Performance of BBR and AQMs in Different Scenarios

**Abstract - Bufferbloat is a problem caused by excess buffering of data which can cause high latency and jitter. Nowadays, more and more interactive applications run in the Internet, thus this problem becomes even more severe and gains more widespread attention. The new congestion control based on measuring bottleneck bandwidth and round-trip propagation time of a path, or BBR, as well as the Active Queue Management (AQM) algorithms are effective schemes to address the bufferbloat problem. We study the performance of BBR and two AQMs ---- PIE (Proportional Integral controller Enhanced) and CoDel (Controlled Delay), using GENI testbed. We find that in single bottleneck link network with 10mbit link capacity, both BBR and AQMs work well. However, in a parking-lot network, the throughput of BBR is degraded. Then we find that it is because of the congestion window (cwnd) size. Perhaps Cubic with PIE is a good alternative for BBR when link capacity is relatively small.**

***Key words: bufferbloat, BBR, AQMs, single bottleneck link network, parking-lot network***

1.Introduction

Most of our traditional congestion control algorithms, including TCP (New) Reno, High Speed, BIC, and CUBIC, are based on packet loss, and works near buffer-limited area, which will achieve high throughput but also cause high latency. Here we present a new congestion control, BBR, which is based on real congestion rather than packet loss. It tries to work at the optimum operating point, which will reach maximum bandwidth with minimum latency. It measures two parameters that characterize a path: bottleneck bandwidth and round-trip propagation time. However, these two parameters cannot be measured at the same time. When we measure the bandwidth, the system works in the bandwidth-limited area, which causes a queue that increase the measured RTT. BBR seeks high throughput with small queues by probing bandwidth and RTT sequentially [1].

PIE (Proportional Integral controller Enhanced) and CoDel (Controlled Delay) are two effective AQMs. PIE randomly drops an incoming packet with a dropping-probability, just like RED (Random early detection), which is implemented in a wide variety of network devices. The difference is that the dropping-probability of PIE is based on queueing latency instead of queue length. What's more, PIE will use the derivation of latency to determine its behavior. [2]

CoDel (Controlled Delay) is another effective bufferbloat solution. Although it is said to be "parameter-free", it actually uses two variables: target and interval. Usually the interval is set to be 100ms and the target 5ms. When the maximum delay in the interval is above target, the controller enters the drop state where a packet is dropped, and the next drop time is set. the initial next drop spacing is 100ms. If the minimum delay in the interval falls below target, the controller cancels the next drop and exits the drop state. If next drop time is reached while the controller is in drop state, the packet being dequeued is dropped and the interval used for the next group of packets is shortened. [3]

We already know that BBR and AQMs work well under certain situations, yet we have to study their performance in different scenarios to put them into more widespread use. We compare their behaviors in both single bottleneck link network and multiple bottleneck links network, in the latter case different flows will have different RTT. We find BBR does not always work well. In the following section I will show my results in detail.

2.Related Work

BBR was developed by Google, and now is widely used in our Internet. Google Cloud Platform’s Front-Ends, Load-Balancers and managed services already have TCP BBR enabled. Google Cloud Platform (GCP) customers automatically benefit from BBR in two ways: From GCP services to cloud users; From Google Cloud to internet users [4]. BBR has greatly improved the throughput of Google and YouTube network. All Google/YouTube servers and datacenter WAN backbone connections use BBR. Also, Code is available as open source in Linux TCP (dual GPLv2/BSD), QUIC (BSD) [5].

AQMs are used more widely today to solve bufferbloat problem. RFC 2309 strongly recommends the adoption of AQM schemes in the network to improve the performance of the Internet. PIE and CoDel are new schemes seeking to provide both low latency and high goodput, without requiring the extensive parameter tuning that was needed for earlier schemes like Random Early Detection (RED). [6]

3.Experiment Setup

*3.1 Dumbbell Topology*

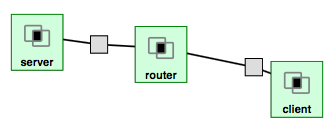


Figure 1: dumbbell topology

The client will send packets to server, passing router.

The bottleneck link capacity is set to be 10mbit. In the first experiment there is no delay added; And in the following experiment, there is 100ms delay and 30ms jitter added to the link.

*3.2 Parking-lot Topology*

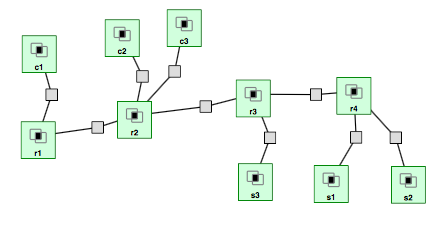


Figure 2: parking-lot topology

The client nodes c1, c2, and c3 send packet to server nodes s1, s2, and s3, passing three hops, two hops, and one hop, respectively.

In the first experiment, 20ms delay is added to each hop so different connections will have different RTT. In the next experiment, I add 40ms delay jitter in both interfaces in r1, so the total delay jitter of c1-s1 connection is 80ms. The bottleneck link capacity is 10mbit.

4. Result

*4.1 Dumbbell Network*

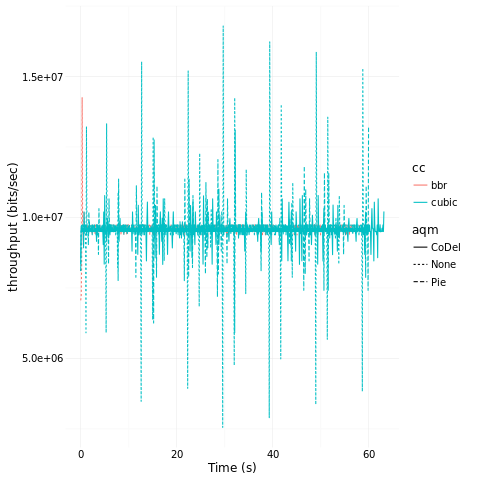


Figure 3: throughput of dumbbell network when no delay added

From the figure above, we can see that in single bottleneck link network, both BBR and Cubic with AQMs can reach high throughput.

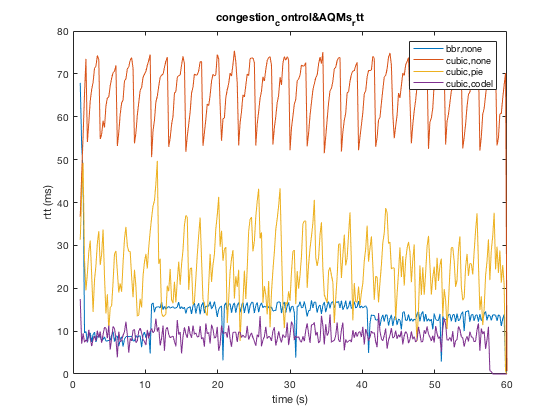


Figure 4: RTT of dumbbell network without added delay

However, the BBR performs much better than Cubic in controlling delay. And Cubic with PIE or CoDel can also control delay well.

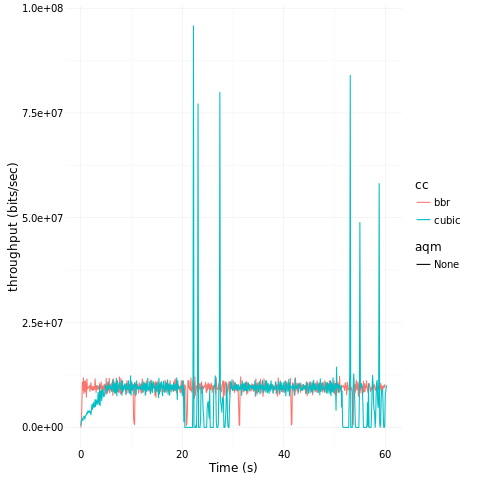


Figure 5：throughput when huge delay jitter added

It can be seen that under this configuration, the delay jitter doesn't have a significant effect on the throughput of BBR or Cubic.

*4.2 Parking-lot Network*

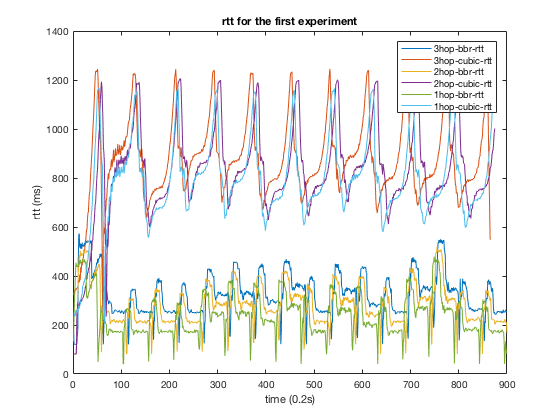
*4.2.1 No delay jitter added*

Figure 6: RTT when no delay jitter added

BBR still performs better than Cubic in controlling delay.

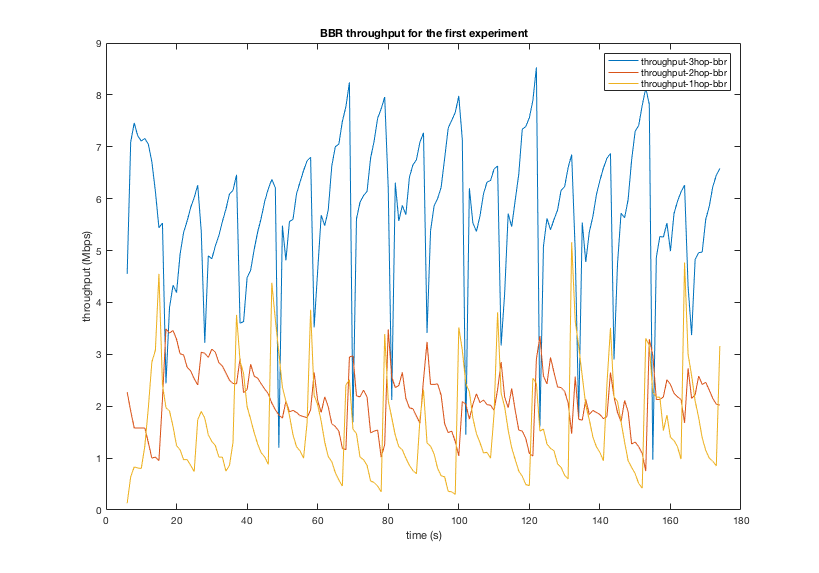


Figure 7:BBR throughput when no jitter added

However, BBR provides really bad throughput with poor fairness.

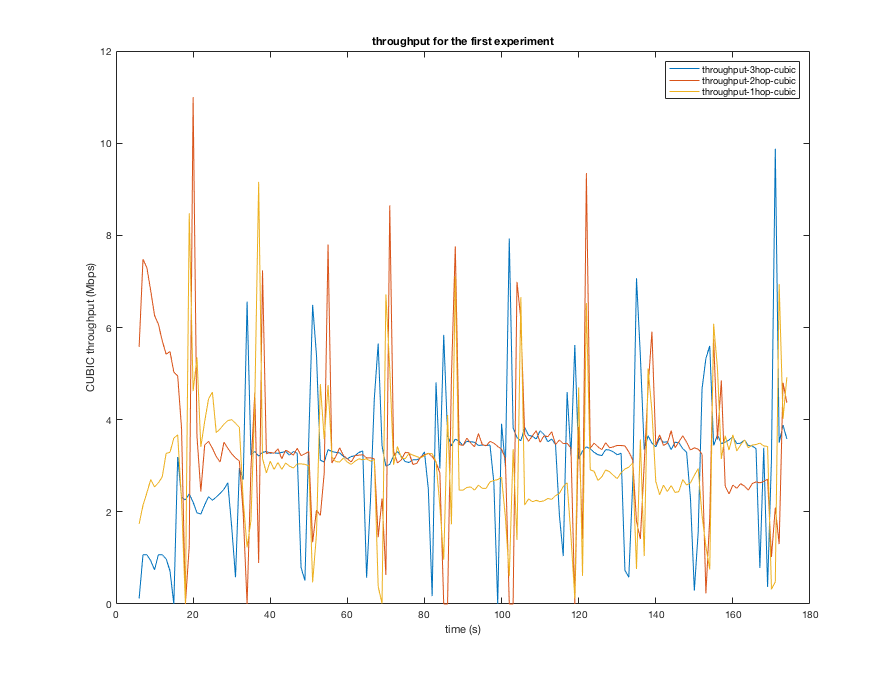


Figure 8:Cubic throughout

Cubic provides high throughput with good fairness.

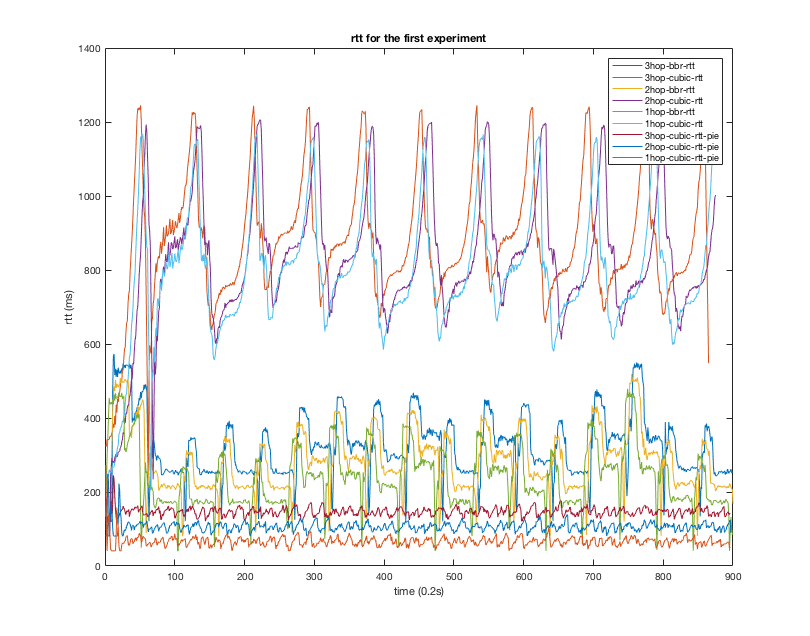


Figure 9: RTT when no delay jitter added

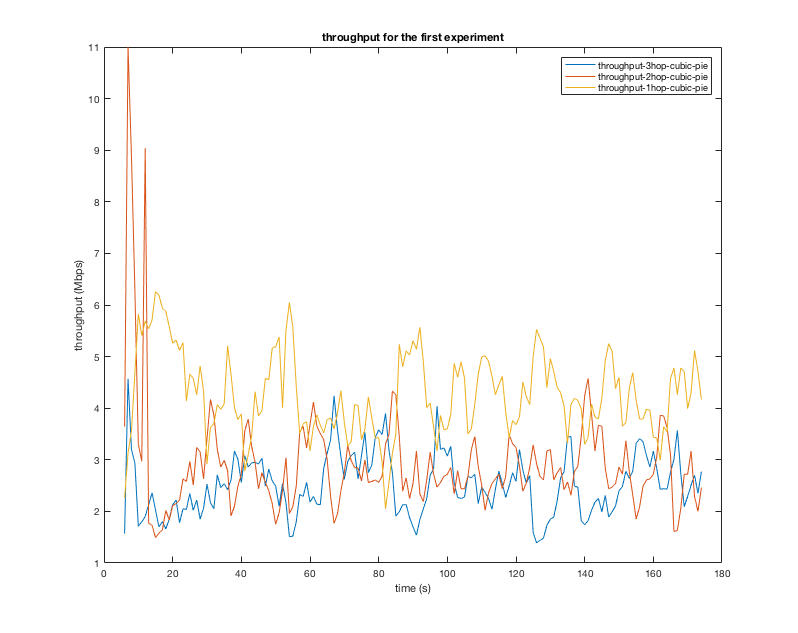


Figure 10:throughput of Cubic with PIE

Cubic with PIE can provide both relative high throughput and low latency.

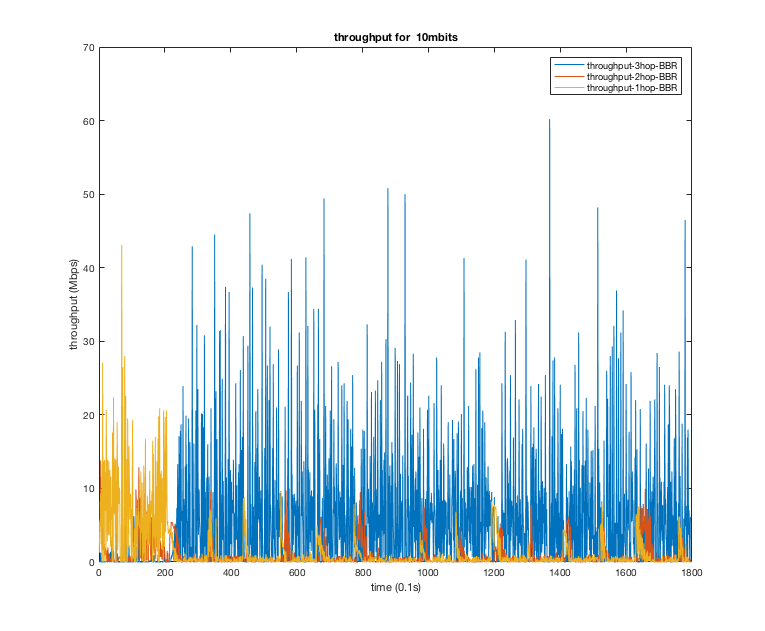
*4.2.2 add delay jitter in 3-hop connection*

Figure 11:BBR throughput

The 3-hop flow still occupies most of the bandwidth.

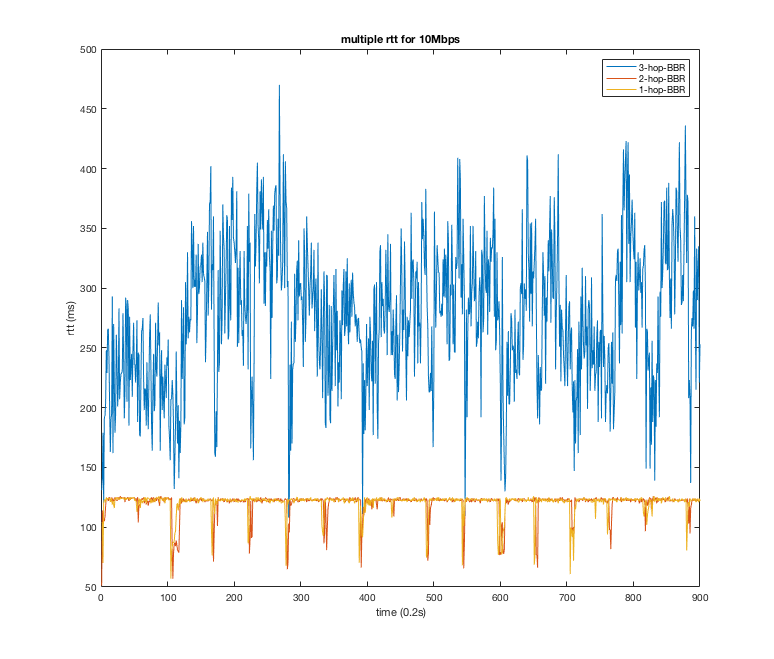


Figure 12: RTT of BBR flows

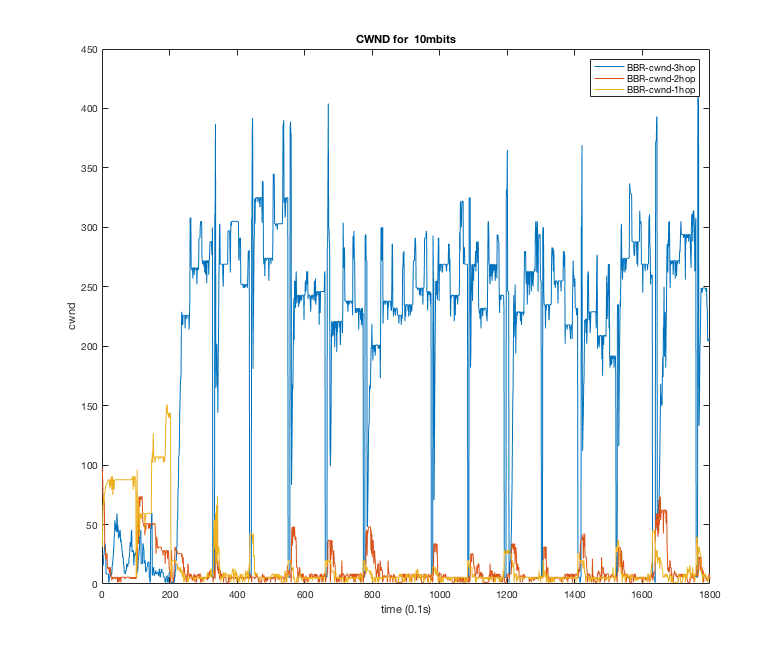


Figure 13: cwnd of three flows

It seems that the RTT variation is not the main cause: The throughput relay more on RTT than on RTT jitter.

Then I plot the RTT and cwnd of the 3-hop connection.

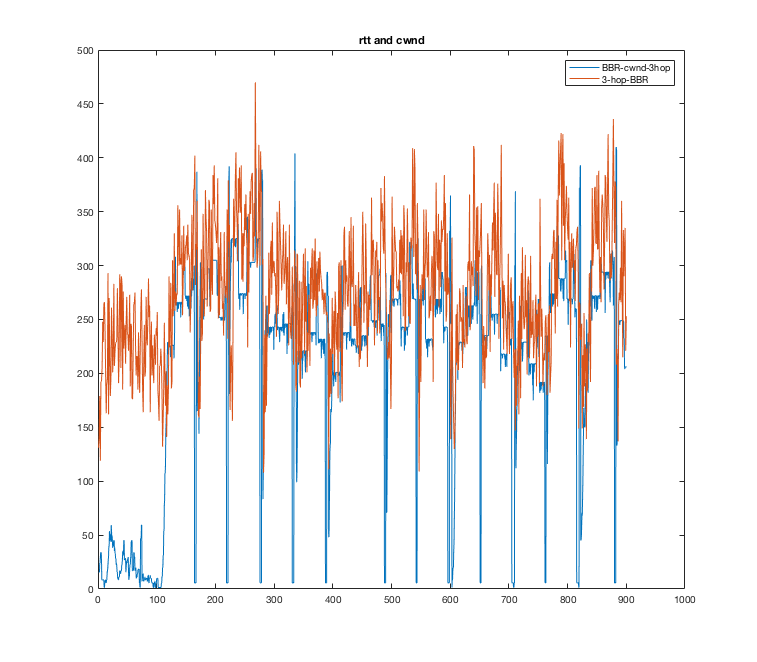


Figure 14:RTT and cwnd of 3-hop flow

when RTT goes up, cwnd goes up. The RTT determines the cwnd size, and cwnd determines the throughput. That's why the flow with larger RTT will occupy more bandwidth.

5. Conclusion and Future Work

For the single bottleneck link network with single flow, both BBR and CUBIC with AQMs work well in providing high throughput and controlling delay. In multiple bottleneck links network that different flows have different RTprop, BBR cannot provide good throughput. But Cubic with PIE can still provide good throughput as well as low latency. It suggests that in some scenarios, we can use CUBIC with AQM to replace BBR to get better performance.

The main cause of the unfairness of BBR is the cwnd size. The cwnd size (data in flight) is the product of cwnd\_gain and BDP, which is the product of the bottleneck bandwidth and minRTT. Thus, the flow with larger RTprop will havd larger cwnd size and get higher throughput.

Currently we only do the experiment when the link capacity is 10mbit. This link rate is still too low since bufferbloat becomes a big problem only when the link rate is large enough. In the following experiments, I will set the link capacity to be 50mbit, 90mbit, 100mbit, and 200mbit and study the performance of BBR and CUBIC with AQMs.

Reference

[1] Neal Cardwell, Yuchung Cheng, C. Stephen Gunn, Soheil Hassas Yeganeh, and Van Jacobson, "BBR: Congestion-Based Congestion Control ---- Measuring bottleneck bandwidth and round-trip propagation time".

[2] Rong Pan, Preethi Natarajan, Chiara Piglione, Mythili Suryanarayana Prabhu,

Vijay Subramanian, Fred Baker and Bill VerSteeg, " PIE: A Lightweight Control Scheme to Address the Bufferbloat Problem", Advanced Architecture & Research Group, Cisco Systems Inc., San Jose, CA 95134, U.S.A.

[3] K. Nichols, V. Jacobson, A. McGregor, ed., J. Iyengar, ed., "Controlled Delay Active Queue Management".

[4] Cardwell, Yuchung Cheng, C. Stephen Gunn, Soheil Hassas Yeganeh, and Van Jacobson, " BBR congestion control comes to GCP – your Internet just got faster", Google Cloud Platform Blog

[5] Neal Cardwell, Yuchung Cheng, C. Stephen Gunn, Soheil Hassas Yeganeh Ian Swett, Jana Iyengar, Victor Vasiliev Priyaranjan Jha, Yousuk Seung, Van Jacobson," BBR Congestion Control Work at Google", IETF 101: London, March 2018

[6] Toke Høiland-Jørgensen, Per Hurtig, Anna Brunstrom, " The Good, the Bad and the WiFi: Modern AQMs in a residential setting", Computer Networks, Volume 89, 4 October 2015, Pages 90-106