On the Effectiveness of CoDel for Active Queue Management

Dipesh M. Raghuvanshi, Annappa B., Mohit P. Tahiliani
Department of Computer Science and Engineering
National Institute of Technology Karnataka,
Surathkal, Karnataka, India 575025
dmraghuvanshi.erandol@gmail.com, annappa@ieee.org, tahiliani.nitk@gmail.com

***Abstract*—Internet in the present scenario has become a gigantic source of information. There has been a tremendous rise in the variety of Internet applications, with each application demanding a specific performance criteria to be satisfied. Routers presently use Passive Queue Management (PQM) mechanisms and hence, merely have any control over the queue occupancy. Therefore, there has been an increased interest in exploring Active Queue Management (AQM) in Internet routers so as to reduce the queue latency and meet the demands of time sensitive applications. In this paper, we mainly focus on analyzing the effectiveness of a recently proposed AQM mechanism called Controlled Delay (CoDel). We study the effectiveness of CoDel by carrying out simulations in ns-2 and comparing its performance with existing AQM mechanisms in variety of Internet scenarios. Based on the simulation results obtained, we discuss the advantages and shortcomings of CoDel in terms of bottleneck link utilization, mean queue length and packet drop rate.**

***Index Terms*—Bufferbloat, CoDel, AQM, tail-drop**

I. Introduction

Over the past few years, there has been a phenomenal growth in the Internet usage and the diversity of its applications. Transmission Control Protocol (TCP) has been the major protocol of the Internet ever since its inception. The success of the Internet, in fact, can be partly attributed to the congestion control mechanisms implemented in TCP [1]. However, tremendous growth in the range of bandwidth, increase in Bit-Error Rates (BER) and increased diversity in the applications of the next-generation networks have challenged the congestion control mechanisms of TCP.

The performance of TCP based applications, apart from the congestion control mechanisms, critically depends on the choice of queue management scheme implemented in the routers. Queue management schemes control the length of the queues by proactively dropping packets when necessary. Passive Queue Management (PQM) (e.g., tail-drop) is the most widely deployed queue management mechanism in Internet routers [2]. PQM does not employ any proactive packet drop before the router queues get full and hence, is easily deployable. However, due to inherent problems of PQM such as global synchronization [3] and lock-out [2], IETF recommends the deployment of Active Queue Management (AQM) for the next-generation Internet routers [4]. Moreover, another limitation of PQM called *persistently full buffer problem* (recently exposed as a part of *Bufferbloat* [5][6]) has proved the necessity of wide spread deployment of AQM.

As a result, AQM mechanisms have been extensively studied in the recent past to monitor and limit the growing queues at router. These mechanisms *avoid* congestion by proactively informing the TCP sender about congestion, either by dropping or by marking a packet. Random Early Detection (RED) [3] is the most widely deployed AQM mechanism in the routers. However, it has been shown that the effectiveness of RED largely depends on appropriately setting at least four parameters, namely: minimum threshold (min_{th}), maximum threshold (max_{th}), queue weight factor (w_q) for exponential weighted moving average and maximum drop probability (max_p) [7]. Optimal values for these parameters differ for different scenarios and hence, setting appropriate values for these parameters has been a critical issue ever since the inception of RED. Although a lot of RED variants have been proposed in the literature [8] [9] [10], there is still a lot of reluctance in the widespread acceptance of RED because these variants further complicate its mechanism.

Recently, a new AQM mechanism called Controlled Delay (CoDel) [11] has been proposed to overcome the shortcomings of PQM and RED. Unlike RED, CoDel is parameterless AQM mechanism that adapts to varying link rates and can be easily deployed [11]. Moreover, unlike RED and its variants that use *average queue size* as a predictor of congestion, CoDel uses *packet sojourn time* to predict congestion.

In this paper, we carry out an experimental study to evaluate the effectiveness of CoDel in a wide variety of Internet scenarios. The performance of CoDel is compared to that of RED and Adaptive RED (ARED) [8]. ARED is a promising variant of RED that minimizes the need for manually setting the RED parameters. The major focus of the study is to evaluate the bottleneck link utilization, mean queue length at the bottleneck router and the overall packet drop rate.

Rest of the paper is organized as follows: Section II briefly describes the working of CoDel algorithm. Section III provides the necessary details about the simulation configuration, simulation scenarios, performance evaluation metrics, etc. Section IV discusses the simulation results and provides a comparative analysis of RED, ARED and CoDel. Section V presents the inferences and a few open issues based on the comparative study. Section VI concludes the paper with possible future directions.

978-0-7695-4941-5/13 $26.00 © 2013 IEEE
DOI 10.1109/ACCT.2013.27

II. Controlled Delay (CoDel) Algorithm

A. Overview

Controlled Delay (CoDel) is the most recent AQM mechanism proposed by Nichols and Jacobson [11] and is believed to be the best to handle Bufferbloat [12]. Unlike other RED based AQM mechanisms, CoDel is independent of various network parameters such as queue size, queue size averages, queue size thresholds, rate measurements, link utilization, drop rate, queue occupancy time or round trip delays [11].

CoDel relies on the packet sojourn time i.e. the actual queue delay experienced by a packet as a metric to predict congestion in the network. If the packet sojourn time is above the *target value* for a specified *interval* of time, CoDel starts proactively dropping/marking the packets to control the queue length. However, CoDel avoids the underutilization of outgoing link by not dropping/marking the packets proactively in case if current queue size is less than one MTU.

B. Algorithm

Algorithm 1: CoDel Algorithm

```
On arrival of every packet :
if current_queue_size < queue_limit then
    Enqueue the packet
    Attach a timestamp in packet header
end
else
    Drop the Packet
end
On departure of every packet :
dequeue_time = timestamp for dequeue time
sojourn_time = dequeue_time − enqueue_time
if inside the dropping state then
    if sojourn_time < target  or
    current_queue_size < MTU then
        Do not drop packets
        Come out of dropping state
    end
    else
        while dequeue_time ≥ next_drop_time do
            Drop the packet
            count = count + 1
            next_drop_time += interval / √count
        end
    end
end
else if outside dropping state and first packet is
being dropped then
    Enter the dropping state
end
```

The algorithm works in two phases: (i) at the time of enqueuing the packet and (ii) at the time of dequeuing the packet. On arrival of every packet, the current queue size is checked. If it is less than the queue limit, the packet is enqueued and the timestamp is added in the header. This timestamp indicates *enqueue time*. On departure of every packet, the timestamp is extracted from the header and is subtracted from the current time to obtain the *packet sojourn time*.

The CoDel algorithm remains either in the dropping state or not in the dropping state. If the packet sojourn time remains above the *target* for a specified interval of time, CoDel enters into the dropping state and starts proactively dropping/marking the packets. Note that the packets are proactively dropped while dequeuing rather than during enqueuing. The time duration between the two proactive packet drops is calculated by the following equation:

$next\_drop\_time += interval/\sqrt{count}$

The *count* indicates the total number of packets dropped since the dropping state is entered.

While the algorithm is in dropping state, if the packet sojourn time becomes lesser than target or if queue does not have sufficient packets to fill the outgoing link, the algorithm leaves the dropping state.

There are two most important CoDel parameters to be set to achieve optimal results: *target* and *interval*. These are fixed parameters and their values are chosen based on the observations from several experiments.

Following are the values for *target* and *interval*:

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

III. Simulation Scenario

A. Simulation Setup

The performance of RED, ARED and CoDel is evaluated in a wide range of scenarios like: varying the bottleneck bandwidth, varying the number of FTP flows and varying the RTT values. We have used network simulator ns-2 [13] to carry out a comparative study of three AQM mechanisms viz. RED, ARED, CoDel. TCP Evaluation Suite for ns-2 is used to simulate a wide range of Internet scenarios. In general, a single bottleneck dumbbell topology with two-way traffic is designed for all the simulations. The bottleneck bandwidth is set to 10Mbps with bottleneck round trip delay set to 32ms unless specified. Non-bottleneck bandwidth is set to 20Mbps with round trip delay set to 4ms. The bottleneck buffer is sized to 8xBDP. The traffic includes audio traffic, video traffic, file-transfer, and web traffic. The scenario consists of five forward-FTP flows unless specified, five reverse-FTP flows, fifteen HTTP flows generated using PackMime generator, five audio flows, five forward-streaming flows and five reverse-streaming flows.

Based on the recommended values in [11], the values of *interval* and *target* queue delay for CoDel are set to 100ms and 5ms respectively.

B. Metrics

In this paper, we mainly concentrate on analyzing three parameters viz. link utilization of bottleneck link, queue size at bottleneck router and packet drop rate. We have selected the

prominent TCP variants like: Reno, Reno with Selective ACK (SACK), High-Speed TCP (HSTCP), Scalable TCP (STCP), Hamilton TCP (HTCP), Binary Increase Congestion Control TCP (BIC) and CUBIC TCP in this study.

IV. Results and Analysis

A. Varying Bottleneck Bandwidth

In this scenario, the bottleneck bandwidth is varied from 1Mbps to 1Gbps, the queue limit is fixed to 8xBDP and the RTT is fixed to 32ms. Fig. 1 through Fig. 3, Fig. 4 through Fig. 6 and Fig. 7 through Fig. 9 show the results for bottleneck link utilization, mean queue length at the bottleneck router and packet drop rate at the bottleneck queue respectively.

The desirable properties of an optimal AQM mechanism are: high link utilization, minimum queue occupancy and least packet drop rate. It can be observed from the above mentioned graphs that CoDel satisfies all the desirable properties whereas RED and ARED fail to retain high link utilization and control the queue occupancy.

When the bandwidth >10Mbps, the link utilization of almost all TCP flavors is severely degraded with RED and ARED. Link utilization with CoDel remains fairly better even when the bottleneck bandwidth is around 100Mbps. Apart from the bottleneck link utilization, CoDel's performance in terms of mean queue length is significantly better than that of RED and ARED. The queue occupancy at an average is around 5%. When bandwidth is less, delay resulting from bursts of packets is expected to be more and hence, the queue occupancy is also expected to be more. When bandwidth is more, delay resulting from bursts of packets is expected to be less and hence, the queue occupancy is also expected to be less. Fig. 3 and 6 show that the CoDel's behavior is as expected. Moreover, the oscillations in the queue length are also smaller with CoDel. Less queue occupancy reduces the overall latency and smaller oscillations in the queue minimize the jitter - both of which are desirable properties for time sensitive applications like DNS queries, Voice over IP (VoIP) and other multimedia applications.

Similarly, when bandwidth is less, packet drop rate is expected to be high because of the bursts of packets and when bandwidth is more, packet drop rate is expected to be low. The packet drop rate behavior, however, remains almost similar for all three mechanisms. The packet drop rate approaches zero once the bottleneck bandwidth >10Mbps.

B. Varying the number of FTP connections

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. 10 through Fig. 12, Fig. 13 through Fig. 15 and Fig. 16 through Fig. 18 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

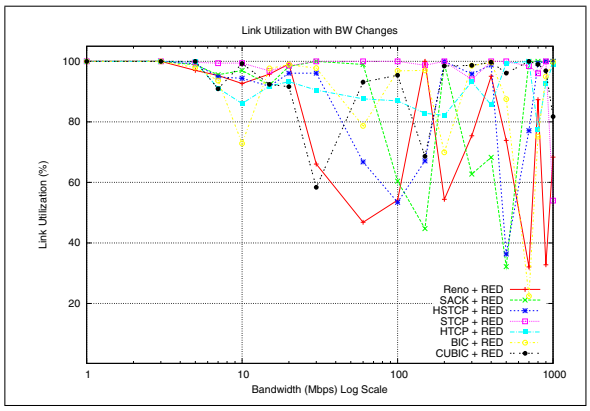

Fig. 1. Bottleneck Link Utilization with Original RED for varying bottleneck bandwidth

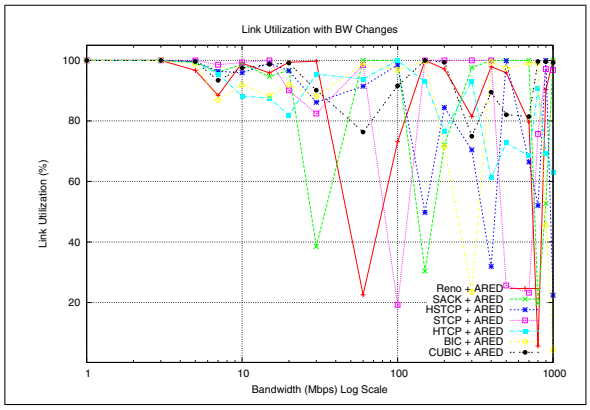

Fig. 2. Bottleneck Link Utilization with ARED for varying bottlenck bandwidth

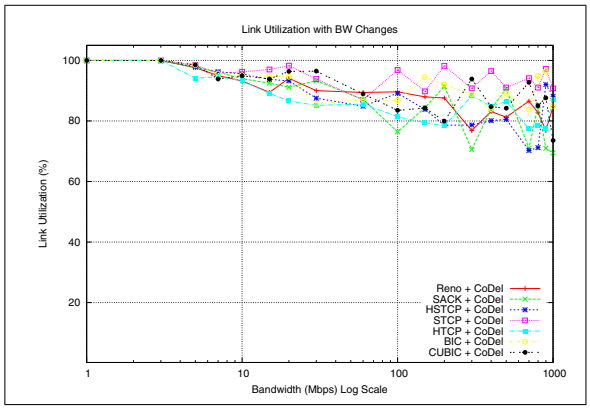

Fig. 3. Bottleneck Link Utilization with CoDel for varying bottlenck bandwidth

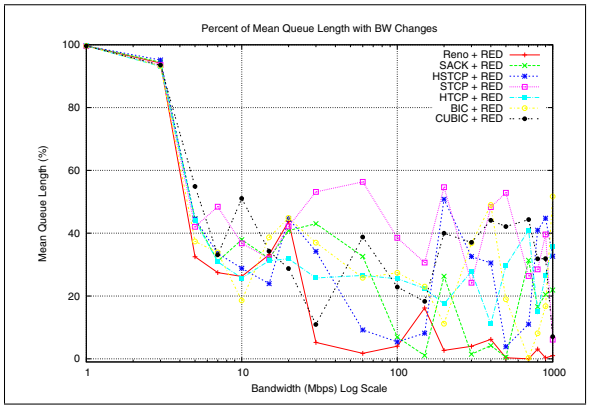

Fig. 4. Bottleneck Queue Length with Original RED for varying bottlenck bandwidth

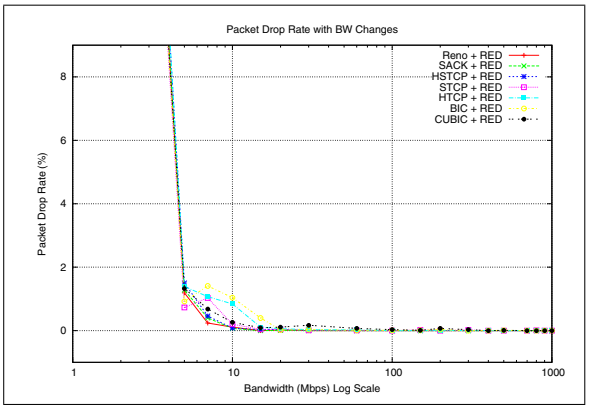

Fig. 7. Packet Drop Rate with Original RED for varying bottlenck bandwidth

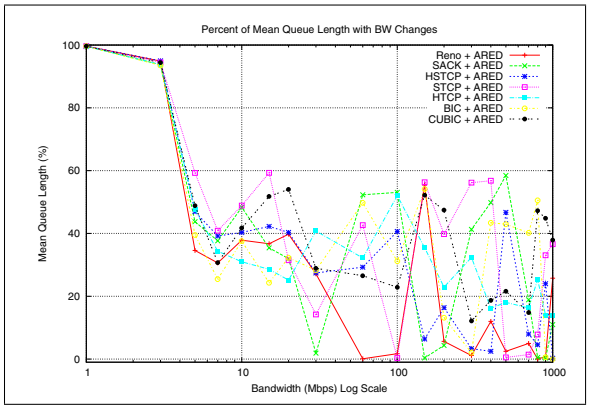

Fig. 5. Bottleneck Queue Length with ARED for varying bottlenck bandwidth

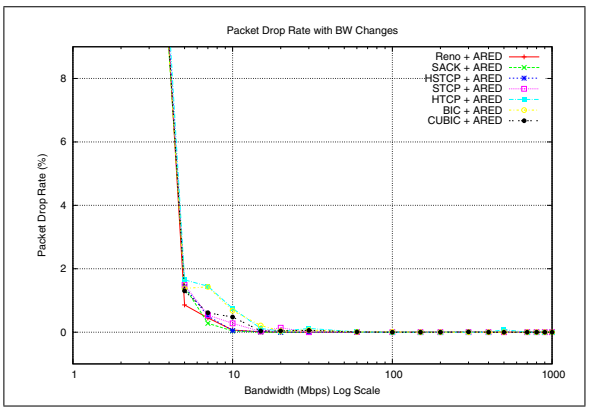

Fig. 8. Packet Drop Rate with ARED for varying bottlenck bandwidth

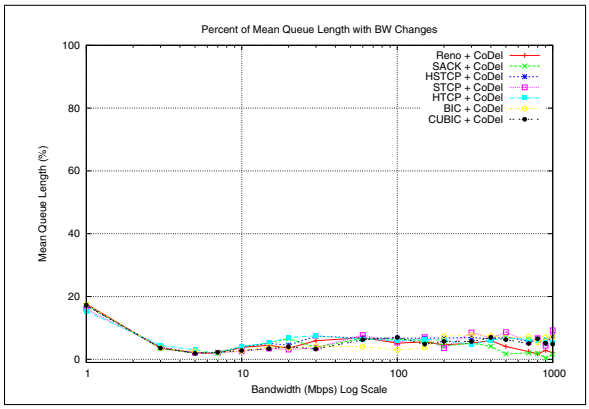

Fig. 6. Bottleneck Queue Length with CoDel for varying bottlenck bandwidth

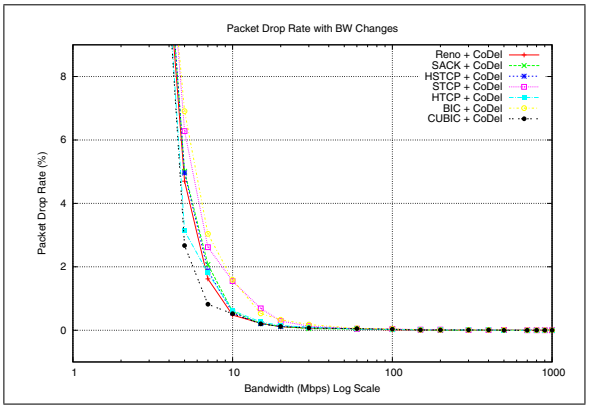

Fig. 9. Packet Drop Rate with CoDel for varying bottlenck bandwidth

expected to be high because of increase in the traffic load. The bottleneck link utilization behavior remains almost similar for all three mechanisms and approaches 100% once the number of forward FTP-flows is more than 15.

In this scenario too, CoDel's performance in terms of mean queue length is significantly better than that of RED and ARED. The queue occupancy at an average is around 8%. 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. Fig. 13 and 14 show that the behavior of RED and ARED is as expected. 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. 15 that with CoDel, the mean queue length remains almost constant since packets are proactively dropped/marked to provide an early congestion notification to the sender. Moreover, it is expected that the packet drop rate would be more than RED and ARED since it is successful in controlling the mean queue length. Fig. 16 through Fig. 18 depict the expected behaviors of RED, ARED and CoDel.

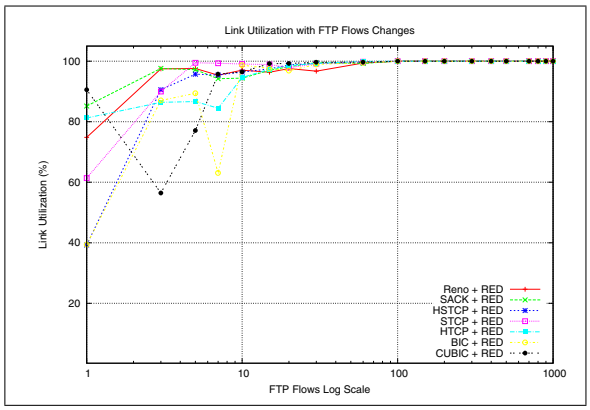

Fig. 10. Bottleneck Link Utilization with Original RED for varying number of FTP connections

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. 19 through Fig. 27 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. For RTT values $>$100ms, the link utilization of all AQMs is slightly affected but that of CoDel is affected more. Moreover, link utilization degrades further as the RTT values approach 1 second (see Fig. 21). This is mainly because the traffic load cannot keep the larger pipe full.

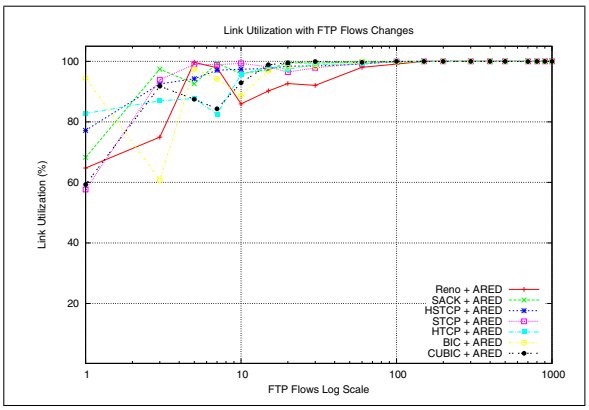

Fig. 11. Bottleneck Link Utilization with ARED for varying number of FTP connections

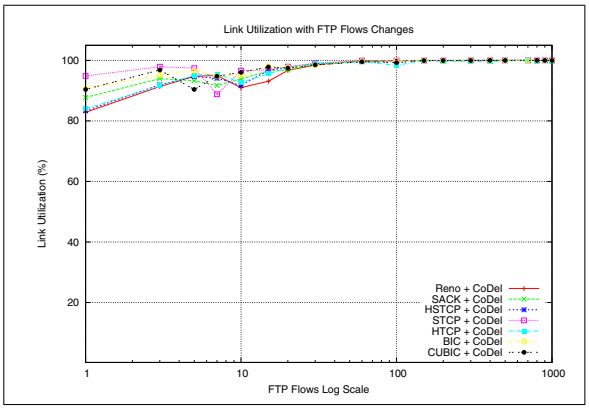

Fig. 12. Bottleneck Link Utilization with CoDel for varying number of FTP connections

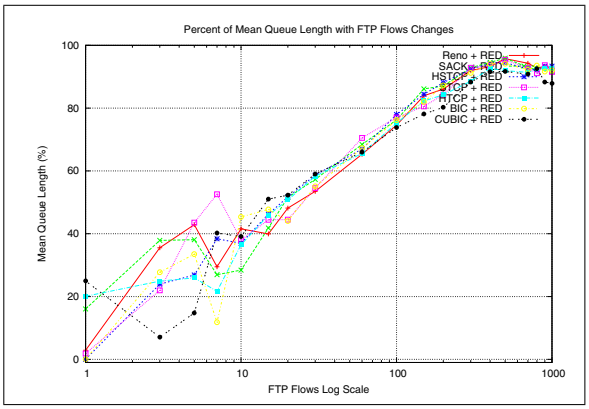

Fig. 13. Bottleneck Queue Length with Original RED for varying number of FTP connections

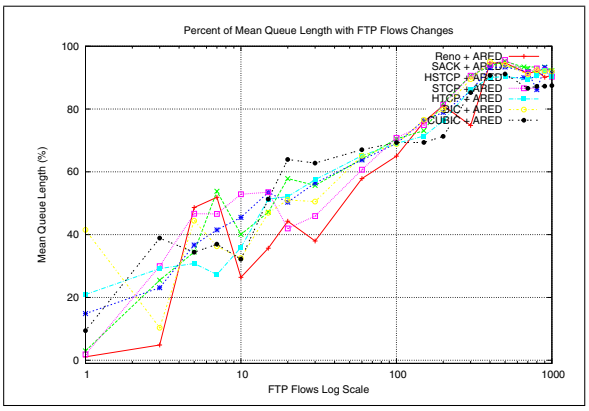

Fig. 14. Bottleneck Queue Length with ARED for varying number of FTP connections

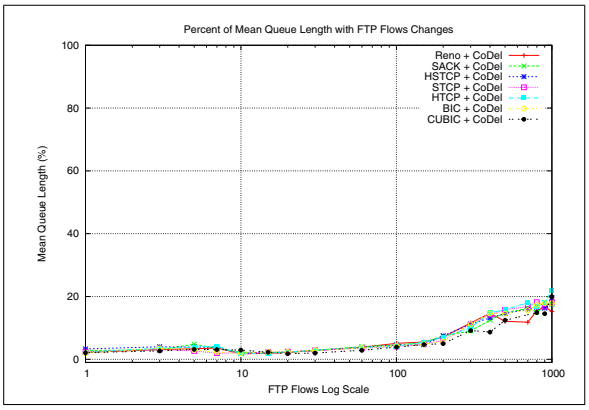

Fig. 15. Bottleneck Queue Length with CoDel for varying number of FTP connections

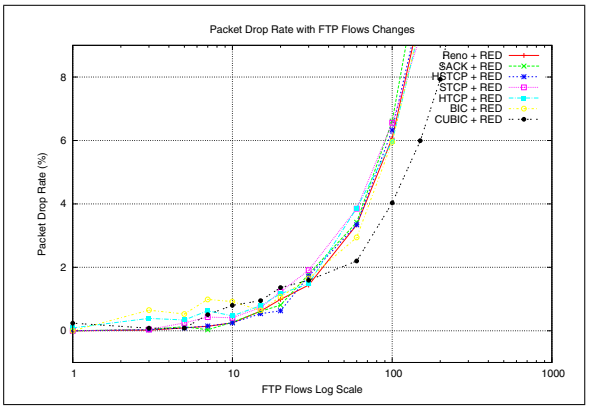

Fig. 16. Packet Drop Rate with Original RED for varying number of FTP connections

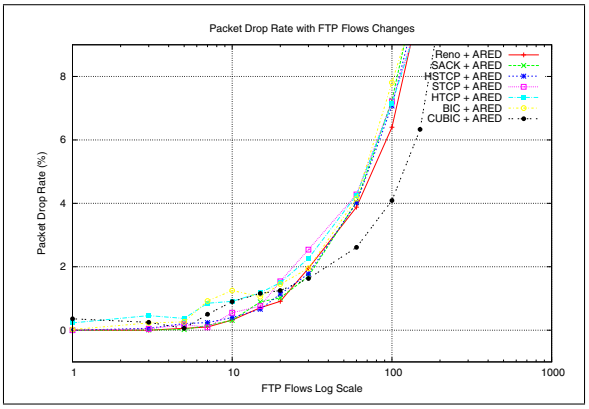

Fig. 17. Packet Drop Rate with ARED for varying number of FTP connections

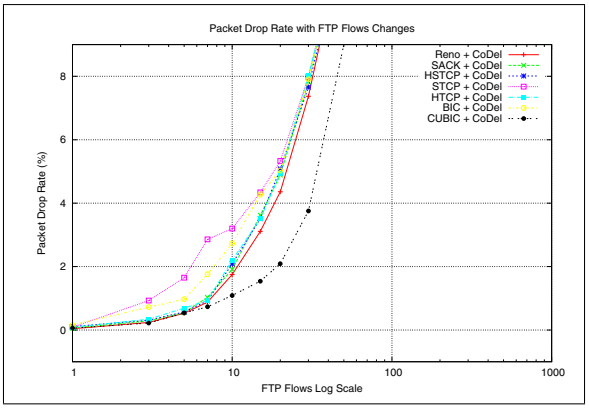

Fig. 18. Packet Drop Rate with CoDel for varying number of FTP connections

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. 22 through Fig. 24 depict the expected behavior. Although queue occupancy reduces for all three AQMs, CoDel's performance is signicantly and consistently better. The average queue size in RED and ARED is calculated based on exponential weighted moving average [3]. It takes time for the average queue size to reduce within the specified thresholds until which the RED and ARED mechanisms keep proactively dropping/marking the packets. As a result, we observe more packet drop rate for RED and ARED than that of CoDel (see Fig. 25 through Fig. 27).

V. Inferences and Open Issues

Based on the simulation study carried out above, we list out the advantages of CoDel over RED and ARED:

- CoDel has efficient link utilization in a wide range of scenarios as compared to RED and ARED.
- CoDel significantly outperforms RED and ARED in terms of mean queue length in all the above scenarios.
- The packet drop rate with CoDel is almost similar to that of RED and ARED.

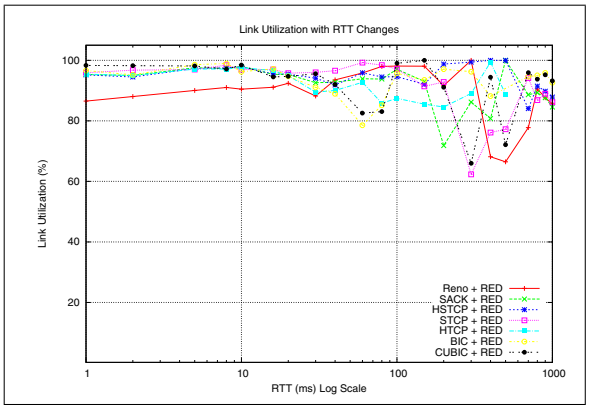

Fig. 19. Bottleneck Link Utilization with Original RED for varying RTT

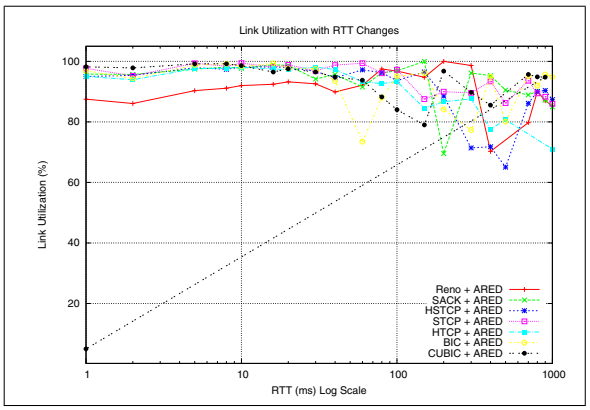

Fig. 20. Bottleneck Link Utilization with ARED for varying RTT

- CoDel is parameterless and hence can be configured with ease.
- CoDel solves the primary problem of Bufferbloat by minimizing the queue occupancy.

Though CoDel seems to be a promising solution to Bufferbloat, there are a few observations that attract attention towards further optimizations required in CoDel:

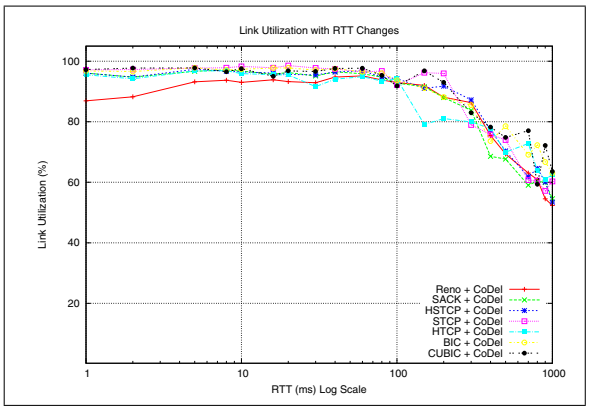

Fig. 21. Bottleneck Link Utilization with CoDel for varying RTT

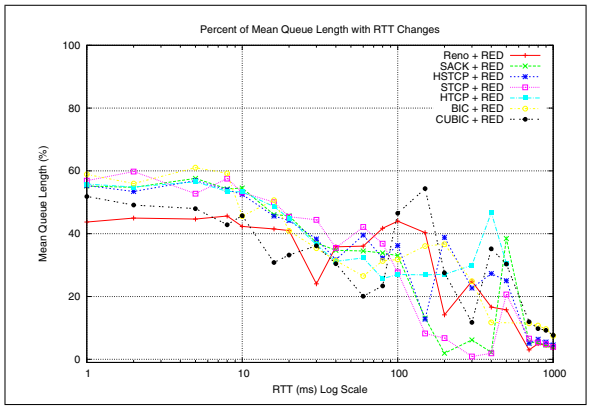

Fig. 22. Bottleneck Queue Length with Original RED for varying RTT

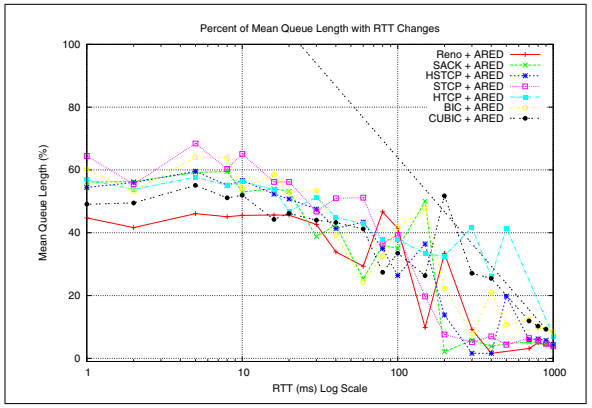

Fig. 23. Bottleneck Queue Length with ARED for varying RTT

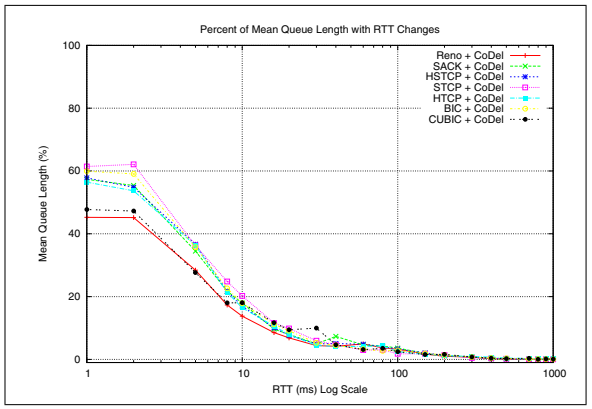

Fig. 24. Bottleneck Queue Length with CoDel for varying RTT

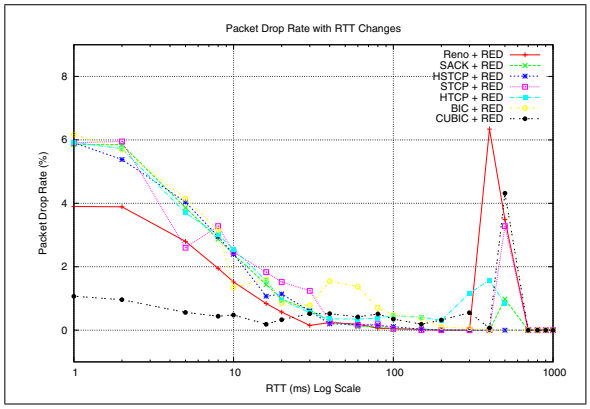

Fig. 25. Packet Drop Rate with Original RED for varying RTT

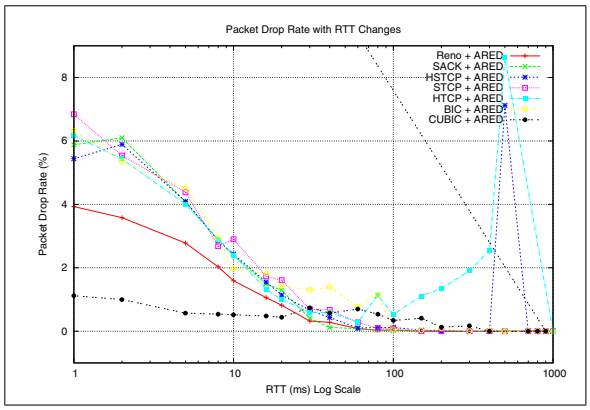

Fig. 26. Packet Drop Rate with ARED for varying RTT

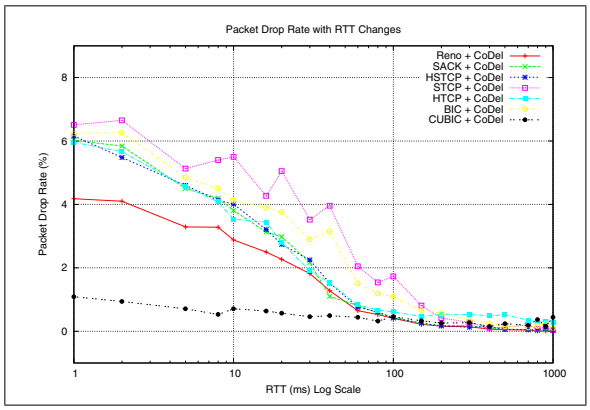

Fig. 27. Packet Drop Rate with CoDel for varying RTT

- For larger RTTs and smaller bandwidths, CoDel has poor link utilization than that of RED and ARED.
- CoDel has slightly higher packet drop rate for more number of FTP-flows.

VI. Conclusions and Future Work

With a rapid increase in the diversity of Internet applications, the inherent problems of PQM have become increasingly apparent. As a result, there has been an active interest towards deploying the efficient AQM mechanisms in the Internet. Although RED has been a popular AQM mechanism over last two decades, it has failed to gain the confidence of the network providers largely due to its parameter sensitivity. Recently, a new parameterless AQM mechanism named CoDel has been proposed to overcome the drawbacks of RED and its variants.

In this paper, we have carried out a simulation study to analyze the effectiveness of CoDel. The performance of CoDel is compared with RED and ARED in a wide range of Internet scenarios like: varying bottleneck bandwidth, varying number of FTP-flows and varying RTT values. The performance parameters analyzed are: bottleneck link utilization, mean queue length of bottleneck queue and overall packet drop rate.

Based on the simulation results it is observed that CoDel is independent of queue size, queue size averages, queue size thresholds, rate measurements, link utilization, drop rate, queue occupancy time or round trip delays. It achieves high link utilization and reduces the queue occupancy in most of the simulated scenarios. Nevertheless, a few more optimizations to CoDel are highly required to further increase its robustness. Moreover, an in-depth investigation is also required to analyze the effectiveness of CoDel in wireless scenarios.

References

[1] W. S. M. Allman, V. Paxson, "TCP Congestion Control," April 1999, RFC 2581.

[2] M. Hassan and R. Jain, "High Performance TCP/IP Networking: Concepts, Issues and Solutions," 2004, Pearson Education, Inc.

[3] S. Floyd and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance," *IEEE/ACM Transactions on Networking*, vol. 1, pp. 397–413, August 1993.

[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] V. Jacobson, "Kathie Nichols CoDel," Vancouver, Canada, July 2012, IETF-84 Transport Area Open Meeting.

[13] "The Network Simulator - ns-2 Project." [Online]. Available: http://www.isi.edu/nsnam/ns/