Congestion Control Algorithms: A Comparative Study of Cubic and BBR

Yaosen Zhang
University of California, San Diego
La Jolla, CA, USA
yaz001@ucsd.edu

ABSTRACT

1 INTRODUCTION

Efficient congestion control algorithms are crucial for optimizing network performance, particularly in diverse applications such as bulk data transfer and video streaming. Cubic[2] and BBR[1] are two widely studied congestion control algorithms, each with unique mechanisms for managing congestion and utilizing available bandwidth. Cubic employs a logarithmic growth model to ensure stability and fairness, while BBR aggressively probes for available bandwidth to maximize throughput.

In this study, I evaluate the performance of Cubic and BBR under various network conditions using a carefully designed experimental setup. The experiments are conducted on three AWS EC2 t2.micro instances running Amazon Linux 2023 within the same Virtual Private Cloud (VPC). Tools such as iperf3, FFmpeg, and the ss utility are used for performance measurement, while the tc tool is employed to simulate network conditions, including delays and packet losses. The evaluation involves testing with different traffic patterns, such as bulk data transfer and video streaming, under a consistent network delay of 100 ms, with added packet loss rates of up to 0.01% for certain scenarios.

This report provides a comprehensive analysis of the throughput, round-trip time (RTT), flow completion time, and fairness of these algorithms under varying conditions. The source code used for this study is available on my Github repository: **https://github.com/Neb345/cse222a_project_cca_analysis**. By understanding their strengths and limitations, this study aims to provide insights into the suitability of Cubic and BBR for different network applications.

2 EXPERIMENT

2.1 Setup

I deploy three AWS EC2 t2.micro instances running the Amazon Linux 2023 operating system within the same VPC to serve as the sending and receiving nodes on the network. The private IP addresses of the instances are designated for communication, and ports 8001 and 8002 are selected for use. The security group for each instance is configured to allow traffic specifically for the specified private IP addresses and ports.

2.2 Congestion Control on Bulk Traffic

I use iperf3 to simulate large file transfers, although it does not actually transfer real data. I use the tc tool to change the congestion setting and add a delay of 100 ms to effectively create a bottleneck bandwidth. Later I used the systemic tool to switch between the congestion control algorithm in use, either Cubic or BBR.

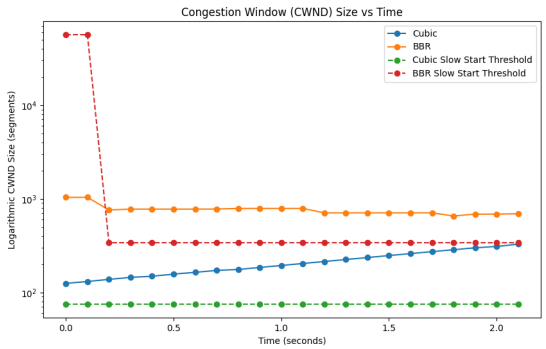

Figure 1: Congestion Window Size (Log-scale) vs Time during 100 ms network delay on the bulk traffic scenario.

The first scenario involves a sender transferring 100 MB of data to a receiver using iperf3. The congestion window (CWND) and the sender's slow start threshold (SSTHRESH) are measured using the ss tool, with an observation interval of 0.1 seconds. This setup allows tracking TCP statistics during the active connection.

Figure 1 illustrates the changes in the congestion window over time for two different congestion control algorithms (CCAs), along with the slow start threshold. Notably, the CWND for Cubic begins at a low value and logarithmically increases toward a threshold and stabilizes. In contrast, the CWND for BBR starts at a relatively high value, decreases gradually, and remains above its slow start threshold. the BBR's CWND drops significantly after the initial high value but recovers when close to its slow start threshold.

I examine the behavior of the two CCAs under packet loss conditions, specifically with loss rates of 0.005% and 0.01%, as shown by Figure 2 and Figure 3. In the presence of packet loss, as shown , Cubic's CWND tends to stabilize at a certain level, while BBR's CWND decreases in stages but later recovers, attempting to utilize more available bandwidth when possible. Cubic's high sensitivity to packet loss results in conservative bandwidth usage as losses increase. Although its logarithmic CWND growth ensures stability, it limits the algorithm's ability to aggressively utilize bandwidth in lossy networks. In contrast, BBR maintains higher CWND values and more efficient bandwidth utilization, even under packet loss conditions. This lower sensitivity to packet loss makes BBR a better fit for lossy networks.

Figure 4, 5, and 6 show the throughput of the data flow in the network for the two CCAs. We see that BBR demonstrates better initial bandwidth utilization and higher overall throughput. Meanwhile,

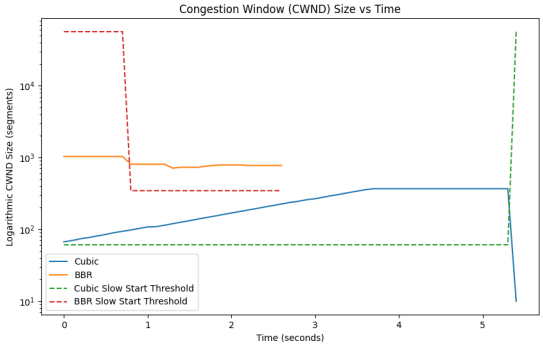

Figure 2: Congestion Window Size (Log-scale) vs Time during 100 ms network delay with 0.005% packet loss rate on the bulk traffic scenario.

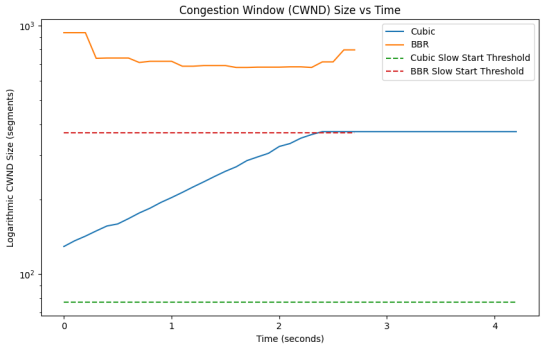

Figure 3: Congestion Window Size (Log-scale) vs Time during 100 ms network delay with 0.01% packet loss rate on the bulk traffic scenario.

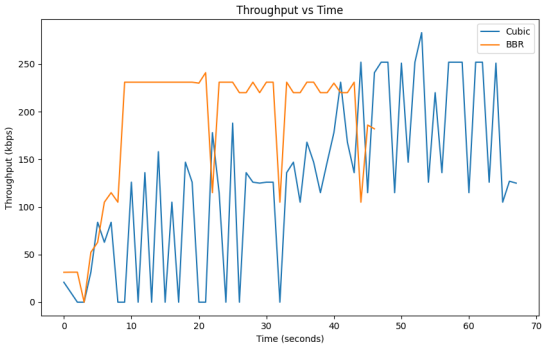

Figure 4: Throughput vs Time during 100 ms network delay on the bulk traffic scenario.

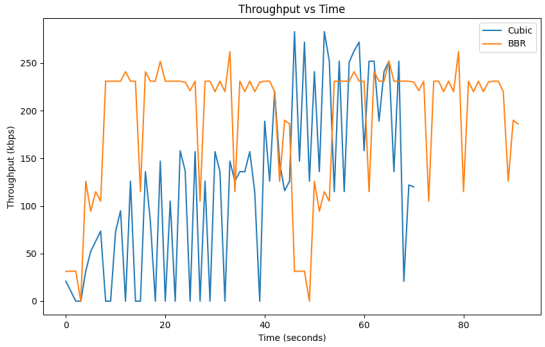

Figure 5: Throughput vs Time during 100 ms network delay with 0.005% packet loss rate on the bulk traffic scenario.

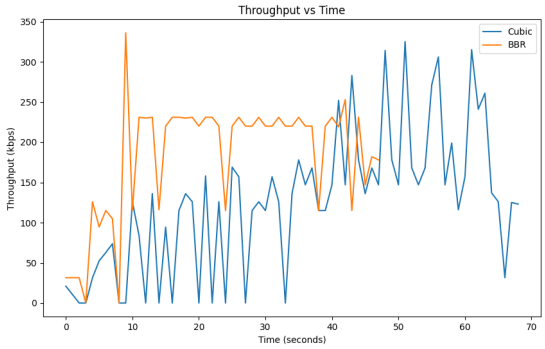

Figure 6: Throughput vs Time during 100 ms network delay with 0.01% packet loss rate on the bulk traffic scenario.

Table 1: Performance Comparison of Cubic and BBR Across Different Settings

Setting	Average Throughput (kbps)	Average RRT (ms)	Flow Completion Time (s)
Cubic1	125	121.261	6.7
BBR1	182	100.608	4.6
Cubic2	120	115.988	7.0
BBR2	186	100.377	4.5
Cubic3	123	123.517	6.8
BBR3	178	100.492	4.7

Cubic struggles with stability, potentially making it less efficient in high-latency environments.

2.3 Congestion Control on Video Streaming

To explore the behavior of the two CCAs in a different application, I use FFmpeg to conduct video streaming from a server to a client for measurement. With a network delay of 100ms, a sample video

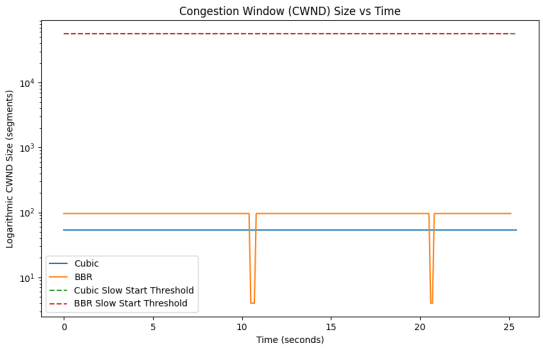

Figure 7: Congestion Window Size (Log-scale) vs Time during 100 ms network delay on the video streaming scenario.

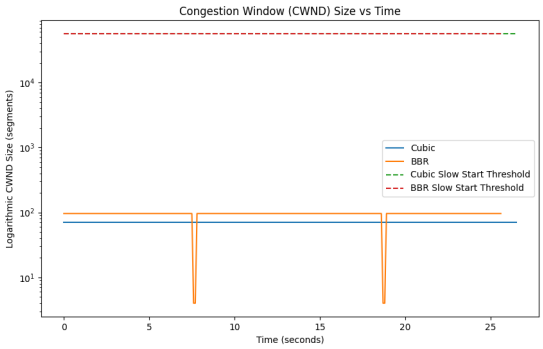

Figure 8: Congestion Window Size (Log-scale) vs Time during 100 ms network delay with 0.005% packet loss rate on the video streaming scenario.

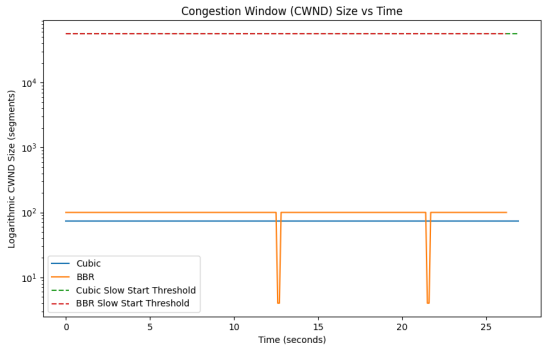

Figure 9: Congestion Window Size (Log-scale) vs Time during 100 ms network delay with 0.01% packet loss rate on the video streaming scenario.

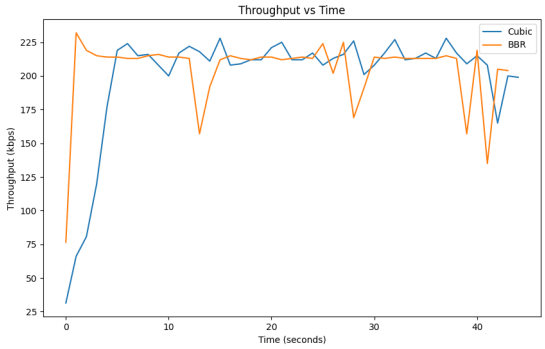

Figure 10: Throughput vs Time during 100 ms network delay on the video streaming scenario.

of size 20.7 MB was streamed, lasting approximately 34 seconds. The average RRT for Cubic is ms and that for BBR is ms. The streaming bitrate is 2916.8 kbps and remaines nearly consistent across experiments, with the streaming speed closely matching the original video playback speed. This set of experiments is comparable to those conducted for bulk traffic scenarios.

Unlike bulk traffic, video streaming consists of smaller, periodic bursts of data designed to ensure smooth playback and low latency. As shown by figure 3, 5, and 6, the constant CWND values for both Cubic and BBR indicate that they are maintaining a steady flow of data with minimal congestion or loss. Since the data demand in video streaming is steady and predictable, the CWND does not need to grow aggressively. BBR, however, starts with a higher CWND value and periodically drops to zero before recovering quickly. These periodic drops suggest that BBR is probing the network to assess available bandwidth, which may cause brief interruptions in video delivery. Nevertheless, BBR's quick recovery demonstrates its effort to maximize bandwidth utilization efficiently.

2.4 Network Fairness

In this experiment, a server transfers a 1 GB file to another server under a network delay of 100 ms. For a different configuration, a packet loss rate of 0.01% is introduced. Unlike the previous experiments, the observation interval is set to 1 second instead of 0.1 seconds.

From Figure 7, we observe that both CCAs achieve a similar average throughput of approximately 210 kbps over time in the absence of packet loss. BBR starts with a higher initial throughput, being unfair to Cubic, but experiences more frequent and brief drops. In contrast, Cubic's gradual throughput growth makes it more stable, though slower to fully utilize available bandwidth in the early phase.

In Figure 8, Cubic's throughput stabilizes around 210 kbps but shows slightly larger fluctuations compared to Figure 7, indicating the impact of packet loss. BBR, on the other hand, becomes less fair in this lossy network, particularly between 30 and 40 seconds, due to its instability and aggressive bandwidth probing during recovery phases. During this same period, Cubic's throughput drops

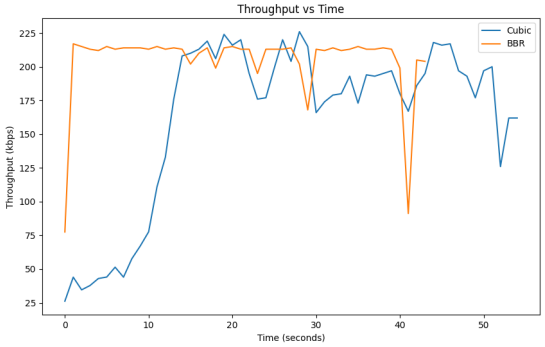

Figure 11: Throughput vs Time during 100 ms network delay with 0.01% packet loss rate on the video streaming scenario.

by about 30 kbps, reflecting its conservative approach to bandwidth utilization.

3 CONCLUSION

The comparative analysis of Cubic and BBR highlights significant differences in their performance and adaptability under varying network conditions. BBR consistently achieves higher throughput due to its aggressive bandwidth probing, making it ideal for scenarios where maximizing resource utilization is critical. However, its instability and reduced fairness, especially in lossy networks, limit its effectiveness in shared environments. In contrast, Cubic demonstrates greater stability and fairness but often underutilizes bandwidth due to its conservative growth model.

For bulk traffic, BBR's aggressive nature allows it to dominate, while Cubic lags in early utilization. In video streaming scenarios, both algorithms maintain steady throughput, but BBR's periodic bandwidth probing can cause brief interruptions. Regarding fairness, Cubic's conservative approach ensures more equitable bandwidth sharing, whereas BBR's aggressive strategy can disrupt balance, particularly in lossy environments.

Ultimately, the choice between Cubic and BBR depends on the specific application and network conditions. This study underscores the need for careful algorithm selection to optimize network performance while balancing throughput, stability, and fairness.

REFERENCES

[1] Neal Cardwell, Yuchung Cheng, C. Stephen Gunn, Soheil Hassas Yeganeh, and Van Jacobson. Bbr: Congestion-based congestion control. In *Proceedings of the 13th ACM Conference on Emerging Networking Experiments and Technologies (CoNEXT)*, pages 213–226. ACM, 2016.

[2] Sangtae Ha, Injong Rhee, and Lisong Xu. Cubic: A new tcp-friendly high-speed tcp variant. *SIGOPS Operating Systems Review*, 42(5):64–74, 2008.