2000, Stockholm, Sweden, Aug.-Sep. 2000.

[6] `www.aciri.org/floyd/REDparameters.txt`

[7] `www.isi.edu/nsnam/ns/`