

Active Queue Management Algorithms: BLUE vs. RED

Bahri Okuroğlu and Sema Oktuğ
Department of Computer Engineering, Istanbul Technical University
okuroglu@itu.edu.tr , oktug@itu.edu.tr

Abstract: Usage of the active queue management algorithms are recommended by IETF in order to prevent congestion and to decrease packet loss rate. The main principle of these algorithms is to drop packets systematically to inform the end nodes about the congestion on gateways. BLUE, a new active queue management algorithm, has many advantages over the most popular active queue management algorithm RED in some particular conditions. In this paper these two active queue management algorithms are compared using simulations.

Keywords: Active Queue Management, Congestion Control, TCP

A. INTRODUCTION

Active queue management mechanisms are congestion control algorithms which are run on the gateways to detect congestion earlier and to send implicit or explicit feedback to the end-nodes. Due to the advantages, usage of the active queue management architectures on the gateways are recommended by IETF [1].

These algorithms has many advantages over classical drop-tail queues. Although the main advantage is the detection of the congestion earlier and reduction in the drop rate, these mechanisms also avoid the global synchronization and bias against bursty flows.

Active queue management algorithms either drop or just mark packets in order to control the queue length. When packets are dropped to control congestion, end nodes do not need to cooperate with the active queue management algorithms. To prevent high loss rates caused by the packet drops to control the queue length, IETF is considering deployment of explicit congestion notification (ECN) [9].

By the use of ECN, gateways can mark packets instead of dropping them, and end nodes can take the proper action to decrease the size of the congestion window which indicates how much data the network can handle and is managed by the end node TCP.

In this paper, the performance of RED and the performance of BLUE are compared by changing RTT values and ECN support under heavy bursty traffic. It is shown that with ECN support BLUE is superior to RED. When ECN support disappears, the performance of BLUE becomes closer to the performance of the RED algorithm.

The rest of the paper is organized as follows: Section 2 gives the description of the RED algorithm and the BLUE algorithm. The performance of RED and the performance of BLUE are compared for different queue sizes, RTT values, and ECN support in Section 3. Finally, Section 4 concludes the paper.

B. ACTIVE QUEUE MANAGEMENT ALGORITHMS

RED (Random Early Detection) is the most popular and widely used active queue management algorithm. Recently an alternative algorithm called BLUE is developed. In this section these two active queue management algorithms are briefly described.

1.. RED

RED is the first widely used active queue management algorithm [2]. It calculates the average queue length using both the actual queue length and the average queue length, and *marks* packets with a probability proportional to the average queue length. By using average queue length at the marking probability calculations, instant bursts can be tolerated. Marking packets provides feedback information to source nodes on the congestion level of the gateways through the path.

```
for each incoming packet
    calculate the average queue length (avg)
    if  $\min_{th} < avg < \max_{th}$ 
        calculate packet dropping probability  $p_a$ 
        mark packets with probability  $p_a$ 
    if  $\max_{th} \leq avg$ 
        mark packet

 $(avg \leftarrow (1 - w_q) * avg + w_q * q)$ 
```

Figure 1 RED Algorithm

The effectiveness of the algorithm on congestion control and reduction of the loss rate is proven by both simulations and real-world experiences [2, 3]. However, the algorithm has parameter selection problems [4, 5]. In order to benefit from the algorithm its parameters must be arranged properly. Moreover, the marking aggressiveness of the algorithm is insensitive to the load on the gateway. Hence, this may result in the loss of queue control. There are

studies on modifying the RED algorithm in order to adjust its parameters dynamically according to the load on the gateway [6].

When large number of flows are active at gateways, RED causes high loss rates and inefficient use of link capacity even with the use of ECN and adaptive parameter adjustment mechanisms [7].

2.. BLUE

BLUE [7] is a recently developed active queue management mechanism which has a completely different marking strategy compared to RED. BLUE assumes that queue length does not directly reflect the congestion level. Hence it does not update the packet marking probability with the queue length. Instead it uses queue overflow and idle event history to update the packet marking probability (p_m). Packet loss due to the queue overflow means that the marking is not aggressive enough and p_m should be increased. Similarly, the queue idle event occurs as a result of the aggressive marking policy therefore the p_m parameter should be decreased. This mechanism effectively allows BLUE to *learn* the correct rate it needs to send back congestion notification.

```

For each packet loss:
    if ((now - last_update) > freeze_time)
         $p_m = p_m + d_i$ 
        last_update = now;
For link idle event:
    if ((now - last_update) > freeze_time)
         $p_m = p_m + d_d$ 
        last_update = now;

```

Figure 2 BLUE Algorithm

BLUE uses three parameters: The first two parameters determine the amount by which p_m is incremented in case of the queue overflows (d_i) or is decremented when the link is idle (d_d). The last parameter is the minimum time interval between two successive updates (*freeze_time*).

BLUE is a simple algorithm compared to RED.

C. PERFORMANCE COMPARISON

Performance of BLUE algorithm is evaluated by simulations and an experimental testbed at [7]. Even it was shown that BLUE beats RED when ECN is used in [7], BLUE has not been evaluated for different levels of RTTs and without ECN support.

In this paper, performance of the BLUE algorithm and the RED algorithm is evaluated and compared by simulations using ns simulation tool [8]. The

simulations are repeated for different RTTs considering networks with and without ECN support.

The network architecture used in the simulations is shown in Figure 3. 20 source nodes are connected through a bottleneck link to the 20 destination nodes. Each node runs 50 simultaneous exponential traffic applications with 2msn ON, 3msn OFF times. All the nodes are connected to the 10Mb/s, 10msn bottleneck with 100Mb/s links. The delays of the links between the sources and the bottleneck are selected differently for each link to differentiate the RTTs of flows. Flows are started randomly in the first 70 seconds of a 200-second simulation and the measurements are taken during the second half of the simulation duration. All nodes employ TCP Reno.

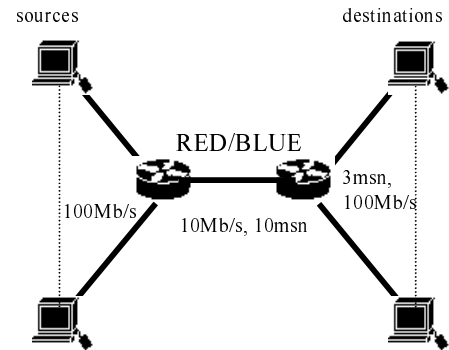


Figure 3 Simulation network architecture

RED or BLUE queue is used at the bottleneck link. Simulations are repeated for various queue sizes. RED queue is configured as $min_{th} = q_size/4$ and $max_{th} = 3min_{th}$ where q_size is the queue capacity. The marking limit of RED, max_p , is selected as 1 which is the best selection for this quantity of flows [7]. BLUE queue is configured as $d_i = 0.0005$, $d_d = 0.001$ and $freeze_time = 10msn$. These BLUE parameters are the optimized values to increase the throughput and decrease the loss rate [10].

Performance of the RED and BLUE algorithms is evaluated for two different levels of RTTs, with and without ECN support. First, in Case 1, the large RTT, ECN capable network environment is used. Next, in Case 2, the RTTs of the flows are decreased while keeping ECN support. Finally, in Case 3, large RTT, and ECN incapable network environment is used.

Case 1: Large RTT flows with ECN support

In this case, the link delays between the sources and the bottleneck are selected as $(2i+200)msn$ where i is the source node index between 0 and 19. All the nodes are ECN capable and gateways mark the packets instead of dropping them.

Figure 4 reveals that BLUE performs better than RED for each queue size, when throughput is considered. BLUE converges to the 100% link utilization using small queues. However, RED can reach that efficiency only for large queues.

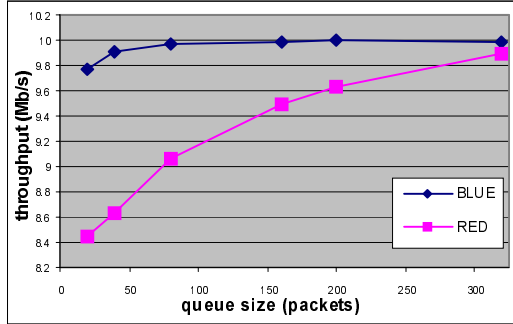


Figure 4 Total throughput over bottleneck link for both RED and BLUE (large RTT)

The inefficient link usage of RED is caused by its inability to control the queue length for high number of active bursty flows. As it is seen in Figure 5, both the instant queue length and the average queue length of RED oscillate between the queue limits. As a result, queue overflows just after a link idle event and becomes empty just after a queue overflow.

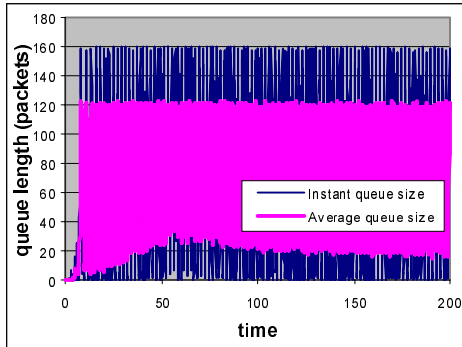


Figure 5 RED average and instant queue size graph for $q_{\max} = 160$ (large RTT)

Figure 6 shows that BLUE can control the queue successfully over time once the behavior of the traffic is learned. When new sources are activated, BLUE suffers from the changing traffic load. Once the traffic is stabilized, BLUE can control the queue successfully since it *knows* how to mark the packets. This control keeps the link at the high throughput condition.

In Figure 7, drop rates obtained for RED and BLUE can be compared. The queue instability occurring in RED causes large drop rates besides low link utilization. These drops are mainly forced drops which is caused by queue overflows. However, as shown Figure 7, when queue size is increased BLUE converges to zero drop rate. Since no queue overflow

is occurred and ECN is used to mark packets, packet loss can be kept at minimum rate.

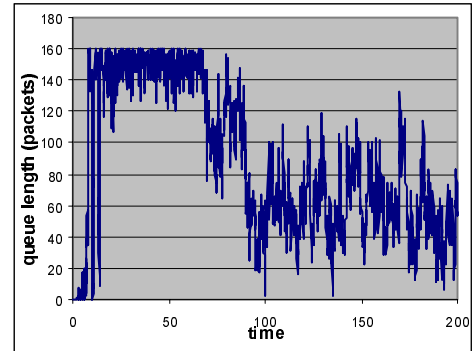


Figure 6 Queue size for BLUE when $q_{\max} = 160$ (large RTT)

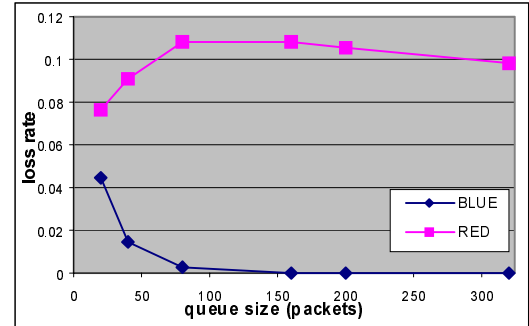


Figure 7 Total drop rate over bottleneck link for both RED and BLUE (large RTT)

Case 2: Small RTT flows with ECN support

Here, the Case 1 environment is modified by decreasing RTTs. The link delays of the sources to the bottleneck are set to $(2i+5)\text{msn}$ where i is the node index between 0 and 19.

Figure 8 shows that RED performs better under this configuration when throughput is concerned. The reason for such a result is again RED's inability to control the queue successfully. As seen in Figure 9, RED behaves like a drop-tail queue and causes too much forced drops to appear under this configuration. Since the queue is full most of the time, the throughput is much better than the one in the previous case. However, this means benefits of active queue management algorithms cannot be observed.

The throughput of BLUE is close to the results obtained in the previous case. As it is shown, RED and BLUE cannot beat each other on the throughput basis when queue size greater than 100 packets.

The most important observation of this simulation case is that RED cannot control the queue even it marks packets at its maximum rate which is 1. This is caused by the marking rate calculation algorithm of

RED, which uses queue length as the congestion measure.

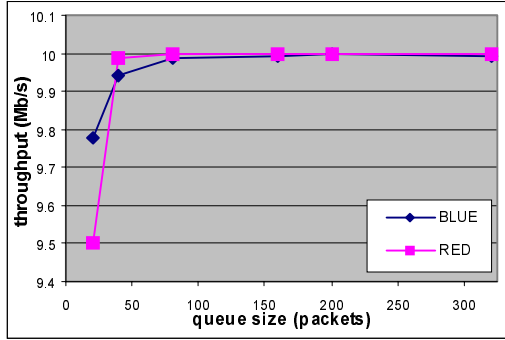


Figure 8 Total throughput over bottleneck link for both RED and BLUE (small RTT)

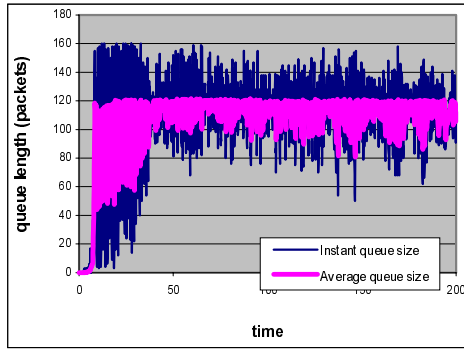


Figure 9 RED average and instant queue size graph for $q_{\max} = 160$ packets (small RTT)

Figure 9 shows that RED cannot control the queue and high number of forced drops caused by the queue overflow result in high loss rate as seen in Figure 10. However, BLUE can still control the queue successfully, it can keep the link at low loss rate.

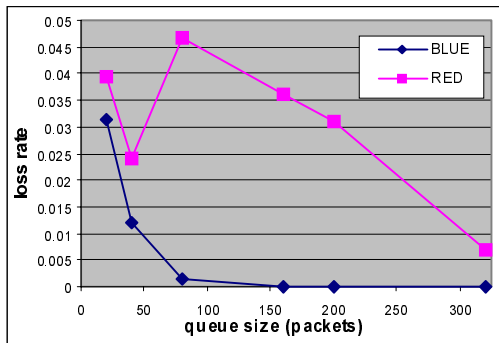


Figure 10 Total packet loss rate over bottleneck link for both RED and BLUE (small RTT)

The Case 1 and Case 2 simulations show that BLUE achieves queue control much better than RED in the case of high number of active flows with ECN support. For large RTTs, the RED queue overflows

and idles many times. When the RTTs are decreased, the queue only overflows.

Case 3: Large RTT flows without ECN support

In order to investigate the behavior of BLUE using the nodes which do not support ECN, the Case 1 configuration is re-run without ECN support.

As seen in Figure 11 and Figure 12, BLUE loses its superiority against RED observed in Case 1 and Case 2. The throughput values obtained for RED and BLUE are close to each other and the drop rate of BLUE is a bit higher than that of RED.

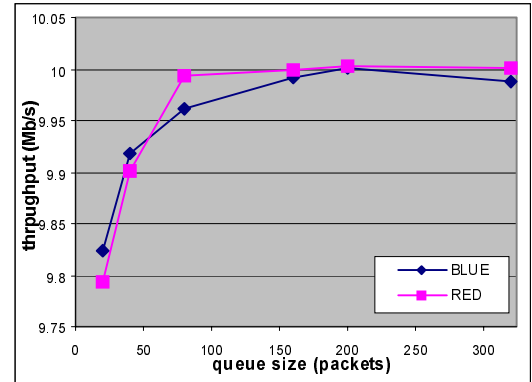


Figure 11 Total throughput over bottleneck link for both RED and BLUE (large RTT, without ECN)

Both RED and BLUE suffer from high loss rates, but large portion of these losses are unforced drops which are caused by active queue management mechanisms to control the queue.

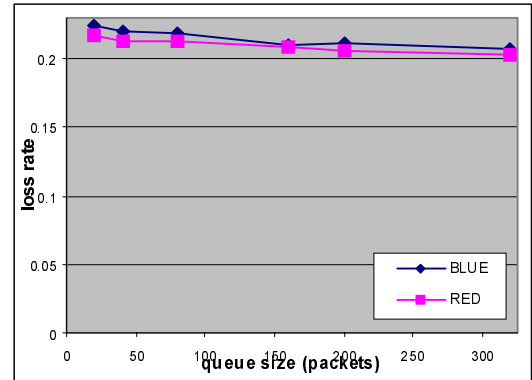


Figure 12 Total drop rate over bottleneck link for both RED and BLUE (large RTT, without ECN)

Figure 13 shows that RED can control the queue successfully when large RTT is used and without ECN support. As a result, the overflow loss for RED decreases. Without the ECN support, the RED queue does not oscillate as in Case 1.

BLUE behaves worse than it behaves in Case 1 on queue control as it is seen in Figure 14. Since it cannot

use packet marking to send feedback, it also suffers from high loss rates as shown in Figure 12.

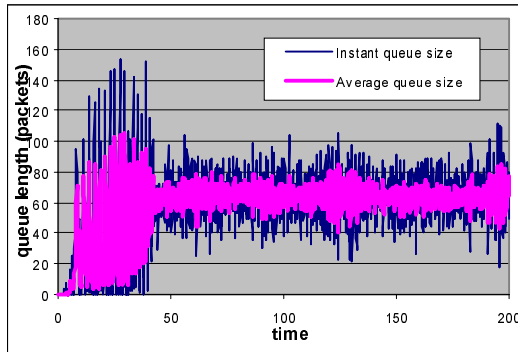


Figure 13 RED average and instant queue size graph for $q_{\max} = 160$ packets (large RTT, without ECN)

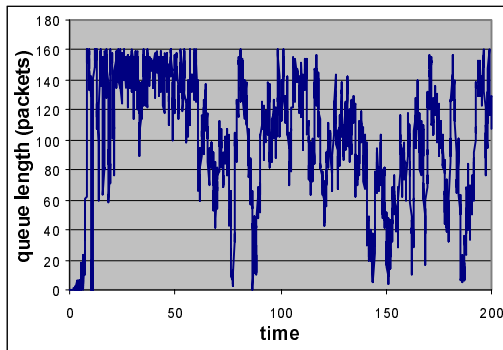


Figure 14 BLUE queue size graph for $q_{\max} = 160$ (large RTT)

The major difference between the results of Case 1 and Case 3 simulations is the behavior of BLUE. The performance of BLUE degrades as ECN support disappears. However, the performance obtained for BLUE is still very close to that obtained for RED.

These simulations are also repeated for smaller RTTs and similar results are obtained.

D. CONCLUSION

Active queue management algorithms detect congestion earlier and convey notification to sources by dropping or marking packets. These mechanisms increase throughput and decrease loss rate.

RED, the most popular active queue management algorithm, has inherent problems besides its proven benefits. A new active queue management algorithm, BLUE, can perform better with the use of ECN in the case high number of flows are active at the gateways.

In this paper, the performance of RED and the performance of BLUE are compared by changing RTT values and ECN support under heavy bursty traffic. It is shown that: When ECN, which helps in the reduction of the loss rate, is used with RED, it loses

the queue control in case of large number of active flows. For some conditions RED behaves like a drop-tail queue, for others queue size oscillates between the queue limits. However, ECN supported BLUE algorithm can control queue successfully and decrease the loss rate for central gateways with high number of flows active. When ECN support disappears, the performance of BLUE becomes closer to that of RED.

Today ECN is not deployed widely, thus BLUE's benefits cannot be observed in real-world. Even if BLUE cannot surpass the performance of RED without ECN support, it can still race with it, and for ECN capable hosts and networks it will help to decrease the packet loss rate.

E. REFERENCES

- [1] Braden, B., et. al., 1998, Recommendations on Queue Management and Congestion Avoidance in the Internet, Request For Comments: 2309
- [2] Floyd, S., Jacobson, V., 1993, Random Early Detection Gateways for Congestion Avoidance, IEEE/ACM Transactions on Networking, <http://www.aciri.org/floyd/papers/early.pdf>
- [3] Doran, S., 1998, EBONE Interface Graphs, <http://adm.ebone.net/~smd/red-1.html>
- [4] Labrador, M., Banerjee, S., 1999, Packet Dropping Policies for ATM and IP Networks, IEEE Communications Surveys, vol 2, no 3, <http://www.comsoc.org/pubs/surveys>
- [5] May, M., Bolot, J., Diot, C., Lyles, B., 1999, Reasons not to deploy RED, INRIA Sophia-Antipolis, ENSIM, Sprintlabs, <http://www-sop.inria.fr/rodeo/personnel/mmay/may-red.html> <http://www-sop.inria.fr/rodeo/personnel/mmay/papers/slides-iwqos-sprint.ps>
- [6] Feng, W., Kandlur, D., Shin, K., 1999, A Self-configuring RED Gateway, INFOCOM '99, <http://www.eecs.umich.edu/~wuchang/work/CSE-TR-349-97.ps.Z>
- [7] Feng, W., Kandlur, D., Saha, D., Shin, K., 2000, BLUE: A new class of active queue management, <http://www.eecs.umich.edu/~wuchang/blue/>
- [8] NS: Network Simulator, <http://www.mash.cs.berkeley.edu/ns/>
- [9] Ramakrishnan, K., Floyd, S., 1999, A Proposal to add Explicit Congestion Notification (ECN) to IP, Request for Comments: 2481
- [10] Okuroğlu, B., 2000, TCP/IP'de Aktif Kuyruk Yönetim Mekanizmaları ve İnternet İçin Kaliteli Hizmet, MS Thesis, İstanbul Technical University, Control and Computer Engineering