An algorithm for adapting RED parameters to TCP traffic

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the ICC 2008 proceedings.

An Algorithm for Adapting RED Parameters to TCP Traffic

Wu CHEN, Shuang-Hua YANG
Department of Computer Science
Loughborough University, Loughborough, LE11 3TU, UK
Emails: {w.chen2, s.h.yang}@lboro.ac.uk

Abstract— **Random Early Detection (RED) can stabilize the queue within a given target range and simultaneously achieve high throughput in the routers. However, the average queue length is quite sensitive to the network scenarios and it is difficult to adapt RED parameters to the changing network traffic. This paper develops an algorithm for systematically adapting RED parameters to variable network conditions such as link capacity, round-trip time and the number of TCP flows. Simulations demonstrate that this algorithm can stabilize the queue length within a target range and maintain high link utilization in a wide variety of network traffic conditions.**

Keywords-RED; Parameter tuning; Congestion control

I. INTRODUCTION

Congestion control plays an important role in the Internet to ensure good network performance. Internet congestion control comprises two main parts: TCP and Active Queue Management (AQM). Although many AQM algorithms have been proposed over the last decade, only RED has been implemented in major commercial routers. One of the main goals of RED is to stabilize the queue length at a given target and maintain high link utilization [1]. If RED is successfully deployed in all the edge routers, i.e. achieves the goal of high link utilization and low loss rate, it is unnecessary to deploy RED in the core routers as well. However, the average queue length varies with traffic loads as well as round-trip times. Parameterizing RED to obtain good performance under variable congestion scenarios is very difficult. Such difficulties discourage network administrators from turning on RED in their routers [2].Certainly, there are many parameter tuning techniques for RED proposed in the literature [3]-[6], but they were either developed on the basis of empirical investigations and analysis, or only applicable under certain assumptions. In particular, Adaptive RED (ARED) [3] provided a simple guideline for adjusting RED parameters in response to the traffic load, but it was only applicable under a narrow range of round-trip times and link capacity. As demonstrated in other AQM or AQM-based approaches, such as Random Exponential Marking (REM) [7], the Proportional Integrator (PI) controller [8], and BLUE [9], parameter setting still remained a critical unsolved problem.

In [6] the authors proposed a method to set initial RED parameters from a Control Theoretic standpoint. However, they did not illustrate how to adapt the RED parameters to changing network scenarios. By improving their Control Theoretic analysis on TCP/RED system, we developed a simple, scalable and systematic rule of parameter tuning as a function of network traffic conditions such as link capacity, round-trip time, and the number of TCP flows, with the key RED parameters decoupled from other RED parameters. Theoretical analysis and ns-2 simulations demonstrate that it is robust, adaptive to TCP dynamics, and produces desirable transient performance.

The rest of the paper is organized as follows. In Section II, we illustrate the pre-developed model of RED. In Section III, we develop our parameter tuning algorithm and stability analysis. In Section IV, we introduce the calculation method for network scenario parameters. We next present simulation results obtained by using ns-2 that verifies our analysis in Section V. Finally, we present our conclusions in Section VI.

II. MODEL

According to the fluid-flow and stochastic differential equation analysis of TCP behavior [10], a linear feedback control model of RED was presented in [6]. The linear control model is shown in Fig. 1.

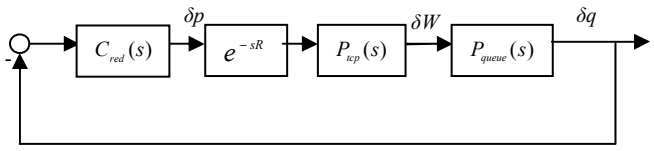


Fig. 1. Feedback control model of RED

In the model $C_{red}(s)$ denotes RED control strategy, $P_{tcp}(s)$ denotes the linearized TCP dynamics, $P_{queue}(s)$ denotes the queue dynamics, and e^{-sR} denotes the delay term. They were given in the following equations:

$$\begin{cases} P_{tcp}(s) = \dfrac{\dfrac{RC^2}{2N^2}}{s + \dfrac{2N}{R^2C}} \\ P_{queue}(s) = \dfrac{\dfrac{N}{R}}{s + \dfrac{1}{R}} \\ C_{red}(s) = \dfrac{L_{red}}{s/K + 1} \end{cases} \quad (1)$$

where

R = Round-trip time (secs);

N = Number of TCP flows;

C = Link capacity (packets/sec);

$\delta W = W - W_0$;

$\delta p = p - p_0$;

$\delta q = q - q_0$;

W = expected TCP window size (packets);

p = probability of packet mark/drop;

q = expected queue length (packets);

(W_0, q_0, p_0) = the equilibrium point defined by $\dot{W} = 0$ and $\dot{q} = 0$;

$L_{red} = \dfrac{p_{\max}}{\max_{th} - \min_{th}}$;

$K = -\dfrac{\log_e(1-\alpha)}{T_s}$;

$p_{\max}$ = Maximum drop probability;

$\max_{th}$ = Maximum threshold;

$\min_{th}$ = Minimum threshold;

α = Average queue weight;

T_s = Sampling interval.

III. PARAMETER TUNING ALGORITHM AND STABILITY ANALYSIS

A. Algorithm

The objective of our algorithm is to maintain the average queue size within target $\max_{th}$ and $\min_{th}$, and thus to provide high link utilization in widely varying network scenarios. The rationale behind setting buffer size (B), maximum threshold ($\max_{th}$) and minimum threshold ($\min_{th}$) as a function of bandwidth-delay product is to follow the rule-of-thumb in accommodating bursty traffic [11], the rational behind setting the maximum drop probability is to achieve a desirable equilibrium point based on theoretical analysis, and the rationale behind setting average queue weight is to maintain the stability of the system from a Control Theoretic standpoint.

Thus, for an initial network scenario of TCP load N_0, round-trip time R_0 and link capacity C_0 with RED parameters α_0, $p_{\max 0}$, $\max_{th0}$, $\min_{th0}$, and buffer size B_0 that stabilize the queue, we have the following algorithm for setting RED parameters when the network scenario varies.

$$B = k_r k_c B_0 \quad (2)$$

$$\max_{th} = k_r k_c \max_{th0} \quad (3)$$

$$\min_{th} = k_r k_c \min_{th0} \quad (4)$$

$$p_{\max} = \left(\frac{k_n}{k_r k_c}\right)^2 p_{\max 0} \quad (5)$$

$$\alpha = \begin{cases} \dfrac{k_n}{(k_r k_c)^2}\alpha_0 & N \le RC/2 \\ \dfrac{1}{k_r k_c}\alpha_0 & N > RC/2 \end{cases} \quad (6)$$

where

$k_r = R / R_0$

$k_c = C / C_0$

$k_n = N / N_0$

$R_0 C_0 \le B_0 \le 2R_0 C_0$

and constraint $p_{\max}$ within the range [0.01,0.5]

i.e. if $p_{\max} > 0.5$ $\quad p_{\max} = 0.5$

if $p_{\max} < 0.01$ $\quad p_{\max} = 0.01$

Compared with the automatic setting method of queue weight α based on link capacity in [3], our tuning algorithm of α is based not only on link capacity, but also on round-trip time and traffic load. A larger α can improve transient response while a smaller α provides a sufficient stability margin. On the other hand, when the buffer size is reduced according to the changing network parameters, the queuing delay is decreased correspondingly. In contrast a larger buffer size can still stabilize the queue within a given target range.

B. Adapting buffer size

The purpose of setting a buffer size is to adapt the buffer size to 1-2 times the bandwidth-delay product based on the experiences of [11].

Supposing (2) we have

$$RC = k_r k_c R_0 C_0 \le B \le 2k_r k_c R_0 C_0 = 2RC$$

Then we always have the buffer size 1-2 times the bandwidth-delay product. At the same time we adjust $\max_{th}$ and $\min_{th}$ in proportion to B and we get equations (3) and (4).

Essentially, the appropriate buffer size is nonlinear with the bandwidth-delay product and traffic loads. Our algorithm can guarantee that RED works well in a wide range of scenarios.

C. Determining the Equilibrium Point

The purpose of setting $p_{\max}$ is to keep the queue at a desirable equilibrium point based on the dynamic model of TCP behavior. At a new equilibrium point where $\dot{W} = 0$ and $\dot{q} = 0$, from [6] we have

$$W^2 p = 2 \text{ and } W = \frac{RC}{N}$$

Then we obtain

$$p = \frac{q - \min_{th}}{\max_{th} - \min_{th}} p_{\max} = 2(\frac{N}{RC})^2 \quad (7)$$

Similarly, at the initial equilibrium point q_0 we obtain

$$p_0 = \frac{q_0 - \min_{th0}}{\max_{th0} - \min_{th0}} p_{\max 0} = 2(\frac{N_0}{R_0 C_0})^2 \quad (8)$$

Supposing we tune $p_{\max}$ in terms of equation (5), from (7) and (8) we have

$$q = k_r k_c q_0$$

So the equilibrium point q moves proportionally to the initial point q_0 and still stays between $\max_{th}$ and $\min_{th}$ when network conditions change.

D. Stability Analysis

1) Stability Proposition

According to [6] we derive a simplified version of stability proposition in this section.

Stability Proposition: Let L_{red} and α satisfy:

$$L_{red}\alpha \le \begin{cases} \frac{0.8N^3}{(RC)^5}, & N \le RC/2 \\ \frac{0.4N^2}{(RC)^4}, & N > RC/2 \end{cases} \quad (9)$$

where

$$\alpha << 1$$

Then the linear feedback control system shown in Fig. 1 is stable.

Proof:

Consider the frequency response of the compensated loop transfer function

$$L(j\omega) = C_{red}(j\omega)P(j\omega)e^{-j\omega R} = \frac{L_{red}\frac{(RC)^3}{(2N)^2}e^{-j\omega R}}{(\frac{j\omega}{K}+1)(\frac{j\omega}{\frac{2N}{R^2C}}+1)(\frac{j\omega}{\frac{1}{R}}+1)}$$

$$\approx \frac{L_{red}\frac{(RC)^3}{(2N)^2}e^{-j\omega R}}{\frac{j\omega}{K}+1} \quad \forall \omega \in [0, \omega_g] \quad (10)$$

where

$$\omega_g = 0.1\min\left\{\frac{2N}{R^2C}, \frac{1}{R}\right\} \quad (11)$$

Thus we obtain

$$\left|L(j\omega_g)\right| \approx \frac{L_{red}\frac{(RC)^3}{(2N)^2}}{\sqrt{\frac{\omega_g^2}{K^2}+1}} < \frac{L_{red}\frac{(RC)^3}{(2N)^2}}{\frac{\omega_g}{K}} \le 1 \quad (12)$$

where

$$\frac{L_{red}(RC)^3}{(2N)^2} \le \frac{\omega_g}{K} \quad (13)$$

We again use (10) to obtain

$$\angle L(j\omega_g) \approx \angle \frac{L_{red}\frac{(RC)^3}{(2N)^2}}{\frac{j\omega_g}{K}+1} - \omega_g R \ge -90^0 - 0.1\frac{180^0}{\pi} > -180^0$$

This and (12) indicate that the closed-loop system is asymptotically stable according to the Nyquist stability criterion.

Given that $\alpha << 1$, we have

$$Log_e(1-\alpha) \approx -\alpha$$

For a stable congested queue we have

$$T_s = \frac{1}{C}$$

Then from (1) we have

$$K = \alpha C$$

Consequently, given any $N \le RC/2$, from (13) we have

$$\frac{L_{red}(RC)^3}{(2N)^2} \le \frac{\frac{0.2N}{R^2C}}{\alpha C}$$

Then

$$L_{red}\alpha \le \frac{0.8N^3}{(RC)^5}$$

Similarly, given any $N > RC/2$, we obtain

$$L_{red}\alpha \le \frac{0.4N^2}{(RC)^4}$$

We can see that the derivation of our stability proposition is similar to that proposed in [6]. We have an intuitive sense that our algorithm provides a similar performance to that in [6]. However, the RED parameters in [6] are tightly coupled with network parameters and a complex calculation is needed to obtain suitable RED parameters for every changed network scenario. In contrast RED parameter tuning is much easier in our proposition as shown in equations (2)-(6).

2) Stability analysis

The purpose of setting α is to stabilize the queue and improve transient response based on Control Theory.

For the initial RED parameters α_0, $p_{\max 0}$, $\max_{th0}$, $\min_{th0}$ that satisfy (9), given $N_0 > R_0C_0/2$, we have

$$\frac{p_{\max 0}}{\max_{th0}-\min_{th0}}\alpha_0 < \frac{0.4N_0^2}{(R_0C_0)^4} < \frac{0.8N_0^3}{(R_0C_0)^5}$$

and given $N_0 \le R_0C_0/2$, we still have

$$\frac{p_{\max 0}}{\max_{th0}-\min_{th0}}\alpha_0 < \frac{0.8N_0^3}{(R_0C_0)^5}$$

Supposing (3)-(6) we have

$$L_{red}\alpha = \frac{p_{\max}}{\max_{th}-\min_{th}}\alpha = \frac{k_n^3}{(k_rk_c)^5}\frac{p_{\max 0}}{\max_{th0}-\min_{th0}}\alpha_0$$

$$< \frac{k_n^3}{(k_rk_c)^5}\frac{0.8N_0^3}{(R_0C_0)^5} = \frac{0.8N^3}{(RC)^5}$$

where

$$N \le RC/2$$

Similarly we have

$$L_{red}\alpha < \frac{0.4N^2}{(RC)^4}$$

where

$$N > RC/2$$

So the stability proposition in (9) can always be satisfied when we tune RED parameters according to the equations (3)-(6). In addition a larger value of α can improve transient response [6], in contrast with a smaller value of α just for the purpose of stabilizing the queue. This provides a useful guideline for tuning the RED parameters whilst maintaining system stability.

IV. Network Traffic Measurement

The online statistic characteristics of heterogeneous network traffic used in the above algorithm include round-trip time (R), number of TCP flows (N), and link capacity (C) in a router. They are measurable and do not change dynamically based on the packet-level traces and flow-level statistics. For example the characteristics of R, N, C and average package size etc have been observed in OC3MON [12] and the IPMON system [13]. It is feasible for us to use the method described in these papers to obtain the network parameters.

In [13] the number of flows is calculated per minute. Those packets with the same 5-tuple information (source address, source port, destination address, destination port, and protocol) are classified as the same flow. The start time of a flow is the first time when a packet with new 5-tuple information is observed, and the flow ends when no packets with the same 5-tuple information are seen for 60s. On the other hand we can use a shorter timeout and a shorter time interval to calculate the traffic loads. Although there is some difference for the calculated value of traffic loads with different time intervals, in terms of [14] the variation of proportionality factor k_n is small with different time intervals.

In [13] round-trip time is measured as the time elapsed between a SYN packet and the first ACK packet that completes the three-way handshake. We only compute the round-trip time for flows of which we observe the SYN/ACK pair and consider the round-trip time of a flow for only one direction. The impact of these limitations on the value of proportionality factor k_r needs further investigation in practical implementation.

Moreover the link capacity C in bytes is a known value, so it is unnecessary to calculate it in most cases when our algorithm is run in byte mode.

The measurement algorithm can be deployed in routers. For example Cisco NetFlow is configured on most Cisco routers to provide a highly condensed and detailed view of all real time network traffic. Because the statistic characteristics of network parameters are not changing dynamically we do not need to predict the network scenario parameters R, N and C in advance and could measure these parameters less frequently. In practice we need to set a suitable time interval to obtain these network parameters on a trial-and-error basis. Because it is easy to obtain these network parameters in edge routers, the feasibility of our RED tuning algorithm is ensured, especially in edge routers.

V. Simulations

We use the ns-2 simulator with a single bottleneck link in a dumbbell topology to verify our algorithm. In all experiments we look at a queue with FTP (greedy) flows, as well as HTTP sessions for introducing noise to the queue. The number of HTTP sessions is twice that of FTP flows in all experiments and the average packet size is 500 bytes. We use our algorithm to design RED parameters with changed scenarios, and then use ARED for comparison.

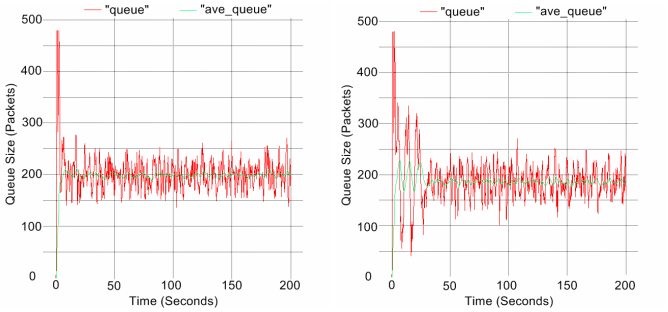
In the first experiment the initial RED parameters are derived from the stability proposition (9) as shown in Table 1. Correspondingly we set ARED parameters $\max_{th}$ = 320, $\min_{th}$ =80, and α=8e-5 the same as the initial value of our algorithm. We plot the queue size of our RED algorithm and ARED in Fig. 2(a) and Fig. 2(b) respectively. We can see that our algorithm provides a faster response time than ARED because it is unnecessary to adapt its p_{max} step by step.

In the following experiments we use the proposed algorithm to tune the RED parameters from the initial parameters used in experiment 1 in response to the changes in network conditions N, C and R as shown in Table 1, while keeping ARED parameters unchanged. In experiment 2 the link capacity is low and the traffic load is reduced. Because RED can keep stable under a lower link capacity according to (9) and ARED is good at dealing with changed traffic load, the queue size is expected to keep stable with ARED. In Fig. 3(b) we indeed see that the queue is stabilized within its target using ARED. In contrast we can use a smaller buffer size in our algorithm in response to the lower link capacity and thus lower queuing delay can be achieved. We can see both a lower queuing delay and a stable queue size in Fig. 3(a). When a larger queue weight is used in our algorithm we see a better transient response in Fig. 3(a). In experiment 3 ARED is unable to keep stable because of the increase in link capacity and round trip time and we can see definite oscillation in Fig.

4(b). On the other hand, because the corresponding RED parameters are tuned according to our algorithm in response to the changed network scenarios, the average queue size converges between min_{th} and max_{th}, and the instantaneous queue size remains greater than zero and smaller than the total buffer size as desired in Fig. 4(a).

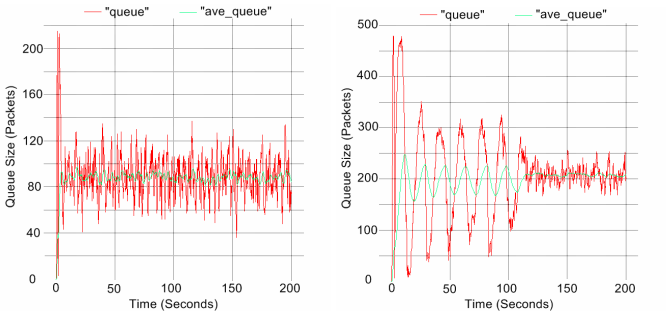
Table 1: Dynamic RED parameters of our RED algorithm

No	N	C	R	B	max_{th}	min_{th}	p_{max}	α
1	100	2500	200	480	320	80	18	8e-5
2	50	1250	180	216	144	36	14.4	2e-4
3	200	5000	300	1440	960	240	40	1.9e-5



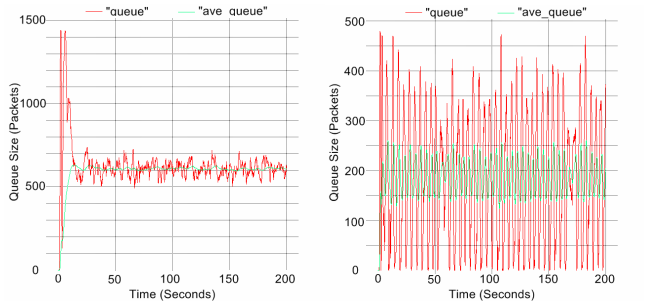
a) Our RED algorithm b) ARED

Fig. 2 Queue size for experiment 1



a) Our RED algorithm b) ARED

Fig. 3. Queue size for experiment 2



a) Our RED algorithm b) ARED

Fig. 4. Queue size for experiment 3

VI. CONCLUSIONS

In this paper we have provided a simple, scalable and systematic algorithm based on Control Theory. This algorithm explains how to adapt RED parameters to the changing network conditions from the initial RED parameters that stabilize the queue. We are investigating its performance in multi-service situations involving networks with multiple bottlenecks and connections.

REFERENCES

[1] S. Floyd and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance," IEEE/ACM Transactions on Networking, vol. 1, No. 4, pp. 397–413, August 1993.

[2] M. May, J. Bolot, C. Diot and B. Lyles, "Reasons not to deploy RED," in Proceedings of 7th International Workshop on Quality of Service, London, UK, pp. 260-262, 31 May-4 June 1999.

[3] S. Floyd, R. Gummadi and S. Shenker, "Adaptive RED: an algorithm for increasing the robustness of RED's active queue management," [Online] Available: http://www.icir.org/floyd/papers/adaptiveRed.pdf.

[4] H. Sirisena, A. Haider and K. Pawlikowski. "Auto-Tuning RED for Accurate Queue Control," in Proceedings of IEEE GLOBECOM, Taipei, Taiwan, November 2002.

[5] T. Ziegler, S. Fdida, and C. Brandauer, "Stability Criteria for RED with Bulk-data TCP Traffic," 2001. Technical Report, August 1999. http://www-rp.lip6.fr/publications/production.html.

[6] C. Hollot, V. Misra, D. Towlsey, and W. Gong, "A Control Theoretic Analysis of RED," in Proceedings of IEEE INFOCOM, Anchorage, USA, April 2001.

[7] S. Athuraliya, V. H. Li, S. H. Low, and Q. Yin, "REM: Active Queue Management," IEEE Network, vol. 15, No. 3, pp 48-53, May 2001.

[8] C. Hollot, V. Misra, D. Towlsey, and W. Gong, "Analysis and design of controllers for AQM routers supporting TCP Flows," IEEE Transactions on Automatic Control, vol. 47, No. 6, pp 945-959, June 2002.

[9] Wu-chang Feng, Kang G. Shin, Dilip D. Kandlur, Debanjan Saha "BLUE Active Queue Management Algorithms," IEEE/ACM Transactions on Networking, vol. 10, No. 4, pp. 513-528, August 2002.

[10] Vishal Misra, Wei-Bo Gong, and Don Towsley, "Fluid-based Analysis of a Network of AQM Routers Supporting TCP Flows with an Application to RED," in Proceedings of IEEE ACM/SIGCOMM, Stockholm, Sweden, 28 August-1 September 2000.

[11] C. Villamizar and C. Song, "High Performance TCP in ANSNET," ACM Computer Communications Review, vol 24, No. 5, pp 45-60, October 1994.

[12] K. Thompson, G. Miller, and R. Wilder, "Wide Area Internet Traffic Patterns and Characteristics," IEEE Network, vol. 11, No. 6, pp. 10-23, November-December 1997.

[13] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seely, and C. Diot, "Packet-Level Traffic Measurements from the Sprint IP Backbone," IEEE Network, vol. 17, No. 6, pp. 6-16, November-December 2003.

[14] G. Iannaccone, C. Diot, I. Graham, N. Mckeown, "Monitoring Very High Speed Links," in Proceedings of IEEE ACM/SIGCOMM, San Diego, USA, August 2001.