INTERNATIONAL JOURNAL OF COMMUNICATION SYSTEMS
Int. J. Commun. Syst. 2008; **21**:987–1008
Published online 13 March 2008 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/dac.932

A framework to determine the optimal weight parameter of RED in next-generation Internet routers[‡]

Bing Zheng[1] and Mohammed Atiquzzaman[2,*,†]

[1]*Bookham Technology, San Jose, CA 94124, U.S.A.*
[2]*School of Computer Science, University of Oklahoma, Norman, OK 73019-6151, U.S.A.*

SUMMARY

Active buffer management can improve the performance of Transmission Control Protocol/Internet Protocol-based networks. Random early detection (RED), an active queue management scheme, has been proposed by the Internet Engineering Task Force for next-generation Internet routers. RED uses a number of parameters, such as buffer thresholds, a packet drop probability and a weight parameter. RED suffers from low throughput and large delay/jitter and induces instability in networks. Previous attempts to improve the performance of RED were based on modifying the thresholds and drop probabilities. In this paper, we show that an optimal value of the weight parameter can improve the performance of RED, and then develop a framework to determine the optimal value of the weight parameter. We show that the optimal weight parameter obtained from our framework improves the performance of RED. Copyright © 2008 John Wiley & Sons, Ltd.

Received 9 October 2007; Revised 30 January 2008; Accepted 30 January 2008

KEY WORDS: RED; queue management; modeling; parameter selection

1. INTRODUCTION

Active queue management (AQM), such as random early detection (RED) [1], can improve the performance of Transmission Control Protocol/Internet Protocol, and has therefore been recommended by the Internet Engineering Task Force for the next-generation Internet routers [2, 3]. *RFC 2309* [2] *requires low delay/jitter service to users as one of the goals of AQM.*

RED probabilistically drops packets when congestion starts to build up at a RED gateway. It uses a linear drop function to calculate the drop probability of a packet at a RED gateway, and uses four parameters to regulate its performance based on a calculated *average queue size*

*Correspondence to: Mohammed Atiquzzaman, School of Computer Science, University of Oklahoma, Norman, OK 73019-6151, U.S.A.
†E-mail: atiq@ou.edu
‡This work was carried out when the first author was at University of Dayton, Ohio.

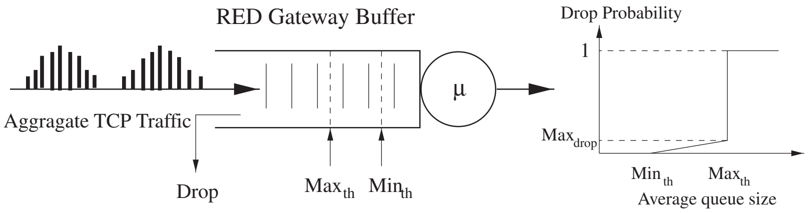


Figure 1. Illustration of the RED gateway queue.

(see Figure 1). The parameters Min_{th} and Max_{th} represent buffer thresholds for packet drop at the gateway, Max_{drop} is the maximum drop probability at Max_{th} and w is a weight parameter to calculate the average queue size from the instantaneous queue size.

The performance of RED has been widely studied (see [4–7] and the references contained therein). The behavior of RED is still not completely understood [6]; especially the relationship between its parameters and performance is still under research [4, 5, 8–11]. Studies have shown that RED has several problems such as low throughput [4, 12] and utilization [13], unfairness to connections [7, 14–17] and large delay/jitter [6, 18]. Several authors have proposed solutions to increase the throughput [4, 10, 12], ensure fairness among connections [5, 7, 14, 15], reduce the delay and jitter [10, 13] and stabilize router queue [19].

Studies have shown that the performance of RED depends, to a large extent, on the *appropriate selection of the RED parameters* [20]. Firoiu and Borden [21] observed that RED will induce network instability and cause major traffic disruption if the RED parameters are not properly configured. The effect of RED on network stability [22–26] and input traffic parameters [27] have also been studied by a number of authors. May *et al.* [6] have shown that RED exhibits a large delay variance that is very sensitive to the weight parameter (w) of RED. Christiansen *et al.* [18] and Wang *et al.* [28] found that the performance of RED in terms of delay is very poor under certain well-known parameter values of RED. They have shown that under heavy Web traffic, the response times perceived by Web users are very sensitive to the choice of RED parameters [29]. Moreover, the RED parameters that provide the best link utilization produce poor response times. They conclude that for links carrying only Web traffic, RED does not appear to provide any clear advantage over tail drop for end user response times.

To overcome some of the limitations of RED, a number of authors have proposed alternative AQM schemes. Feng *et al.* [30] proposed the BLUE algorithm, which uses packet loss and link idle events (instead of average queue length) to measure the severity of congestion at the routers. They have shown that BLUE performs better than RED in terms of packet loss rate and buffer size requirements. Hollot *et al.* [31] have applied classical control theory to come up with Proportional and Proportional-Integral controllers that have faster response time and stability as compared with RED. Most of the existing AQM schemes cannot provide accurate fair bandwidth sharing while being scalable. Chan and Hamdi [32] have proposed a scheme that estimates the number of flows in the buffer and the data source rate of a flow, in order to improve the fairness in bandwidth sharing among connections. Jie *et al.* [17] proposed to improve the bandwidth fairness of RED by identifying the high throughput flows according to its ‘drop-weight’ and raising the drop probability by increasing the value of Max_{drop} parameter to control the high throughput flows.

Previous work (see Section 2) on the selection of RED parameters have mainly focused on setting Min_{th}, Max_{th}, and Max_{drop} according to connection properties (such as bandwidth and delay) and traffic type (such as TCP and User Datagram Protocol (UDP)), *without paying much attention to optimizing the value of w*. RED uses exponentially weighted moving averages, which acts as a low-pass filter [1], to calculate the average queue length. These filters require a very small value of the weight (w) in order to effectively control the nonlinear instabilities in RED and to filter out bursty increases in the queue size [33].

The weight should be chosen in a manner such that short-term increases in the instantaneous queue length due to transient congestion should not significantly increase the average queue length. On the other hand, long-term increases in the queue length, resulting from persistent congestion, should have reasonable impact on the change in the average queue length. The value of w determines the sensitivity of the average queue length as a function of the instantaneous queue length. Too small a value of w will make the average queue length to respond too slowly to long-term increases in the gateway queue, and also fail to closely track rapid instantaneous queue length depletion, leading to under-utilization of links [33]. On the contrary, too large a value of w will result in the average queue length to respond too fast, and will be unable to filter out transient congestion at the gateway queue [1]. As the weight parameter is used to calculate the average queue length, it is *very important to chose an optimal value of w*.

The smallest value of w, below which the average queue length will not catch up with the long-term changes in the gateway queue, is defined as the *lower bound* of w. Similarly, the largest value of w, above which the average queue length will respond to the short-term changes of the gateway queue, is defined as the *upper bound* of w. Although the authors in [1] presented a method to determine the lower and upper bounds of w, their method was based on the following assumptions:

- The queue changes from empty to one packet and remains at one packet;
- The packets arrive and depart at the same rate.

As the average queue length attempts to track long- and short-term congestion, and a bottleneck link usually has a lower rate than the rate at which traffic arrives at the RED gateway, the assumptions are restrictive. We believe that the above assumptions result in improper (small) estimation of w, resulting in large jitter [6] in RED gateways. In addition to the assumptions mentioned above, the method described in [1] *lacks a general framework to calculate the weight parameter based on the long- and short-term congestion in the queue.*

The *objective* of this paper is to develop a framework to determine the lower and upper bounds of RED's weight parameter. The *aim* is to improve the delay and jitter performance of RED queues in the next-generation Internet routers. Our framework is based on the self-similar nature of the instantaneous queue size variation of RED queues. We propose *an analytical model to determine the weight parameter based on the self-similar model of the queue length variation*. We then evaluate the *delay/jitter performance of RED queues using the weight parameter calculated by our model*, and show that the weight parameter obtained from our model results in better performance than the value of the weight parameter mentioned in [1]. Our framework is very suitable for RED gateways that handle applications requiring low delay/jitter. It can also be used for Differentiated Services gateway routers that use RED with In and Out as the AQM scheme.

The rest of the paper is organized as follows. Previous work on RED is described in detail in Section 2. Assumptions and notations required to develop the framework are described in Section 3, followed by the framework to estimate the lower and upper bounds of the weight parameter in Section 4. In Section 5 we compare and analyze the performance of RED gateways using results obtained from our framework and previous work in the literature. Finally, conclusions are given in Section 6.

2. PREVIOUS WORK ON RED

To put our work in the context of previous work, in this section, we describe in detail the previous work on the various performance aspects of RED such as throughput, fairness and delay/jitter. A comprehensive survey on various aspects of RED can be found in [34].

2.1. Throughput of RED

Lakshman *et al*. [12] carried out a simulation of TCP/IP over Asynchronous Transfer Mode (ATM) to study the throughput of RED, and found that an exponential drop function is better than the linear drop function of RED. However, an exponential drop function may require more computing power, and may not be easily implemented in hardware except in certain limiting conditions. Suter *et al*. [4] studied the throughput of RED under per-flow queue management, and found that the throughput of RED is generally low for a large number of TCP connections. Moreover, with a mixture of bursty and greedy sources, RED suffers from low throughput and unfairness among connections. When TCP has to compete with more aggressive sources, or used in asymmetric networks with a perpetually congested reverse path, RED's throughput is very low.

Feng *et al*. [20] showed that the effectiveness of RED depends, to a large extent, on the appropriate selection of the RED parameters. They also showed that there is no single set of RED parameters that work well under all congestion scenarios. They therefore proposed an adaptive RED gateway that self-parameterizes itself based on the traffic mix. The result shows that such traffic-dependent parameterizations of RED gateways can effectively reduce packet loss while improving link utilization under a range of network loads. However, adaptive determination of the RED parameters complicates gateway management in high-speed routers.

Parris *et al*. [35] pointed out that RED is not suitable in the case of UDP transmissions. They proposed an extension to RED, called *Classed Based Threshold*, which uses different sets of buffer thresholds for different traffic types and priority classes. For example, UDP traffic uses buffer thresholds that are different from those used by TCP. TCP traffic is thus protected from UDP traffic.

Firoiu and Borden [21] modeled RED as a feedback control system, and pointed out that RED will induce network instability and major traffic disruption if the parameters are not properly chosen.

2.2. Fairness of RED

It is known that RED is not fair among connections [7], resulting in unfair throughput for different connections. To solve the problem of unfairness among connections, Kim and Lee [14] studied the *Fair Buffering RED* (FB-RED) for TCP over ATM. The main idea of FB-RED is to use the bandwidth delay product of a link to calculate the drop probability of the RED queue associated with the link. They implemented two cases: (i) the first one uses the inverse of bandwidth delay product to calculate the maximum drop probability and (ii) the second one uses the inverse of the square root of the bandwidth delay product to calculate the maximum drop probability. Although FB-RED results in fairness among connections, it however needs to track the information for all the connections, resulting in *scalability problems* that makes it unsuitable routers in the core of the network.

To solve the scalability problem of FB-RED, Ott *et al*. [5] proposed *Stabilized Random Early Drop* (SRED) that, similar to RED, preemptively discards packets with a load-dependent drop probability when a buffer gets congested. SRED however has an additional feature to estimate the number of active connections or flows; the estimate is obtained without collecting or analyzing state information of individual flows. Instead of calculating the average queue size, it uses the number of active flows and the instantaneous queue size to determine the packet drop probability. It divides the queue into three segments with a constant drop probability for each segment. SRED can stabilize, over a wide range of load levels, the buffer occupancy at a level that is independent of the number of active connections. Although SRED overcomes the drawback of [14], it suffers from *low throughput* even for a small number of traffic flows.

Lin and Morris [15] proposed *Fair RED*, which tracks a flow's use of buffer space (per-active-flow accounting) to determine the drop rate of the flow. Although it achieves a fair drop rate for different flows, it needs to track the state of each flow, which results in scalability problems in the core of the network, as in [14].

Arce *et al*. [33] used medians, instead of averages (used in RED), to calculate the queue size for dropping packets. The method has been shown to provide a fairer treatment to bursty traffic than the RED algorithm does. To solve the scalability problem of RED for large number of connections, Joo and Bahk [36] proposed the adjustment of RED's drop probability curve, for varying network conditions.

2.3. Delay and jitter of RED

May *et al*. [6] studied the *queuing delay and delay variance* (jitter) of RED. It was found that the delay variance in a RED buffer is large and very sensitive to the weight parameter (w). The smaller the value of w, the larger the delay variance.

For Web browsing under low to medium levels of network congestion, it has been found that RED has minimal effect on Hypertext Transfer Protocol response time regardless of the setting of its parameters [18]. Under heavy congestion, RED can be carefully tuned [37] to yield higher throughput than tail drop at the expense of sacrificing the delay. Therefore, for Web traffic, RED provides no clear advantage over tail drop.

From the above discussion, we observe that RED suffers from *low throughput*, *large delay/jitter*, *and induces instability in the network*. Moreover, for real-time application, RED's delay is also large. In Section 4, we will develop a framework to determine an optimal value of w, which will increase the throughput of RED.

3. MODELING ASSUMPTIONS AND NOTATIONS

In this section, we describe the assumptions and notations that will be used to develop our framework for determining the upper and lower bounds of the weight parameter in Section 4.

Research has shown that local area network traffic is much better modeled by a *self-similar* process [38–40] rather than a Poisson distribution. Self-similar traffic pattern over a wide range of time scale has a similar form to that of the traffic pattern over a short time scale. It has been shown in [1] that if the input traffic pattern to a buffer with deterministic service time is self-similar,

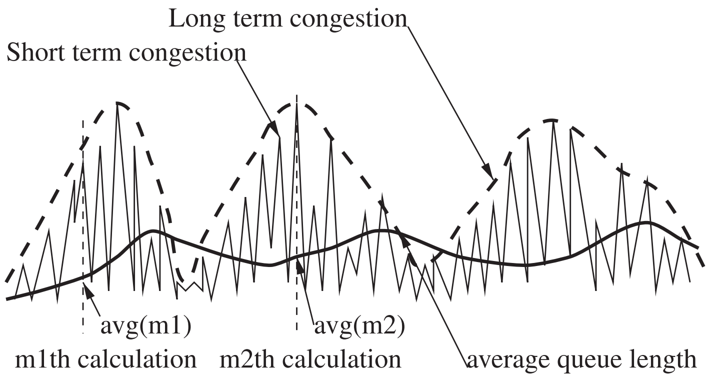


Figure 2. Self-similar model of the instantaneous queue length at a RED gateway.

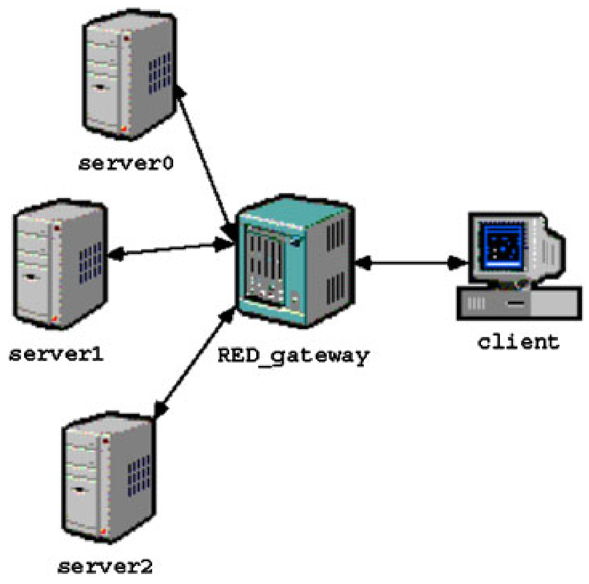


Figure 3. Simulation configuration to test the performance of a RED gateway for different values of w.

then the queue length variation in a RED buffer exhibits a self-similar phenomenon as shown in Figure 2. The *queue length variation* of a RED gateway can therefore be described by long- and short-term congestion as described below:

- *Long-term congestion* is the variation (burstiness) in the queue length resulting from persistent congestion in the buffer over a large time scale [2].
- *Short-term congestion* is the variation in the queue length resulting from transient congestion in the buffer over a short time scale.

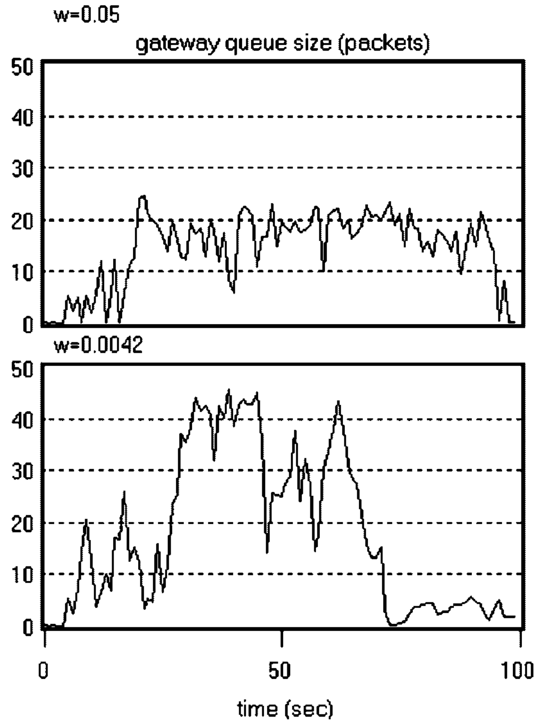


Figure 4. Instantaneous RED gateway queue size for $w=0.05$ and 0.0042.

3.1. Assumptions

We make the following assumptions that will be used to describe the model for determining the weight parameter of a RED queue in Section 4. Note that some of the assumptions have been used in previous work as indicated below.

- The RED gateway queue is initially empty (also assumed in [1]).
- The average queue length is initially zero (also assumed in [1]).
- Long-term congestion, which varies slowly, is described by a slow function $g(i)$, where i corresponds to the ith calculation of the average queue length.
- Short-term congestion, which varies fast, is described by a fast function $f(i)$ corresponding to the ith calculation of the average queue length.
- The instantaneous queue length, $q(i)$, is described by the modulation of the fast function ($f(i)$) by a slow function ($g(i)$), i.e.

$$q(i)=g(i)f(i) \tag{1}$$

3.2. Notations

We define the following variables that are used in our model in Section 4:

- w, weight parameter for calculation of average queue length at a RED gateway;
- $q(i)$, instantaneous queue size of the RED gateway during the ith calculation of the average queue length. From our assumptions in Section 3.1, $q(0)=0$;

- avg(i), average queue length of the RED gateway at the ith calculation. It is defined as

$$\text{avg}(i) = (1-w)*\text{avg}(i-1) + w*q(i) \tag{2}$$

  where avg(0) = 0 from our assumptions in Section 3.1;
- K_l, minimum buffer threshold for RED gateway to perform active packet drop;
- K_h, maximum buffer threshold for RED gateway to perform packet drop with probability of one;
- $\delta(m_2, m_1)$, difference in the average queue lengths between the calculations at m_1 and m_2, i.e.

$$\delta(m_2, m_1) = \text{avg}(m_2) - \text{avg}(m_1) \tag{3}$$

4. DETERMINATION OF WEIGHT PARAMETER, w

In this section, we develop a model to determine the lower and upper bounds of w by considering the variation of the queue length. The lower and upper bounds provide a range of w values that can be used by network operators to optimize the performance of next-generation routers employing

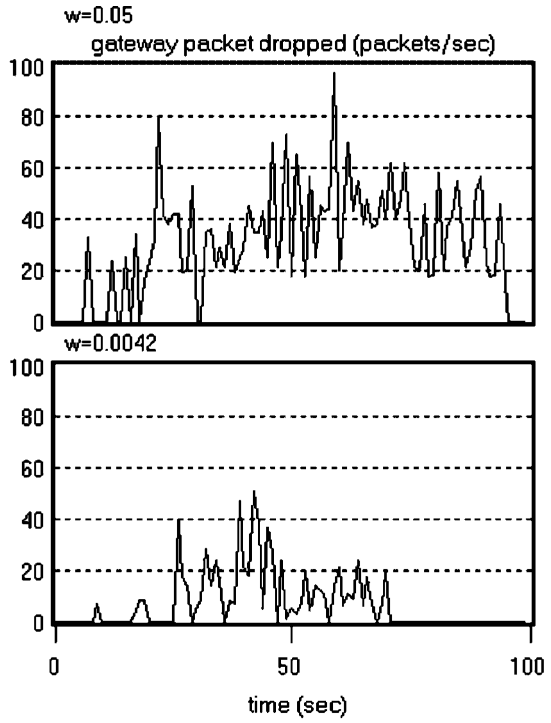


Figure 5. Packet drop at a RED gateway queue for $w = 0.05$ and 0.0042.

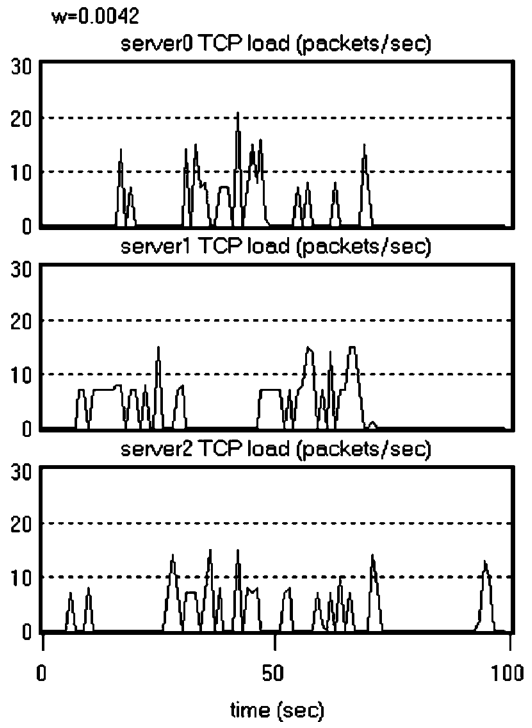


Figure 6. TCP load at the three servers for $w = 0.0042$.

RED queues. From Equation (2) we obtain

$$\begin{aligned}
\text{avg}(1) &= (1-w)*\text{avg}(0) + w*q(1) \\
&= w*q(1) \qquad (4) \\
\text{avg}(2) &= (1-w)*\text{avg}(1) + w*q(2) \\
&= (1-w)*w*q(1) + w*q(2) \qquad (5) \\
\text{avg}(3) &= (1-w)*\text{avg}(2) + w*q(3) \\
&= (1-w)^2*w*q(1) + (1-w)*w*q(2) + w*q(3) \qquad (6) \\
\vdots \;\; & \vdots \\
\text{avg}(m_1) &= \sum_{i=1}^{m_1} (1-w)^{m_1-i} w*q(i) \qquad (7)
\end{aligned}$$

Using Equation (7), $\delta(m_2, m_1)$ can be expressed as

$$\delta(m_2, m_1) = \sum_{i=1}^{m_2-m_1} (1-w)^{m_2-i} w g(i) f(i) \qquad (8)$$

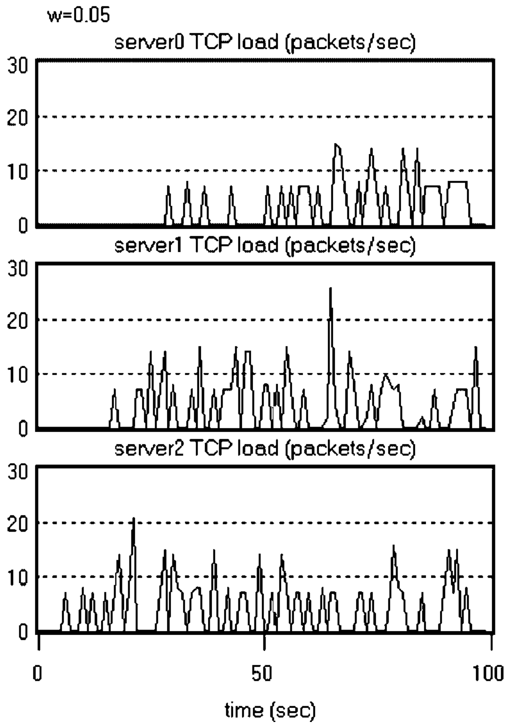


Figure 7. TCP load at the three servers for $w = 0.05$.

In the following two subsections, we estimate the *lower bound* and *upper bound* of w based on Equation (8).

4.1. Lower bound of w

As mentioned in Section 1, too small a value of w cannot track the short-term congestion of the queue. The lower bound of w is therefore determined by the ability of the RED algorithm to track short-term congestion. If $\delta(m_2, m_1) \geqslant K_l$, packets will definitely be dropped irrespective of the value of $\text{avg}(m_1)$. Therefore, to obtain the lower bound of w, $\delta(m_2, m_1)$ should be less than K_l to track short-term congestion.

As we describe the instantaneous queue size by modulating a fast varying function $f(i)$ by a slow varying function $g(i)$ (see Section 3.1), the difference $m_2 - m_1$ should be small to enable tracking of short-term congestion. Therefore, $g(i)$ is almost constant between the two calculation points (m_1 and m_2), which are very close.

If we denote $\overline{m} = (m_1 + m_2)/2$, Equation (8) can be expressed as

$$\delta(m_2, m_1) = (1-w)^{m_2} w g(\overline{m}) \sum_{i=1}^{m_2 - m_1} (1-w)^{-i} f(i) \tag{9}$$

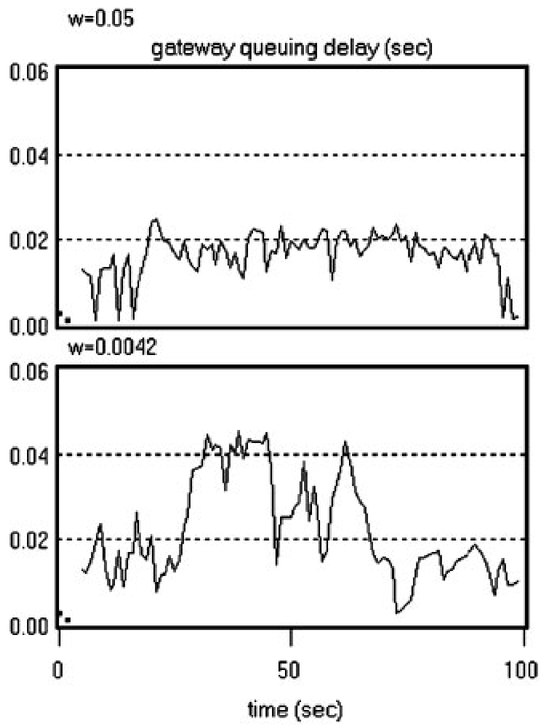


Figure 8. Queuing delay at a RED gateway queue for $w=0.05$ and 0.0042.

As the short-term queue size variation can be described by an ON–OFF model [40], $f(i)$ can be expressed as

$$f(i)=q_0\left(\frac{1-(-1)^i}{2}+b\frac{1-(-1)^{i-1}}{2}\right) \tag{10}$$

where b is the burst ratio between the ON and OFF states, and q_0 is a scale parameter representing the queue length during the OFF period of long-term congestion. For the above ON–OFF model, $f(i)$ and $f(i+2)$ represent two adjacent ON (or OFF) periods. Therefore, $m_2-m_1=2$ in the calculation of $\delta(m_2,m_1)$ for two adjacent short-term peaks. Then, we have

$$\delta(m_2,m_2-2)=(1-w)^{m_2}wg(\overline{m})q_0\sum_{i=1}^{2}(1-w)^{-i}\left(\frac{1-(-)^i}{2}+b\frac{1-(-1)^{i-1}}{2}\right) \tag{11}$$

By simplifying Equation (11) with two adjacent burst arrivals, we obtain

$$\begin{aligned}\delta(m_2,m_2-2)&=(1-w)^{m_2-1}wg(\overline{m})q_0\left(1+\frac{b}{1-w}\right)\\&=(1-w)^{m_2-1}wg(\overline{m})q_0\frac{1-w+b}{1-w}\end{aligned} \tag{12}$$

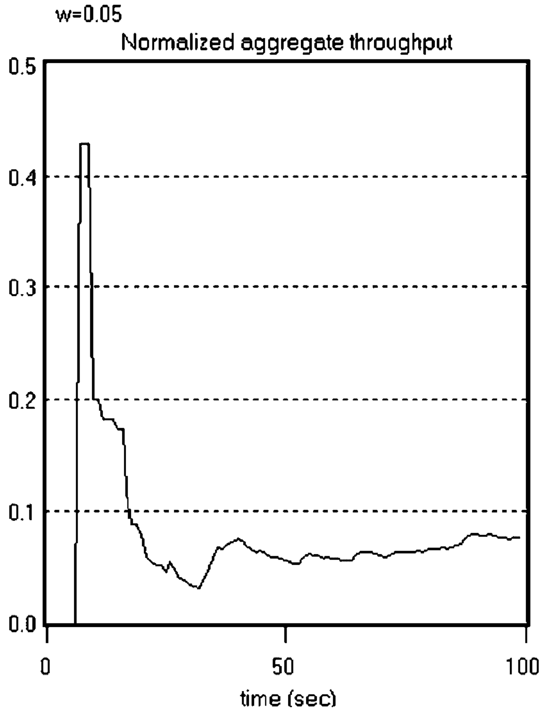


Figure 9. Normalized end-to-end throughput for $w = 0.05$.

As $w \ll 1$ and $b \gg 1$, we have $1 - w + b \simeq 1 + b$. Therefore,

$$\delta(m_2, m_2 - 2) \simeq (1 - w)^{m_2 - 2} w g(\overline{m}) q_0 (b + 1) \tag{13}$$

As explained in the beginning of this section, the condition $\delta(m_2, m_1) \leqslant K_l$ must be satisfied to accommodate adjacent burst arrivals without packet drop. Using this condition, Equation (13) gives

$$(1 - w)^{m_2 - 2} w \leqslant \frac{K_l}{g(\overline{m}) q_0 (b + 1)} \tag{14}$$

Equation (14), therefore, gives the *lower bound* for w.

4.2. Upper bound of w

To derive the upper bound for w, we need to consider the variation of the slow function $g(i)$ over a period that is featured by long-term congestion. Here we assume that $g(i)$ increases and decreases linearly [1] and periodically [40]. During this period, the change in average queue length must be large enough to capture the long-term behavior of network congestion. Let us denote $l_1 = m_2 - m_1$

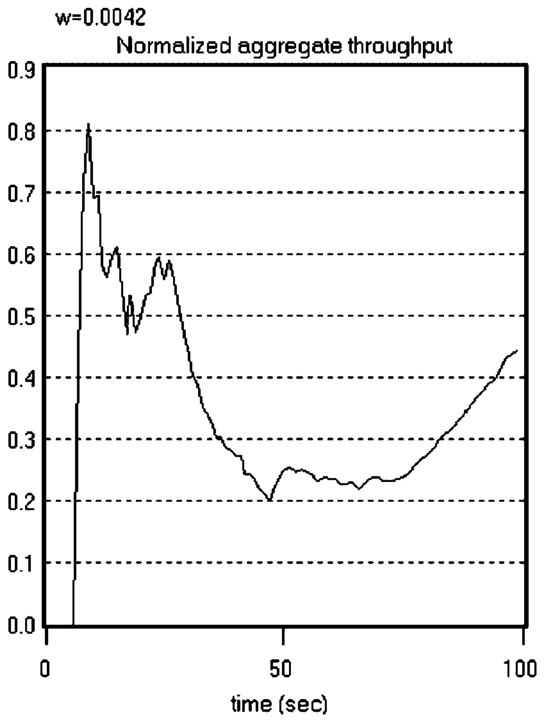


Figure 10. Normalized end-to-end throughput for $w = 0.0042$.

to be half of a long-term congestion period as shown in Figure 2. By substituting Equation (10) into Equation (8), and using $l_1 = m_2 - m_1$, we obtain

$$\delta(m_2, m_2 - l_1) = (1-w)^{m_2} w q_0 \sum_{i=1}^{l_1} (1-w)^{-i} i \left(\frac{1-(-)^i}{2} + b \frac{1-(-1)^{i-1}}{2} \right) \tag{15}$$

Note that for the series $S1 = 1x + 3x^3 + 5x^5 + \cdots$ and $S2 = 2x^2 + 4x^4 + 6x^6 + \cdots$, the following approximations hold when $x > 1$:

$$S1 = x \frac{(2n-1)x^{2n+2} - (2n+1)x^{2n}}{(1-x^2)^2} \simeq \frac{2nx^{2n+1}}{x^2 - 1} \tag{16}$$

$$S2 = x \frac{2nx^{2n+3} - (2n+2)x^{2n+1}}{(1-x^2)^2} \simeq \frac{2nx^{2n+2}}{x^2 - 1} \tag{17}$$

where n is the term number in the series. Using Equations (16) and (17) in Equation (15), we obtain

$$\delta(m_2, m_2 - l_1) \simeq (1-w)^{m_2} w q_0 \left(b \frac{l_1 (1-w)^{-l_1 - 2}}{(1-w)^{-2} - 1} + \frac{l_1 (1-w)^{-l_1 - 1}}{(1-w)^{-2} - 1} \right) \tag{18}$$

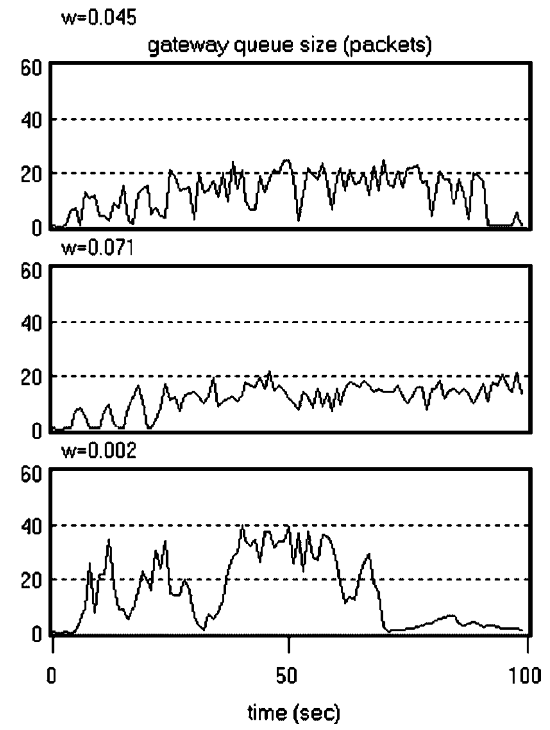


Figure 11. Instantaneous RED gateway queue size for $w=0.045$, 0.071 and 0.002.

For $w \ll 1$, we obtain

$$\frac{1}{1-w}+1 \simeq \frac{2}{1-w}$$

Then

$$\delta(m_2, m_2-l_1) \simeq \frac{q_0 l_1 (b+2)}{2}(1-w)^{m_2-l_1} \tag{19}$$

To track long-term congestion, it is necessary that the change in average queue length between two calculations should be no less than K_l, i.e. $\delta(m_2, m_1) \geqslant K_l$. This gives us

$$K_l \leqslant \frac{q_0 l_1 (b+2)}{2}(1-w)^{m_2-l_1} \tag{20}$$

From the above equation, we obtain the *upper bound* of the weight parameter w as

$$w \leqslant 1-\mathrm{e}^{\frac{\ln \frac{2K_l}{(b+2)q_0 l_1}}{m_2-l_1}} \tag{21}$$

Equations (14) and (21) set the *lower bound* and *upper bound*, respectively, for the weight parameter w.

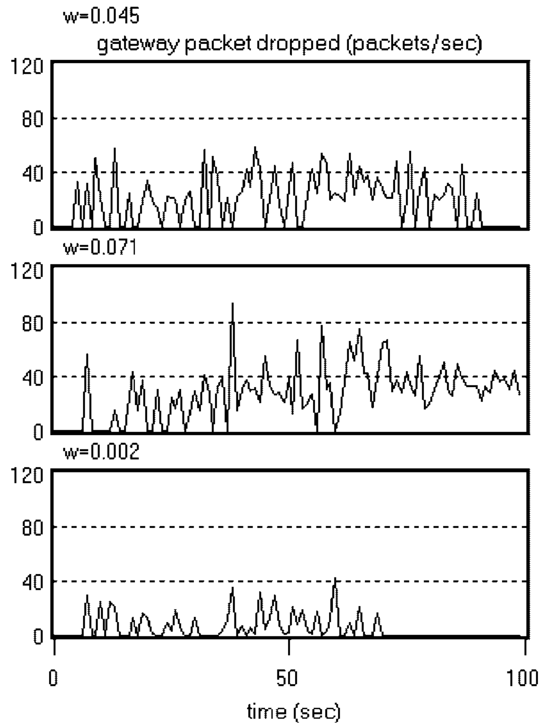


Figure 12. Packet drops at a RED gateway queue for $w = 0.045, 0.071$ and 0.002.

4.3. *Numerical bounds of w*

Having developed the expressions for the lower and upper bounds of w in the above sections, we use the expressions to determine some representative numerical bounds of w.

From Figure 3(a) in [1], we can extract the following values. $b = 4$, $K_l = 5$, $q_0 = 5$, $l_1 = 14$, $g(\overline{m}) = 16$ and $m_2 = 30$ and 65 for calculating the lower and upper bounds, respectively. By using the above parameters in Equations (14) and (21), the lower and upper bounds of w are obtained as 0.045 and 0.071, respectively. In Section 5, we will use these two boundary values of w, in addition to $w = 0.05$ (which represents a typical value satisfying the bounds obtained from our model), to determine the performance of RED and compare it with previous studies.

5. SIMULATION RESULTS AND DISCUSSION

In Section 4, we proposed a model that takes into account the variation of the queue length to determine the lower and upper bounds of the weight parameter of a RED queue. To check the effectiveness of our model, we ran simulations of RED queues with five values of w: three obtained from our model (see Section 4.3) and two proposed in [1]. In this section, we compare the performance of RED using the five different values of w.

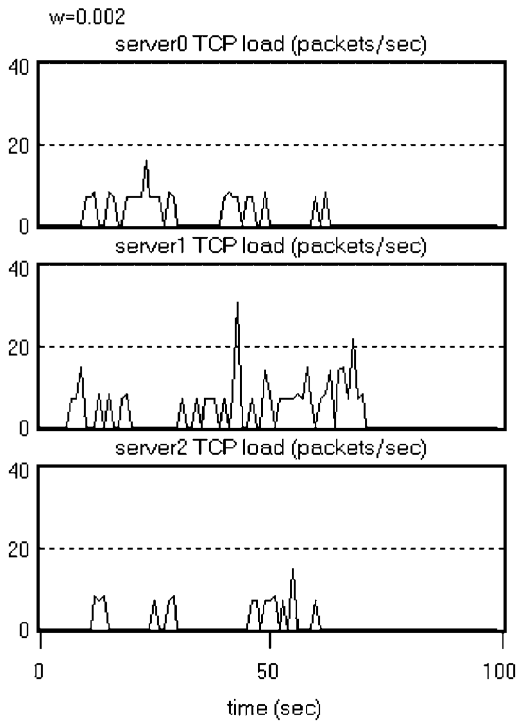


Figure 13. TCP load at the three servers for $w = 0.002$.

5.1. Simulation configuration

To enable a fair comparison of our work with previous work, we have used the simulation configuration shown in Figure 3, which has been used by previous researchers [1, 6, 20]. Three TCP-based servers, using File Transfer Protocol at the application layers, send traffic to a client through a RED gateway that feeds a bottleneck link. Simulations were carried out with the OPNET 5.1 network simulation tool for different values of w as described above. Parameters for the network configuration are given below:

- *Server0 to RED gateway link*: Propagation delay 1 ms, link rate 100 Mbps.
- *Server1 to RED gateway link*: Propagation delay 5 ms, link rate 100 Mbps.
- *Server2 to RED gateway link*: Propagation delay 3 ms, link rate 100 Mbps.
- *Client to RED gateway link*: Propagation delay 5 ms, bottleneck link rate 10 Mbps. To induce congestion at the RED queue, the bottleneck link rate has been chosen to be 30 times smaller than the sum of link rates feeding the bottleneck link.
- *Gateway processing speed*: 1 ms per packet.
- *Gateway queue size*: 50 packets.
- $K_l = 5$, $K_h = 15$: These values have been selected using the rule given in [1, 41].
- $w = 0.045, 0.05, 0.071$ (as calculated by our model in Section 4.3) and 0.0042, 0.002 (as given in [1]).
- $\text{Max}_{\text{drop}} = 0.1$ [41].

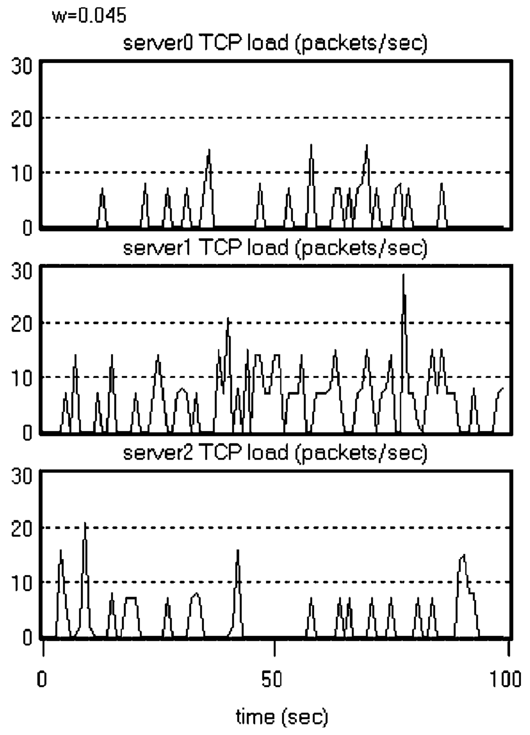


Figure 14. TCP load at the three servers with $w = 0.045$.

The above values of the link bandwidths were chosen to induce congestion at the gateway. To test the fairness of TCP connections having different Round Trip Times, the delays of the three links connecting the servers to the RED gateway were chosen to be different.

5.2. *Results and discussions*

For the network configuration given in Figure 3, we ran simulations for 100 s of simulation time in order to cover enough time for the results to reach the steady state. In this section, we compare the performance of RED for two sets of parameter values:

- $w = 0.05$ (an arbitrary value within the range suggested by our model in Section 4.3) *versus* $w = 0.0042$ (as suggested in [1]);
- $w = 0.045$ and 0.071 (the upper and lower bounds suggested by our algorithm) *versus* $w = 0.002$ (as suggested in [41]).

5.2.1. RED performance for $w = 0.05$ and 0.0042. Figure 4 shows the instantaneous queue size at the RED gateway for $w = 0.05$ and 0.0042. It is seen that for $w = 0.05$, the queue size and queue size variation are both much smaller than for $w = 0.0042$. Figure 5 shows packet drops at the RED gateway. Although the queue drops more packet for $w = 0.05$ than for $w = 0.0042$, it is important to note that *the drops for $w = 0.05$ are active packet drop rather than drops due to buffer overflow (tail drop) occurring for $w = 0.0042$*. The fact that $w = 0.0042$ causes tail drops is

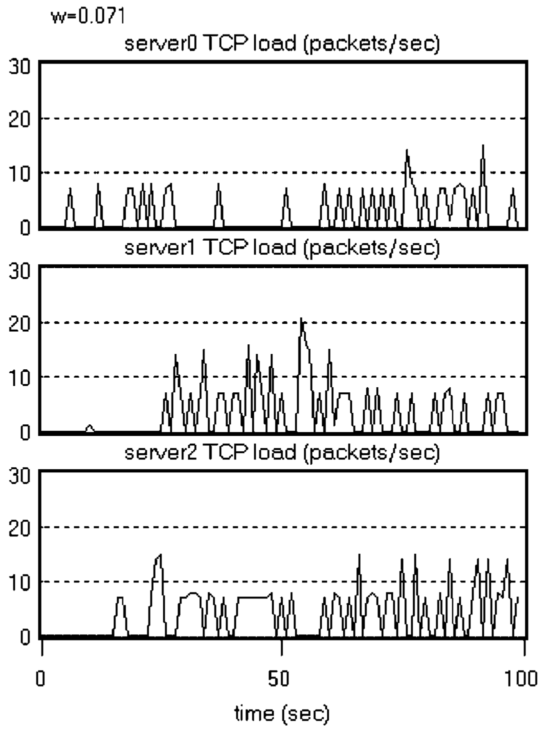


Figure 15. TCP load at the three servers with $w=0.071$.

evident from global synchronization as shown in Figure 6, where the three TCP servers back off (no transmission of data) simultaneously during the time interval between 72 and 95 s. The global synchronization results from an improper choice of $w=0.0042$, which is too small to perform AQM (resulting in drops being predominantly due to tail drop) in this experiment. (Note that May *et al.* [6] also observed global synchronization when using RED.) On the contrary, for $w=0.05$ (as obtained from our proposed algorithm), global synchronization is avoided because of AQM over the entire simulation time as shown in Figure 7, where there is no global synchronization among the TCP servers.

Figure 8 shows the instantaneous queuing delay experienced by packets at the RED gateway queue. As the gateway queue for $w=0.05$ is smaller than for $w=0.0042$ (see Figure 4), the instantaneous queuing delay is also smaller. Furthermore, as the instantaneous gateway queue size for $w=0.05$ has a low variance (see Figure 4), it also results in a smaller delay variance (jitter).

Figures 9 and 10 show the aggregate end-to-end TCP throughput for $w=0.05$ and 0.0042, respectively. As $w=0.05$ results in a higher packet drop than for $w=0.0042$, it is reasonable that $w=0.05$ results in a lower normalized throughput.

5.2.2. RED performance for $w=$ 0.045, 0.071 and 0.002. In Figures 4–8, we have shown that $w=0.05$ (an arbitrary value within the range of w suggested by our algorithm) results in a higher performance of RED as compared with using $w=0.0042$ (as suggested in [1]). However, recently Floyd [41] has suggested using $w=0.002$. To further test the effectiveness of our algorithm, we

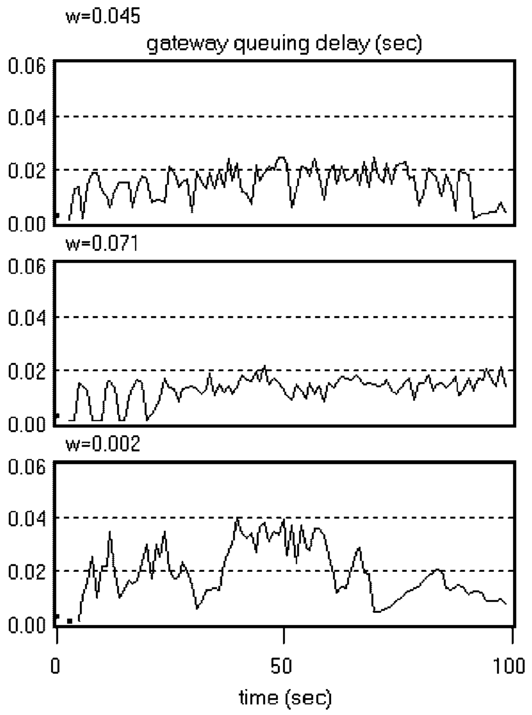


Figure 16. Queuing delay at a RED gateway queue for $w = 0.045, 0.071$ and 0.002.

have carried out simulations with $w = 0.002$, 0.045 and 0.071, where the last two values are the lower and upper bounds as suggested by our algorithm (see Section 4.3). Figure 11 shows that both the bounds ($w = 0.045$ and 0.071) suggested by our algorithm result in a lower instantaneous queue size. Although the packet drop is higher for $w = 0.045$ and 0.071 as seen in Figure 12, it should be noted that the drops are due to active packet drops as mentioned above. This is confirmed by global synchronization happening at $w = 0.002$ (Figure 13) and the lack of global synchronization for $w = 0.045$ (Figure 14) and 0.071 (Figure 15). Figure 16 shows that the bounds of w suggested by our algorithm results in a lower queuing delay than $w = 0.002$.

6. CONCLUSION

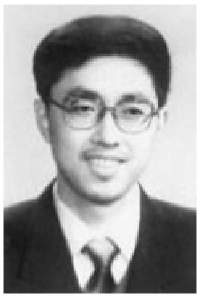
In this paper, we have developed a model to calculate the *lower* and *upper* bounds of the weight parameter in a RED gateway. The model takes into account the short- and long-term congestion at the gateway to calculate the bounds of the weight parameter of a RED queue. The bounds on the weight parameter calculated from our model allow the RED algorithm to trace persistent congestion at the gateway, while filtering out short-term transient congestion. We have shown that a RED gateway configured with the weight parameter obtained from our model provides lower delay/jitter for TCP traffic when compared with previous models. Therefore, our model is suitable for real-time applications that require low delay/jitter, such as IP Telephony and video conferencing. Our model can also be used to dynamically configure the weight parameter of a RED gateway based on the TCP traffic pattern.

REFERENCES

1. Floyd S, Jacobson V. Random early detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking* 1993; **1**(4):397–413.
2. Braden B, Clark D, Crowcroft J, Davie B, Deering S, Estrin D, Floyd S, Jacobson V, Minshall G, Partridge C, Peterson L, Ramakrishnan K, Shenker S, Wroclawski J, Zhang L. Recommendations on queue management and congestion avoidance in the Internet. *RFC 2309*, April 1998.
3. Heinanen J, Baker F, Weiss W, Wroclawski J. Assured forwarding PHB. *RFC 2597*, June 1999.
4. Suter B, Lakshman TV, Stiliadis D, Choudhury AK. Buffer management schemes for supporting TCP in gigabit routers with per-flow queuing. *IEEE Journal on Selected Areas in Communications* 1999; **17**(6):1159–1169.
5. Ott TJ, Lakshman TV, Wong L. SRED: stabilized RED. *IEEE INFOCOM99*, New York, March 1999; 1346–1355.
6. May M, Bonald T, Bolot J-C. Analytic evaluation of RED performance. *IEEE INFOCOM*, Tel-Aviv, Israel, 26–30 March 2000.
7. Hasegawa G, Matsuo T, Murata M, Miyahara H. Comparisons of packet scheduling algorithms for fair service among connections on the Internet. *IEEE INFOCOM*, Tel-Aviv, Israel, 26–30 March 2000; 1253–1262.
8. Zheng B, Atiquzzaman M. Active queue management. In *High Performance TCP/IP Networking*, Hassan M, Jain R (eds), Chapter 12. Prentice-Hall: Englewood Cliffs, NJ, 2004; 281–307.
9. Feng WC, Kandlur DD, Saha D. Adaptive packet marking for maintaining end to end throughput in a differentiated service Internet. *IEEE/ACM Transactions on Networking* 1999; **7**(5):685–697.
10. Zheng B, Atiquzzaman M. DSRED: an active queue management scheme for next generation networks. *LCN*: *The IEEE Conference on Local Computer Networks*, Tampa, FL, 8–10 November 2000; 242–251.
11. Sirisena H, Haider A, Pawlikowski K. Auto-tuning RED for accurate queue control. *Globecom 2002*, Taipei, Taiwan, 2002; 2010–2015.
12. Lakshman TV, Neidhardt A, Ott TJ. The drop from front strategy in TCP and in TCP over ATM. *IEEE INFOCOM*, San Francisco, CA, 26–28 March 1996; 1242–1250.
13. Athuraliya S, Low S, Li VH, Yin Q. REM: active queue management. *IEEE Network* 2001; **15**(3):48–53.
14. Kim W-J, Lee BG. The FB-RED algorithm for TCP over ATM. *IEEE GLOBECOM*, Sydney, Australia, 8–12 November 1998; 551–555.
15. Lin D, Morris R. Dynamics of random early detection. *ACM SIGCOMM*, Cannes, France, 14–18 September 1997; 127–137.
16. Feng WC, Feng WC. The impact of active queue management on multimedia congestion control. *ICCCN*: *IEEE International Conference on Computer and Communication Networks*, Louisiana, October 1998.
17. Jie Y, Jun L, Zhenming L. Bf-red: a novel algorithm for improving bandwidth fairness of red. *IEEE International Conference on Networking*, *Sensing and Control*, Ft. Lauderdale, FL, 2007; 1001–1005.
18. Christiansen M, Jeffay K, Ott D, Smith FD. Tuning RED for web traffic. *ACM SIGCOMM*, Stockholm, Sweden, August 2000.
19. Zhu L, Ansari N. Local stability of a new adaptive queue (AQM) management scheme. *IEEE Communications Letters* 2004; **8**(6):406–408.
20. Feng WC, Kandlur DD, Saha D, Kang DG. Self-configuring RED gateway. *IEEE INFOCOM*, New York, March 1999; 1320–1328.
21. Firoiu V, Borden M. A study of active queue management for congestion control. *IEEE INFOCOM*, Tel-Aviv, Israel, 26–30 March 2000; 1435–1444.
22. La RJ. Instability of a tandem network and its propagation under RED. *IEEE Transactions on Automatic Control* 2004; **49**(6):1006–1011.
23. Low SH, Paganini F, Wang J, Adlakha S, Doyle JC. Dynamics of TCP/RED and a scalable control. *IEEE INFOCOM 2002*, New York, NY, 2002; 239–248.
24. Ranjan P, Abed EH, La RJ. Nonlinear instabilities in TCP-RED. *IEEE/ACM Transactions on Networking* 2004; **12**(6):1079–1092.
25. Wang L, Cai L, Liu X, Shen X. Stability and tcp-friendliness of AIMD/RED systems with feedback delays. *Computer Networks* 2007; **51**(15):4475–4491.
26. Chen X, Wong S, Tse C, Trajkovic L. Stability analysis of RED gateway with multiple TCP connections. *IEEE International Symposium on Circuits and Systems*, New Orleans, LA, 27–30 May 2007.
27. Feng G, Agarwal AK, Jayaraman A, Siew CK. Modified RED gateways under bursty traffic. *IEEE Communications Letters* 2004; **8**(5):323–325.
28. Wang C, Li B, Hou T, Sohraby Y, Lin Y. LRED: a robust active queue management scheme based on packet loss ratio. *IEEE INFOCOM*, Hong Kong, China, 7–11 March 2004; 12–16.

29. Christiansen M, Jeffay K, Ott D, Smith F. Tuning RED for Web traffic. *IEEE/ACM Transaction on Networking* 2001; **9**(3):249–264.
30. Feng WC, Kandlur D, Saha D, Shin KG. BLUE: a new class of active queue management algorithms. *Technical Report CSE-TR 387-99*, University of Michigan, Ann Arbor, MI, April 1999. Also appears in *IEEE/ACM Transactions on Networking* 2002; **10**(4):513–528.
31. Hollot CV, Misra V, Towsley D, Gong W-B. On designing improved controllers for AQM routers supporting TCP flows. *IEEE INFOCOM 2001*, Anchorage, Alaska, April 2001; 1726–1734.
32. Chan MK, Hamdi M. An active queue management scheme based on a capture–recapture model. *IEEE Journal on Selected Areas in Communications* 2003; **21**(4):572–583.
33. Arce GR, Barner KE, Ma L. RED gateway congestion control using median queue size estimates. *IEEE Transactions on Signal Processing* 2003; **51**(8):2149–2164.
34. Zheng B, Atiquzzaman M. Active queue management in TCP/IP networks. *Performance of TCP over Satellite Networks*. Prentice-Hall: Englewood Cliffs, NJ, 2003.
35. Parris M, Jeffay K, Smith FD. Lightweight active router queue management for multimedia networking. *Proceedings of SPIE*, San Jose, CA, U.S.A., 25–27 January 1999; 162–174.
36. Joo C, Bahk S. Scalability problems of RED. *Electronics Letters* 2002; **38**(21):1297–1298.
37. Qiang C, Yang OWW. On designing self-tuning controllers for AQM routers supporting TCP flows based on pole placement. *IEEE Journal on Selected Areas in Communications* 2004; **22**(10):1965–1974.
38. Leland WE, Taqqu MS, Willinger W, Wilson DV. On the self-similar nature of Ethernet traffic. *IEEE/ACM Transactions on Networking* 1994; **2**(1):1–15.
39. Paxson V, Floyd S. Wide area traffic: the failure of Poisson modeling. *IEEE/ACM Transactions on Networking* 1995; **3**(3):226–244.
40. Kalampoukas L, Varma A, Ramakrishnam KK. Two way TCP traffic over rate controlled channels: effects and analysis. *IEEE/ACM Transactions on Networking* 1998; **6**(6):729–743.
41. Floyd S. RED: discussions of setting parameters, November 1997 (Available from: www.aciri.org/floyd/REDparameters.txt.)

AUTHOR'S BIOGRAPHIES



Bing Zheng received his PhD degree in Electrical Engineering in August 2001, from University of Dayton. He received a BSc in Physics in 1984 and an MSc in Electro–Optics in 1990. Before joining the Department of Electrical and Computer Engineering of University of Dayton in August 1997, he was an Associate Professor working in tunable WDM technology, optical fiber technology, integrated optoelectronic technology, and flat panel display technology with the University of Electronic Science and Technology of China (UESTC). His working areas include TCP, IP, ATM, multimedia transmission, VHDL/FPGA/Digital system design, tunable WDM technology, fiber optical technology, and photonic/optoelectronic technology, etc.

Mohammed Atiquzzaman received his MSc and PhD degrees in Electrical Engineering and Electronics from the University of Manchester Institute of Science and Technology, U.K. Currently, he is a faculty in the School of Computer Science at the University of Oklahoma, U.S.A. Dr Atiquzzaman has over 140 publications in leading journals and conferences in the areas of communications & networking, parallel/distributed computing, and image processing. He has received funding from state and federal agencies such as National Science Foundation (NSF), National Aeronautics and Space Administration (NASA), and the United States Air Force.

Mohammed is the Editor-in-Chief of *Journal of Network and Computer Applications* Co-editor-in-chief of *Computer Communications Journal*, and serves on the editorial board of the *IEEE Communications Magazine*, *Journal of Real-Time Image Processing, International Journal on Wireless and Optical Communications*, and *International Journal of Sensor Networks*. He has chaired a number of international conferences, such as 2003 Workshop on High Performance Switching and Routing (HPSR2003), SPIE Quality of Service over Data Networks conference, IEEE Globecom and IEEE ICC. He has been involved in the program committees of many leading international conferences such as INFOCOM, Globecom, ICCCN, LCN, etc. He has been invited for seminars at various organizations and served as panelist in conferences and funding organizations. He is a senior member of the IEEE. Mohammed's email address is atiq@ou.edu, and his homepage is at www.cs.ou.edu/∼atiq.