Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2010, Article ID 329035, 14 pages
doi:10.1155/2010/329035

Research Article

An Improved ARED Algorithm for Congestion Control of Network Transmission

Jianyong Chen, Cunying Hu, and Zhen Ji

Shenzhen City Key Laboratory of Embedded System Design, College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China

Correspondence should be addressed to Jianyong Chen, cjyok2000@hotmail.com

Received 19 December 2009; Accepted 8 February 2010

Academic Editor: Massimo Scalia

Copyright © 2010 Jianyong Chen et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In order to achieve high throughput and low average delay in computer network, it is necessary to stabilize the queue length and avoid oscillation or chaos phenomenon. In this paper, based on Adaptive Random Early Detection (ARED), an improved algorithm is proposed, which dynamically changes the range of maximum drop probability $p_{\max}$ according to different network scenarios and adjusts $p_{\max}$ to limit average queue size q_{ave} in a steady range. Moreover, exponential averaging weight w is adjusted based on linear stability condition to stabilize q_{ave}. A number of simulations show that the improved ARED algorithm can effectively stabilize the queue length and perform better than other algorithms in terms of stability and chaos control.

1. Introduction

In the current internet, congestion control has been a serious problem. Lack of effective management of congestion significantly affects the performance of internet, such as degradation of link utilization, more round-trip time, and even makes the network inaccessible. At the same time, various approaches have been proposed to solve the issue. These schemes can be divided into two categories. One category strengthens congestion management at the end while making few changes to internet [1, 2]. The other one aims to adapt Active Queue Management (AQM) at each router, and eventually control congestion level, such as Random Early Detection (RED) [3], Random Early Marking (REM) [4] and Virtual Queue (VQ) [5, 6]. Among them, RED is the most prominent and well-studied AQM scheme. Though RED can prevent global synchronization, reduce packet loss, and achieve high throughput, it induces oscillation or chaos to degrade the performance of the system in some network scenarios. The stability of RED gains more and more researchers' attentions.

The main goal of AQM is to keep queue length stable in order to make a good tradeoff between high throughput and low delay. As a queue management policy, RED gateway manages queue by monitoring the average queue size q_{ave}. When q_{ave} exceeds the preset lower threshold $q_{\min}$, arriving packets are randomly dropped or marked with a certain probability, so that some connections can sense the early congestion, and then adjust their window sizes to avoid serious congestion and packet lost. Once the average queue size is larger than the upper threshold $q_{\max}$, the RED gateway drops/marks every arriving packet to keep q_{ave} below $q_{\max}$. Although the RED algorithm is an effective mechanism to achieve high throughput and low delay, some researches note that RED is quite sensitive to traffic load and RED parameters [7–9], and that the average queue length exhibits nonlinear instability and chaos [10–13]. In particular, when the average queue becomes larger than the upper threshold $q_{\max}$, q_{ave} fluctuates seriously, leading to low throughput and large packet loss rate. Many modified RED algorithms are proposed to eliminate oscillations or chaos to improve its robustness by revising the drop probability function or by adjusting control parameters.

Among these algorithms, one set of approaches are to modify the packet drop probability function to improve the stability of RED and its performances [14–18]. Particularly, Hollot et al. proposed a PI-controller (Proportional-Integral controller) to improve reaction of TCP-RED dynamics [15]. Besides that, the authors proposed LRED (Loss Ratio-based RED) in [18], which takes into account the dropping probability based on loss rate and queue length rather than average queue size. These modified algorithms outperform original RED in terms of better stability and higher throughput. However, these methods always induce additional parameters that are needed to be optimized to stabilize the queue length in different network scenarios. The setting of parameters is still an unsolved problem. Another set of algorithms does not change the basic idea of RED, but tunes the control parameters to improve the stability of the queue length [8, 9, 12, 13, 19–23]. Feng et al. have argued that there is no single set of RED parameters that can work well under different congestion scenarios. In [8], they proposed original ARED, which tunes the parameter $p_{\max}$ based on network traffic to keep the average queue size q_{ave} between the upper threshold $q_{\max}$ and lower threshold $q_{\min}$. Floyd et al. present a revised version, also called Adaptive RED [9], which adopts AIMD mechanism to change $p_{\max}$ slowly. Although ARED can stabilize the queue length at a given target and achieve high throughput in some networks, ARED does not provide any systematic methods to set parameters and the target. Furthermore, the control parameter selection is based only on empirical observation and simulation analysis. As a result, it performs well in one simulation, but fails in another. Besides that, some researches propose other variants of RED, such as SARED (Stabilized ARED) [19], PD-RED [20], while others discuss the bound of $p_{\max}$ [21–23]. These algorithms keep the queue length stable in some network scenarios by dynamically tuning parameters, but they fail to control the fluctuation and chaos in others.

In the paper, we analyze the steady range of q_{ave} and its linear stability condition, and propose a novel improved ARED algorithm. The improved ARED constrains $p_{\max}$ in a more reasonable range according to both the steady range and the target range of ARED. What is more, the improved ARED also decreases w to stabilize q_{ave} based on linear stability condition. By stabilizing q_{ave} within target range, the improved ARED algorithm could get high link utilization, more stable delay with less delay jitter. The rest of the paper is organized as follows. In Section 2, RED algorithm and a discrete-time TCP-RED model are introduced. Then we analyze linear stability condition of the average queue size. In Section 3, an improved ARED algorithm is proposed based on ARED algorithm, and the settings of the control parameters $p_{\max}$ and w are discussed in detailed. Section 4 presents simulations

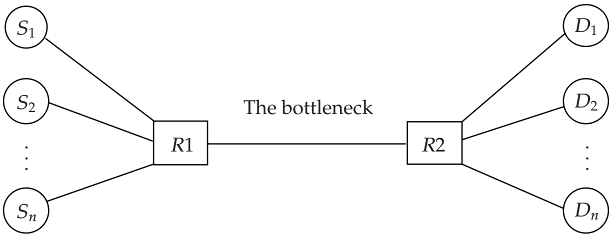


Figure 1: Structure of the network.

results and discusses the performance of various AQM schemes. Finally, we present the conclusions in Section 5.

2. Discrete-Time TCP-RED Model

In this paper, the network topology used is a dumbbell with a single bottleneck as shown in Figure 1. There are N connections sharing a single link with the capacity C. It is assumed that these connections are identical, long-lived, and have the same round-trip propagation delay d.

2.1. Random Early Detection (RED)

RED is a recommended scheme of AQM by the Internet Engineering Task Force (IETF) [3]. The main goal of RED is to achieve low average delay and high throughput. In order to achieve the goal, the RED gateway always measures q_{ave}, and drops/marks the arriving packets with the probability p to notify the TCP end of the incipient congestion when $q_{\text{ave}} > q_{\min}$. Once $q_{\text{ave}} > q_{\max}$, RED gateway drops every arriving packet so that it can control the average queue size even in the absence of cooperating sources. The drop probability p with RED can be described by the following piecewise function [3]:

$$p = \begin{cases} 0 & q_{\text{ave}} < q_{\min}, \\ 1 & q_{\text{ave}} > q_{\max}, \\ \dfrac{q_{\text{ave}} - q_{\min}}{q_{\max} - q_{\min}} p_{\max} & q_{\min} < q_{\text{ave}} < q_{\max}. \end{cases} \tag{2.1}$$

Here $p_{\max}$ is the drop probability when q_{ave} is equal to $q_{\max}$, and also called the maximum drop probability. $q_{\max}$ and $q_{\min}$ are the expected upper and lower thresholds, respectively. q_{ave} can be got with the following equation [3]:

$$q_{\text{ave}} = (1 - w)q'_{\text{ave}} + wq. \tag{2.2}$$

Here, q'_{ave} is average queue size at the previous time. q is instantaneous queue length. w is exponential averaging weight which is limited at $[0, 1]$. q_{ave} is updated at the time of packet arrival [3].

2.2. Discrete-Time Feedback Model for TCP-RED

To better understand the nonlinear instability in TCP-RED, Ranjan et al. have developed a discrete-time dynamical feedback TCP-RED model [11]. It is assumed that RED queue management mechanism with Explicit Congestion Notification (ECN) capability is implemented at each node to provide congestion signal. According to the model, the instantaneous queue size q can be determined by some network parameters, such as the buffer size B, mean packet size M, the number of connections N, and a constant K in [1, $(8/3)^{0.5}$]. q_{ave} can be expressed mathematically as [11]

$$q_{\text{ave}} = f\left(q'_{\text{ave}}\right) = \begin{cases} (1-w)q'_{\text{ave}} & q_{\text{ave}} \geq q1, \\ (1-w)q'_{\text{ave}} + wB & q_{\text{ave}} \leq q2, \\ (1-w)q'_{\text{ave}} + w\left(\dfrac{NK}{\sqrt{p}} - \dfrac{Cd}{M}\right) & \text{otherwise}. \end{cases} \tag{2.3}$$

Here $q1 = p1\ (q_{\max} - q_{\min})p_{\max} + q_{\min}$, where $p1 = (NMK/(dC))^2$ is the minimum drop rate that leads to an empty queue at next time. $q2 = p2(q_{\max} - q_{\min})/p_{\max} + q_{\min}$, where $p2 = (NMK/(dC+BM))^2$ is the maximum drop probability with full queue at next time. C is the capacity of the shared link, and d is round-trip propagation delay. The parameter p can be calculated by (2.1):

In the discrete-time model, it is noted that the average queue size is updated over time scale of approximately one RTT (round-trip time) [10, 11]. It is much longer than typical interarrival times of packets in practice. Therefore, w in the model is much larger.

2.3. Instability in TCP-RED

The interaction between TCP and RED is proved to be rather complex and nonlinear [11, 24]. In recent years, some novel approaches are proposed to better understand the complex and nonlinear phenomena in practice [25–27], which may be used to explain the nonlinear phenomena in TCP-RED. In this section, the complex behavior with RED parameters variation is analyzed based on the associated eigenvalue that is presented as follows.

If the average queue size stabilizes at a value q_{ave}, that is, $q_{\text{ave}} = f(q_{\text{ave}})$, the fixed point q_{ave} must lie in the range $[q2, q1]$. That can be obtained from (2.3). Considering the associated eigenvalue λ corresponding to q_{ave}, the linear stability condition is $|\lambda| < 1$ [11].

$$|\lambda| = \left| \frac{\partial f(q_{\text{ave}})}{\partial q_{\text{ave}}}\bigg|_{q_{\text{ave}}} \right| = \left| 1 - w - \frac{wNK}{2\sqrt{v}\left(q_{\text{ave}} - q_{\min}\right)^{3/2}} \right| < 1. \tag{2.4}$$

Here, $v = p_{\max}/(q_{\max} - q_{\min})$ means the increase factor of p. Figures 2 and 3 plot the average queue size and the associated eigenvalue for a fixed point $q^*_{\text{ave}} = 345.1$, as w varies from 0.12 to 0.2. In simulation, the parameters are the same as in [11]: $C = 75\,\text{Mb/s}$, $K = (3/2)^{0.5}$, $M = 4$, 000b, $d = 0.1\,\text{s}$, $N = 250$, $q_{\max} = 750$ packets, $q_{\min} = 250$ packets, $B = 3750$ packets, $p_{\max} = 0.1$.

From Figures 2 and 3, it is evident that for $w < 0.1578$, q_{ave} converges to the fixed point $q^*_{\text{ave}} = 345.1$. At the same time, $|\lambda|$ also stays below 1. As w increases, PDB

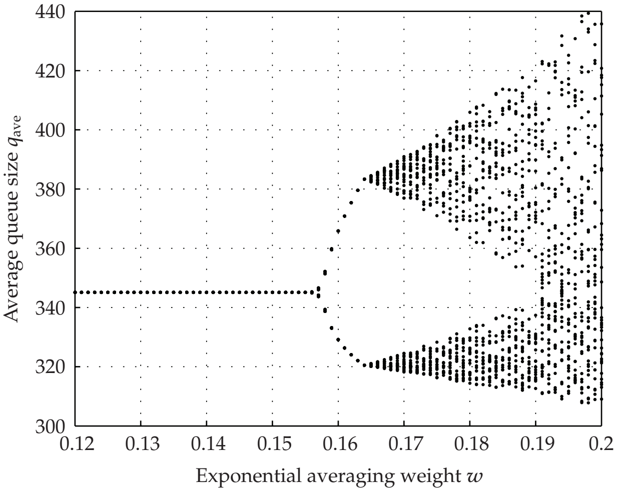


Figure 2: Bifurcation diagram of average queue size.

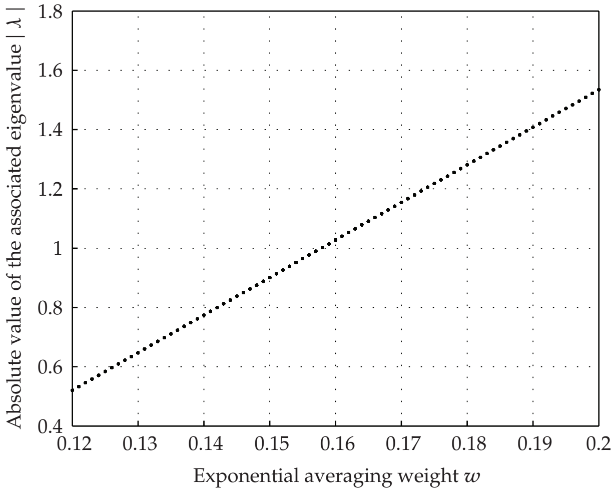


Figure 3: Absolute value of the associated eigenvalue.

(period doubling bifurcation) emerges from the fixed point, and then chaos appears with $|\lambda| > 1$. It implies that if $|\lambda| > 1$ for q_{ave}, the average queue size is instable at the point. In other words, the point is not a fixed one. The system oscillates and even exhibits chaos phenomenon.

Because all the parameters in (2.4) are positive, λ must be smaller than 1. The linear stability can be rewritten as $\lambda > -1$, that is,

$$\lambda = 1 - w - \frac{wNK}{2\sqrt{v}\left(q_{\text{ave}} - q_{\text{min}}\right)^{3/2}} > -1. \tag{2.5}$$

3. The Improved ARED Algorithm

In this section, we first briefly introduce ARED algorithm, and then propose a novel improved ARED algorithm. At last, we give a theoretic analysis for the proposed algorithm.

3.1. Adaptive RED

In 2001, Floyd et al. proposed a modified version of the original ARED (Adaptive RED), also called ARED [9]. To ensure high throughput and low delay, they set a target range with $[q_{\min} + 0.4\ (q_{\max} - q_{\min}), q_{\min} + 0.6\ (q_{\max} - q_{\min})]$. When the current average queue size is not in the range, the control parameter $p_{\max}$ is revised by AIMD (additive-increase multiplicative-decrease) policy, which can be expressed mathematically as:

$$p_{\max} = \begin{cases} p_{\max} + \alpha, & q_{\text{ave}} > q_{\text{target}},\ p_{\max} < 0.5, \\ p_{\max}\beta, & q_{\text{ave}} < q_{\text{target}},\ p_{\max} \geq 0.01. \end{cases} \tag{3.1}$$

Here, q_{target} is the target range. ARED changes $p_{\max}$ slowly, and the time scale is larger than interarrival time in original ARED. In general, the parameter *interval* = 0.5 seconds. Based on the principle that the change of q_{ave} should not be larger than the range of target in single *interval* [9], Floyd et al. advised to set $\alpha = \min(0.01, p_{\max}/4)$ and $\beta = 0.9$. Besides that, $p_{\max}$ is constrained within the range [0.01, 0.5], so that the overall performance of RED could be still acceptable in the transition period, even though the average queue size is out of the target range. A lot of experiments demonstrate that ARED has a good performance in terms of stability and robustness [9]. However, the parameter w and the bound of $p_{\max}$ are configured according to some experiment results. Therefore, ARED works well in some network scenarios but fails in others.

3.2. The Improved ARED Algorithm

Since ARED adjusts $p_{\max}$ by monitoring whether q_{ave} is within the target range, it is sensitive to the change of q_{ave}. Besides that, it causes oscillation when w is improper. In the section, we propose a novel ARED, named improved ARED, to reduce the parametric sensitivity and traffic sensitivity. The algorithm is described in Algorithm 1.

There are two main differences between ARED and the improved ARED: (1) $p_{\max}$ is constrained in the range $[p2/0.4, p1/0.6]$ in the improved ARED, instead of the range [0.01, 0.5] in ARED, (2) w is also adjusted in the improved ARED, when the linear stability condition is not satisfied and q_{ave} stays in the steady range.

3.3. Maximum Drop Probability $p_{\max}$

Base on the discussion in Section 2.3, one can see that q_{ave} is stable only when it is in the range $[q2, q1]$ and λ at the point is below -1. In order to stabilize q_{ave}, it is necessary to ensure $q_{\text{ave}} \in [q2, q1]$. If the target range in ARED is constrained in the steady range $[q2, q1]$,

```
Every interval seconds:
  if ($q_{\text{ave}} > q_{\text{target}}$ && $p_{\max} < p_{\max\_}u$)
        $p_{\max} = p_{\max} + \alpha$;
  else
        if ($q_{\text{ave}} < q_{\text{target}}$ && $p_{\max>}p_{\max\_}l$)
            $p_{\max} = p_{\max} \times \beta$;
        if ($q_{\text{ave}} \in q_{\text{target}}$)
          Calculate $\lambda$ by (2.4)
            if ($\lambda < -1$ && $w > w_{\min}$)
                $w = w \times \gamma$;
Parameters:
$p_{\max\_}u$: the upper threshold of $p_{\max}$; $p1/0.6$
$p_{\max\_}l$: the lower threshold of $p_{\max}$; $p2/0.4$
$\gamma$: decrease factor of $w$
$w_{\min}$: the lower threshold of $w$; $1 - \exp(-1/(10Cd/M))$
Other parameters are the same as those with Adaptive RED
```

Algorithm 1: Improved ARED algorithm.

$p_{\max}$ could be adjusted effectively to keep q_{ave} in the range. In the case, the two following inequalities are obtained:

$$\begin{aligned} q2 &< q_{\min} + 0.4\left(q_{\max} - q_{\min}\right), \\ q1 &> q_{\min} + 0.6\left(q_{\max} - q_{\min}\right). \end{aligned} \tag{3.2}$$

Substituting $q1 = p1(q_{\max} - q_{\min})/p_{\max} + q_{\min}$ and $q2 = p2(q_{\max} - q_{\min})/p_{\max} + q_{\min}$ into (3.2), we obtain the range of $p_{\max}$:

$$\frac{p2}{0.4} < p_{\max} < \frac{p1}{0.6}. \tag{3.3}$$

By the way, the drop mechanism with ARED can keep q_{ave} within the steady range $[q2, q1]$, so the stability of q_{ave} depends on the value of λ from (2.5). Figure 4 shows the drop probability with the improved ARED algorithm.

3.4. *Exponential Averaging Weight w*

The inequality (2.5) shows that λ depends on control parameters $(q_{\max}, q_{\min}, w, p_{\max})$, system parameter N, the constant K and q_{ave}. Since the number of connections N is out of control, a network manager usually changes control parameters to stabilize q_{ave}. The thresholds $q_{\min}$ and $q_{\max}$ refer to the tradeoffs between throughput and delay, and they are constants in general. Thus, they are set in advance. The control parameters $p_{\max}$ is adjusted by ARED scheme. Therefore, w becomes the only parameter to be chosen to achieve the stability of q_{ave}. Since λ is a monotony decrease function of w from (2.5), $\lambda > -1$ can be ensured by decreasing w when λ is below -1. In other words, the linear stability of q_{ave} can be obtained by decreasing w.

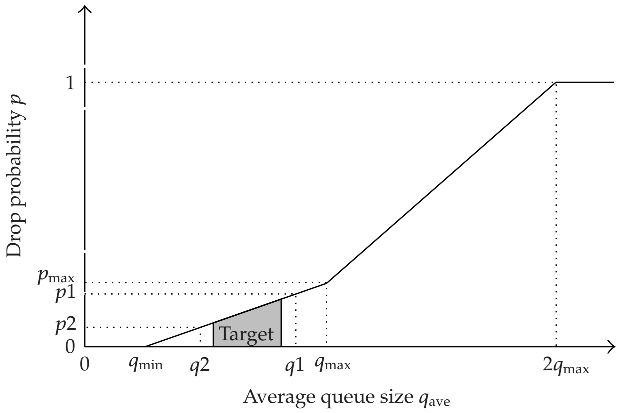


Figure 4: Drop probability of the improved ARED algorithm.

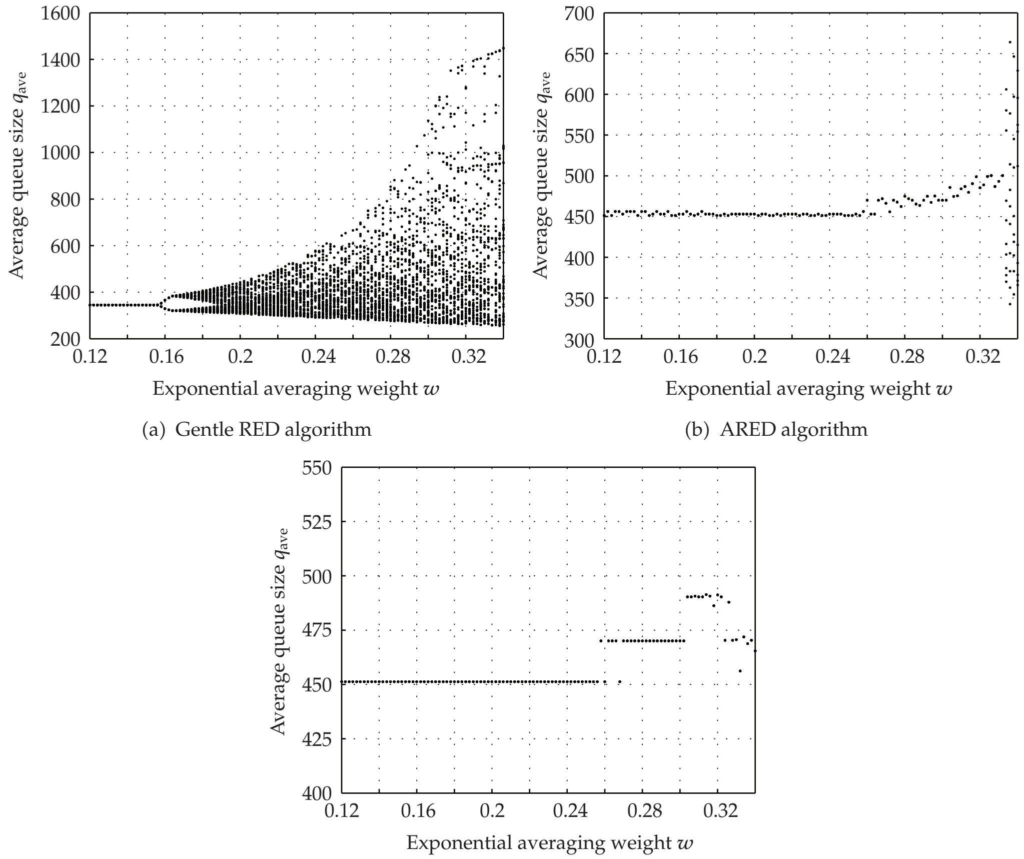
On the other hand, if w is too small, the average queue size cannot reflect the change of instantaneous queue size, so that RED gateway cannot detect the early congestion. In order to avoid too mall of w, a lower threshold $w_{\min} = 1 - \exp(-1/(10Cd/M))$ is used [9]. Besides that, multiplicative-decrease policy is used to reduce w quickly.

4. Simulations

The instability of RED schemes comes from its parametric sensitivity and traffic sensitivity. In other words, RED scheme is very sensitive to control parameters (w, $q_{\min}$) and system parameters (N, d). In this section, we test the performance of the improved ARED, the gentle RED, and ARED when these parameters vary in a large range. The three algorithms use the discrete-time TCP-RED model for simulation, and are evaluated by the stability of q_{ave}. The parameters in simulation are the same as in [11]: $C = 75\,\text{Mb/s}$, $K = (3/2)^{0.5}$,$M = 4{,}000\text{b}$,$d = 0.1\,\text{s}$,$N = 250$,$q_{\max} = 750$ packets,$q_{\min} = 250$ packets,$B = 3750$ packets, $w = 0.15$, $p_{\max} = 0.1$, $\gamma = 08$ except special descriptions. Besides that, because the instant of observation is about one RRT [10, 11], $p_{\max}$ is updated after every 5 iterations. It is roughly 0.5 seconds for both the improved ARED and ARED.

4.1. *Control Parameters* $(w, q_{\min})$

In the subsection, we test the parametric sensitivity of various algorithms. The plots in Figure 5 present the average queue size when w varies from 0.12 to 0.34 for the gentle RED, ARED and the improved ARED, respectively. Figure 5(a) shows that for gentle RED, q_{ave} can be stabilized at 345.1 with a small w. As w increases, q_{ave} becomes instable, and then PDB appears. When w is larger than 0.164, q_{ave} exhibits chaos phenomenon. ARED cannot stabilize q_{ave} within the target range until $w > 0.334$ as shown in Figure 5(b). It implies that ARED is more stable than gentle RED when w varies. However, it also leads to PDB and chaos. From Figure 5(c), it is found that the improved ARED keeps q_{ave} stable at a point within target range when w is set at different values. The experiments show that the robustness of the improved ARED is better than Adaptive RED and gentle RED in terms of w.



(a) Gentle RED algorithm

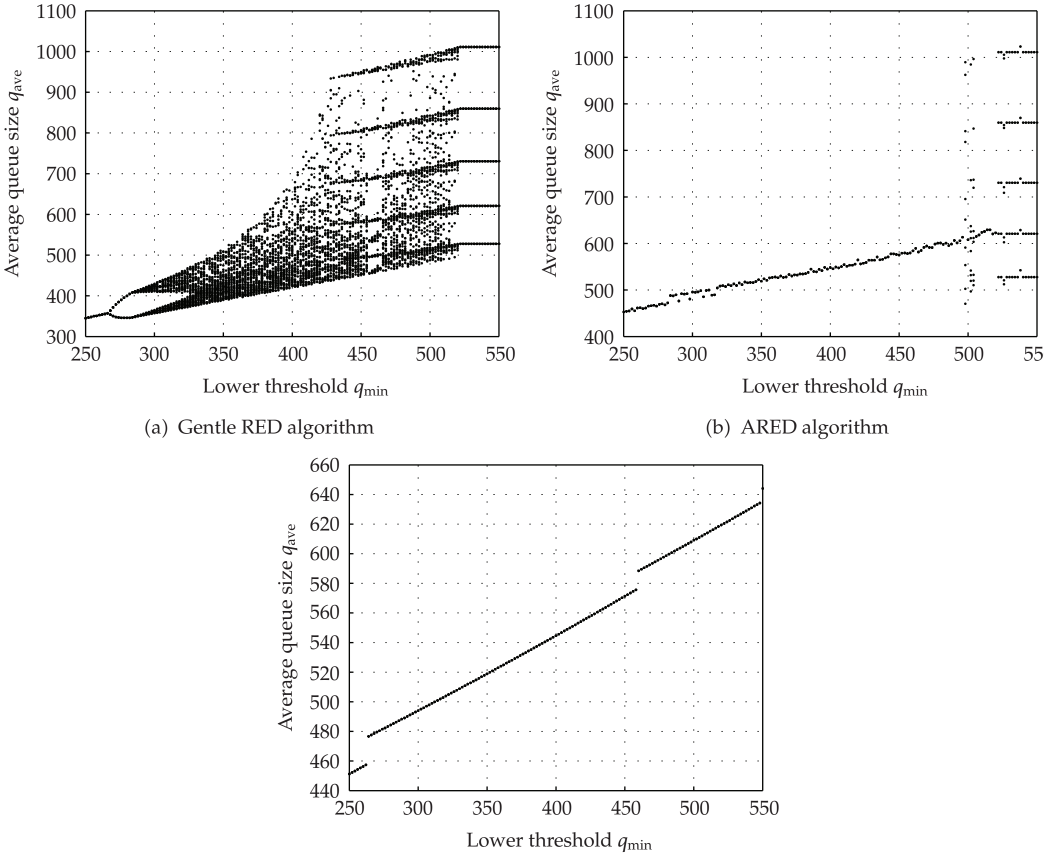
(b) ARED algorithm

(c) Improved ARED algorithm

Figure 5: Average queue size q_{ave} versus w.

In simulations, ARED can stabilize q_{ave} with a larger w compared with gentle RED because $p_{\max}$ with ARED is adjusted dynamically. However, when w is larger than some critical values, q_{ave} is more sensitive to the change of queue length, and thus changes fast. As a result, ARED gateway frequently adjusts $p_{\max}$ that leads to oscillation. In the same case, the improved ARED decreases w according to the linear stability condition besides adjusting $p_{\max}$. Therefore, the performance of the improved ARED outperforms ARED.

Figure 6 demonstrates the change of the average queue size for various algorithms with the variation of $q_{\min}$. From these figures, it is evident that gentle RED does a poor job in terms of stability due to the constant $p_{\max}$. However, both ARED and the improved ARED perform better in eliminating oscillations and chaos when the lower threshold changes from 250 to 550. When $q_{\min}$ is near 500 packets, ARED exhibits oscillations as shown in Figure 6(b). That is because the target range $[q_{\min} + 0.4(q_{\max} - q_{\min}), q_{\min} + 0.6(q_{\max} - q_{\min})]$ shrinks as $q_{\min}$ increases, which means that q_{ave} stays within the range with a smaller probability. Under this condition, $p_{\max}$ fluctuates frequently, so q_{ave} fluctuates too. However, the improved ARED changes the value of w based on linear stability condition to reduce the effect of the queue length on q_{ave}, and then reduce the fluctuation of $p_{\max}$. By doing so, the improved RED



(a) Gentle RED algorithm

(b) ARED algorithm

(c) Improved ARED algorithm

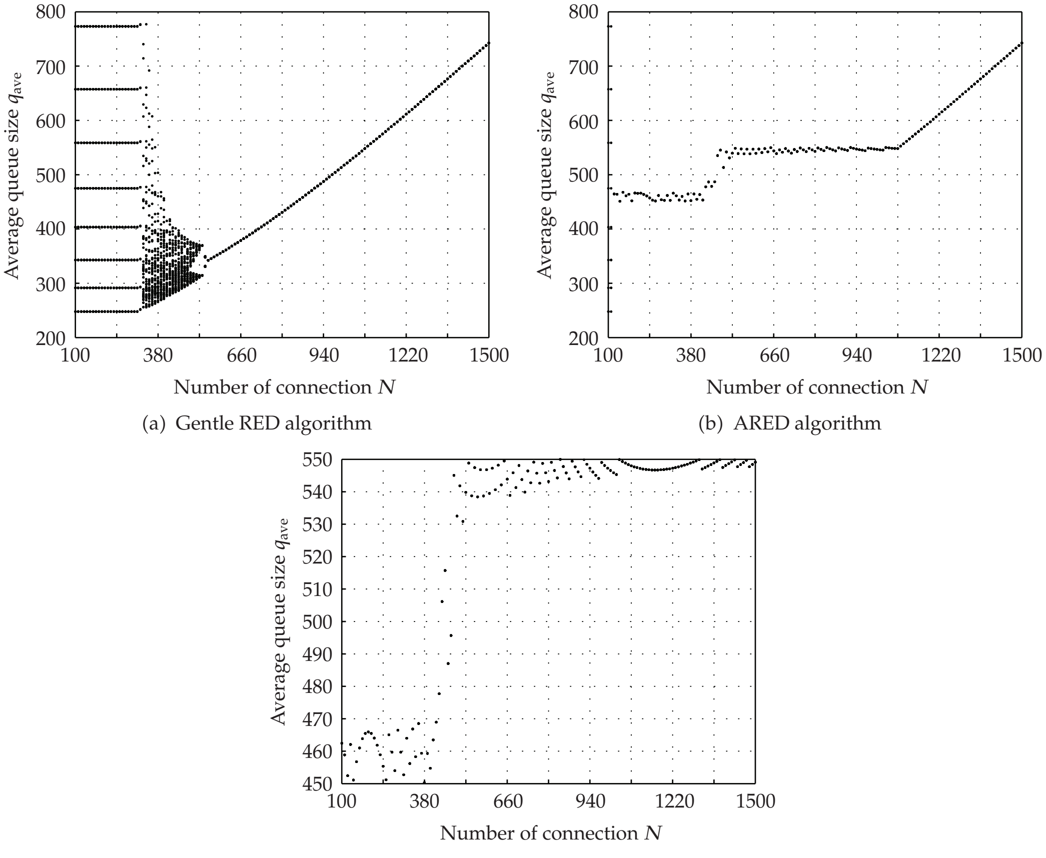
Figure 6: Average queue size q_{ave} versus q_{min}.

degrades the sensitivity of q_{min} to some extent, and improves the stability of q_{ave} compared with ARED and gentle RED, as shown in Figure 6(c).

4.2. System Parameters (N, d)

In this subsection, we test the stability and robustness of different algorithms under different network load N and different round-trip propagation delay d.

Figure 7 shows the average queue size when N increases from 100 to 1500. From Figure 7(a), it is evident that the system is difficult to stabilize q_{ave} with small N. As N increases, the system is more stable and q_{ave} stabilizes at a larger value. These results are consistent with that in [28]. However, if N continues to increase without constrain, there exists a value of N with q_{ave} near $2q_{\mathrm{max}}$, which corresponds to the drop probability of 1 for gentle RED. Once N exceeds the value, q_{ave} oscillates [8]. Therefore, the system can stabilize q_{ave} only if N varies in some range when other parameters are determined. Figure 7(b) shows that q_{ave} of ARED is stable when $N \in [520, 1080]$. However it increases when $N > 1080$ and oscillates when $N < 520$. In the simulations, the improved ARED can keep q_{ave} stable in the target range with $N \in [100, 1500]$, as shown in Figure 7(c).



(a) Gentle RED algorithm

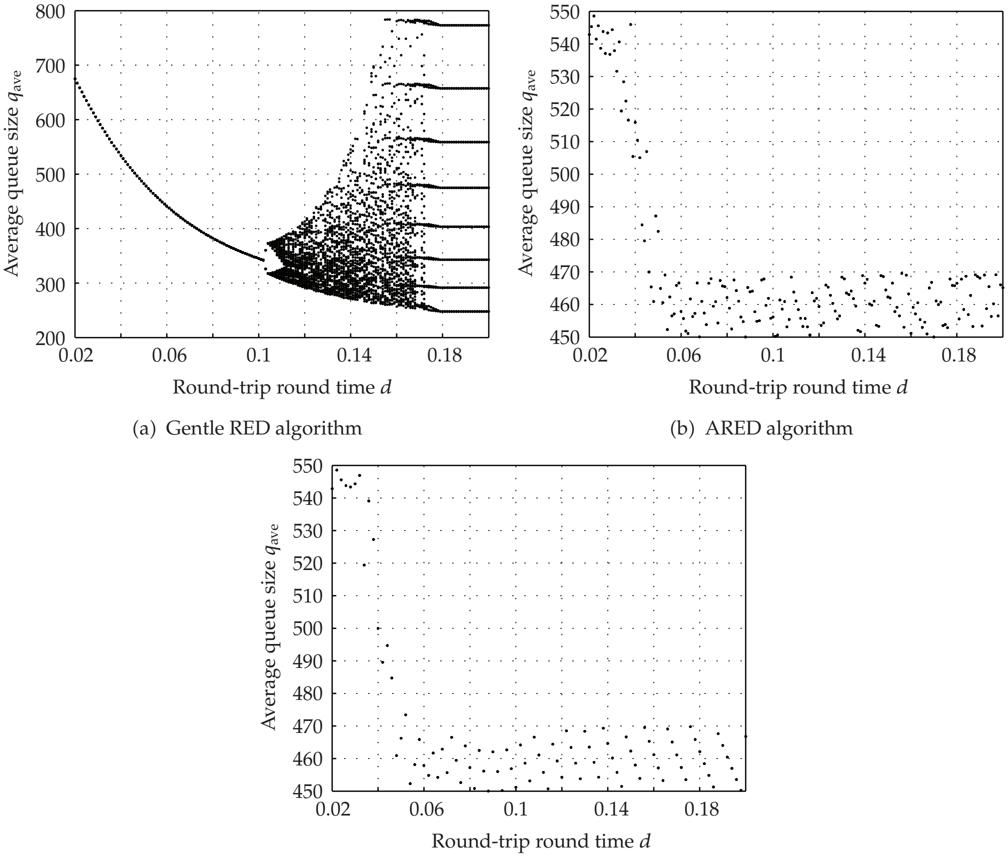
(b) ARED algorithm

(c) Improved ARED algorithm

Figure 7: Average queue size q_{ave} versus N.

For small N, ARED and the improved ARED decrease $p_{\max}$ to reduce the effect of single source on traffic so they can achieve good stability. For large N, a lot of packets need to be dropped to trigger enough congestion signals with a large drop probability. For the improved ARED, both $p1$ and $p2$ will increase accordingly, and the bounds of $p_{\max}$ will be enlarged as shown in (3.3). Therefore, the improved ARED can provide enough congestion signal by adapting a larger $p_{\max}$. However, the upper threshold of $p_{\max}$ is set to 0.5 for ARED, so it is ineffective to provide a large enough drop probability with $N > 1080$. Therefore, the improved algorithm can get better performance in the simulations.

The plots in Figure 8 show the average queue size with respect to the round-trip propagation delay. For gentle RED, larger d results in instability as shown in Figure 8(a), because it takes more time to transmit an early congestion signal. After the ends receive the congestion signals, a lot of packets have been sent out, caused more serious congestion with a long queue length. On the other hand, when the system recovers from congestion, the ends may still decrease sending rate, due to the delay arrival of congestion signals sent at the previous time. In this case, the average queue size will be very small and the network resource cannot be fully utilized. In the experiments, both ARED and the improved ARED can reduce aggressiveness of drop mechanism through decreasing $p_{\max}$ that makes q_{ave} enter



(a) Gentle RED algorithm

(b) ARED algorithm

(c) Improved ARED algorithm

Figure 8: Average queue size q_{ave} versus d.

into stability slowly. Figures 8(b) and 8(c) also verify that both algorithms are effective in control of oscillation and chaos.

5. Conclusions

In this paper, we study the instability of TCP-RED based on a discrete-time feedback system model and propose an improved ARED algorithm in which bounds of the maximum drop probability $p_{\max}$ are optimized and the lower threshold of the exponential averaging weight w is adjusted based on linear stability condition. Simulations based on the discrete-time model show that the improved ARED algorithm can stabilize q_{ave} effectively in the target range with various values of control parameters $(w, q_{\min})$ and system parameters (N, d). Compared with the gentle RED and ARED, the improved ARED algorithm is less sensitive to parameters, and performs better than the two schemes in terms of stability. The simulation results imply that the improved ARED can achieve high throughput and less delay.

Acknowledgments

The work was supported by National Natural Science Foundation of China (Grant no. 60573125, 60703112, and 60873264).

References

[1] L. S. Brakmo and L. L. Peterson, "TCP Vegas: end to end congestion avoidance on a global internet," *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 8, pp. 1465–1480, 1995.

[2] R. J. La and V. Anantharam, "Utility-based rate control in the Internet for elastic traffic," *IEEE/ACM Transactions on Networking*, vol. 10, no. 2, pp. 272–286, 2002.

[3] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, pp. 397–413, 1993.

[4] S. Athuraliya, S. H. Low, V. H. Li, and Q. Yin, "REM: active queue management," *IEEE Network*, vol. 15, no. 3, pp. 48–53, 2001.

[5] R. J. Gibbens and F. Kelly, "Resource Pricing and the Evolution of Congestion Control," http://www.statslab.cam.ac.uk/~frank/evol.html, 1998.

[6] S. Kunniyur and R. Srikant, "Analysis and design of an adaptive virtual queue (AVQ) algorithm for active queue management," in *Proceedings of the ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM '01)*, San Diego, Calif, USA, August 2001.

[7] M. May, J. Bolot, C. Diot, and B. Lyles, "Reasons not to deploy RED," in *Proceedings of the 7th International Workshop on Quality of Service (IWQoS '99)*, pp. 260–262, 1999.

[8] W. Feng, D. Kandlur, D. Saha, and K. G. Shin, "A self-configuring RED gateway," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM '99)*, vol. 3, pp. 1320–1328, 1999.

[9] S. Floyd, R. Gummadi, and S. Schenker, "Adaptive RED: an algorithm for increasing the robustness of RED's active queue management," *Technical Report*, 2001, http://www.icir.org/floyd/papers/adaptiveRed.pdf.

[10] R. J. La, P. Ranjan, and E. H. Abed, "Analysis of adaptive random early dection (Adaptive RED)," in *Proceedings of the 18th International Teletraffic Congress (ITC '03)*, Berlin, Germany, 2003.

[11] P. Ranjan, E. H. Abed, and R. J. La, "Nonlinear instabilities in TCP-RED," *IEEE/ACM Transactions on Networking*, vol. 12, no. 6, pp. 1079–1092, 2004.

[12] L. Chen, X. F. Wang, and Z. Z. Han, "Controlling bifurcation and chaos in Internet congestion control model," *International Journal of Bifurcation and Chaos*, vol. 14, no. 5, pp. 1863–1876, 2004.

[13] J. H. C. Nga, H. H. C. Iu, S. H. Ling, and H. K. Lam, "Comparative study of stability in different TCP/RED models," *Chaos, Solitons and Fractals*, vol. 37, no. 4, pp. 977–987, 2008.

[14] S. Flyod, "Recommendation on using the "gentle" variant of RED," http://www.icir.org/floyd/red/gentle.html, 2000.

[15] C. V. Hollot, V. Misra, D. Towsley, 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, 2002.

[16] S. Liu, T. Basar, and R. Srikant, "Exponential-RED: a stabilizing AQM scheme for low- and high-speed TCP protocols," *IEEE/ACM Transactions on Networking*, vol. 13, no. 5, pp. 1068–1081, 2005.

[17] S. Guo, X. Liao, C. Li, and D. Yang, "Stability analysis of a novel exponential-RED model with heterogeneous delays," *Computer Communications*, vol. 30, no. 5, pp. 1058–1074, 2007.

[18] C. Wang, J. Liu, B. Li, K. Sohraby, and Y. T. Hou, "LRED: a robust and responsive AQM algorithm using packet loss ratio measurement," *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, no. 1, pp. 29–43, 2007.

[19] H. Javam and M. Analoui, "SARED: stabilized ARED," in *Proceedings of the International Conference on Communication Technology (ICCT '06)*, pp. 1–4, 2006.

[20] J. Sun, K. Ko, G. Chen, S. Chan, and M. Zukerman, "PD-RED: to improve the performance of RED," *IEEE Communications Letters*, vol. 7, no. 8, pp. 406–408, 2003.

[21] L. Tan, W. Zhang, G. Peng, and G. Chen, "Stability of TCP/RED systems in AQM routers," *IEEE Transactions on Automatic Control*, vol. 51, no. 8, pp. 1393–1398, 2006.

[22] B. Zheng and M. Atiquzzaman, "A framework to determine bounds of maximum loss rate parameter of RED queue for next generation routers," *Journal of Network and Computer Applications*, vol. 31, no. 4, pp. 429–445, 2008.

[23] B. Zheng and M. Atiquzzaman, “A framework to determine the optimal weight parameter of RED in next-generation Internet routers,” *International Journal of Communication Systems*, vol. 21, no. 9, pp. 987–1008, 2008.

[24] N. Bigdeli and M. Haeri, “Time-series analysis of TCP/RED computer networks, an empirical study,” *Chaos, Solitons and Fractals*, vol. 39, no. 2, pp. 784–800, 2009.

[25] M. Li and W. Zhao, “Representation of a stochastic traffic bound,” IEEE Transactions on Parallel and Distributed Systems, IEEE Computer Society Digital Library, IEEE Computer Society, http://doi.ieeecomputersociety.org/10.1109/TPDS.2009.162, 2009.

[26] M. Li, “Fractal time series—a tutorial review,” *Mathematical Problems in Engineering*, vol. 2010, Article ID 157264, 26 pages, 2010.

[27] M. Li and P. Borgnat, “Forward for the special issue on traffic modeling, its computations and applications,” *Telecommunication Systems*, vol. 43, no. 3-4, pp. 181–195, 2010.

[28] C. V. Hollot, V. Misra, D. Towsley, and W. Gong, “A control theoretic analysis of RED,” in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM ’01)*, vol. 3, pp. 1510–1519, 2001.