15th Triennial World Congress, Barcelona, Spain

# ACTIVE QUEUE MANAGEMENT ALGORITHM WITH A RATE REGULATOR

**Hyuk Lim, Kyung-Joon Park, Eun-Chan Park and Chong-Ho Choi**

*School of Electrical Engineering and Computer Science*
*Seoul National University, Seoul 151-744, Korea*
*E-mail {hyuklim, kjpark, ecpark, chchoi}@csl.snu.ac.kr*

Abstract: In this paper, we propose an efficient control scheme for active queue management (AQM) supporting TCP flows. The proposed controller consists of two parts: a rate controller and a queue size controller. The rate controller is a proportional-integral (PI) controller, which improves the response to dynamic traffic variation and keeps the packet arrival rate around the link capacity. The queue size controller is a proportional (P) controller like the Random Early Detection (RED) algorithm. The rate controller gains are obtained by minimizing the performance index, either the integral square error (ISE) or the integral absolute error (IAE). We compare the performances of the proposed algorithm, the RED algorithm and the PI controller for AQM through *ns* simulations.

Keywords: Computer networks, Closed queueing networks, Queueing network models

## 1. INTRODUCTION

The congestion control has been widely studied to enhance the performance of the network. The congestion control with drop-tail queues in TCP networks has some problems. First of all, the TCP sources of drop-tail queues reduce their rates only after detecting packet losses due to a buffer overflow. Therefore a considerable time may have passed between the packet drop and its detection. At the same time, a large number of packets may have been dropped as the sources continue to transmit at a rate that the network cannot support. In addition, the packet drops at a drop-tail queue could result in the global synchronization of sources (Floyd and Jacobson 1992).

To alleviate these problems, the RED gateways were proposed for AQM (Floyd and Jacobson 1993). The RED algorithm makes network operate with high throughput and low average delay. However, the resulting average queue size is sensitive to TCP traffic load and the RED parameter settings. (Feng *et al.* 1999*a*)(Feng *et al.* 1999*b*)(Floyd *et al.* 2001).

Many variants of RED were proposed to resolve these problems (Feng *et al.* 1999*b*)(Ott *et al.* 1999)(Wang and Shin 1999)(Cnodder *et al.* 2000)(Athuraliya *et al.* 2001). Recently, several researchers have proposed control theoretic approaches (Hollot *et al.* 2001*a*)(Hollot *et al.* 2001*b*)(Aweya *et al.* 2001*a*)(Aweya *et al.* 2001*b*). In (Hollot *et al.* 2001*a*) and (Hollot *et al.* 2001*b*), the authors gave a control theoretic analysis of the RED and designed a PI controller which outperformed the RED significantly. The authors of (Aweya *et al.* 2001*a*) and (Aweya *et al.* 2001*b*) focused on stabilizing the queue size and proposed an integral controller for AQM.

In this paper, we propose an efficient control scheme for active queue management. In order to improve the response to dynamic traffic variation, we add a rate controller to a queue size controller. To obtain the control gains, we model the AQM as a first order lag plus delay (FOLPD) system, which is a simplified version of (Hollot *et al.* 2001*a*), and apply an optimization method to the rate control system. The rate control gains are obtained by minimizing the performance index, either the integral square error (ISE) or the integral absolute error (IAE). The rate control can compensate for the rate fluctuation even before the it affects the queue size. The proposed scheme with a

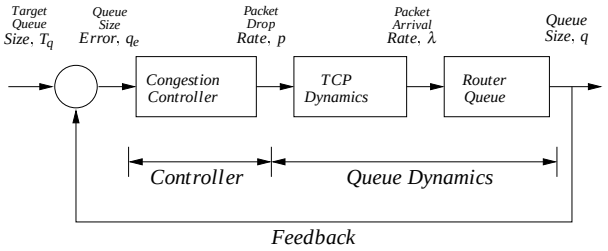



Fig. 1. TCP congestion avoidance as a closed-loop feedback control system

rate control responses quickly to dynamic traffic, and reduces the queue size variance and the packet loss rate. We compare the proposed scheme with the RED and the PI controller for AQM.

The rest of the paper is organized as follows. In Section 2, we present a linearized model for the AQM system. In Section 3, we describe the RED and the PI algorithm for AQM. In Section 4, we propose an efficient control scheme for enhancing the network performance. We compare the proposed algorithm with the RED and the PI controller via simulations using an *ns*-2 network simulator (Fall and Varadhan 2001) in Section 5. Finally, we present the conclusion in Section 6.

## 2. FOLPD MODEL

AQM algorithms control network congestion by dropping or marking packets with ECN (Explicit Congestion Notification) (Ramakrishnan and Floyd 1999). When TCP sources detect that their packets are dropped or marked with ECN, they reduce their sending rates, and the queue size of the router decreases. This process constitutes a closed loop feedback control system as shown in Fig. 1 (Aweya *et al.* 2001*a*). The system consists of TCP sources, a router queue, and a congestion controller. The congestion controller regulates the queue size of the router by changing the drop probability.

We adopt the dynamic model of TCP behavior presented in (Hollot *et al.* 2001*a*), which was developed using fluid-flow and stochastic differential equation analysis. This model relates the average value of key network variables and is described by the following coupled, nonlinear differential equations:

$$\dot{W}(t) = \frac{1}{R(t)} - \frac{W(t)W(t-R(t))}{2\,R(t-R(t))}p(t-R(t))$$
$$\gamma = \frac{W(t)}{R(t)}N(t) - C \qquad (1)$$

where

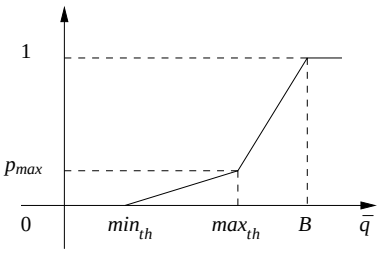



Fig. 2. The drop probability function of the *gentle* RED

$W$ = TCP window size (packets)

$\gamma$ = Queue occupation rate (packets/sec)

$R$ = round-trip time = $\frac{q}{C} + T_p$ (sec)

$C$ = link capacity (packets/sec)

$T_p$ = propagation delay (sec)

$N$ = load factor (number of TCP connections)

$p$ = probability of packet mark/drop.

If we assume $N(t) = N$ and $R(t) = R_0$, then we can linearize the dynamic model about the operating point ($W_0$, $\gamma_0$, $p_0$) as the following:

$$\delta\dot{W}(t) = -\frac{2N}{R_0^2 C}\delta W(t) - \frac{R_0 C^2}{2N^2}\delta p(t-R_0)$$
$$\delta\gamma(t) = \frac{N}{R_0}\delta W(t) \qquad (2)$$

where

$$\delta W = W - W_0,\ \delta\gamma = \gamma - \gamma_0,\ \delta p = p - p_0$$

Instead of considering the queue size as in (Hollot *et al.* 2001*b*), we consider the queue occupation rate $\gamma$. Then, the dynamic equations are written as a first-order lag plus delay (FOLPD) model. The FOLPD model is characterized by three parameters, i.e., the gain $K_g$, the time constant $T_g$ and the delay $\tau$ (Zhuang and Atherton 1993).

$$G(s) = \frac{\delta\gamma(s)}{\delta p(s)} = \frac{K_g e^{-s\tau}}{T_g s + 1} \qquad (4)$$

where

$$K_g = \frac{R_0^2 C^3}{4N^2},\ T_g = \frac{R_0^2 C}{2N},\ \tau = R_0.$$

## 3. RED AND PI CONTROLLER

The RED algorithm manages the queue size by randomly dropping packets with an increasing probability as the average queue size increases. The drop probability $p$ is a linear function of the average queue size $\bar{q}$. Fig. 2 shows a packet drop probability function of a *gentle* RED (Floyd 2001). The *gentle* RED has two slopes. One is

$$s_1 = \frac{p_{max}}{max_{th} - min_{th}} \qquad (6)$$

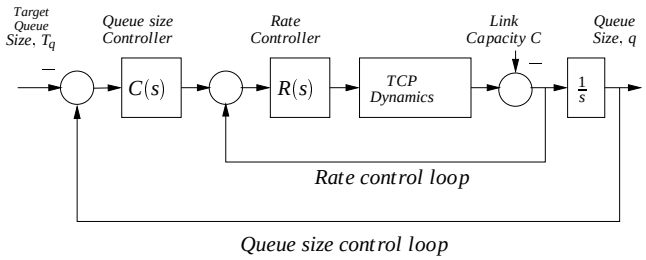



Fig. 3. The block diagram of the proposed P-PI control scheme

when the average queue size is between the minimum and the maximum thresholds [$min_{th}$, $max_{th}$], and the other is

$$s_2 = \frac{1 - p_{max}}{B - max_{th}} \tag{7}$$

when the average queue size is between the maximum threshold and the maximum buffer size [$max_{th}$, $B$]. The drop probability function can be expressed as

$$p = sat(u), \tag{8}$$

where $u$ is the congestion controller output as

$$u = \begin{cases} s_1\,(\bar{q} - min_{th}) & \bar{q} < max_{th} \\ s_2\,(\bar{q} - max_{th}) + p_{max} & \text{otherwise} \end{cases} \tag{9}$$

and $sat(\cdot)$ is a saturation function defined as

$$sat(x) = \begin{cases} 1 & x > 1 \\ 0 & x < 0 \\ x & \text{otherwise.} \end{cases} \tag{10}$$

The above equation shows that the RED is basically a proportional controller with two gains. Its input is the average queue size, and the target queue size is $min_{th}$. When the network is lightly congested, this proportional controller can keep the average queue size around $min_{th}$. However, if the network is heavily congested, this controller cannot regulate the queue size properly. This drawback of the RED algorithm will be shown in Section 5.

The PI controller consists of a proportional and an integral controller. The controller input is not the average queue size but the instantaneous queue size $q$. While the RED algorithm updates its drop probability at every packet arrival, the PI controller updates it periodically with a fixed time interval. The drop probability is written as

$$p = sat(u) \tag{11}$$

and the controller output $u$ is

$$u = k_p \left[ (q - q_{ref}) + \frac{1}{T_i} \int (q - q_{ref}) dt \right] \tag{12}$$

where $k_p$, $T_i$ and $q_{ref}$ are the proportional gain, the integral time constant and the target queue size, respectively. The proportional controller computes the drop probability based on the queue size. The proportional gain is expressed as

$$k_p = \frac{p'_{max}}{B - q_{ref}} \tag{13}$$

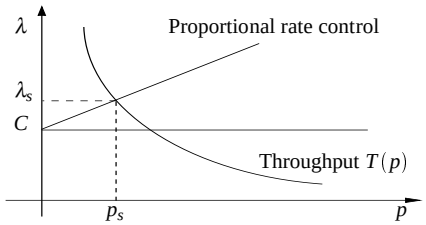



Fig. 4. Equilibrium point for a proportional rate control

where $p'_{max}$ is the drop probability when $q = B$. The probability of the proportional controller is bound by $p'_{max}$. The integral controller computes the drop probability based on traffic load. If the traffic load increases, the probability of the integral controller increases. If the traffic load decreases, the probability also decreases. The PI controller integrates the queue size error in order to make the drop probability change according to the traffic load. The integral controller in the queue size control loop can produce a large overshoot, which may cause a buffer overflow.

## 4. PROPOSED ALGORITHM

Here we propose an efficient control scheme for active queue management system. Because a change of the packet arrival rate affects the queue size directly, the response to dynamic traffic can be improved by compensating for the rate change. In order to regulate the queue size, the packet arrival rate should be also kept around the link capacity. Therefore, we propose a multiple loop control scheme, which has another inner loop for rate control as shown in Fig. 3.

We adopt a proportional-integral (PI) controller as the rate controller. The proportional rate control can achieve fast response to fluctuating traffic load, because it compensates for a rate change even before the change has effect on the queue size. However the proportional rate control $p = k_r(\lambda - C)$ cannot ensure the packet arrival rate $\lambda$ converges to the link capacity $C$.

To explain this discrepancy between the packet arrival rate and the link capacity in steady state, we consider the overall steady-state TCP behavior by a graphical method. Fig. 4 illustrates the equilibrium of the drop probability and the throughput when the proportional rate control $p = k_r(\lambda - C)$ is applied to congestion control. In Fig. 4, we assume that the steady-state throughput $T(p)$ is a strictly decreasing function of the drop probability $p$. The equilibrium point ($p_s$, $\lambda_s$) is at the intersection of the proportional control and the throughput function. We can observe from Fig. 4 that the equilibrium arrival rate $\lambda_s$ is deviated from the link capacity $C$, and consequently there always exists a rate error in equilibrium. Therefore, instead of the proportional control, we use the proportional-integral control to make the rate error become zero in equilibrium.

The transfer function of the PI rate controller $R(s)$ is

$$R(s) = k_r \left(1 + \frac{1}{T_r s}\right) \tag{14}$$

where $k_r$ and $T_r$ are the proportional gain and the integral time constant of the rate controller, respectively. From the FOLPD model in (4) and the above PI controller, the rate control system has the following transfer function:

$$P(s) = \frac{k_r(T_r s+1)K_g e^{-s\tau}}{T_r s(T_g s+1) + k_r(T_r s+1)K_g e^{-s\tau}} \tag{15}$$

To optimize the PI controller gains, $k_r$ and $T_r$, we consider two popular performance indexes as follows:

$$ISE(\theta) = \int_0^\infty e(\theta,t)^2 dt \tag{16}$$

$$IAE(\theta) = \int_0^\infty |e(\theta,t)| dt \tag{17}$$

where $\theta$ denotes the gain vector which can be chosen to minimize the performance indexes (O'Dwyer 2000). To obtain the controller gains, an optimization is carried out using the third order Padé approximation for the delay (Franklin *et al.* 1995). The formulas for $k_r$ and $T_r$ are

$$k_r = \frac{a_1}{K_g}\left(\frac{\tau}{T_g}\right)^{b_1}$$
$$T_r = \frac{T_g}{a_2 + b_2\left(\frac{\tau}{T_g}\right)} \tag{18}$$

and the optimal values for $a_i$ and $b_i$ are listed in Table 1 (O'Dwyer 2000).

The queue size controller regulates the buffer properly. Because the rate controller produces the load dependent drop probability, a proportional controller is used as the queue size controller $C(s)$, i.e.,

$$C(s) = k_q \tag{19}$$

where $k_q$ is the proportional gain of the queue size controller. The proportional gain should be selected to guarantee the system stability. The open loop transfer function of the AQM system can be written as

$$Q(s) = k_r \left(1 + \frac{1}{T_r s}\right) \frac{K_g e^{-s\tau}}{T_g s+1} \frac{s+k_q}{s} \tag{20}$$

To get a proper range of $k_q$, we use the frequency-response design method (Franklin *et al.* 1995). We consider the first three terms as a plant and the last term $\frac{s+k_q}{s}$ as an unit gain proportional integral controller, which has the infinite gain at zero frequency and a phase decrease below the break point at $\omega = k_q$. Because the phase margin of a system means the system's tolerance to time delay, the congestion control system should have a sufficient phase margin. Therefore, $k_q$ should be set as a frequency substantially less than the crossover frequency so that the system's phase margin is not affected very much (Franklin *et al.* 1995).

Table 1. PI tuning formulas

| Criterion | ISE | IAE |
|---|---|---|
| $a_1$ | 0.980 | 0.758 |
| $b_1$ | -0.892 | -0.861 |
| $a_2$ | 0.690 | 1.020 |
| $b_2$ | -0.155 | -0.232 |

## 5. SIMULATION

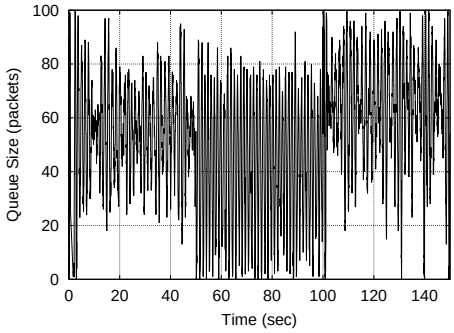
To compare the performance of the proposed scheme with other AQM algorithms, we conducted simulations using the *ns*-2 network simulator. A simple bottleneck network configuration was implemented with two routers and a number of TCP connections as in (Aweya *et al.* 2001*a*). The routers are connected through a link of capacity 10Mbps. We used the TCP-Reno as the default transport protocol and assumed that the average packet size is 1000 bytes. Each TCP connection has a propagation delay between 50 ms and 150 ms. The target queue size at the bottleneck link is set to be 50 packets. In the first simulation, we compared the responsiveness of the AQM schemes to dynamic traffic. In the second simulation, we compared the performance criteria such as the average queue size, the standard deviation (std) of the queue size, and the packet loss rate for the different number of TCP connections.

### 5.1 *Simulation I*

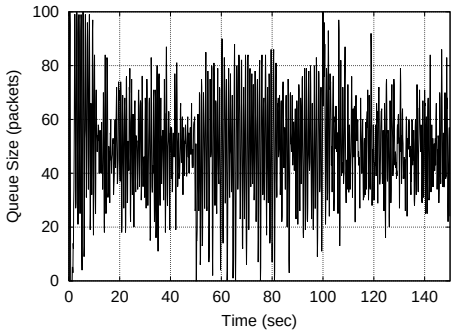
In this simulation, we compared the response of the AQM algorithms when some of the TCP connections were off and then on after some period. Initially at t = 0 s, the number of TCP connections was 50, and then 25 TCP connections were dropped at t = 50 s. At t = 100 s, another 50 TCP connections were established. We compared four algorithms: the RED, the PI, the proposed P-PI with ISE, and the proposed P-PI with IAE. As we can see in Fig. 5(a), the queue size of the RED changes according to the traffic load and fluctuates severely. This fluctuation can cause problems such as a large queueing delay, an under-utilization, and a buffer overflow. Fig. 5 shows that the queue sizes of the proposed P-PI schemes with the gains obtained from the ISE or the IAE remain around 50, and fluctuate less than those of the RED and the PI controller.

### 5.2 *Simulation II*

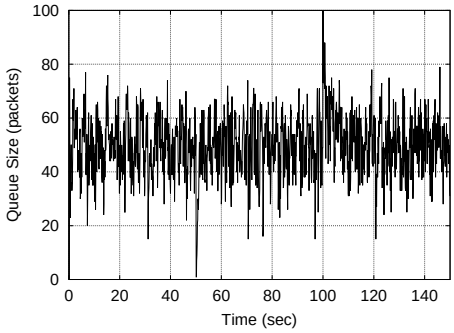
Here we conducted simulations to compare the performances such as the average queue size, the standard deviation of the queue size and the packet loss rate for the different number of the TCP connections. Fig. 6 shows the simulation results. In Fig. 6(a), we can see that the average queue size of the RED becomes larger as the TCP connections increase, while the other



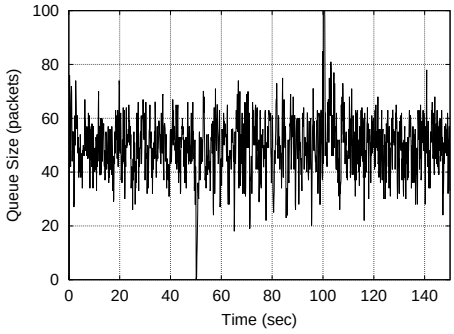

(a) RED




(b) PI control

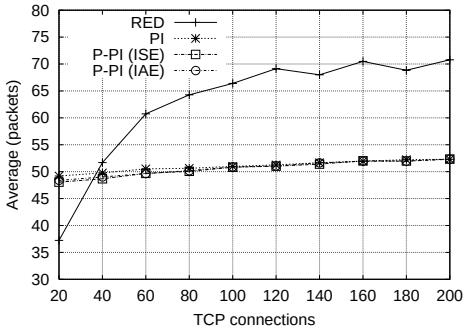



(c) The proposed P-PI control with ISE

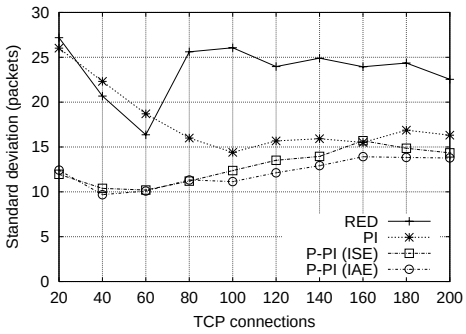



(d) The proposed P-PI control with IAE

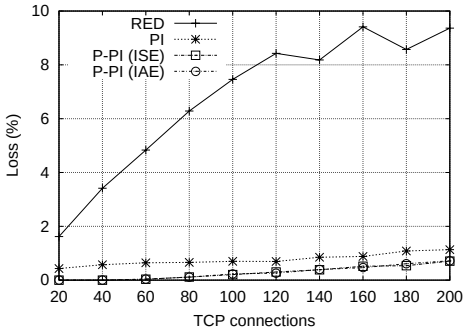
Fig. 5. Simulation I




(a) Queue size average




(b) Queue size standard deviation




(c) Packet loss rate

Fig. 6. Simulation II

algorithms keep their average queue size around the target queue size, 50 packets. Fig. 6(b) shows that the standard deviations of the P-PI with either ISE or IAE are smaller than those of the RED and the PI in almost all cases. This can be also explained by Fig. 5. The queue size of the PI controller fluctuates much more than the P-PI schemes. We can see that the schemes based on control theoretic approach give better performance than the RED in Fig. 6. Except the RED, all the three algorithms show very small packet loss rates. This is mainly because the three algorithms maintain the queue size around 50 packets and prevent a buffer overflow. Meanwhile, the RED fails to prevent a buffer overflow and this results in a large packet loss rate. Furthermore as we can see from Fig. 6(c), the loss rates of two proposed P-PI controllers are much smaller than that of the PI algorithm. Actually until the number of the TCP connections becomes 60, there is little loss under the proposed schemes.

## 6. CONCLUSION

In this paper, we proposed an efficient control scheme for active queue management to achieve fast response to dynamic traffic. We modeled the AQM as a first order lag plus delay (FOLPD) system based on the dynamic equations which were developed in (Hollot *et al.* 2001*b*). Using this FOLPD model, we designed a proportional-integral rate controller, and obtained the control gains minimizing the performance indexes. By adding a rate control loop to AQM system, we can regulate the queue size, reduce the variance of the queue size, and reduce the packet loss rate. The

simulation results show that the performance of the proposed P-PI schemes are much better than those of the RED and the conventional PI algorithm.

## 7. REFERENCES


Athuraliya, S., S. H. Low, V. H. Li and Q. Yin (2001). REM: Active queue management. *IEEE Network* **15**, 48–53.

Aweya, J., M. Ouellette and D. Y. Montuno (2001*a*). A control theoretic approach to active queue management. *Computer Networks* **36**, 203–235.

Aweya, J., M. Ouellette and D. Y. Montuno (2001*b*). An optimization-oriented view of random early detection. *Computer Communications* **24**, 1170–1187.

Cnodder, S. D., O. Elloumi and K. Pauwels (2000). RED behavior with different packet sizes. In: *Proceedings of ISCC*. pp. 793–799.

Fall, J. and K. Varadhan (2001). The ns manual. http://www.isi.edu/nsnam/ns/ns-documentation.

Feng, W., D. Kandlur, D. Saha and K. Shin (1999*a*). Blue: A new class of active queue management algorithms. *Tech. Rep., UM CSE-TR-387-99*.

Feng, W., D. Kandlur, D. Saha and K. Shin (1999*b*). A self configuring RED gateway. In: *Proceedings of IEEE INFOCOM*. Vol. 3. pp. 1320–1328.

Floyd, S. (2001). RED queue management. http://www.aciri.org/floyd/red.html.

Floyd, S. and V. Jacobson (1992). On traffic phase effects in packet switched gateways. *Internetworking: Research and Experience* **3**, 115–156.

Floyd, S. and V. Jacobson (1993). Random early detection gateways for congestion avoidance. *IEEE/ACM Trans. on Networking* **1**, 397–413.

Floyd, S., R. Gummadi and S. Shenker (2001). Adaptive RED: an algorithm for increasing the robustness of RED's active queue management. *submitted for publication*.

Franklin, G. F., Powell, J. D. and Emami-Naeini, A., Eds.) (1995). *Feedback control of dynamic systems*. Addsion-Wesley.

Hollot, C. V., V. Misra, D. Towsley and W. Gong (2001*a*). A control theoretic analysis of RED. In: *Proceedings of IEEE INFOCOM*. Vol. 3. pp. 1510–1519.

Hollot, C. V., V. Misra, D. Towsley and W. Gong (2001*b*). On designing improved controllers for AQM routers supporting TCP flows. In: *Proceedings of IEEE INFOCOM*. Vol. 3. pp. 1726–1734.

O'Dwyer, A. (2000). PI and PID controller tuning rules for time delay process: a summary. *Tech. Rep., AOD-00-01, Edition 1*.

Ott, T. J., T. V. Lakshman and L. H. Wong (1999). SRED: Stabilized RED. In: *Proceedings of IEEE INFOCOM*. Vol. 3. pp. 1346–1355.

Ramakrishnan, K.K. and S. Floyd (1999). Aproposal to add explicit congestion notification (ECN) to IP. RFC 2481.

Wang, H. and K. Shin (1999). Refined design of random early detection gateways. In: *Proceedings of GLOBECOM*. Vol. 1b. pp. 769–775.

Zhuang, M. and D.P. Atherton (1993). Automatic tuning of optimum PID controllers. **140**, 216–224.