2012 International Conference on Information and Network Technology (ICINT 2012)
IPCSIT vol. 37 (2012) © (2012) IACSIT Press, Singapore

Analyzing Marking Mod RED Active Queue Management Scheme on TCP Applications

G.A. Ramachandra [1], Reshma Banu [2] and G.F.Ali Ahammed [3 +]

[1] Department of computer science & Technology, S.K. Universty, Anantapur. A.P, India
[2] Department of Information Science & Engg, Ghousia college of Engg. Ramanagaram
[3] Department of Electronics & Communication, Ghousia college of Engg. Ramanagaram

Abstract. We propose modified random early detection(ModRED) active Queue Management Scheme for congestion control in TCP/IP networks. Our AQM aims at providing better control over the burstiness level while remaining the advantages of Basic RED. We use ns2 to simulate a series of network configurations and the numerical results demonstrate that Our AQM can achieve higher link utilization ,less loss rate& high Throughput compared to RED without sacrificing its advantages in all simulation scenarios. The improvement becomes more significant under bursty traffic.

Keywords: active queue management (AQM), congestion control, random early detection (RED), TCP.

1. Introduction

When there are too many coming packets contending for the limited shared resources, such as the queue buffer in the router and the outgoing bandwidth, congestion may happen in the data communication. During congestion, large amounts of packet experience delay or even be dropped due to the queue overflow. Severe congestion problems result in degradation of the throughput and large packet loss rate. Congestion will also decrease efficiency and reliability of the whole network, furthermore, if at very high traffic, performance collapses completely and almost no packets are delivered.

Traffic on the Internet tends to fluctuate and to be greedy. Ideally, a router queue management algorithm should allow temporary bursty traffic, and penalize flows that persistently overuse bandwidth. Also, the algorithm should prevent high delay by restricting the queue length, avoid underutilization by allowing temporary queueing, and allocate resource fairly among different types of traffic [1]. In practice, most of the routers being deployed use simplistic Drop Tail algorithm, which is simple to implement with minimal computation overhead, but provides unsatisfactory performance.

Active Queue Management (AQM) policies attempt to estimate the congestion at a node and signal by dropping packet(s) before the buffer is full. A responsive congestion control strategy then reduces its transmission rate. This helps in avoiding further congestion and is expected to reduce the packet loss rate and keep the average queue size low. When packets are dropped aggressively, the capacity of the node may remain underutilized. An AQM policy thus has two components; one component estimates the congestion and another component takes the packet drop decision. The performance, therefore, depends upon how aggressive or conservative the estimation of the congestion is and also on how aggressively the packets are dropped based on this estimate.

Active Queue Management (AQM)[2] aims to detect congestion in the network before it becomes severe by overfilling the router queue. It means that the router tries to reduce the sending rate of the traffic sources

+ Corresponding author. Tel.: + 919886191379.
E-mail address: ali_ahammed@rediffmail.com.

by dropping or marking packets. There exist two approaches to indicate congestion: Packets can be dropped and packets can be marked. First strategy requires cooperation of the endpoints and latter generates additional overhead through re-sending. Endpoints have to react on marked packets as they have been dropped and decrease their throughput. With this the same improvement of bandwidth utilization can be achieved, but without additional overhead costs. Additionally some AQM mechanisms aim to reduce the bandwidth of greedy flows by dropping their packets at higher rates. In this chapter we give a survey on Active Queue Management algorithms that are suitable for Peer-to-Peer networks.

2. Classification of AQM Schemes

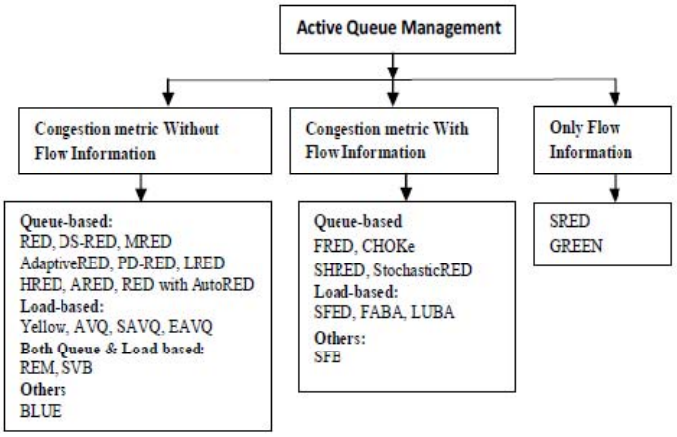

Fig. 1: Classification of AQM

2.1. Queue-based AQM

RED: The first well known AQM scheme proposed is RED[2]. It is one of the popular algorithms. It tries to avoid problems like global synchronization, lock-out, bursty drops and queuing delay that exists in the traditional passive queue management i.e Droptail scheme.

The main idea of RED is to observe the average length of the queue to detect congestion. If the congestion is coming, the congestion information is sent to any source randomly so that the rate of the source is reduced before the queue is full. There are two steps in RED. Firstly, compute the average queue length. Secondly, compute the drop probability of the packet according to first step. RED uses the weighted method like lowpass filter to calculate the average queue length avg Q

$$\text{avg} \quad (1 - \text{wq})\text{avg} + \text{wqq} \tag{1}$$

2.2. Load-based AQM

AVQ: The virtual queue is updated [8], when a packet arrives at the real queue to indicate the new arrival of the packet. As in Fig 1 when the virtual queue or buffer overflows, the packets are marked / dropped. The virtual capacity of the link is modified such that total flow entering each link achieves a desired utilization of the link.

This is done by aggressive marking when the link utilization exceeds the desired utilization and less aggressive when the link utilization is below the desired utilization. As a result this provides early feedback than the RED.

2.3. Queue and load-based AQM

REM: Random Exponential Marking (REM) achieves high utilization with negligible loss or queuing delay even as the load increases. This scheme stabilizes both the input rate around link capacity and the queue around a small target independent of the number of users sharing the link. REM[2] maintains a variable called price as a congestion measure. This variable is used to determine the marking probability and is updated periodically based on rate mismatch and queue mismatch . REM uses an exponential curve instead of a liner curve to determine the packet marking probability against price. Even though REM considers the

queue size increasing rate indirectly in the process of marking probability calculations, it ignores the RTT in the marking probability calculation.

3. ModRED AQM Algorithm

The main idea of the approach is to restrict the TCP transmission window with the flow control window instead of the congestion control window, thus controlling the transmission window with a finer granularity. Results from simulations show that the our algorithm modified queues improve on the average queue size, one-way packet delay, delay jitter, number of packet drops and throughput as compared to RED queues, especially in paths that have non- ECN compliant routers. Moreover, Modified RED (ModRED) does not require modification to TCP implementations at servers or clients Mod RED is based on the two-drop precedence policy. A packet is marked at the edge of the network as IN or OUT of its service contract and it is treated differently inside the network, on the basis of this priority classification. The router inside the network keeps just one queue for IN and OUT packets and apply to them two different RED algorithms as we can see in Figure 2. Instead of using the same average queue size for both priorities, it uses the average queue size for OUT (out of profile) packets, and the average queue size without taking into account the queued OUT packets for IN (in profile) packets. In time of congestion the router starts to drop OUT packets and eventually, if congestion persists, will start to discard IN packets, as well.

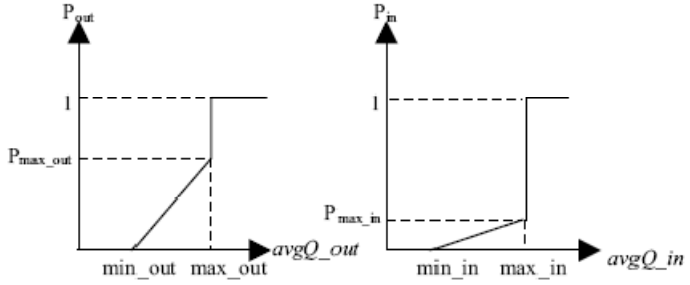

Fig. 2: The marking/dropping behavior of ModRED

We are proposing to use flow control feedback to reduce congestion at routers. In this thesis, we will deal with congestion occurring at the ingress and gateway routers two major congestion areas within the network. Our algorithm sets the window field to one maximum segment size (MSS) in the ACK packets 1 that are going towards the sender from the receiver, instead of dropping or marking the packets at the queue. (The only exception that it will not modify the receiver window field is when the field has a value of 0. This occurs when a TCP application wants to tell its peer not to send any more data). We denote our algorithm as modification to RED schemes by affixing “mod(M)” before the name of the RED. In our algorithm the field is set to 1 only if the average queue length is between *Thmin* and *Thmax*, where *Thmin* and *Thmax*, are the minimum and maximum threshold values respectively. However, if the average queue length is greater than *Thmax,* the packet is dropped

4. Performance Evalution

To evaluate the performance of ModRED, we use Network Simulator ns2 to simulate a series of scenarios. Simulation results demonstrate that in all scenarios performance improvements can be achieved by ModRED in terms of link utilization/node throughput without sacrificing other advantages of RED, such as congestion avoidance as opposed to congestion reaction, avoiding global synchronization of source backoff, etc. This performance gain depends on the bursty nature of traffic at the gateway. Significant higher link utilization and/or node throughput is found for some cases of bursty traffic at the Router

A Dumbbell network topology is chosen to make it easier to understand the congestion network environment. As shown in Figure 3, there are n connections in the network n is a variable parameter that means how many connections share the bottleneck link. We choose n within 16, 32 & 64. The larger is the number of connections, the worse is the congestion in the bottleneck. The bottleneck link's bandwidth is 10Mbps, and the n connections' output are all 10Mbps. The data transfer time (Tp) is 150ms.

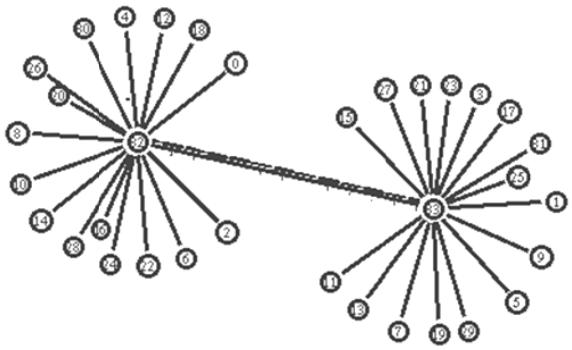

Fig. 3: Simulation model of for 16 node Network

Table. 1: Mod RED Simulation Parameters

Bottleneck link bandwidth	10 [Mbit/s]
Propagation delay of the bottleneck link	150 [ms]
Packet Size	1,000 [byte]
Buffer Size	100 [packet]

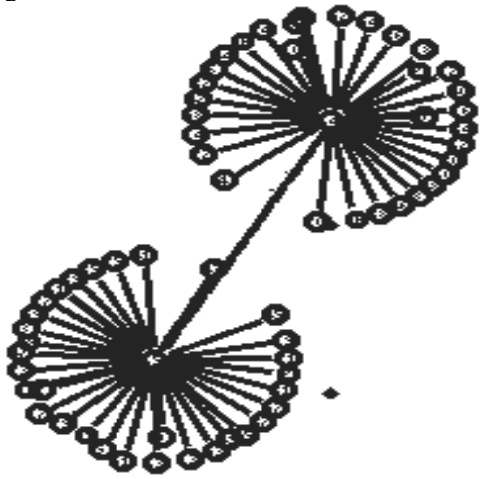

Fig. 4: Simulation model of for 32 node Network

Table. 2: LossRate with !0sec Simulation Time

Nodes	Load	Loss Rate with 10 sec simulation Time								
		Queue Size =25			Queue Size =100			Queue Size =250		
		Drop Tail	RED	Mod RED	Drop Tail	RED	Mod RED	Drop Tail	RED	Mod RED
16	25%	0.36	0.57	0.39	0.0	0.64	0.0	0.0	0.64	0.0
	50%	0.93	1.27	0.85	0.42	1.56	0.44	0.1	1.62	0.1
	75%	1.36	1.87	1.36	0.66	2.23	0.64	0.1	2.82	0.1
	100%	1.93	2.43	1.9	1.38	2.90	1.36	0.1	3.24	0.1
32	25%	0.93	1.22	0.85	0.42	1.52	0.44	0.0	1.64	0
	50%	1.89	2.38	1.87	1.35	2.88	1.36	0.0	3.18	0
	75%	2.82	3.65	2.77	1.50	4.01	1.49	0.0	4.21	0
	100%	3.64	4.2	3.51	2.01	4.82	2.06	0.1	5.07	0.1
64	25%	1.89	2.38	1.09	1.35	2.88	0.40	0.06	2.17	0.04
	50%	3.64	4.28	3.11	2.01	4.82	0.88	0.12	4.02	0.21
	75%	4.74	5.70	4.11	2.87	5.15	1.40	1.45	4.57	0.57
	100%	6.23	6.60	4.57	3.58	5.21	2.15	1.65	5.06	0.78

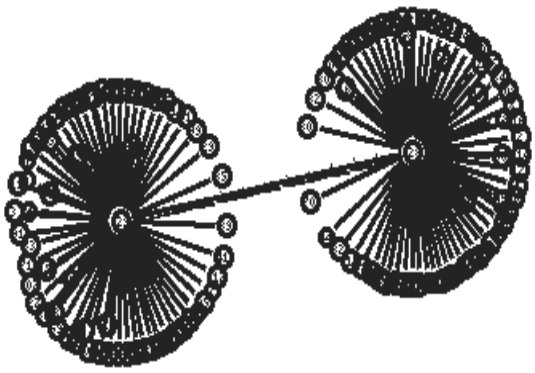

Fig. 5: Simulation model of for 64 node Network

Table. 3: Throughput with 10sec Simulation Time

Nodes	Load	Through put for 10 sec simulation Time								
		Queue Size =25			Queue Size =100			Queue Size =250		
		Drop Tail	RED	Mod RED	Drop Tail	RED	Mod RED	Drop Tail	RED	Mod RED
16	25%	99	98.8	98.9	98.8	99.01	99.6	98.8	99.01	99.0
	50%	98.3	98.2	98.4	97.9	98.07	97.9	97.7	97.9	97.7
	75%	98.2	97.63	98.2	97.6	97.33	97.6	97.3	96.6	97.3
	100%	97.3	97.01	97.4	97.0	96.66	96..9	97.3	96.2	97.3
32	25%	98.3	98.29	98.4	97.9	96.04	97.9	97.3	99.6	97.3
	50%	97.4	97.17	97.4	97.0	96.6	96.9	97.3	96.1	97.3
	75%	96.6	96.26	96.56	97.5	95.4	97.5	97.0	95.2	97.0
	100%	95.7	95.22	95.83	96.4	94.7	96.8	96.2	94.4	96.2
64	25%	97.4	97.1	98.8	97.0	96.6	99.5	97.3	97.7	99.7
	50%	95.7	95.2	96.8	96.4	94.7	99.0	96.2	95.9	99.5
	75%	94.6	93.7	95.8	95.5	83.7	98.5	95.7	95.3	99.2
	100%	93.1	92.9	95.3	95.0	94.7	97.8	95.2	94.8	99.0

The Drop tail and RED, Mod RED AQMs were tested under different traffic loads, by varying the number of active saturated TCP sessions. This section describes the NS-2 simulation details associated with our comparison study of the performance of RED ModRED. The NS-2 simulator provides the ability to simulate drop-tail, RED and ModRED routers. With different queue size 25,100 and 250 packets, different loads like 25% ,50%,75 % and 100% ,where 25% load means we have divided the number nods with 4,if 50% load means we have divided the number nods with 2. Our simulation has variable simulation time and, NS-2 includes code to simulate TCP protocols. Mod RED- was simulated by extending the RED implementation.

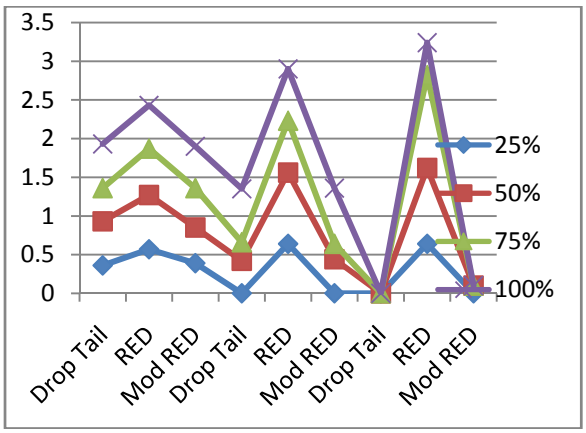

Fig. 6: Loss Rate with 10 sec simulation Time with 16 Nodes

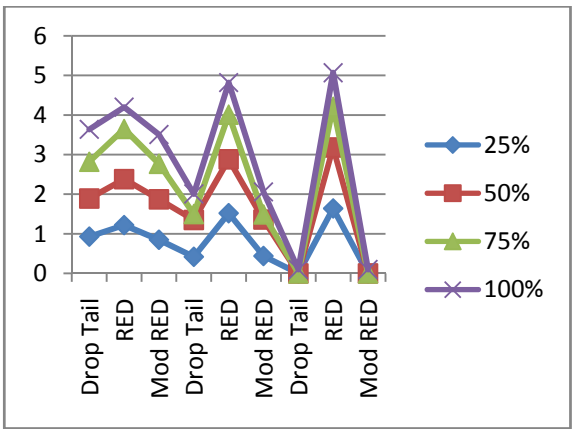

Fig. 7: Loss Rate with 10 sec simulation Time with 32 Nodes

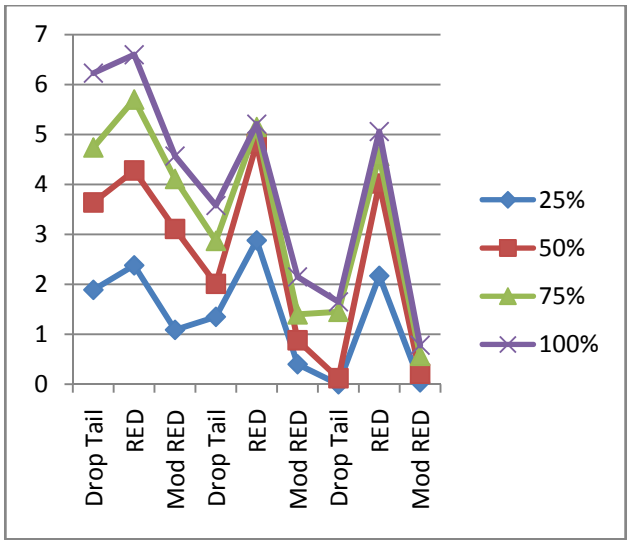

Fig. 8: Loss Rate with 10 sec simulation Time with 64Nodes

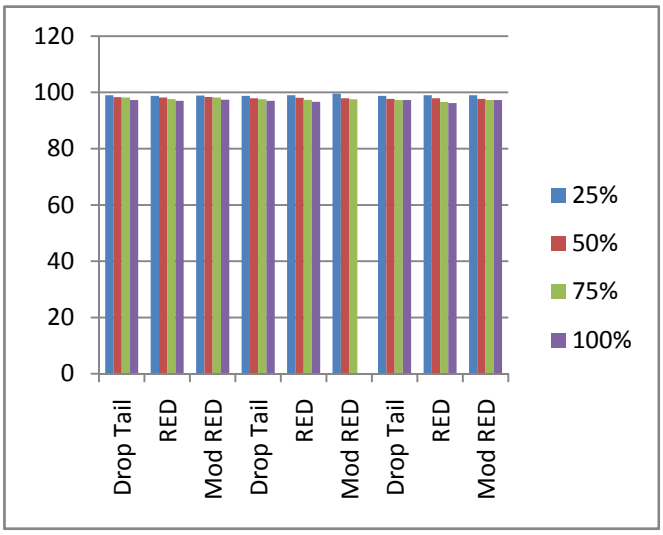

Fig. 9: Throughput with 10 sec simulation Time with 16 Nodes

Figures 6 to 11 shows the loss rate and Throughput of Simulation results of drop-tail, RED and Mod RED queues of 25, 100, and 250 packets at 25%, 50%, 75%, and 100% load. As the queue length was increased from 25 to 100 and to 250 packets, the packet loss rate was reduced from 0.36 % to 0.0%. A packet loss rate of 0.0% is an indication that the queue never overflowed in the experiment. This result implies that a queue of 250 packets is over provisioned for 50% offered load. These results show the trade-off between small queues (25 and 100 packets) and large queues (250 packets). A small drop-tail queue obtains small queuing delay but incurs a high packet loss rate (for example, the packet loss rate was 3.58 % for a queue of 100 packets but only 1.6 % for a queue of 250 packets at 75 % offered load at 64 node network). Because of this, flows that do not experience any packet loss enjoy good response times with a small drop-tail queue, especially those that are small. On the other hand, a large drop-tail queue can reduce the packet loss rate but subjects flows to more queuing delay.

Hence, for large flows that dominate the links (in terms of number of packets) and are likely to experience some packet losses (assuming that packets are dropped randomly and uniformly), the impact of increased queuing delay is out weighted by the effects of reduced packet losses. Thus, large flows receive better performance under a large drop-tail queue. Overall, the drop-tail queue with 100 packets is chosen as a baseline for comparison with AQM algorithms because it appears to achieve a good trade-off for drop-tail queues between improving response times for a large number of small objects and a small number of large objects.

The performance of ModRED at 25% load was undistinguishable from that of the uncongested network and of drop-tail with a queue length of 100 packets. At 50% offered load, ModRED delivered equally good performance with both queue reference values of 25 and 100 packets and the performance for ModRED

came very close to the performance of the congested network. As the offered load increased to 75% and 100%, ModRED obtain significant performance improvement over packet dropping.

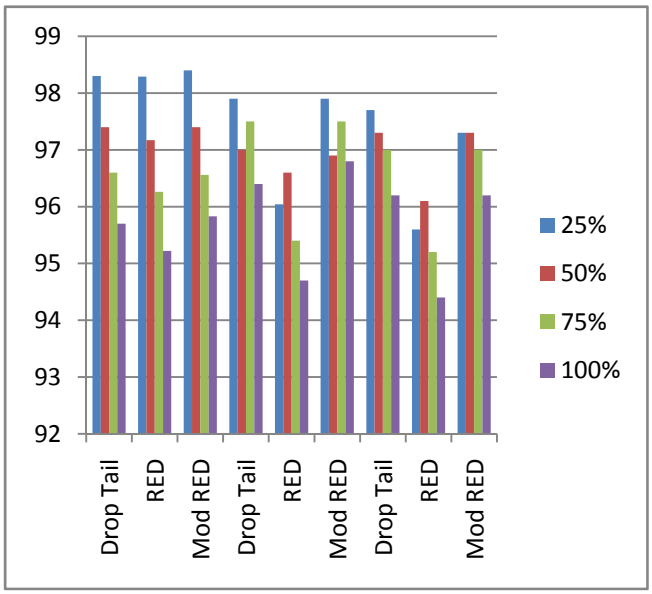

Fig. 10: Throughput with 10 sec simulation Time with 32 Nodes

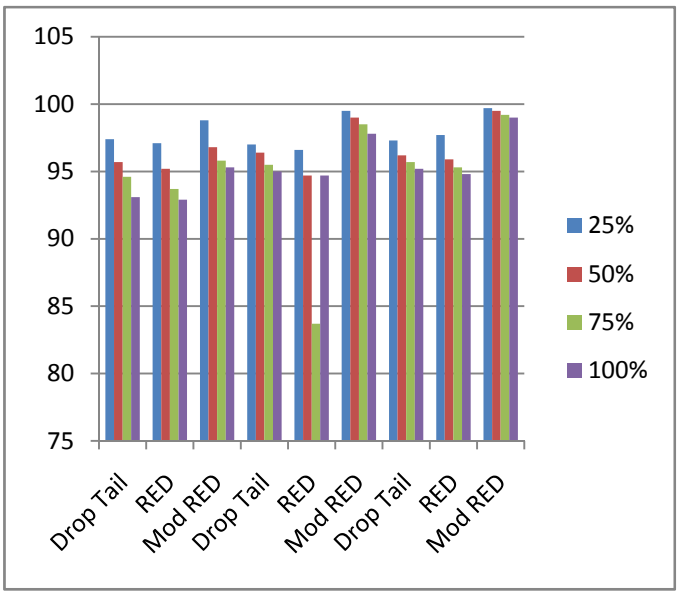

Fig. 11: Throughput with 10 sec simulation Time with 64 Nodes

5. Conclusion

In this paper simulation results with different traffic and a uniform RTT distribution were presented for Drop tail, RED and Mod RED of AQM algorithms that have been proposed recently in research literature.. we have proposed a modification to existing RED algorithm called Mod RED does not require modification to all end system TCP/IP stacks but can be solely implemented in routers (especially for ingress and gateway routers). This means that our algorithm does not require both the sources and receivers to be "compliant" as was the case for ECN-compliant queues. Our scheme helps to reduce the average queue sizes of the RED queues. By reducing the average queue sizes, queues reduce the queuing delay resulting in significant improvements in one-way end-to-end packet delays and dropped packets. For large flows that dominate the links (in terms of number of packets) and are likely to experience some packet losses (assuming that packets are dropped randomly and uniformly), the impact of increased queuing delay is out weighted by the effects of reduced packet losses.

In contrast to existing AQM algorithms, ModRED provides a framework for balancing loss rate and queuing delay. ModRED is flexible enough to absorb transient bursts where the input rate temporarily exceeds the link capacity. On the other hand, ModRED can control and stabilize router's queue when persistent congestion occurs. Mod RED gave significant performance improvement over the all other RED & AQM algorithm.

6. References

[1] Xiaoming Li," Performance Comparison of BQM and Normal Queue Management Techniques",ACMSE'06 March 10-12,2006 Melbourne,florida,USA.pp350-354.

[2] R G.Thiruchelvi , J.Raja. "A Survey On Active Queue Management Mechanisms". In IJCSNS,Vol.8 No.12,Dec.2008.pp.130-145.

[3] NS-2 Simulator, available at http://www.isi.edu/nsnam/ns/

[4] Sally Floyd. Recommendation on using the "gentle" variant of RED http://www.icir.org/floyd/red/gentle.html, March 2000.

[5] Long Le, Kevin Jeffay, and F. Donelson Smith. A loss and queuing delay controller for buffer management. In Proceedings of the IEEE International Conference on Distributed Computing Systems (ICDCS), July 2006

[6] Alberto Medina, Mark Allman, and Sally Floyd. Measuring the evolution of transport protocols in the Internet. ACM Computer Communication Review, 35(2), April 2005.

[7] K.Chitra, G.Padamavathi, “Classification and Performance of AQM-Based Schemes for Congestion Avoidance “ , IJSCIS, Vol. 8 No. 1, April 2010

[8] S. Kunniyur and R. Srikant, Analysis and Design of an AdaptiveVirtual Queue (AVQ) Algorithm for Active Queue Management,Proc., ACM SIGCOMM 2001, pp. 123-134.

[9] Hari Balakrishnan, Venkata Padmanabhan, SrinivasanSeshan, and Randy Katz. \Eectiveness of loss labelingin improving TCP performance in wired/wireless networks. " In Proceedings of ICNP'2002: The 10thIEEE International Conference on Network Protocols, , Paris, France, November 2002.

[10] Rong Pan, Lee Breslau, Balaji Prabhakar, and Scott Shenker. Approximate fairness through differential dropping. ACM Computer Communication Review, 33(2):23–39, April 2003

[11] Alberto Medina, Mark Allman, and Sally Floyd. Measuring the evolution of transport protocols in the Internet. ACM Computer Communication Review, 35(2),April 2005.

[12] Sanjeewa Athuraliya, Victor H. Li, Steven H. Low, and Qinghe Yin. REM: Active queue management. IEEE Network, 15(3):48–53, May 2001.

[13] G.F.Ali Ahammed and Reshma Banu ”Analyzing the Performance of Active Queue Management Algorithems ’’ in (IJCNC) International Journal of Computer Networks & Communications,Vol2,No2,March 2010.