Performance Analysis of BBR Congestion Control Protocol Based on NS3

Hao Zhang*, Haiting Zhu*, Yu Xia*, Lu Zhang†, Yuan Zhang*, Yingying Deng*
*School of Internet of Things, Nanjing University of Posts and Telecommunications, Nanjing, China
†School of Information Engineering, Nanjing University of Finance and Economics, Nanjing, China
*Corresponding author: htzhu@njupt.edu.cn

***Abstract*—With the increasing complexity of the network communication environment. Research on TCP congestion control algorithm is one of the effective ways to alleviate network congestion. Google published the bottleneck bandwidth and round-trip time (BBR) congestion control algorithm in 2016. Unlike many currently loss-based congestion control algorithms (like BIC), BBR uses an estimate of the available bottleneck link bandwidth and RTT to govern its pacing rate. BBR has a good performance on Google's internal network and gains widespread attention. In this paper, we use NS3 to establish different network communication models. By analyzing the simulation data, we find that BBR outperforms BIC on high-latency, high-bandwidth network; BBR could periodically reduce the pacing rate to alleviate the bufferfloat problem; and there is a fairness problem between BBR and other versions of TCP protocols. In addition, we confirm that BBR flows have RTT fairness problem: a BBR flow with longer RTT dominates a competing flow with shorter RTT. Finally, we analyze the adaptability between BBR and AQM algorithms.**

***Keywords*—TCP; Congestion control; NS3; BBR;**

I. Introduction

Computer networks have experienced an explosive growth over the past few years and with that growth have come severe congestion problems[1]. Congestion control is still a topic of ongoing research[2]. But the primary congestion control mechanisms are no longer applicable to the current network environment.TCP based on AIMD (Additive Increase Multiplicative Decrease) algorithm have low performance in high-bandwidth and high-latency networks. The reason is that TCP continuously try to slowly increase the transmission window during the congestion avoidance phase. When the packet is lost, the congestion window is quickly reduced, and the effective bandwidth could not be fully utilized. Additionally, as the cache space of the core router continues to increase, the congestion control mechanism based on packet loss causes bufferbloat[3], which also affects network performance.

BBR is a new congestion control algorithm proposed by Google in 2016[4]. It is neither delay-based nor loss-based and it ignores packet loss as congestion signal. It builds a model of the network path consisting of its bottleneck bandwidth and RTT to adjust its pacing rate[5]. BBR is committed to solving two problems: Firstly, it could fully utilize the bandwidth on the network with high packet loss; Secondly, It could alleviate the buffer occupancy on the network. Google already deployed BBR in its own production platforms like the B4 wide-area network at Google.com and YouTube servers in the Internet to develop and evaluate BBR and provided quick integration of BBR with the Linux kernel (available since version 4.9)[6].

The remainder of the paper is organized as follows: Section II gives more detail about the BBR mechanisms. In section III, we presentd the characteristics of the network used in our experiments. The experiments and results described in section IV, we analyze the efficiency and fairness of BBR, and also analyze the applicability of BBR protocol, the compatibility between BBR and different AQM algorithms. SectionV summarizes the results of this evaluation.

II. Background and Related Work

A. TCP BBR Congestion Control Mechanism

As a new congestion control mechanism, BBR creates a model of the network by periodically estimating the available bottleneck bandwidth (BtlBw) and round-trip propagation delay (RTprop). The product is defined as the Bandwidth Delay Product (BDP), which are used to govern the rate packets are sent into the network and the maximum amount of data allowed in-transit[7]. BBR governs its sending behavior through two control parameters: Pacing rate and congestion window (Cwnd).

$$BDP = \mathrm{Btl}BW * RTprop \tag{1}$$

$$Cwnd = G * BDP \tag{2}$$

$$Pacing\ rate = G * BtlBW \tag{3}$$

The BBR congestion control algorithm has four states: Startup, Drain, Probe Bandwidth, and Probe RTT. BBR performs state switching based on the values of BtlBW and RTT. In the Startup phase, BBR uses a slow start mode like standard TCP to exponentially increase the transmission rate. When it finds that the effective bandwidth no longer grows, BBR will enters the Drain phase. In the Drain phase, the BBR slowly empties the data packets in the buffer that avoids taking up the buffer, and reduces the transmission rate until the RTprop is no longer reduced. In the Probe Bandwidth phase, BBR alternately detects bandwidth and delay. Bandwidth detection is based on a positive feedback system. The BBR periodically attempts to increase the packet transmission rate. When the received ACK rate increases, the packet transmission rate is further increased. At this stage, BBR sets 8 gain coefficients(G) of 5/4, 3/4, 1, 1, 1, 1, 1, 1,

978-1-7281-5141-0/19/$31.00 ©2019 IEEE
DOI 10.1109/CBD.2019.00071

respectively. When $G > 1$, BBR increases the transmission rate; when $G < 1$, BBR converges to reduce the transmission rate; when $G = 1$, keeps the current rate to send packets. When the link is congested, the BBR adjusts the gain coefficient while keeping the BDP unchanged. Once the congestion is relieved, the BBR could be restored to the original state in the first time. The BBR enters Probe RTT phase without a decrease in RTprop within 10s, and the send window is fixed to 4 MSS and the new RTprop value is estimated.

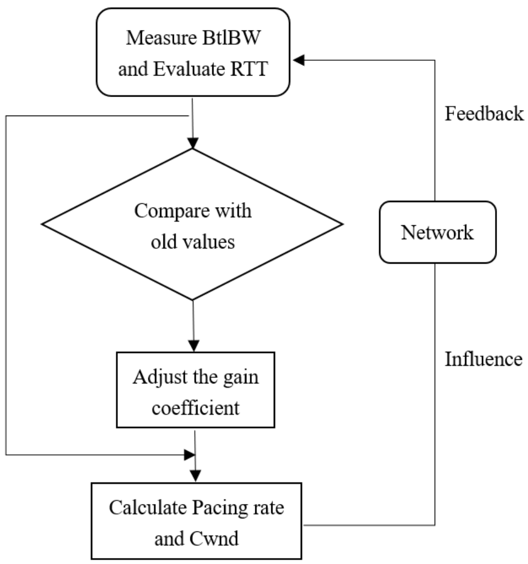


Fig. 1. The working principle of the TCP-BBR.

B. Related Work

Traditional TCP congestion control mechanisms could be divided into loss-based congestion control mechanisms and delay-based control mechanisms. Loss-based congestion controls (such as BIC-TCP[8] or TCP-New Reno[9]) use packet loss as congestion signal. Delay-based congestion controls (such as Vegas-TCP[10])use network latency as a signal of congestion.

In[11], Hurtig analyzed the impact of BBR's startup and steady-state mechanisms on variable-sized flows of a widely deployed loss-based CCA, CUBIC. Li et al.[12]showed that BBR is able to achieve better throughput than most proposed cellular network oriented CCAs while maintaining comparable or lower RTT than CUBIC. In this paper, we compare the throughput of BIC and BBR and analyze the four states of BBR in the experiment.

Brown[13] showed loss-based or delay-based congestion control favors short RTT flows, but BBR presents an opposite bias against short RTT flows. Ma's[14] in-depth analysis revealed the root cause of BBR's bias against short RTT flows. Tao et al.[15] modeled the bandwidth dynamics of BBR and presented the closed-formed for mulationon BBR's RTT fairness. Different from the above authors, we explain the cause of BBR's fairness problem with other TCP protocols from the principle of congestion control mechanism.

AQM algorithms share the same goal of congestion mitigation with TCP, the most effective congestion control occurs when AQM and TCP work together. Grazia et al.[16] analysis of the performance of different couples of TCP and AQM algorithms, and confirmed that CoDel has provided excellent performance balance with only a high queue occupancy. In[14] , the authors used two BBR flows competing on a bottleneck with different RED parameters to help the short RTT flow to regain slightly more bandwidth share. To the best of our knowledge, we are the first to evaluate BBR combined with different AQM algorithms. we find that the couple of BBR and CoDel could achieve excellent performance.

III. Experiments

We use the NS3 to build experimental models for simulation. NS3 is an open source, scalable network simulator. It provides a variety of Application Programming Interfaces for network emulation. The experimental scenario is the uplink and downlink simulation. We use N clients and N servers network model, and set different TCP protocols on clients. The experimental topology is shown in Fig. 2.

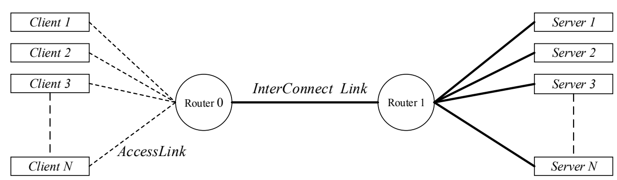


Fig. 2. Network simulation experiment topology.

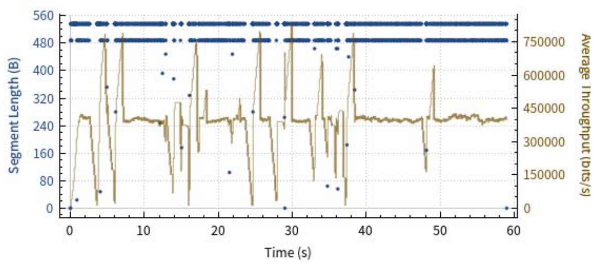
Since the NS3 does not support TCP-BBR for simulation experiments, NS3 needs to be extended. In this paper, we use a specific version of TCP-BBR that Vivek Jain extends for NS3[17].

In this paper, we conduct 5 simulation experiments. The evaluation considers achieved throughput, queuing delay, packet losses, fairness and different AQM algorithms. In experiment 1, we use a pair of nodes to install the BBR protocol and the BIC protocol on the client to analyze the throughput in the same network. In experiment 2, we use four pairs of nodes(clients send data to servers) and install BBR, BIC, New Reno, and Vegas on the clients to discuss the fairness of BBR. In experiment 3, we analyze the effect of RTT on BBR performance. In experiment 4, we analyze the effect of queue size on BBR by adjusting the queue length of intermediate nodes. In experiment 5, we evaluate the performance of BBR by selecting different AQM algorithms.

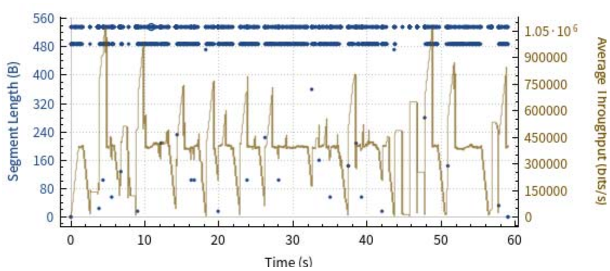
IV. Results

A. Throughput analysis

In experiment 1, BBR is deployed on the client. The link bandwidth is set to 10 Gbps, the delay is 0.02 ms, bandwidth from the client to the router0 is set to 1 Gbps, and bandwidth from the server to the router1 is set to 10 Gbps. Set the packet loss function to randomly drop packets. After deploying the BIC to the client in the same simulation network. The results are shown in Fig. 3. The throughput of the BBR is stable at 440 Mbps for most of the time, while the throughput of the BIC has been fluctuating around 440 Mbps.



(a) BBR throughput changes with time



(b) BIC throughput changes with time

Fig. 3. BBR and BIC throughput change with time.

It could be seen in Fig. 3(a) that the throughput of the BBR exhibits a periodic changing. Within 0 to 40s, the BBR detects the maximum bandwidth in a period of 10s, and calculates the pacing rate by the gain coefficient when the maximum bandwidth is no longer increased. Because the convergence of BBR is slow, the adjustment range of each round is 25% of the current window, resulting in an increase or decrease in throughput that is biased toward the peak. Within 40s to 50s, BBR continuously detects the maximum bandwidth and minimum delay during the stable operation phase, but the measured RTprop does not decrease compared to the minimum delay of the previous round evaluation, and it takes the initiative at 49s. Entering the Drain phase, the BBR quickly reduces the transmission rate, and then calculates the bandwidth delay product BDP according to the latest estimated minimum delay. Within 50s to 60s, BBR enters Probe BW phase to stably occupy bandwidth. Compared with the BBR, the throughput of the BIC is low and the throughput fluctuation is large. The result is shown in Fig. 3(b). Through the above analysis, the performance of the BBR is less affected by the packet loss link, and transmission rate could be stabilized at most times. But the BIC is affected by the packet loss and could not maintain the throughput stability with time.

B. Fairness analysis

Different clients deploy the different TCP protocol: Fairness refers to the ratio of TCP flows to network resources such as bandwidth and cache between different protocols. In experiment 2, BBR is deployed on the client1 to send data to the server1. BIC is deployed on the client2 to send data to the server2. The client3 deploys the Vegas and the client4 deploys the NewReno to send data to the server3 and server4. The interconnect link sets random packet loss, the link bandwidth of the client to the router0 is set to 1 Mbps, the delay is set to 20 ms, the bandwidth of the interconnect link is set to 2 Mbps, and the delay is 10 ms, the link bandwidth from the server to the router1 is set to 1 Mbps. The simulation results are shown in Fig.4.

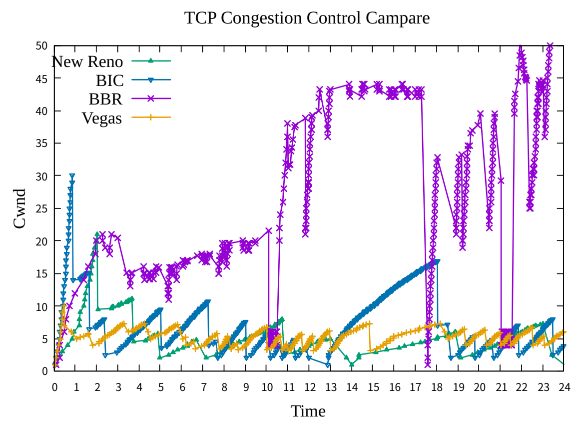


Fig. 4. Different TCP congestion windows change with time.

BBR has the highest link utilization. The goal of loss-based CCAs is designed to increase Cwnd and fill the cache space, but different TCP protocol has different threshold, so NewReno window growth time remains the longest in the slow start phase. Four clients send data to the corresponding servers. After the link cache space is full, NewReno and BIC rapidly reduce the Cwnd according to the AIMD algorithm. Although BBR finds the effective bandwidth no longer grows according to the received ACK, BBR will continue to occupy the bandwidth by using the previously estimated BtlBW and RTprop. As the link cache is full, the packet loss rate increases, NewReno has lower link utilization than BBR. Due to the preemption of link resources by different TCP protocols, the link delay increases, and the RTT will increase sharply. Faced with such false congestion, Vegas will still actively reduce the transmission window according to the increased RTT value. Therefore, Vegas has the lowest link utilization. Because BIC changes the Cwnd according to the idea of binary search, caused it's maximum congestion window is susceptible to packet loss.

By comparing the resource preemption of BBR and other TCP protocols , BBR could ensure that the larger Cwnd is continuously sents packets; When the link has a high packet loss rate, BIC needs to adjust the cwnd for a long time; Vegas stream has the weakest ability to seize resources; Compared to NewReno, BBR could make full use of bandwidth on network link.

Different clients deploy the same TCP protocol: The BBR is deployed on the client1 and client2, and the BIC is deployed on the client3 and client4. The interconnect link sets random packet loss, the link bandwidth of the client to the router0 is set to 1 Mbps, the delay is set to 20 ms, the bandwidth of the interconnect link is set to 2 Mbps, and the delay is 10 ms, the link bandwidth of the server to the router1 is set to 1 Mbps, the delay is set to 20 ms. The simulation results are shown in Fig. 5.

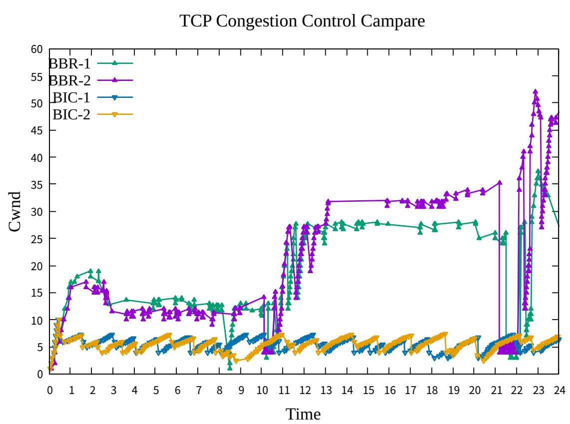


Fig. 5. Same TCP congestion windows change with time.

Client1 and Client2 of the BBR protocol have good fairness under the same link delay. However, BBR and BIC are unfair in terms of seizing resources. According to this experiment, we analyze four states of BBR congestion control algorithm. BBR is capable of detecting the maximum available bandwidth and minimum delay during the STARTUP phase, which is a fast acceleration phase. Based on the received ACK, the BBR confirms that the BtlBW is no longer increasing and then enters the Drain phase. Within 1s to 2s, BBR reduces the pacing rate detection minimum RTprop in the Drain phase. Within 2s to 10s, BBR enters the Probe BW phase, alternately detects BtlBW and RTprop, calculates BDP based on these two parameters, and adjusts Cwnd and pacing rate through gain. Therefore, BBR could stably Cwnd. When the BBR does not detect a lower RTprop within 10s, the BBR enters the robe RTT phase, in which the Cwnd is fixed to 4 MSS while the minimum delay is detected as the new delay estimate. Within 10s to 11s, the window size of the BBR drops sharply, in order to reduce the link delay and prevent the BBR from occupying the link buffer and causing bufferbloat. This process takes up 2% of the time. BBR entered a new round of state control mechanism within 11s to 23s. The Cwnd of BBR could be kept stable in a round state, and the periodicity shows an upward trend.

Through this experiment, we have found that there is a fairness problem between BBR and other TCP. Therefore, we explain the important reasons for the fairness of BBR and other TCP. BBR could increase the transmission rate on a link with a high packet loss rate. Whether New Reno or BIC congestion control algorithm based on packet loss or the Vegas congestion control algorithm based on delay, external events are required to prompt the TCP link to be in a congested state. The TCP congestion control state machine directly dominates the congestion algorithm adjustment window as long as it detects packet loss. However, BBR is a feedback driver, built-in independent speed regulation mechanism, actively adjusts the transmission window, which is not controlled by the TCP congestion control state machine, and independently completes the switching of four states. This is an important cause of the fairness between BBR and other TCP protocols.

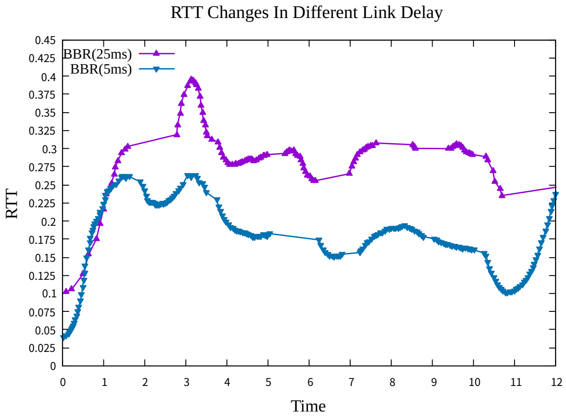
C. *Queue and RTT analysis*

Effect of RTT on performance of BBR: To verify whether the BBR flow has RTT fairness problem. In experiment 3, the link delay is set to 5 ms for the client1 to router0, and the bandwidth is set to 1 Mbps; the link delay from the client2 to the router0 is set to 25 ms. The bandwidth is set to 1 Mbps; the bandwidth of the interconnect link is 2 Mbps, and the delay is 20 ms, the bandwidth of the server to the router1 is set to 1 Mbps, the delay is set to 20 ms. The BBR is deployed on both client1 and client2.

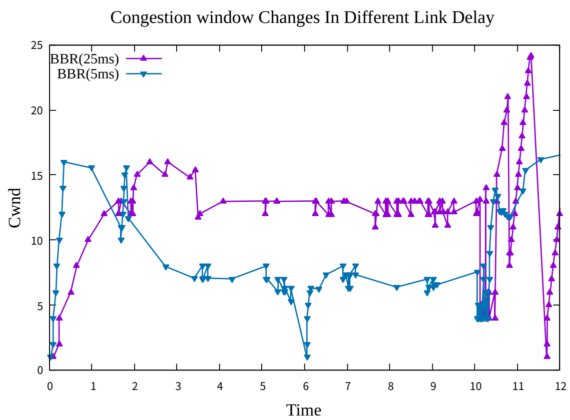
In the previous section we have verified that BBR maintains good fairness with the same RTT value, as in Fig. 5. The traditional AIMD-based congestion control algorithms could better utilize the link bandwidth when the RTT of the data stream is small. However, the BBR congestion control algorithm is contrary to the traditional TCP congestion control algorithms. A larger RTT could make the BBR occupy more link bandwidth. The result is shown in Fig. 6(a). The RTT value of client1 is large, which results in a high link utilization rate. The data flow on client2 has a smaller RTT value, which results in a low link utilization rate. The result is shown in Fig. 6(b).

This simulation verified the existence of RTT unfairness in BBR. In the same bottleneck link, a BBR flow with a large RTT value could obtain most of the bandwidth, while a BBR flow with a small RTT value could obtain less bandwidth. The short RTT flows result in limited BBR performance. The RTT fairness is the future direction of BBR improvement.

Effect of queue size on performance of BBR: When the TCP protocol is in the slow start phase, the instantaneous burst of data packets will exceed the actual carrying capacity of the network. Therefore, a queue cache is needed to temporarily store these over-sent packets. However, the traditional TCP congestion algorithms do not recognize the existence of the queue and considers that the queue cache is the remaining bandwidth and fill up until an overflow occurs. In experiment 4, the link bandwidth of the client to the router0 is set to

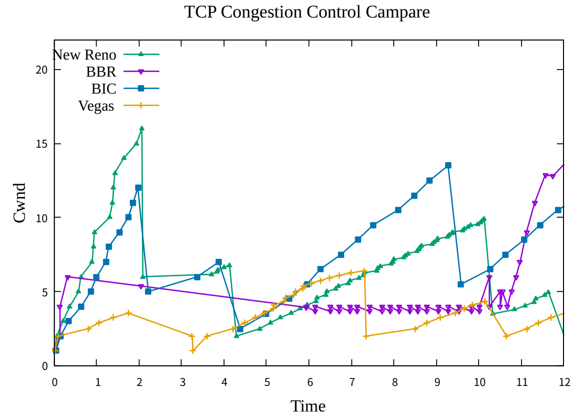


(a) BBR protocol RTT value changes with time

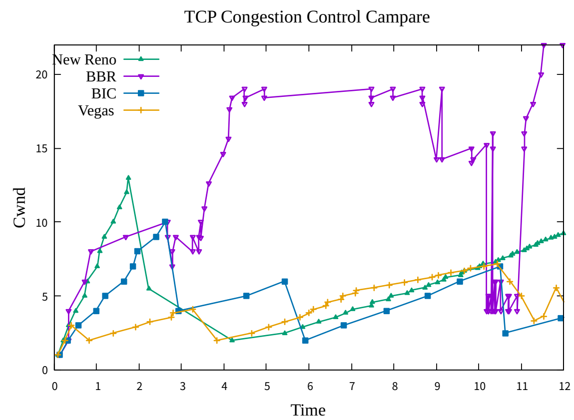


(b) BBR protocol Cwnd value changes with time

Fig. 6. Effect of different RTT values on the performance of BBR.



(a) Setting the queue size to 40



(b) Setting the queue size to 25

Fig. 7. Effect of different queue sizes on the performance of BBR.

1 Mbps, the delay is set to 20 ms, the bandwidth of the interconnect link is set to 2 Mbps, and the delay is 10 ms, the bandwidth of the server to the router1 is set to 1 Mbps, the delay is set to 20 ms. We use the default active queue management algorithm FIFO to set the queue size(threshold). Random packet loss after the queue size exceeds the packet loss threshold (L).

As in Fig. 7(a). when the queue size is set to 40, the Cwnd of BBR is maintained at 5 MSS; the traditional TCP protocols' Cwnd are higher than BBR; The purpose of BBR convergence is to actively reduce the pacing rate to avoid the bufferbloat problem, but the other TCP protocols are to fill the queue buffer space, which directly causes BBR to starve in the deep queue. As shown in Fig.7(b), when the queue size is set to 25, the performance of the BBR returns to normal. Based on the above analysis, a problem has also arisen. In the traditional queue management algorithms, how to set queue size threshold and the packet loss rate for different versions of the TCP protocol is a major challenge. How to associate the packet loss rate with the queue size is an improvement direction. The change in queue size causes a large discrepancy in BBR performance.

D. Different AQM algorithms combined with BBR

The default algorithm performs a Random drop after the queue size reaches a preset threshold. It is used to remind the TCP protocol to reduce the transmission rate. Setting different packet loss rate and packet loss threshold will affect the performance of BBR. In this paper, we set different active management queue algorithms on routing nodes, and analyze different couple of TCP and AQM algorithms. In experiment 5, the BBR is deployed on the client1, and the BIC is deployed on the client2. we set the FIFO on routing nodes, the link bandwidth is 1 Mbps, and the delay is set to 20 ms for simulation. Then we set the CoDel on the routing nodes for simulation. The result is shown in Fig. 8.

Compared with the FIFO, CoDel has provided excellent performance with BBR and BIC. It could seen that the BBR has a stronger bandwidth occupancy capability than BIC. Setting different active queue management algorithms directly affect the performance of the BBR. The packet loss strategy of the CoDel is that the more the data flow is in the queue,

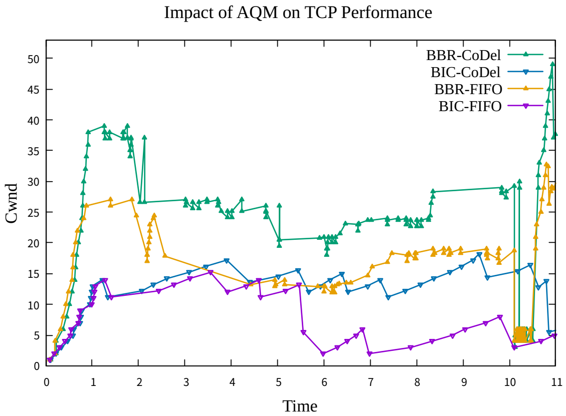


Fig. 8. Effect of different AQM on TCP performance.

the more likely the data packet is discarded. The CoDel itself also adaptively adjusts the packet loss rate without external configuration, and could achieve its performance with lower algorithm complexity. The CoDel has good adaptability with BBR, which mainly depends on the BBR's pacing rate control mechanism. The BBR data stream does not suddenly reach the bottleneck link. We could see from Fig. 5 that even if two BBR flows will adjust the data flow to equalize the bandwidth through Probe RTT, BBR will not maliciously occupy the cache space of the link. From the results, BBR and CoDel achieves good performance comparable to BBR and FIFO.

V. Conclusion

In this paper, we analyzed and evaluated the performance of the TCP-BBR protocol based on the NS3. We use throughput, queuing delay, packet loss, and fairness as the evaluation factors, and set up multiple simulation environments. Firstly, we conducted a throughput comparison experiment between BIC and BBR, and analyzed the reasons for the high throughput of BBR. Secondly, we conducted the fairness analysis of BBR and other TCP protocols. We explained the cause of BBR's fairness problem from the principle of congestion control mechanism, and use one of the experiments to analyze the four states of BBR. Then we analyzed the disadvantage of BBR and verified that BBR has weak competitiveness in short RTT flows. Finally,we found that the couple of TCP-BBR and CoDel could provide excellent performance than FIFO. How to solve short RTT flow problem and the combination of AQM algorithms and BBR could be further studied to improve the performance of BBR.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (Grant No. 61502250, No. 71801123, and No. 61872191) and sponsored by NUPTSF (Grant No. NY214188).

References

[1] V. Jacobson, "Congestion avoidance and control," in *ACM SIGCOMM computer communication review*, vol. 18, no. 4. ACM, 1988, pp. 314–329.

[2] M. Hock, R. Bless, and M. Zitterbart, "Experimental evaluation of bbr congestion control," in *2017 IEEE 25th International Conference on Network Protocols (ICNP)*. IEEE, 2017, pp. 1–10.

[3] J. Gettys, "Bufferbloat: Dark buffers in the internet," *IEEE Internet Computing*, no. 3, p. 96, 2011.

[4] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh, and V. Jacobson, "Bbr: Congestion-based congestion control," 2016.

[5] N. Cardwell, Y. Cheng, S. Yeganeh, and V. Jacobson, "Bbr congestion control," *Working Draft, IETF Secretariat, Internet-Draft draft-cardwell-iccrg-bbr-congestion-control-00*, 2017.

[6] D. Scholz, B. Jaeger, L. Schwaighofer, D. Raumer, F. Geyer, and G. Carle, "Towards a deeper understanding of tcp bbr congestion control," in *2018 IFIP Networking Conference (IFIP Networking) and Workshops*. IEEE, 2018, pp. 1–9.

[7] A. Peterson, "Abbrate: Automating bbr congestion control attack exploration using a model-based approach," Ph.D. dissertation, Northeastern University, 2019.

[8] L. Xu, K. Harfoush, and I. Rhee, "Binary increase congestion control (bic) for fast long-distance networks," in *Ieee Infocom*, vol. 4. INSTITUTE OF ELECTRICAL ENGINEERS INC (IEEE), 2004, pp. 2514–2524.

[9] L. A. Grieco and S. Mascolo, "Performance evaluation and comparison of westwood+, new reno, and vegas tcp congestion control," *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 2, pp. 25–38, 2004.

[10] K. Srijith, L. Jacob, and A. L. Ananda, "Tcp vegas-a: Improving the performance of tcp vegas," *Computer communications*, vol. 28, no. 4, pp. 429–440, 2005.

[11] P. Hurtig, H. Haile, K.-J. Grinnemo, A. Brunstrom, E. Atxutegi, F. Liberal, and Å. Arvidsson, "Impact of tcp bbr on cubic traffic: A mixed workload evaluation," in *2018 30th International Teletraffic Congress (ITC 30)*, vol. 1. IEEE, 2018, pp. 218–226.

[12] F. Li, J. W. Chung, X. Jiang, and M. Claypool, "Tcp cubic versus bbr on the highway," in *International Conference on Passive and Active Network Measurement*. Springer, 2018, pp. 269–280.

[13] P. Brown, "Resource sharing of tcp connections with different round trip times."

[14] S. Ma, J. Jiang, W. Wang, and B. Li, "Towards rtt fairness of congestion-based congestion control," *CoRR*, 2017.

[15] Y. Tao, J. Jiang, S. Ma, L. Wang, W. Wang, and B. Li, "Unraveling the rtt-fairness problem for bbr: A queueing model," in *2018 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2018, pp. 1–6.

[16] C. A. Grazia, N. Patriciello, M. Klapez, and M. Casoni, "A cross-comparison between tcp and aqm algorithms: Which is the best couple for congestion control?" in *2017 14th International Conference on Telecommunications (ConTEL)*. IEEE, 2017, pp. 75–82.

[17] [Online]. Available: http://network-simulator-ns-2.7690.n7.nabble.com