Comparative Analysis of Congestion Control Algorithms: CUBIC vs. BBR in Bulk Traffic and Video Streaming Applications

Sidney Pan

December 2024

1 Introduction

Congestion control algorithms(CCA) prevent the network from becoming overloaded and ensure network stability. Among modern CCAs, CUBIC and BBR represent two different approaches. CUBIC is a loss-based algorithm that uses packet loss as a congestion signal and uses a cubic growth function to help optimize flow on high-speed networks. BBR is a rate-based algorithm that estimates bottleneck bandwidth and round-trip time (RTT) to provide high utilization and low latency in a wide variety of network environments.

This project compares the performance of CUBIC and BBR in bulk traffic and video streaming applications. Metrics like congestion window behavior, throughput, RTT, flow completion times will be examined under different conditions (normal condition and controlled packet loss conditions with 0.005% and 0.01%). Fairness between algorithms will be examined for bulk traffic.

2 Related Work

Traditional CCAs like TCP Reno are based on packet loss as the primary congestion signal and adjust the congestion window using additive increase and multiplicative decrease (AIMD). They show limitations in high-speed and long-delay networks due to the inability to use the available bandwidth efficiently.

CUBIC was introduced in 2008 and was designed to improve the scalability and fairness of TCP in high-speed and long-distance networks [HRX08]. CUBIC is an evolution of its predecessor BIC-TCP in which CUBIC replaces the linear window growth function with a cubic function, achieving more efficient bandwidth utilization and reducing sensitivity to RTT variations. This RTT-independence also ensures fairness among flows with different RTTs, which addressed limitations in TCP-Reno and TCP-SACK. However, CUBIC still has limitations of loss-based algorithms such as bufferbloat.

BBR (Bottleneck Bandwidth and Round-trip propagation time), developed by Google in 2016, is a rate-based CCA that directly measures and adapts to the bottleneck bandwidth and RRT [CCG+17]. BBR optimizes for maximum throughput and minimal delay by maintaining minimal queues at bottlenecks and pacing-based sending. BBR has been tested and deployed on Google's networks and the test showed BBT achieved 2 to 25 times the throughput of CUBIC.

3 Overview of Process

The project analyzes and compares the behavior of CUBIC and BBR under different applications. We set up a controlled network environment where two CCAs are tested against bulk data transfer and video streaming. A bottleneck was created between the two nodes by adding a delay at the sender. Metrics such as congestion window size, slow start threshold, throughput, RTT, flow completion times are measured. Packet loss under 0.005% and 0.01% are simulated to evaluate how the algorithms behave under different network conditions. Fairness is tested using simultaneous bulk traffic flows with both algorithms to analyze bandwidth allocation. Python is used for plotting.

4 Experimental Setup

4.1 Testbed Setup

Use two Linux-based nodes (Ubuntu 22.04 LTS) on AWS using T3-medium instances with 16GB of storage, with one node as the client and the other as the server. Connect to the instances via SSH and ensure network connectivity between them. Simulate a network bottleneck by introducing a 20ms delay at the sender using traffic control tools. Ubuntu 22.04 LTS has inbuilt CUBIC and BBR implementation.

4.2 Experiment Configurations

- Selected Applications:
  - Bulk Traffic: Simulate with iperf3.
  - Video Streaming: Prepare a video for streaming using ffmpeg, host it on the server with an HTTP server, and stream it on the client using VLC.
- Metrics: For Bulk Traffic scenario we analyze:
  - Congestion window behavior and slow start threshold for CUBIC and BBR.
  - Throughput comparison of CUBIC and BBR.

- Flow Completion Time (FCT) comparison of CUBIC and BBR.
- RTT comparison of CUBIC and BBR
- Under 0.005% and 0.01% packet loss rates, comparison of Throughput, Retransmission, Congestion window size for CUBIC and BBR.
- Fairness test on Throughput of two simultaneous flow of CUBIC and BBR.

For Video Streaming scenario we analyze:

- Congestion window behavior and slow start threshold for CUBIC and BBR.
- Throughput comparison of CUBIC and BBR.
- Under 0.005% and 0.01% packet loss rates, comparison of Throughput, Retransmission, Congestion window size for CUBIC and BBR.

5 Results

5.1 Bulk Traffic

5.1.1 Congestion Window and Slow Start Threshold over time

From Figure 1 we observe CUBIC's growth function has a convex phase when the CWND is far from the saturation point, which allows for aggressive growth to probe available bandwidth. The concave phase occurs when CWND approaches the previous maximum value and achieves a more cautious growth. Then after the smooth plateaus there's a sharp CWND growth.

Periodic drops in CWND indicate congestion events or packet loss that make CUBIC to decrease CWND and adjust the slow start threshold. Then the drops are always followed with a gradual recovery following the cubic function pattern we discussed in the last paragraph.

We can identify the slow start phase and congestion avoidance phase: the slow start phase happens at the beginning of the session or after congestion events, for example, at time around 0, 60, and 100 seconds. CWND increases exponentially as it aggressively probes for available bandwidth until reaching SSTHRESH. After this, it transitions to congestion avoidance phase and CWND grows more conservatively. This slow growth ensures network stability.

In Figure 2, unlike loss-based algorithms like CUBIC, BBR has CWND size oscillates around the bottleneck bandwidth-delay product (BDP). CWND in BBR also shows sharp fluctuations that reflects its bandwidth-probing behavior. BBR cycles through phases such as ProbeBW, ProbeRTT and others to find the optimal rate. The steep drops shows BBR reducing CWND during ProbeRTT to measure the minimum RTT.

Unlike CUBIC where SSTHRESH adjusts during congestion events to guide the transition of slow start and congestion avoidance, BBR has a constant SSTHRESH. This reflects BBR's mechanism of making SSTHRESH less relevant but focuses on maintaining a high throughput by probing for bandwidth.

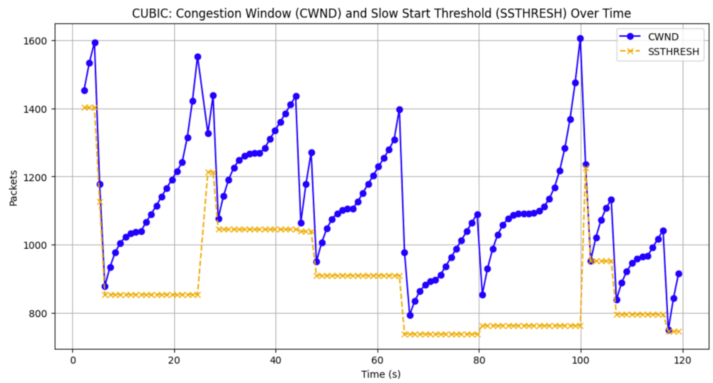

Figure 1: CUBIC: Congestion Window (CWND) and Slow Start Threshold (SSTHRESH) Over Time

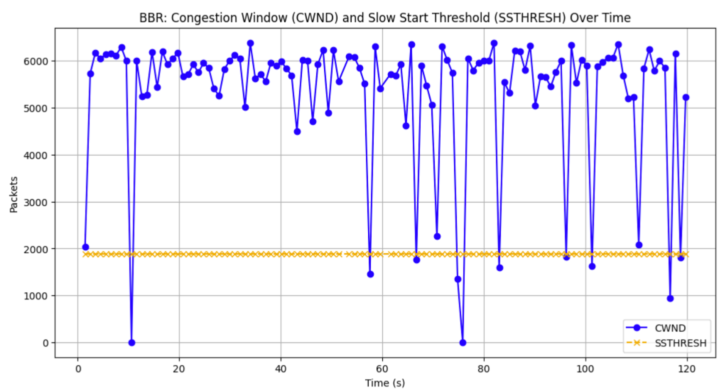

Figure 2: BBR: Congestion Window (CWND) and Slow Start Threshold (SSTHRESH) Over Time

5.1.2 Throughput Comparison

From Figure 3, the throughput of BBR is higher than CUBIC for all time. The throughput of BBR tends to fluctuate around a constant value, while CUBIC has periodic growth and decrease. BBR outperforms CUBIC because it actively estimates the available bandwidth and RTT and use this information to control the flow. This result relates BBR's ability to avoid excessive queuing and better usage of available capacity. The throughput pattern showed by CUBIC such as steep decreases is typical for loss-based algorithms. CUBIC also ramps up more slowly due to probing for capacity, leading to less efficient usage of network capacity comparing with BBR.

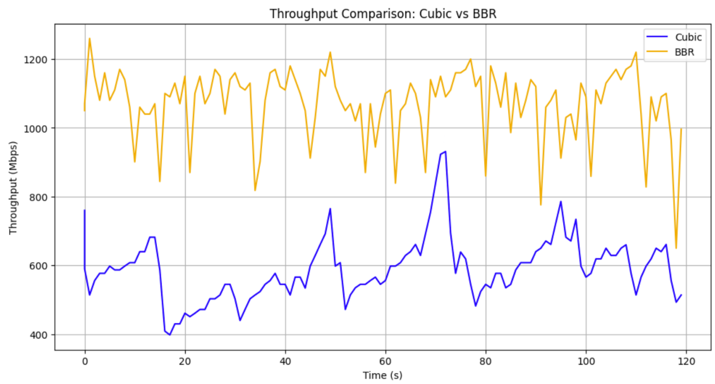

Figure 3: Throughput Comparison: Cubic vs BBR

5.1.3 Flow Completion Time (FCT) Comparison

We experiments on flow sizes of 1, 10M, 50M, 100M, 500M, 1000M. From Figure 4, BBR shows consistently lower FCT compared to CUBIC across all flow sizes. For small flows, the difference between CUBIC and BBR is small, while as the flow size increases, the difference grows significantly. This result shows BBR achieves lower latencies compared to CUBIC. This result aligns with BBR's ability of avoiding oscillatory behavior and high queuing delays in CUBIC.

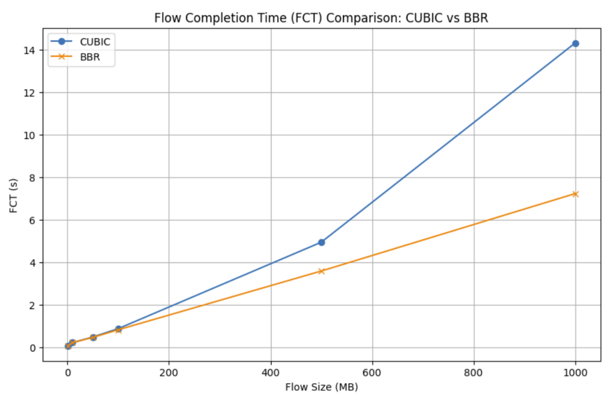

Figure 4: Flow Completion Time (FCT) Comparison: CUBIC vs BBR

5.1.4 RTT comparison

From Figure 5, both CUBIC and BBR have RTT with a narrow range (20-25ms). BBR has sporadic spikes to high RTTs such as 35ms or 50ms. CUBIC's steady RTT shows that its mechanism can void large RTT deviations. BBR's spikes reflects it continuously probes for available bandwidth during ProbeBW. This can cause temporary queuing at bottleneck and cause high RTT.

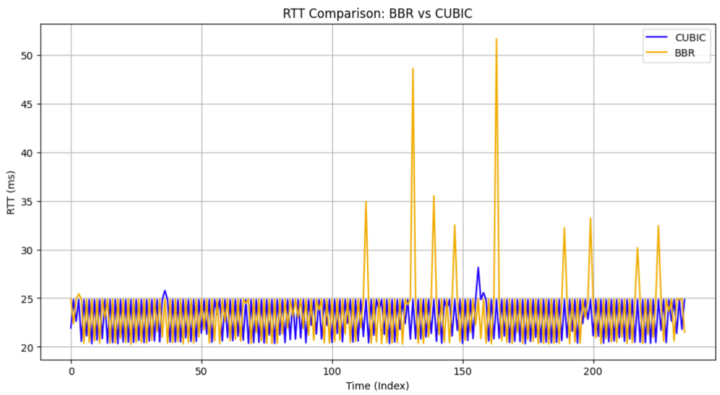

Figure 5: RTT Comparison: BBR vs CUBIC

5.1.5 Packet Loss Situations

In Figure 6, we compare the behavior of BBR and CUBIC under two levels of packet loss (0.01% and 0.005%) using metrics of throughput, retransmissions, and CWND.

For throughput, BBR is consistently higher than CUBIC in both loss levels with a more centered value. CUBIC has lower throughput at 0.01% loss, aligning with its reliance on packet loss for congestion control.

The middle plot shows that BBR has a higher retransmission count than CUBIC because it aggressively probes for available bandwidth. CUBIC has a constant and low retransmission rate in both loss rates. This aligns with CUBIC conservative congestion avoidance phase.

The third plot shows that BBR has larger and more varied CWND in both loss rates while CUBIC has more stable and smaller CWND following the cubic pattern (though not very obvious in the plot due to scale).

In the packet loss situations, BBR outperforms CUBIC more significantly. CUBIC is more stable and conservative but suffers in packet loss environments due to its loss-based mechanism.

5.1.6 Fairness

Figure 7 shows throughput over time for two flow of CUBIC and BBR running simultaneously. At the start, CUBIC has higher throughput than BBR and over time, the throughput of both flows begin to stabilize and fluctuate. The fluctuation reflects competition for the shared bandwidth. However, the plot does not show significant unfair bandwidth allocation. This could due to BBR and CUBIC have different mechanisms to use and respond to network conditions.

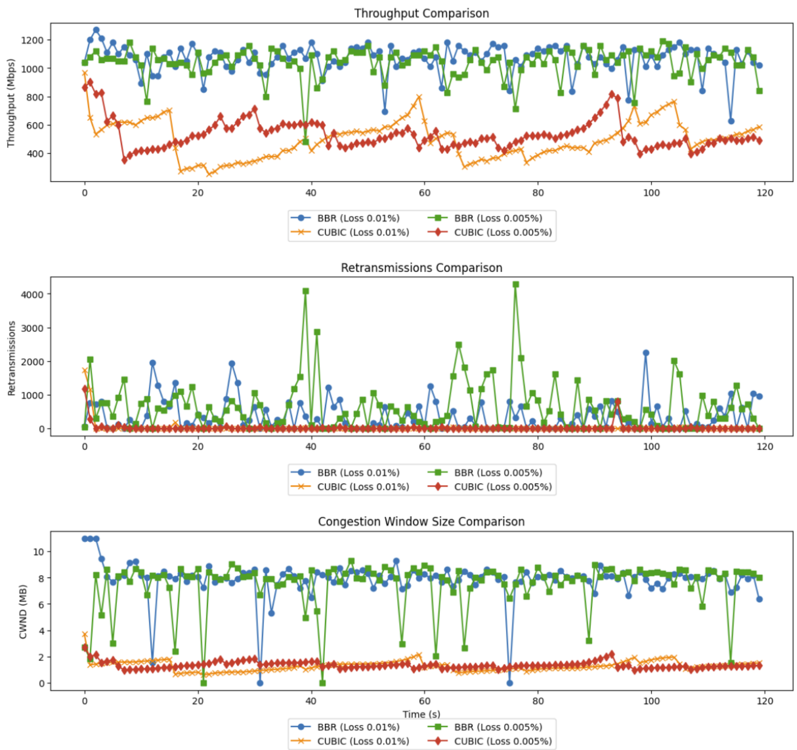

Figure 6: Throughput, Retransmission, Congestion Window Size comparison under packet loss rate 0.005% and 0.01%

5.2 Video Streaming

5.2.1 Congestion Window

Figure 8 and 9 show that both CUBIC and BBR show significant variability in CWND over time, with frequent dips and recoveries, which might represent congestion such as queuing at bottleneck or transient congestion events. The similar behavior might due to high utilization of available bandwidth that they both resides close to the bottleneck capacity.

5.2.2 Throughput Comparison

Figure 10 shows that both CUBIC and BBR have similar average throughput over the test duration, indicating that they are both capable of fully utilizing the available bandwidth for video streaming under the specified network condition. CUBIC is more steady and BBR has greater fluctuations. CUBIC's smoother throughput curve makes it better suited for video streaming applications requiring steady data rates.

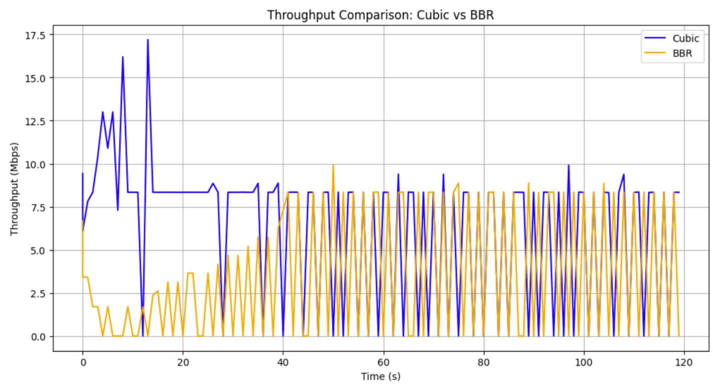

Figure 7: Fairness with Throughput Comparison: Cubic vs BBR

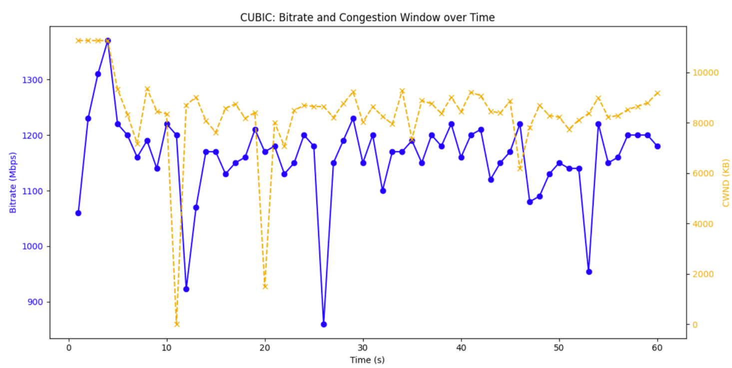

Figure 8: CUBIC: Bitrate and Congestion Window over Time (Video Streaming)

5.2.3 Packet Loss Situations

For video streaming, we can see from previous sections that the difference of CUBIC and BBR is not significant in normal conditions. However, under packet loss rates of 0.005% and 0.01%, from Figure 11 we observe that BBR outperforms CUBIC significantly, although not as significant as in the bulk traffic situations.

BBR achieves higher and more consistent throughput than CUBIC under both packet loss rates. CUBIC's loss-based mechanism struggles to maintain throughput in packet loss environments. The retransmission behavior is the same as we discussed in the bulk traffic section. BBR's CWND remains robust under packet loss, while CUBIC's CWND significant reduction decreases its ability to efficiently use available bandwidth in lossy conditions.

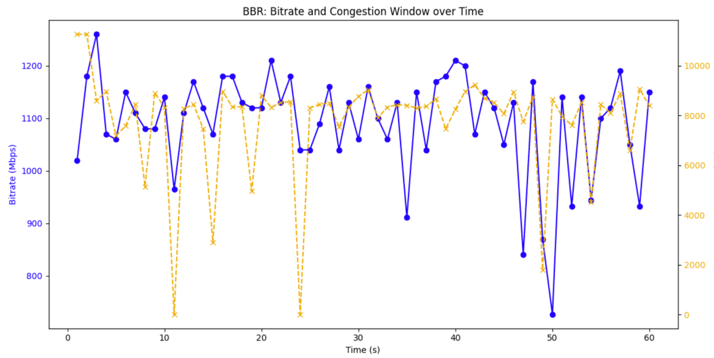

Figure 9: BBR: Bitrate and Congestion Window over Time (Video Streaming)

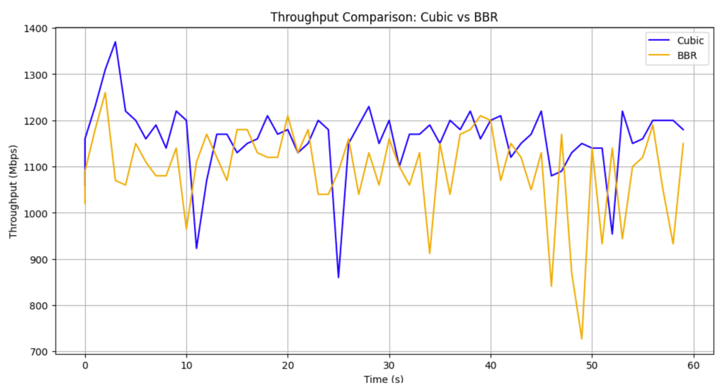

Figure 10: Throughput Comparison (Video Streaming): Cubic vs BBR

5.2.4 Comparison of Bulk Traffic and Video Streaming

BBR generally outperforms CUBIC in maintaining higher throughput and stability under various conditions. However, BBR's advantages are less significant in video streaming as we discussed. BBR test for available bandwidth by periodically increasing sending rate, which can cause bitrate fluctuations that reflected as interruptions or quality changes in video streaming. CUBIC's smoother and more steady behavior might align better with the video streaming's preference of consistent delivery.

6 Conclusions

This project compares CUBIC and BBR in bulk traffic and video streaming under varying conditions including different packet loss rates. BBR outperforms CUBIC in bulk traffic. However, in video streaming, BBR has similar performance but is less steady than CUBIC, while CUBIC's steady behavior makes

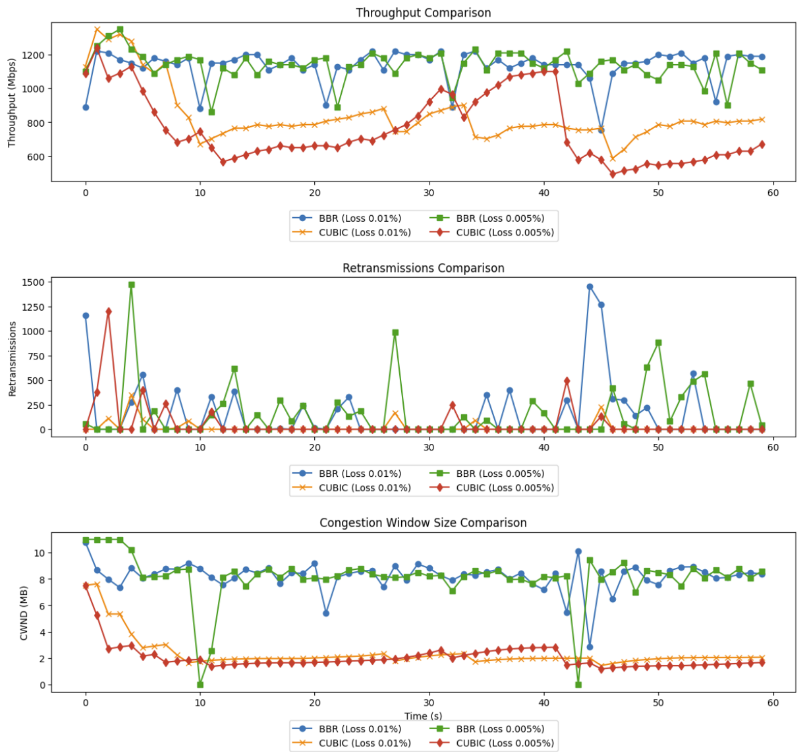

Figure 11: Throughput, Retransmission, Congestion Window Size comparison under packet loss rate 0.005% and 0.01% (Video Streaming)

it more suited for consistent delivery. BBR performs significantly better than CUBIC in both bulk traffic and video streaming, maintaining higher throughput and more steady congestion window, while CUBIC's inherent loss-based mechanism makes it struggles in lossy environments. In summary, BBR is better for high throughput and dynamic network environments, while CUBIC is better for applications that prefer stability like video streaming.

References

[CCG+17] Neal Cardwell, Yuchung Cheng, C. Stephen Gunn, Soheil Hassas Yeganeh, and Van Jacobson. Bbr: congestion-based congestion control. *Commun. ACM*, 60(2):58–66, January 2017.

[HRX08] Sangtae Ha, Injong Rhee, and Lisong Xu. Cubic: a new tcp-friendly high-speed tcp variant. *SIGOPS Oper. Syst. Rev.*, 42(5):64–74, July 2008.

Appendices

Github repo: `https://github.com/sid-0012/network-congestion-control`