Simulation Analysis of Active Queue Management for Internet Congestion Control

Sukant Kishoro Bisoy
Department of Computer Science
C.V.Raman College of Engineering,
Bhubaneswar, India.

Sunil Kumar Mohapatra
Department of Computer Science
C.V.Raman College of Engineering,
Bhubaneswar, India.

Prasant Kumar Pattnaik
School of Computer Engineering,
KIIT University
Bhubaneswar, India.

Pratima Panigrahi
School of Computer Engineering,
KIIT University
Bhubaneswar, India.

ABSTRACT

In order to prevent congestion, the current internet uses end-to-end congestion control protocol like TCP. In congestion control issues, queue management employed by router has been utmost important. Active queue management (AQM) has been proposed as a router-based mechanism for early detection of congestion inside the network. AQM scheme helps for end-to-end congestion control by having routers detecting congestion and notify end-systems, so that the sender adjust transmission rate earlier and avoid unwanted packet drops.

Aim of the paper was to analyze the performance of various active queue management (AQM) techniques like Random Exponential Marking (REM), Gentle RED (GRED) and Nonlinear RED (NLRED) with DropTail (DT). We study different characteristics of various versions of RED using NS2 simulator and outcome indicates that performance of NLRED is better than others in terms of goodput, packet loss rate, delay, link utilization, fairness index, average queue length.

Keywords

AQM, RED, REM, GRED, NLRED, DropTail, Fairness Index and Goodput

1. INTRODUCTION

The most important problem in Internet is congestion. Because of heavy traffic in core network, congestion may occur at router [1]. A Droptail router discards packets when its FIFO queue is full. It was shown in Zhang et al. [2] that under heavy load conditions, Droptail routers cause global synchronization, a phenomenon in which all senders sharing the same bottleneck router/link shut down their transmission windows at almost the same time. The network, in particular the routers in the network, should play an active role in its resource allocation, so as to effectively control/prevent congestion. This is known as active queue management (AQM) [3]. The essence is that an AQM router may intelligently drop packets before the queue overflows.

Aim of this paper was to conduct a comparison study of various active queue management like GRED [4], NLRED [5] and REM [6] with Droptail to study the different characteristics with respect to some parameters.

The rest of the paper is organized as follows. In section 2 we will explain the queue management algorithm. Section 3 will present model and simulation set up and section 4 will present result analysis. Finally, conclusion of the paper is given in section 5 and references in section 6.

2. QUEUE MANAGEMENT ALGORITHM

The idea behind Active Queue Management is to discard a packet before queue overflow according to a drop probability function. By discarding a packet before queue overflow, a TCP sender can detect congestion earlier and react earlier.

Researchers and the IETF proposed active queue management (AQM) as a mechanism for detecting congestion inside the network and they strongly recommended the deployment of AQM in routers as a measure to preserve and improve WAN performance. AQM algorithms [3] run on routers and detect incipient congestion by typically monitoring the instantaneous or average queue size. When the average queue size exceeds a certain threshold but is still less than the capacity of the queue, AQM algorithms infer congestion on the link and notify the end systems to back off by proactively dropping some of the packets arriving at a router. Alternately, instead of dropping a packet, AQM algorithms can also set a specific bit in the header of that packet and forward that packet toward the receiver after congestion has been inferred. Upon receiving that packet, the receiver in turns sets another bit in its next ACK.

Among various active queue management schemes has been proposed we consider mainly three AQM schemes namely gentle random early detection (GRED), nonlinear random early detection (NLRED) and random exponential marking (REM) with Droptail queue. Each of these schemes is discussed below.

2.1 Random Early Detection

Among various algorithm one well known AQM algorithm is random early detection (RED) [7] which has been recommended by the IETF as the default AQM scheme for routers of next generation networks (NGN) [2] , [3]. The basic idea of RED algorithm is that a router detects congestion early by computing the average queue length avg, and sets two

buffer thresholds; maximum threshold (maxth) and minimum threshold (minth) for packet drop. When a packet arrives at router, the avg is updated using an exponentially weighted moving average (EWMA) of the previous queue length, as shown in function [8]

$$avg = (1-w_q)avg^1 + w_q\, q$$

where avg^1 is the calculated average queue size when the last packet arrived, q is the instantaneous queue size, and w_q is the pre-determined weighting factor with a value between 0 and 1. As avg varies from a minimum threshold (min_{th}) to a maximum threshold (max_{th}), the packet dropping probability (p_d) increases linearly from 0 to a maximum packet dropping probability (max_p).

RED was mainly designed to overcome the two problems associated with drop-tail routers, namely, global synchronization and bias against busty sources [7]. Also maintain high link utilization, and remove biases against bursty sources. One of the fundamental problems with RED is that they rely on queue length as an estimator of congestion. Since the RED algorithm relies on queue length, it has an inherent problem in determining the severity of congestion. As a result, RED may need a wide range of parameters to operate correctly under different congestion scenarios. RED performance is highly sensitive to its parameter settings [9] [10] [11]. So it needs tuning the parameters properly so as to get better performance in RED. Such instability is due to the linear packet dropping function adopted by RED, which tends to be too aggressive at light load and not aggressive enough when the average queue size approaches the max_{th}[5]. LIRED algorithm performs better in the bursty network traffic by developing packet drop probability function based on linear interpolation method [12].

2.2 Gentle RED

In the gentle RED option[4], once the averaged queue size exceeds max_th, the drop probability does not jump to 1, but increases linearly (‘‘gently’’) to 1 as the average queue size increases to twice max_th.

2.3 Nonlinear Random Early Detection

The purpose of NLRED is to simply replace the linear packet dropping function in RED by a judiciously designed nonlinear quadratic function. The underlying idea is that, with the proposed nonlinear packet dropping function, packet dropping is gentler than RED at light traffic load but more aggressive at heavy load [5].

In RED when ‘avg’ exceeds the minimum threshold, it drops packets linearly. In same scenario, NLRED immediately adopts the nonlinear quadratic function to drop packets. However considering the same value of max_p in RED, NLRED will be gentler than RED for all traffic load [5]**.** This is because the packet dropping probability of NLRED will always be smaller than that of RED.

2.4 Random Exponential Marking

REM is an AQM schemes that measure congestion by a quantity called ‘price’. Price is computed by distributing local information to each link and feedback to the source through packet marking or dropping. REM uses different definition of congestion measure and a different marking probability function. REM has the following two key features [6].

Match rate clear buffer: attempts to match user rates to network capacity while clearing buffers (or stabilize queues around a small target), regardless of the number of users.

S**um prices:** The *end-to-end* marking (or dropping) probability observed by a user depends in a simple and precise manner on the *sum* of link prices (congestion measures), summed over all the routers in the path of the user. The marking probability of REM is higher than RED from beginning.

REM achieves both high utilization and negligible loss and delay in a simple and scalable manner by decoupling the congestion measure from performance measure such as loss, queue length or delay [6].

3. MODEL AND SIMULATION SETUP

As different algorithms have different assumptions for the network configuration and traffic pattern, the main challenges in designing our simulation is to select a typical set of network topology and parameters (link bandwidth, RTT, and router buffer size), as well as load parameters (numbers of TCP flow, packet size, TCP window size, traffic patterns) as the basis for evaluation.

For this simulation, a simple topology is created, where many persistent TCP flows share a bottleneck router with AQM schemes or DT as shown in Figure 1, which consists of N senders and N sink, connected together via two routers R1 and R2. This simple topology consider a single bottleneck link (R1 – R2) traversed by multiple TCP flows. Different level of congestion is created by varying number of flows (N) in bottleneck link. In this simulations, 20 to 120 TCP flows are running for 100 seconds from all nodes (Source 0 to Source n) to the corresponding destinations node (sink 0 to sink n). In order to analyze the performance, experiment is done using packet level ns-2 simulator [13]. The active queue management is implemented at router R1, whose queue buffer size is 100 packets. The data packet generated by sender is 512 bytes long and simulation time varies from 25 – 150 sec. Here the number of flows varies from 20 – 120 and maximum drop probability from 0.1 - 0.5. The bottleneck capacity, the round trip delay, and the number of flows vary according to the objective of the experiment. The simulation ends at 150 sec to gather sufficient information about different schemes and use the same common arrival process of sessions. In this experiment, some default values are taken for the parameters of RED and REM. The parameter values of RED are minimum threshold (min_th) =5, maximum threshold (max_th) =15 and maximum droping probability (max_p) =0.1. The parameter values of REM are phi (φ) = 1.001, alpha (α) = 0.1 and gamma (γ) = 0.001. For both RED and REM weighted queue length (q_weight) =0.002. Other parameters used in this simulation are shown in Table 1.

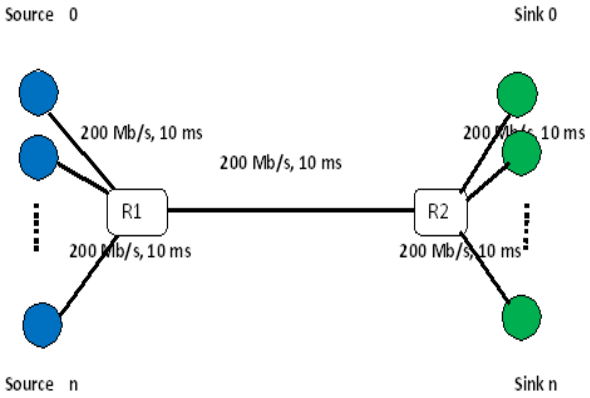

Figure 1: Dumbbell Topology

Table 1. Parameter values in simulation

PARAMETER	***VALUE/SPECIFICATION***
Channel type	Wired channel
Topology	Dumbbell
Packet size	512 bytes
Queue type	RED,REM,NLRED, DROPTAIL
Packet type	FTP
Min Threshold of RED	5
Max threshold of RED	15
q_weight	0.002
Max_p	0.1
Window size	8000
Time of simulation	100 Sec

4. RESULT ANALYSIS

Through the simulation, it has been found that NLRED provides better performance than Droptail, GRED and REM. This simulation allowed 20 to 120 number of TCP flows to pass through bottleneck link keeping simulation time 100 sec to find the performance of all mentioned queuing protocols. Figure 2, indicates that goodput for all algorithms increases as number of flow increase. But NLRED achieves slightly higher goodput than GRED, REM and Droptail. When the number of flows larger than 60 the goodput for NLRED converge to the link bandwidth. Goodput is the ratio of the total number of non-duplicate packets received at all destinations per unit time to link capacity.

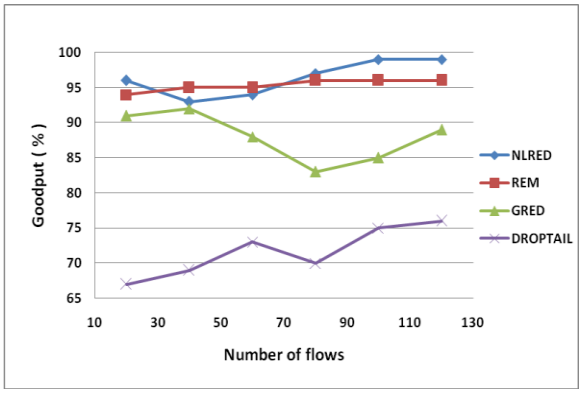

Figure 2: Goodput vs Numer of flows

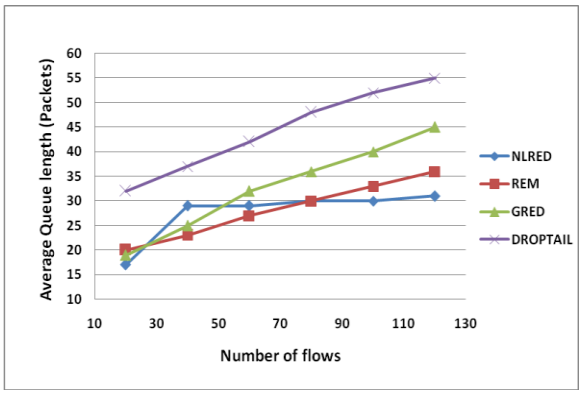

Figure 3: Average queue length vs Number of flows

Figure 3, concludes that NLRED allows the average queue size to grow at a faster rate when the number of flows is small (i.e. 50). Then after, as the number of flows increases, NLRED tends to control the average queue size better than GRED, REM and Droptail. The queuing delay decrease for all the schemes as number of flows increases (see Figure 4).

Loss rate is the ratio of the total number of packets dropped to the total number of packets sent. The packet loss rate for all protocols increases with increase of flows. But NLRED's packet loss falls below 10% for 120 numbers of flows depicted in Figure 5.

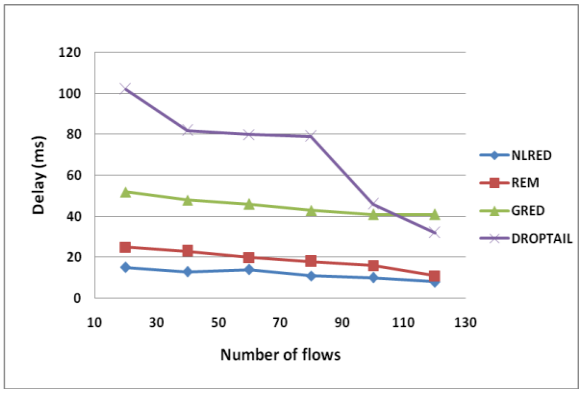

Figure 4: Delay vs Number of flows

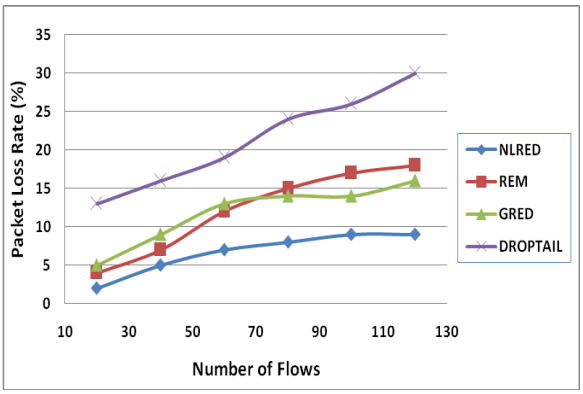

Figure 5: Packet loss rate vs Number of flows

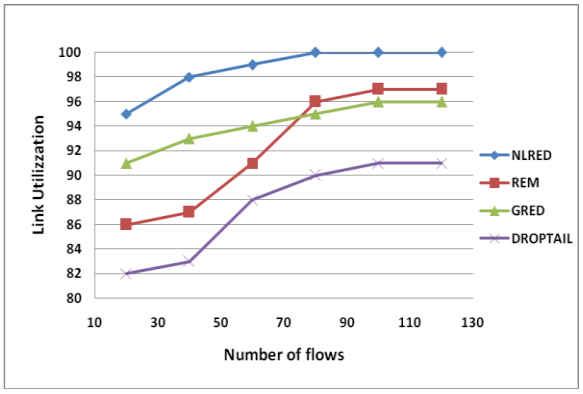

Figure 6: Link utilization vs Number of flows.

With more than 60 flows the loss rate seems to increase more than linearly with number of flows. As Figure 6 says when the number of flows is nearer hundreds, NLRED schemes yield full link utilization and GRED, REM and Droptail achieves more than 90% link utilization. In order to show the ability to maintain equal bandwidth between flows, in this Jain's Fairness Index is used. Hence a higher fairness index indicates better fairness between flows.

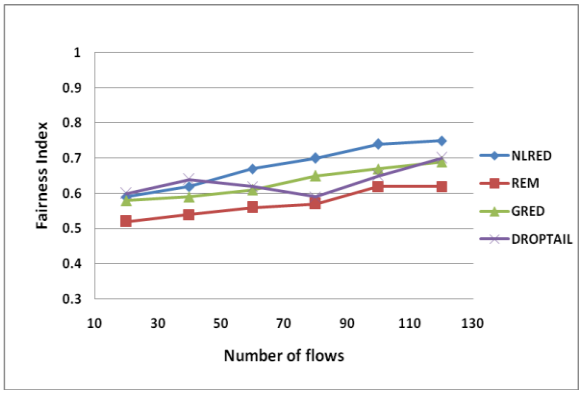

Figure 7: Jain's Fiarness index vs Number of flows.

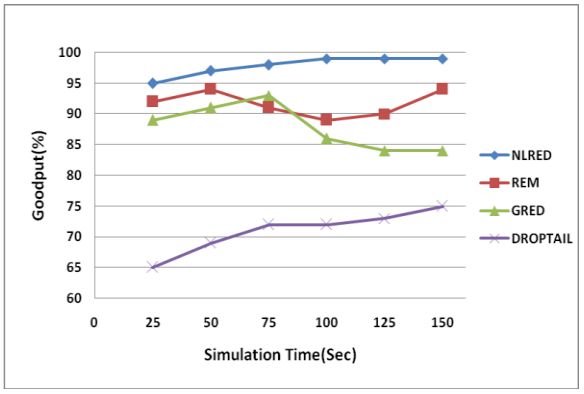

Figure 8: Goodput vs Simulation time.

As Figure 7 suggests that Fairness Index increase for all schemes with number of flows increases but NLRED able to achieve fairness significantly better than GRED, REM and Droptail because it queues two packets of flows before marking a packet from that flow. GRED is able to outperform REM and Droptail. Figure 8 and Figure 9 indicates that NLRED achieves higher goodput and few packet losses than REM, GRED and Droptail.

To illustrate the effect of maximum dropping probability (max_p), fixed number of simulation time(100 sec) has been taken. In order to compare GRED and NLRED, we varied max_p parameters values to 0.1, 0.3, and 0.5 and kept same set of parameters, q_weight =0.002, min_th =5, and max_th=15. As Figure 10 suggest that NLRED is less sensitive to the choice of max_p having different number of flows. NLRED have higher goodput as compared to GRED for max_p (p) values 0.1, 0.3 and 0.5 as shown in fig. 10. This happens only because NLRED uses non linear quadratic packet dropping function. As max_p increases, the goodput of NLRED and GRED schemes decreases.

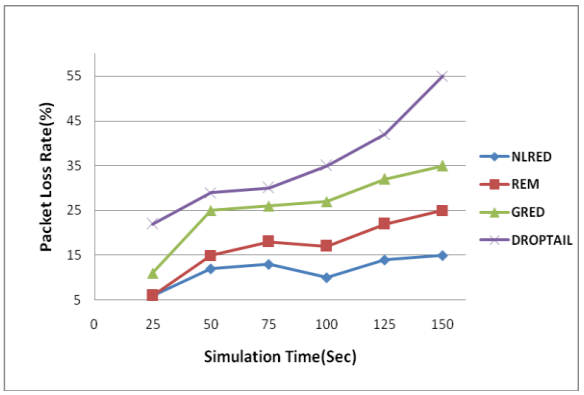

Figure 9: Packet loss rate vs Simulation time.

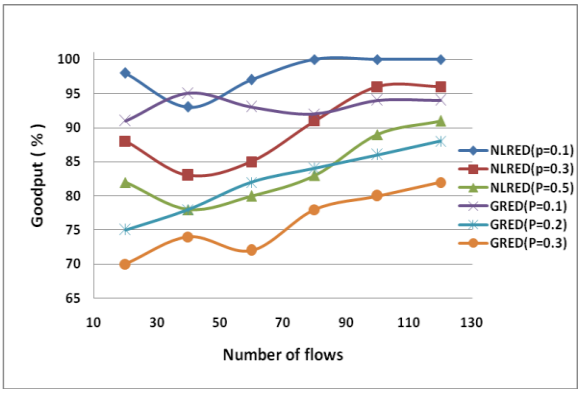

Figure 10: Goodput vs Number of flows.

5. CONCLUSIONS

This paper analyzes the performance of several active queue management techniques namely NLRED, GRED and REM with Droptail based on simulation results. Result shows that, NLRED performs better than GRED, REM and Droptail because it maintains good link utilization (always higher than 90%) and small queue size. Also it maintains low delay, high goodput and low packet loss ratio than others. NLRED achieves fairness significantly better than GRED, REM and Droptail between flows. Also found, NLRED is comparatively less sensitive to parameter like max_p. In general all variants of RED and REM performed better than Droptail. But NLRED Performs better than all these protocols.

6. REFERENCES

[1] V. Jacobson, Congestion avoidance and control, Computer Communication Review 18 (4) (1988) 314–329.

[2] L. Zhang, S. Shenker, D.D. Clark, Observations on the dynamics of a congestion control algorithm: the effects of two-way traffic, in: Proceeding of ACM SIGCOMM '91, the Conference on Communications Architecture and Protocols, Zurich, Switzerland, September 03–06, 1991, pp. 133–147.

[3] B. Braden, D. Clark, J. Crowcroft, et al., Recommendations on queue management and congestion avoidance in the Internet, in: RFC2309, April 1998.

[4] S. Floyd, Recommendation on using the ''gentle_variant of RED,'' March 2000. Available from: <http://www.icir.org/floyd/red/gentle.html>.

[5] Kaiyu Zhou, Kwan L. Yeung, Victor O. K. Li, "Nonlinear RED: A simple yet efficient active queue management scheme", Elsevier Journal of Computer Networks, 50, (2006), pp. 3784-3794.

[6] S. Athuraliya, S.H. Low, V.H. Li, et al., REM: active queue management, IEEE Network 15 (2001) 48–53.

[7] S. Floyd, V. Jacobson, "Random early detection gateways for congestion avoidance", IEEE/ACM Transactions on Networking1, (1993), pp. 397-413.

[8] D. Que, Z. Chen, B. Chen, "An improvement algorithm based on RED and its performance analysis", 9th Int. Conf. on Signal Processing, Oct. 2008, pp. 2006- 2008.

[9] S. Floyd, R. Gummadi, S. Shenker, Adaptive RED: An algorithm for increasing the robustness of RED's active queue management, [i] 2001. Available from: <http://www. icir.org/floyd/papers/adaptiveRed.pdf>.

[10] S. Kunniyur, R. Srikant, Analysis and design of an adaptive virtual queue (AVQ) algorithm for active queue management, ACM SIGCOMM Computer Communication Review 31 (2001) pp. 123–134.

[11] L. Zhu, N. Ansari, Local stability of a new adaptive queue management (AQM) scheme, IEEE Communications Letters 8 (2004) pp. 406–408.

[12] Abbasov, B, "Using linear interpolation method for packet drop probability function of RED algorithm", 5th International Conference on Application of Information and communication Technologies, Oct. 2011, pp 1-4.

[13] NS-2, The ns Manual (formally known as NS Documentation) available at http: //www. isi.edu/nsnam/ns/doc.