Performance Evaluation of Various Active Queue Management for Bufferbloat

Kanu Monga, Krishan Kumar

Abstract - Due to unprotected large buffers in network devices, the Internet is suffering from high latency and jitter which leads to decreased throughput. The perseveringly full buffer problem, recently exposed as "bufferbloat" [1] [2] has been observed for decades, but is still with us. As a solution to this problem, several new AQM algorithms CoDel, sfqCoDel and CoDel-DT have been proposed. This paper aims to evaluate these AQM algorithms by carrying out simulations in ns-2 and compares their performance with that of DropTail. sfqCoDel outperforms various peer solutions in variety of scenarios in terms of bottleneck link utilization, packet drop rate and mean queue length.

Keywords- AQM, RED, CoDel, Droptail, Bufferbloat

I. INTRODUCTION

Over the past few years, the usage of Internet has grown to phenomenal levels and this has led to heavy congestion in routers. Due to the rapid price drop of memory, buffering has been overdone in many network devices which results in problem called bufferbloat. The reason is that TCP congestion control algorithms [1] depends on packet loss to predict congestion but if the buffers are very large, packets sit in the buffer queues instead of being dropped, and congestion signal does not reach to the endpoints in a timely manner; meaning that they do not slow down, and the buffers remain full. Moreover, every new packet that gets in the buffer has to wait for all packets queued before it to be transmitted before it can go on its way. These two effects join to create what is referred to as bufferbloat [5][6]. It can cause many applications to time out, and users to experience huge delays.

As a solution to this problem, several AQM mechanisms have been proposed. AQM limits the buffers from growing by either dropping or by marking a packet i.e. as the queue approaches its threshold, it alerts TCP sender so that sender can reduce the speed at which packets are sent. The necessity for AQM algorithms like RED [2], ARED, BLUE etc. has been evident from decades [4] but none has been widely deployed due to implementation difficulties. Recently, a new AQM mechanism called Controlled Delay (CoDel) [11] has been proposed to overcome the shortcomings of existing AQMs. Unlike RED [7], CoDel is parameterless AQM mechanism that adapts to varying link rates and can be easily deployed [11]. It uses packet sojourn time to predict congestion. It is one of the most simple and efficient AQM algorithm. It helps to fix BufferBloat problem.

In this paper, an experimental study is carried out to evaluate the performance of CoDel and its variants in a wide range of Internet scenarios. The performance of same is also compared to that of DropTail [3]. The evaluation parameters of this study are bottleneck link utilization, mean queue length at the bottleneck router and overall packet drop rate. Rest of the paper is classified as follows: Section II briefly explains the working of these AQM algorithms. Section III presents the details about simulation scenarios, performance evaluation metrics etc. Section IV demonstrates the results and based on the comparative study, inferences have been deduced in Section V. Finally, Section VI concludes the paper.

II. CONTROLLED DELAY (CODEL) ALGORITHM

A. Overview

Controlling Delay (CoDel) is a recently proposed Active Queue Management (AQM) scheme developed by Van Jacobson and Kathleen Nichols [11] to overcome the Bufferbloat problem [12] in the Internet. Unlike RED,ARED and its variants [8] [9] [10], CoDel works on the the basis of minimum time that packets spend in the queue. CoDel detects the standing queue and once sojourn time i.e. time spent by packets in queue remain above predefined threshold (called target) over an entire interval, it sends a congestion signal to TCP by marking/dropping packets. CoDel requires minimal tuning and can be implemented efficiently on all networks [13] [14]. It is not based on link utilization, queue size, round trip delay, queue size thresholds etc.

B. Algorithm

CoDel works by adding a timestamp to each packet as it arrives. When the packet reaches the head of the queue, the time spent in the queue is calculated, if the time spent by a packet within the queue is higher than a defined threshold(target) for the entire interval, the algorithm sets a timer to drop a packet at dequeue. This dropping is only done when queue holds at least one MTU's worth of bytes.
Constant used in CoDel algorithm [13]:

- target = acceptable standing queue delay (constant 5ms)
- interval = time on order of worst case RTT of connections through the bottleneck (constant between 10ms to 1sec).

Revised Version Manuscript Received on April 30, 2016.

Kanu Monga, Department of Computer Science and Engineering, Shaheed Bhagat Singh State Technical Campus, Ferozepur (Punjab). India.
Krishan Kumar, Associate Professor, Shaheed Bhagat Singh State Technical Campus, Ferozepur (Punjab). India.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication Pvt. Ltd.

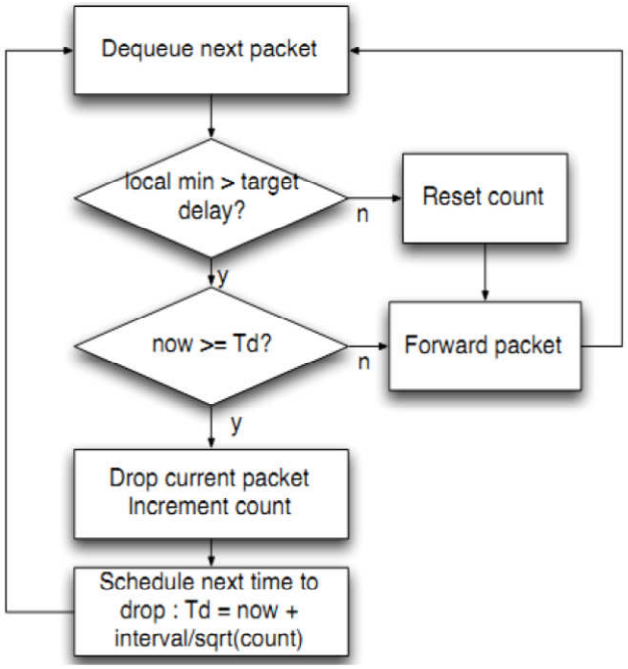

Fig 1: CoDel Algorithm

C. The Stochastic Fair Queue CoDel (sfqCoDel)

The sfqCoDel [15] is a variation of CoDel which drops packets intelligently by proactively dropping the one which occupies extra bandwidth comparing to the others, thus ensuring fair consumption of bandwidth by each packet. The goal of fair queuing is to give each distinct user of the network, a fair share of available bandwidth. Instead of keeping track of all active flows and their share of bandwidth, flows are hashed into a number of buckets, each of which has its own queue. These queues are offered in a round-robin fashion when packets are dequeued [16].

D. CoDel-DT

CoDel-DT [17] is a altered version of the "Controlled Delay" (CoDel) AQM. CoDel-DT utilizes a delay prediction at enqueue time and drop-decision is taken at enqueue. If the prediction of delay is sufficiently accurate, the performance can be shown similar to CoDel. CoDel-DT is more easily implementable in certain situations.

III. SIMULATION SCENARIO

A. Simulation Setup

Network simulator ns2 [18] is used to run the simulations in order to evaluate the performance of AQM techniques. New queue objects like CoDel, sfqCoDel, CoDel-DT are added to ns2.

A single bottleneck dumbbell topology with two-way traffic is designed. In this scenario there are two routers that are n1 and n2 which are connected with bottleneck having a simplex link, which has a capacity of 20Mb and delay of 4ms. The bottleneck bandwidth is set to 10Mbps with bottleneck round trip delay set to 32ms. Size of bottleneck buffer is 8xBDP. The traffic consists of five forward-FTP flows, five reverse-FTP flows, five HTTP flows generated using PackMime generator, five audio flows, five forward-streaming flows and five reverse-streaming flows.

B. Metrics

The major parameters considered for this study are: link utilization of bottleneck link, queue size of the bottleneck router and packet drop rate. TCP Reno is used in this analysis.

IV. RESULTS AND ANALYSIS

A. Varying Bottleneck Bandwidth

The bottleneck bandwidth is varied between 1Mbps to 1Gbps, queue size is set to 8xBDP and RTT fixed to 32ms. Fig. 2, Fig. 3 and Fig. 4 show the results for bottleneck link utilization, mean queue length at the bottleneck router and packet drop rate at the bottleneck queue respectively. Favourable characteristic of an ideal AQM is the one having higher link utilization, least number of packet drops and minimal queue size. sfqCoDel AQM exhibits the aforesaid characteristics as compared to others.

Fig. 2 shows that there is degradation in bottleneck link utilization for all the queue mechanisms as the bottleneck bandwidth is increased. This is the expected behaviour of queue mechanisms.

But with sfqCoDel, link utilization remains fairly better. Increase in bottleneck bandwidth, enables the router to empty its queue more quickly by forwarding the packets through the bottleneck link. Hence the mean queue length is expected to reduce at higher bottleneck bandwidths. From Fig. 3 it is apparent that sfqCoDel shows a tremendous improvement in mean queue length at lower bottleneck bandwidths. Initially, DropTail shows an increase in mean queue length but as the bottleneck bandwidth increases, DropTail also shows reduction in mean queue length. Packet drop rate is expected to reduce at higher bottleneck bandwidth. Fig. 4 shows that packet drop rate is almost similar for all the queue mechanisms with a slight improvement in case of CoDel and sfqCoDel.

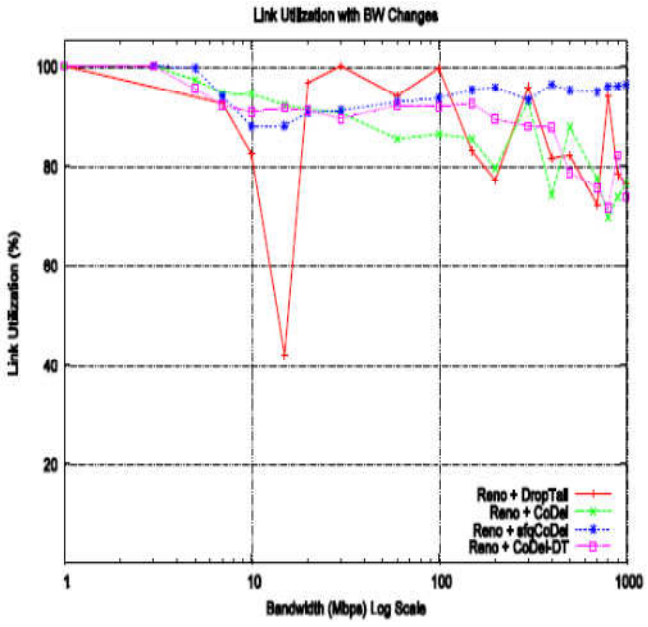

Fig.2: Bottleneck Link utilization by varying bottleneck bandwidth in TCP Reno

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication Pvt. Ltd.

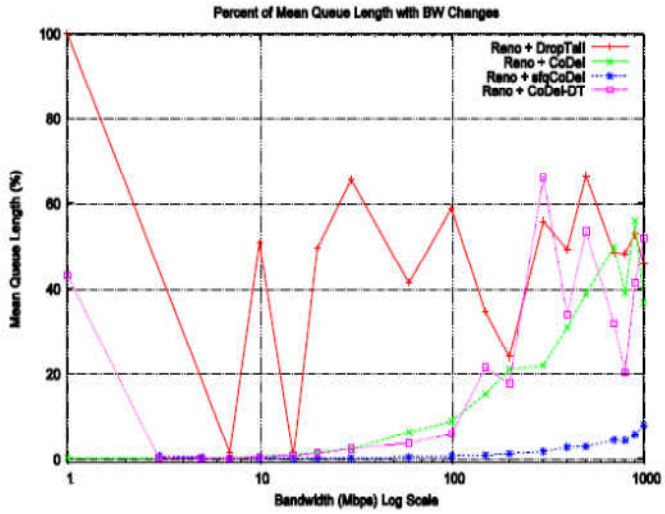

Fig. 3: Mean Queue length by varying bottleneck bandwidth in TCP Reno

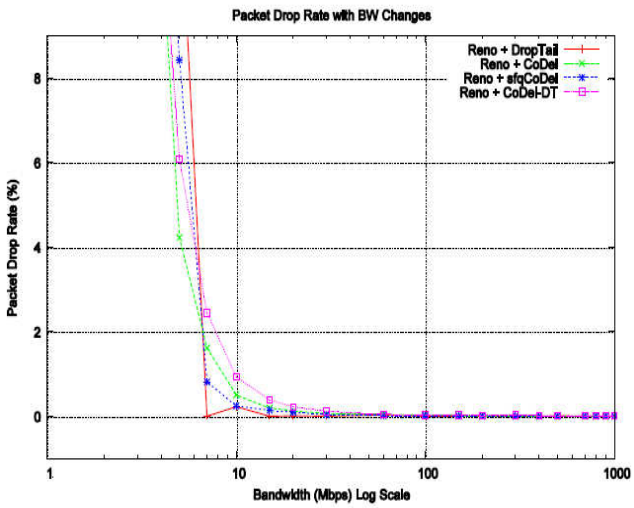

Fig. 4: Packet Drop Rate by varying bottleneck bandwidth in TCP Reno

B. Varying the number of FTP Connection:

In this scenario, the numbers of forward FTP-flows are varied from 1 to 1000, the bottleneck bandwidth is fixed to 10Mbps, the queue limit is fixed to 8xBDP and the RTT is fixed to 32ms. Fig. 5, Fig. 6 and Fig. 7 show the results for bottleneck link utilization, mean queue length at the bottleneck router and packet drop rate at the bottleneck queue respectively. When number of forward FTP-flows are less, the bottleneck link utilization is expected to be low and when number of forward FTP-flows are more, the bottleneck link utilization is expected to be high because of increase in the traffic load. The bottleneck link utilization behavior remains almost similar for all three mechanisms except DropTail. With DropTail, link utilization is severly degraded. When the number of forward FTP-flows are less, the amount of bursts of packets is expected to be less and hence, the queue occupancy is also expected to be less. As the number of forward FTP-flows increases, the amount of bursts of packets is expected to increase sharply and hence, the queue occupancy is also expected to increase sharply. However, the goal of an AQM must be to control the mean queue length even when the amount of bursts of packets is high. It can be observed from Fig 6 that with DropTail mean queue length gets increased whereas with CoDel, sfqCoDel and CoDel-DT, the mean queue length remains almost constant since packets are proactively dropped/marked to provide an early congestion notification to the sender. Number of packet drop is expected to be low with minimal forward FTP flows, but as traffic load approaches 5, there is a spike in packet drop rate. Fig. 7 depict the expected behaviors of DropTail, CoDel, sfqCoDel and CoDel-DT.

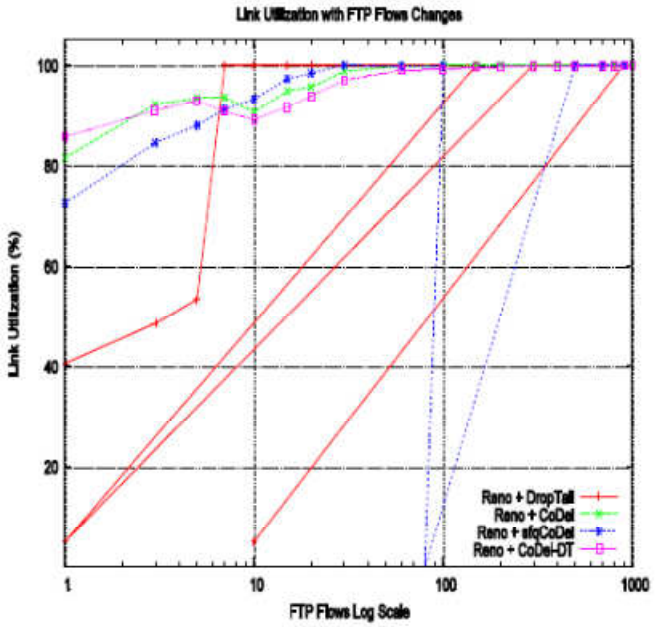

Fig. 5: Bottleneck Link utilization by varying number of FTP connections in TCP Reno

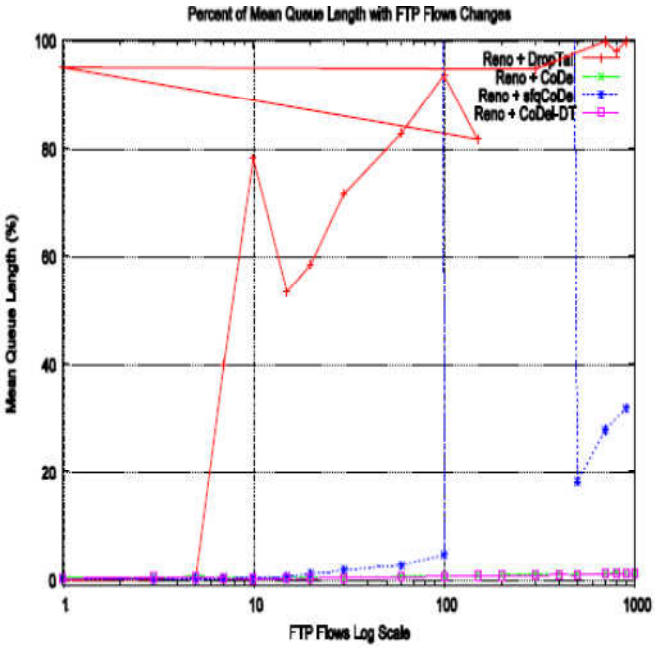

Fig. 6: Mean Queue length by varying number of FTP connections in TCP Reno

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication Pvt. Ltd.

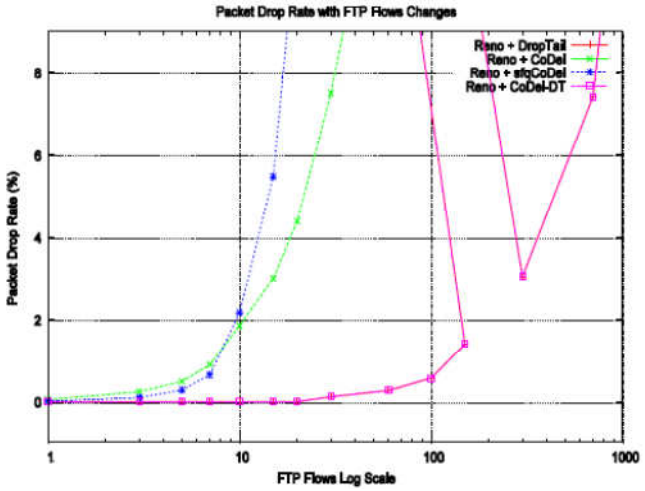

Fig.7: Packet Drop Rate by varying number of FTP connections in TCP Reno

C. Varying the RTT values:

In this scenario, the RTT is varied from 1ms to 1 second, the bottleneck bandwidth is fixed to 10Mbps and the queue limit is fixed to 8xBDP. Fig. 8 through Fig. 10 show the results for bottleneck link utilization, mean queue length at the bottleneck router and packet drop rate at the bottleneck queue respectively.

When the RTT <100ms, the bottleneck link utilization remains fairly good for all the AQM mechanisms. RTT values >100ms are common in the Internet.

Moreover, link utilization degrades further as the RTT values approach 1 second (see Fig. 8).

This is mainly because the traffic load cannot keep the larger pipe full. As the RTT increases, the capacity of the pipe also increases. Hence, it is expected that the queue occupancy decreases with the increase in the RTT. Fig. 9 depict the expected behavior. Although queue occupancy reduces for all AQMs, sfqCoDel's performance is signicantly and consistently better. CoDel, sfqCoDel and CoDel-DT mechanisms keep proactively dropping/marking the packets. As a result, we observe more packet drop rate for them than that of DropTail (see Fig. 10). sfqCoDel shows lesser packet drop than CoDel and CoDel-DT.

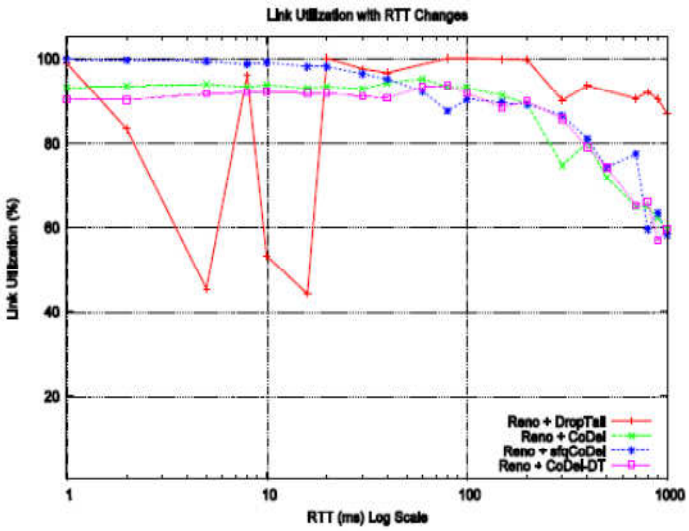

Fig. 8: Bottleneck Link utilization by varying RTT in TCP Reno

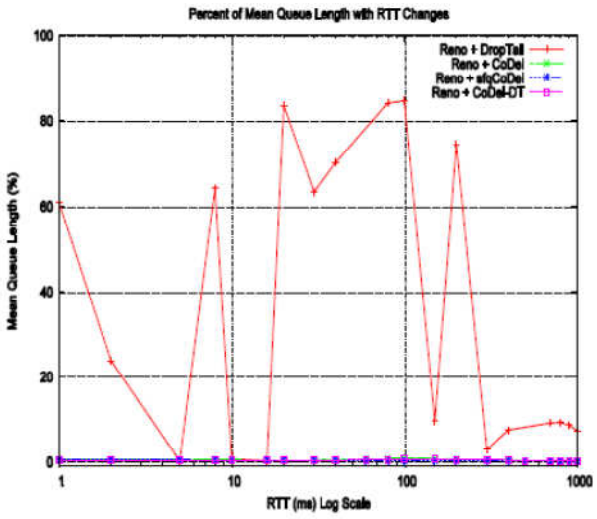

Fig. 9 : Mean Queue length by varying RTT in TCP Reno

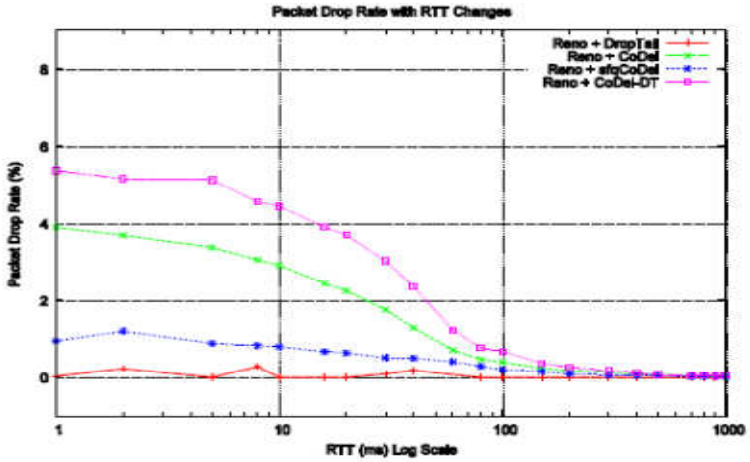

Fig. 10: Packet Drop Rate by varying RTT in TCP Reno

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication Pvt. Ltd.

V. CONCLUSIONS AND FUTURE WORK

Over the past few decades, the usage of Internet has increased at a alarming rate. Nowdays, network is suffering from poor performance and unnecessary latency also called as bufferbloat problem. As a solution to this, many Active Queue Management alogorithms have been developed. This paper compares the performance of these algorithms CoDel, sfqCoDel and CoDel-DT in wide range of scenarios like varying bottleneck bandwidth, varying number of connections, varying RTT values. A comparison is also carried out with traditional algorithm i.e. DropTail. Simulation results show that in almost all scenarios, sfqCoDel outperforms CoDel, CoDel-DT and DropTail in terms of link utilization, mean queue length and packet drop rate.

The results presented here are supporting, but these are performed using simulations. An in-depth investigation required to analyse the behaviour of these algorithms on real world networks. Also we can analyse the behavior of these algorithms on wireless networks. Moreover, interaction of these AQM algorithms with different TCP congestion control algorithms still needs to be explored.

REFERENCES

1. W. S. M. Allman, V. Paxson, “TCP Congestion Control,” April 1999, RFC 2581.
2. S. Floyd and V. Jacobson, “Random Early Detection Gateways for Congestion Avoidance,” IEEE/ACM Transactions on Networking, vol. 1, pp. 397–413, August 1993.
3. M. Hassan and R. Jain, “High Performance TCP/IP Networking: Concepts, Issues and Solutions,” 2004, Pearson Education, Inc.
4. Braden B. et al., “Recommendations on Queue Management and Congestion Avoidance in the Internet,” April 1998, RFC 2309.
5. J. Gettys, “Bufferbloat: Dark Buffers in the Internet,” IEEE Internet Computing Magazine, vol. 15, p. 96, June 2011.
6. “Bufferbloat Project,” 2011. [Online]. Available: http://www.bufferbloat.net
7. S. Floyd, “RED: Discussions of Setting Parameters,” November 1997. [Online]. Available:http://www.icir.org/floyd/REDparameters.txt
8. S. Floyd, R. Gummadi, and S. Shenker, “Adaptive RED: An Algorithm for Increasing the Robustness of RED’s Active Queue Management,” Tech. Rep., August 2001.
9. K. Zhou, K. L. Yeung, and V. O. K. Li, “Nonlinear RED: a simple yet efficient Active Queue Management Scheme,” Computer Networks, vol. 50, pp. 3784–3794, December 2006. [Online]. Available: http://portal.acm.org/citation.cfm?id=1228646.1228661
10. T. O. Lakshman, T. V. Lakshman, and L. Wong, “SRED: Stabilized RED,” in Proceedings of INFOCOM, 1999, pp. 1346–1355.
11. K. Nichols and V. Jacobson, “Controlling Queue Delay,” ACM Queue Magazine: Networks, vol. 10, no. 5, pp. 68–81, May 2012. [Online]. Available: http://queue.acm.org/detail.cfm?id=2209336
12. Jacobson, “Kathie Nichols CoDel,” Vancouver, Canada, July 2012, IETF-84 Transport Area Open Meeting.
13. Dipesh M. Raghuvanshi, B. Annappa, and Mohit P. Tahiliani. ”On the Effectiveness of CoDel for Active Queue Management.” In Proceedings of Third International Conference on Advanced Computing & Communication Technologies, ACCT ’13, pages 107-114. IEEE Computer Society, 2013.
14. T. Sharma, “Controlling queue delay (codel) to counter the bufferbloat problem in internet," INPRESSCO International Journal of Current Engineering and Technology, 2014.
15. Paul E. McKenney and Dave Tht.”SFQ on Steroids” Retrived from http://snapon.lab.bufferbloat.net/d/lwn/SFQ2012/FQ-Codel.htmlx (January 7, 2013)
16. V. P. Rao, M. P. Tahiliani, and U. K. K. Shenoy, “Analysis of sfqcodel for active queue management," in Applications of Digital Information and Web Technologies(ICADIWT), 2014 Fifth International Conference on the, pp. 262-267, IEEE, 2014.
17. G. White, “Active queue management in docsis 3.1 networks,” Communications Magazine, IEEE, vol. 53, no. 3, pp. 126-132, 2015.
18. “The Network Simulator - ns-2 Project.” [Online]. Available: http://www.isi.edu/nsnam/ns/

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication Pvt. Ltd.