Evaluating the bandwidth share model of BBR and CUBIC congestion control protocol

Shibo Xing
shx26@pitt.edu

Qingyang Li
qil77@pitt.edu

***Abstract*—Our paper validates and extends a prior work that models the bandwidth competition between *Bottleneck Bandwidth and Round-trip* (BBR) TCP congestion control protocol and the more common CUBIC protocol. Among all the miscellaneous parameters that affect the bandwidth shares of the two protocols, we have validated that bottleneck buffer queue size and the number of BBR protocols are two most dominant factors. Both factors revolve around the queue status since it determines the loss rate and round-trip time of the data transmission, which are the deterministic measurements of CUBIC and BBR respectively. Number of CUBIC protocols is concluded as an irrelevant variable of the bandwidth allocation, as suggested by our evaluation. We further examined the bandwidth share model by confirming that a network failure would shift bandwidth shares from CUBIC to BBR. To acquire this result, we emulated a lossy transmission environment.**

I. Introduction

BBR TCP congestion control is a RTT-driven transport layer protocol that searches for a suitable data transfer rate without actively incurring lost packets[Car+17]. As this Google-led protocol contributes more and more internet traffic in recent years, network researchers have turned their attention to its impact on the network bandwidth dynamics. Albeit BBR v1's initiative is to alleviate global network congestion phenomenon, researchers have been scrutinizing its fairness of bandwidth allocation to other traditional protocols like CUBIC and Reno[HBZ17; War+19; KC19].

Due to the disparities of bandwidth in different routers in the wide area network, buffer queue has been used to assist data transmissions to prevent extensive discarded packets. Thus, the status of buffer queue is the focal point of TCP protocols sending rate control. A recent work (BBR congestion model) attempts to model BBR and loss-based protocols' bandwidth shares in a simulated network delay environment[War+19], with independent variables like bottleneck buffer queue sizes and number of BBR data-flows along with other variables, as shown in equation 1. Due to the queue size's relation with BBR's estimate of Bandwith-Delay Product(BDP), BBR congestion model claims that bottleneck buffer queue size has a negative impact on BBR's fraction of the bandwidth. As described by the model, number of BBR data-flows increases the lower bound of the in-flight packets during BBR's probe state, affecting its RTT estimate and positively influencing its bandwidth share. The output of the model is a number from 0 to 1 that represents BBR's aggregate share of the bandwdith:

$$Probe_{time} = (\frac{q}{c} + .2 + l) \times \frac{d}{10}$$

$$BBR_{frac} = (\frac{1}{2} + \frac{cl}{2q} + \frac{4N}{q}) \times (\frac{d - Probe_{time}}{d}) \quad (1)$$

Parameter	Description
N	Number of BBR flows sharing bottleneck
q	Bottleneck queue capacity (packets)
c	Bottleneck link capacity (packets per second)
l	RTT when there is no congestion (seconds)
d	Flow completion times after convergence (seconds)

Table 1: BBR congestion model parameters [War+19]

In our paper, we validated the relationships between the independent variables and the output variable as described in the BBR congestion model by replicating parts of the experiments in [War+19]. We would qualitatively examine the modeled outputs in our testing environment and compare it to the observed results.

The BBR congestion model bases some of its modeling on the assumption that the number of CUBIC protocols has no effect on the bandwidth dynamics. Due to the insufficiency of loss-based protocols analysis in [War+19], we added in the number of CUBIC data flows as a new independent variable and used several statistical metrics to examine the correlation between CUBIC data flow count and the output variable.

We further extended the BBR congestion model by modeling a characteristic network scenario of which the bottleneck link failure occurs and causes packets drop in a fixed rate. Due to the complexity of modern wide area networks, congestion is not the only cause of packet loss. An array of other reasons like hardware or software defects, network attacks and overloaded CPU or I/O devices could also lead to lossy transmissions. As this is a type of real circumstance, we are interested to see its effects on the bandwidth allocations of data flows. Through analyzing the CUBIC protocol's congestion window mutation algorithm, we made the expectation that arbitrary loss rate would force CUBIC to yield more bandwidth to BBR. After observing further data transmissions with human-made loss rates on our testbed, we evaluated our expectation by comparing the aggregate BBR bandwidth with bottleneck loss to no loss,

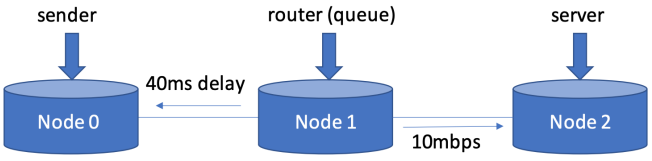

Fig. 1. Experiment testbed

while keeping all other parameters constant.

II. Experimental Setup

We set up our testing environment in a local-area network(LAN) site in Global Environment for Network Innovations(GENI), where we have created three nodes to act as the transmission sender, router and receiver. In order to emulate a network transmission with bandwidth disparities, we added a one-way delay of 40ms from the router to sender that would lag the ACK packets. We limited the bandwidth from router to receiver to 10mbps and deployed a buffer queue on router with size equal to the queuing parameter q. Both the delay in RTT, queue and outflow bandwidth limitation is accomplished with the Linux traffix control (tc) tool. iperf3 is used on sender and receiver to simulate concurrent data flows competing for bandwidth.

III. Experiments and Results

A. Data Collection

The first stage of our experiment consists of data collection. We used the aggregate bandwidth measured by iperf3 in bits per second as the output and collected a total of two datasets:

1) Dataset I: We sampled transmission data with parameters BBR count(1-33) and CUBIC count(1-33) with a fixed step and queue size (0.25, 0.5, 1, 4, 8, 16, 32), resulting in a 567×4 matrix. In order to achieve the convergence effect, we have ensured our experiment duration to be long enough to fill the bottleneck queue. We run the experiment for 60 seconds for the first 5 queue sizes in BDP numbers , 100 seconds for 16 BDP and 200 seconds for 32 BDPs. It took about 14 hours to collect the entire dataset. Automating this data collection task requires synchronization between sender and router. About 10% of our data was corrupted due to early experiment cut-off, for which we recovered using linear interpolation.

2) Dataset II: We repeated our method in Dataset I and added variable loss rate to the data collection process. We sample data transmission results with loss rate percentages as (1, 2, 5, 10, 20, 50), BBR and CUBIC number as combiations of 9 and 13 and queue size as 4 BDP. We ran each sample test for 60 seconds and collected a 24×4 matrix. The instructions on how to execute the data collection process is included in appendix.

B. Results and Evaluation

*1) **queue size** and **BBR number** test:* The two dominant factors are the queue size and number of BBR in BBR congestion model. In order to validate the efficacy of the relationship they express, we set a CUBIC number to a fixed value of 1, and other variables in Table 1. to the testbed default. Then we input our combinations of queue size and BBR number into the model to calculate our modeled outputs. We forsook the 64-BDP queue and experimented with queue size with 0.25 and 0.5 BDP instead. BBR congestion model did not take into account of the queue of less than 1 BDP, of which the output exceeds 1. We had to cap the output to 1 in such cases. Similar to [War+19] we qualitatively and quantitatively compared the modeled output and the observed BBR fraction of the bandwidth.

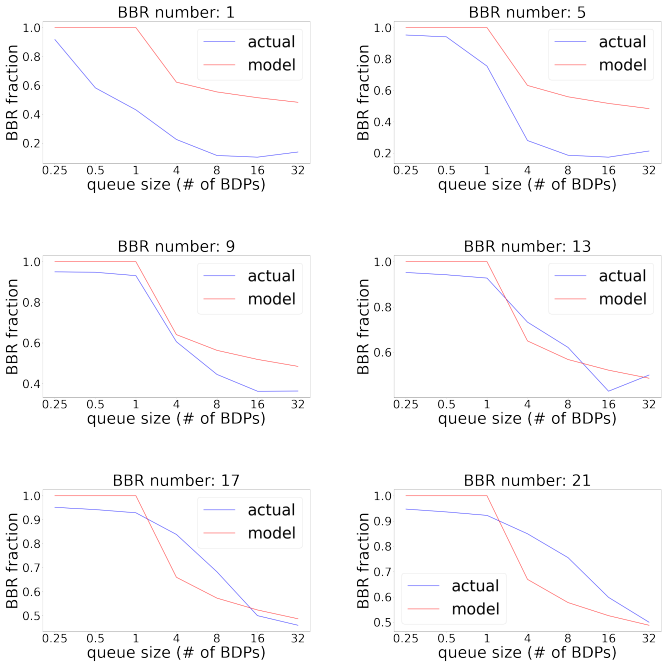

Fig. 2. queue size and BBR number test results

Figure 2 shows the results of our efficacy test. Qualitatively, we can tell that the claims made regarding queue size and BBR number are validated. There are some discrepancy between the model outputs and observed results in the graphs where BBR number is 1 or 5 We attributed this deviation to the lack of convergence stability. Based on our observation during the experiment, small number of data flows are more prone to fluctuating bandwidth allocations. As pointed out the BBR congestion model, BBR keeps the in-flight packets at $4 \times$ BBR number during the probe time. Therefore, competitions with higher BBR number have a higher likelihood of sustaining the congestion and the bandwidth allocations. We quantitatively evaluated the model approximation of the real bandwidth share using L2 norm of errors calculated over the queue sizes for each BBR number, as shown in Table 2. The result tells us that the best model approximation happens when BBR number is 13.

Loss rate	loss sequence error L1 norm
1	-0.058
2	0.122
5	0.271
10	0.514
20	0.466
50	0.927

Table 4: BBR congestion model parameters

BBR num	sequence error L2 norm
1	1.070
5	0.720
9	0.254
13	0.172
17	0.237
21	0.287

Table 2: BBR congestion model parameters

*2) **CUBIC number** test:* According to BBR congestion model, number of loss-based flows is not a factor that affects BBR fraction output since it does not influence the in-flight capacity, which is the auxiliary variable of BBR sending rate. [War+19] presents this assumption by showing that competition between 1 BBR flow and 16 CUBIC flows with a 32-BDP queue yields approximately the same result as a test with 1 CUBIC given all other parameters unchanged. In order to further validate this claim, we compared all loss number significance with other independent variables in BBR congestion model. Given BBR model congestion model and our observations, we can largely deduce that all three main variables have a linear relationship with the output. Thus, to test the correlation of the input parameters of BBR congestion model, we performed Pearson correlation test, single-factor regression and multi-factor regression on these variables. In table 3, we have shown the quantitative results of the statistical tests. The Pearson correlation results largely repeats the multi-factor regression's results since there is no multicollinearity among the variables. In all tests, CUBIC number consistently reports the worst performance in terms of factor correlation. The most convincing evidence would its 0.054 p-value of single-factor regression. It compels us to not reject the hypothesis that CUBIC number's correlation is indeed a result from random noise when using a common confidence interval of 95%. Based on these quantitative evidence, it is safe to say that CUBIC number does not play a factor in the BBR congestion model. This property could highlight BBR's fairness issues in a lot of cases. An overwhelming number of CUBIC flows does not lead to more fair allocation towards themselves. Although BBR is not widely implemented and not used frequently as the default TCP protocol like CUBIC, it can still tilt the network bandwidth dynamics when it is overwhelmed by number.

*3) **loss rate** test:* To analyze the loss rate effect on bandwidth, we visually compare the output result curves across dataset I and II. We selected a subset of parameters where queue size equals 4-BDP and protocol numbers are a combination set of 9 and 13. These are the parameters by which

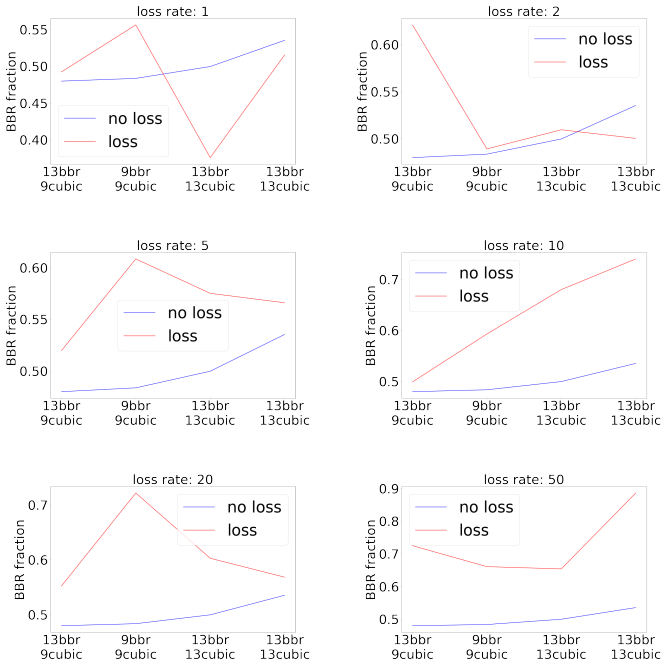

Fig. 3. queue size and BBR number test results

the modeled results have best converged to our observed results. Then, two subsets with these parameters are extracted from dataset I and II, where the latter contains loss rate as an extra parameter. Our expectation for comparison results is that CUBIC would be significantly disadvantageous in human-made non-congestion loss rate scenario. An link-failure induced packet loss may result in duplicated ACK packets or timeout for ACKs. According to the CUBIC congestion protocol, such scenarios could trigger the *Packet loss* or *Timeout* label. Former scenario would reduce the *ssthresh* by a portion and the latter would reset CUBIC congestion window size to 0. Thus, we predicted that BBR's allocation would benefit from this situation.

If we qualitatively analyze the results in Figure 3, we can tell that the large loss rates have caused CUBIC to yield bandwidth to BBR in various degrees. 1% and 2% loss rates did not cause a significant shift in the bandwidth dynamics. Same as our intuition, when the loss rate increases, the BBR acquires more bandwidth share. this relation is also mostly validated by our quantitative results shown in table 4, where we calculated the L1 norm of the error sequence between lossy transmissions' outputs and that of the loss-free transmissions. The metrics indicated that during a network failure scenario, CUBIC flows would be disadvantaged. We can envisioned that during partial network failure, BBR transmission may still suffer from the same loss rate as CUBIC, but it can alleviate the losses by seizing more bandwidth.

IV. Discussion and Future Work

We had some difficulty regarding replicating the results of BBR congestion model. One of the input variables of this model is the duration of the transmission after convergence.

	Pearson correlation	single-factor regression		multi-factor regression	
parameters	coefficient	R^2	p-val	coefficient	p-val
queue size	-0.435	0.190	0	-0.436	0
BBR number	0.228	0.052	0	0.229	0
CUBIC number	**-0.081**	**0.007**	**0.054**	**-0.081**	**0.027**

Table 3: CUBIC number test results

The implication of the definition of convergence in the paper is that queue being filled [War+19]. Yet the precise definition is not given by authors. Based on our observations, numerous competitions between protocols resulted in unsettled bandwidth shares oscillating in long period. Such a competition could casually be mistaken as convergence.

BBR congestion model is a complex one with a handful of variables. Some of the parameters matter less, such as the duration transmission, and some other unrealized parameters, such as loss rate, could be incorporated into the model to make it more complete. Given the complexity of today's network environment and protocols algorithms, there are still room for expanding this congestion model. As in the future work, we might be able to model the bandwidth from a loss-based protocol perspective and discover more factors and relationships related to the bandwidth.

V. Conclusion

We have validated the major claims and the assumption made by the BBR congestion model [War+19]. Their mathematical estimation of how queue size and the number of BBR data flows could affect the bandwidth allocations during a competition with CUBIC is verified by us through implementing their model and comparing the results to observed data. We also validated their claim about the independence between number of CUBIC data flows and the model output through statistical tests. While controlling other variables, we evaluated the effect on bandwidth share caused by the variations of non-congestion loss rate, a dominant factor that navigates the CUBIC sending rate. Our hypothesis that network failure could adversely affect the CUBIC bandwidth is then supported. The complexity of network environment potentially means that unrealized variables could contribute to the bandwidth analysis. Therefore, we wish to extend the model by exploring further relationships that could stem from loss-based protocols.

VI. Appendix

A. testbed setup

1) Create GENI nodes: To know how to create nodes on GENI, check GENI Education website [14] for detailed instruction. The site we were using to deploy our testbed is University of Chicago.

2) Enable BBR: Since September 2016, BBR has been published to Linux 4.9.0 or newer version. To enable BBR on GENI Linux machine, refer to the blog written by Jack Wallen [Wal18] for detailed instruction.

3) Install iperf3: Iperf3 is developed by Lawrence Berkeley National Laboratory and subject to perform active measurements for maximum achievable bandwidth on IP network. GENI has installed iperf2 as default but as developer of iperf3 stated that they rebuilt iperf3 and make it completely different from iperf2. To install iperf3 on GENI node, reference to the iperf3 homepage [EL]for detailed instruction.

4) Install Anaconda and Jupyter Notebook: We have provided the scripts written in .ipynb files for data collection and result analysis. To install the anaconda, please visit anaconda documentation [Anaa] for instruction. After installing complete, active conda environment to install jupyter by command *conda install jupyter*. After installing, follow anaconda documentation for port forwarding and visit jupyter notebook on localhost through port forwarding[Anab].

B. instructions on the experiment scripts

1) collect datasets: please consult Data_Collector_1.ipynb and Data_Collector_2.ipynb for parameter details

```
# on router
cd setup
source testbed_setup.sh
source exp1_setup [params]

# on server
source testbed_setup.sh
source exp1_run.sh 1 1 1 1 1

# on sender
source testbed_setup.sh
[run all cells in Data_Collector_1.ipynb]
# or
[run all cells in Data_Collector_2.ipynb]
```

2) evaluate results: directly run Data_Test.ipynb to use our pre-collected observation data stored in iperf3_results

```
# on sender
[run all cells in Data_Test.ipynb]
```

References

[14] *Geni education*. 2014. URL: https://www.cs.unc.edu/Research/geni/geniEdu/index.html.

[Car+17] Neal Cardwell et al. "BBR: congestion-based congestion control". In: *Communications of the ACM* 60.2 (2017), pp. 58–66.

[HBZ17] Mario Hock, Roland Bless, and Martina Zitterbart. "Experimental evaluation of BBR congestion control". In: *2017 IEEE 25th International Conference on Network Protocols (ICNP)*. IEEE. 2017, pp. 1–10.

[Wal18] Jack Wallen. *How to enable TCP BBR to improve network speed on linux*. Jan. 2018. URL: https://www.techrepublic.com/article/how-to-enable-tcp-bbr-to-improve-network-speed-on-linux/.

[KC19] Geon-Hwan Kim and You-Ze Cho. "Delay-aware BBR congestion control algorithm for RTT fairness improvement". In: *IEEE Access* 8 (2019), pp. 4099–4109.

[War+19] Ranysha Ware et al. "Modeling bbr's interactions with loss-based congestion control". In: *Proceedings of the internet measurement conference*. 2019, pp. 137–143.

[Anaa] Anaconda. *Installing on Linux*. URL: https://docs.anaconda.com/anaconda/install/linux/.

[Anab] Anaconda. *Running Jupyter Notebook on a remote server*. URL: https://docs.anaconda.com/anaconda/user-guide/tasks/remote-jupyter-notebook/.

[EL] Esnet and Lawrence Berkeley National Laboratory. *ESnet/Iperf: IPERF3: A TCP, UDP, and SCTP Network Bandwidth Measurement Tool*. URL: https://github.com/esnet/iperf.