Unraveling the RTT-fairness Problem for BBR: A Queueing Model

Yuechen Tao, Jingjie Jiang, Shiyao Ma, Luping Wang, Wei Wang, and Bo Li
Hong Kong University of Science and Technology

***Abstract*—BBR is a congestion-based congestion control algorithm recently proposed by Google. It proactively measures the bottleneck bandwidth and round trip times (RTTs) of a connection pipe, based on which it governs its sending behaviors. Despite the significant throughput gains and latency reduction, some experimental studies reveal that BBR may result in a salient RTT-fairness problem, in that short-RTT flows can be starved of bandwidth allocation when competing with long-RTT flows.**

In this paper, we study BBR's RTT-fairness problem from a theoretic perspective. We present a closed-form solution that characterizes the intrinsic dynamics of BBR flows and their interactions. Specifically, we model BBR's sending behaviors and bandwidth dynamics, based on which we establish an exponential relationship between the flows' bandwidth shares and their RTTs. We show that the degree of unfairness is dictated by the RTT ratio between two flows, irrespective of the other network parameters, such as the initial sending rates or link capacity. In particular, when the RTT ratio of the two flows is greater than 2, the short-RTT flow is starved of bandwidth allocation ($\leq 0.1\%$). Our theoretical results are corroborated by simulations in a wide range of settings.

I. Introduction

BBR [1] (Bottleneck Bandwidth and RTT) has attracted great attentions due to its high throughput, low latency and loss irrelevancy. Unlike traditional loss-based TCP congestion control (e.g., CUBIC [2]), BBR proactively estimates the bottleneck bandwidth and propagation delay of a connection pipe. Based on this information, BBR governs its sending rates, so as to deliver at the full bottleneck bandwidth without creating an excess queue in the pipe. According to Google [3], BBR achieved $133\times$ throughput gains in B4 network, and reduced the average delay by $\sim 80\%$ in YouTube [1].

However, prior experimental studies [4], [5] revealed that BBR has a significant bias towards long-RTT flows, in that long-RTT flows could take up almost all bandwidth, irrespective of the capacity of the bottleneck links. Despite those findings in experiments, prior works do not expose the intrinsic behaviors of BBR flows or explain the bandwidth dynamics. There are several important questions remained to be answered. First, existing work has not mathematically modeled BBR's bandwidth dynamics. Second, experimental results do not quantitatively capture the bias towards long-RTT flows. Third, it is not conclusive whether short-RTT flows can obtain any bandwidth shares, and under what conditions.

To answer those questions, we develop a theoretical model based on queueing network to rigorously analyze the behaviors of BBR flows. We study the relationship between flow bandwidth share and several key parameters including flow RTT, initial sending rate and network capacity. Our main contributions are summarized as follows:

1) The closed-form model of BBR bandwidth dynamics. We briefly review the BBR algorithm and derive the evolution of a BBR flow's sending behavior. Based on the sending rates and RTTs, we compute the flow's inflight in the next round trip. The inflight then determines the queue sharing and the estimated bandwidth, which in turn decides the next round of the sending rate. Through this feedback loop, we are able to quantify BBR's bandwidth dynamics and prove that analyzing the sending rate is sufficient to characterize BBR's behavior.

2) The quantified relationship between the flows' bandwidth share and their RTT difference. We undertake a rigorous analysis of the parameters that determine the bandwidth share of a long-RTT flow and a short-RTT flow. We find that the bandwidth share of the long-RTT flow increases *exponentially* with the RTT ratio. This formulation further confirms the experimental observation that a marginal increase in RTT difference may result in a huge difference in bandwidth share [4]. Furthermore, the final bandwidth share of the two flows is dictated by the RTT difference. A larger network capacity only accelerates the exponential growth toward the final share but has no influence on the final value, whereas the initial rate of the two flows has no influence at all.

3) The lower bound on the bandwidth share of the short-RTT flow. Although experimental results show that the long-RTT flow quickly obtains 99.9% bandwidth share in most cases [4], we are able to discover the conditions on whether there exists a lower bound ($> 0.1\%$) for the short-RTT flow's bandwidth share.

We find that the RTT ratio of the two flows determines the final bandwidth share. A larger RTT ratio implies a lower possibility of getting more than 0.1% bandwidth share for the short-RTT flow. When the ratio is larger than a derived threshold, the short-RTT flow is essentially drained out. On the other hand, when the ratio is smaller than the threshold, the short-RTT flow can still get some bandwidth ($> 0.1\%$), the amount of which depends on its RTT. For example, a flow with 200-ms RTT will not be starved when competing with a 240-ms flow, where the RTT ratio is 1.2. In contrast, a 5-ms flow only gets 0.1% bandwidth share when competing with a 6-ms flow, though the RTT ratio remains 1.2.

The research was support in part by RGC GRF grants under the contracts 16211715 and 16206417, a RGC CRF grant under the contract C7036-15G.

II. Related work

For loss-based congestion control protocols, these are plenty of theoretic analyses on the RTT fairness problem. In [6] and [7], the authors have modeled the RTT fairness of the traditional AIMD algorithm for synchronous and asynchronous flows. In [8], Poojary and Sharma used a Markov chain to analyze the fairness between CUBIC [9] and TCP NewReno [10] towards RTTs and loss rates. Alizadeh *et al.* applied a fluid model to qualify the bandwidth allocation in DCTCP [11], [12]. For delay-based congestion control protocols, Boutremans *et al.* mathematically analyzed the sensitivity of Vegas [13] towards RTT estimation [14].

Experiment-based works found the severe RTT unfairness problem of BBR flows [4], [5]. In [4], Ma *et al.* further demonstrate that the bias towards long-RTT flows is pervasive under different network settings. However, neither of them analyzed this issue mathematically. To the best of our knowledge, we are the first to model the bandwidth dynamics of BBR and present the closed-formed formulation on BBR's RTT fairness.

III. Modeling BBR's Behavior

A. BBR Behavior

BBR uses the maximum bottleneck bandwidth (`MaxBw`) and the minimum RTT (`MinRTT`) to model the network. By using `MaxBw` as the sending rate, a BBR flow can transfer data at full bottleneck bandwidth without creating an excess queue, hence minimizing the transfer delay.

A BBR flow periodically probes for more bandwidth by pacing faster than the `MaxBw`. If it increases the `pacing_gain` to 1.25, meaning that the flow sends data 25% faster than the current `MaxBw`. Therefore, higher `MaxBw` can be detected by a `pacing_gain` greater than 1. This, however, may create an excess queue in the bottleneck link, which could be drained out by setting the `pacing_gain` less than 1.

There are there phases in BBR, `Startup`, `Drain` and `Steady State`. A BBR flow spends most of the time in the `Steady State`. Therefore, we will focus on the flows' bandwidth dynamics during this phase. The `Steady State` contains two modes, `ProbeBW` and `ProbeRTT`. In the `ProbeBW` mode, BBR cycles through eight phases with the following `pacing_gain` values: 5/4, 3/4, 1, 1, 1, 1, 1, 1. With a `pacing_gain` greater than 1, this BBR flow probes for higher bandwidth. It then drains the queue with a `pacing_gain` below 1, after which it cruises with a `pacing_gain 1.0`. In this way, a BBR flow can transfer data using the full bandwidth, while maintaining the minimal queue at the same time. If a BBR flow does not see smaller `MinRTT` for 10 seconds, it enters the `ProbeRTT` mode, in which it maintains the inflight at 4 packets for at least 200 ms or a round trip time, so as to drain out the queue and expose a new `MinRTT`. The total process is illustrated in Fig. 1.

B. System Model

We consider a simple scenario where two BBR flows compete with each other on a bottleneck link. We assume a deep buffer in the bottleneck switch, meaning that the switch

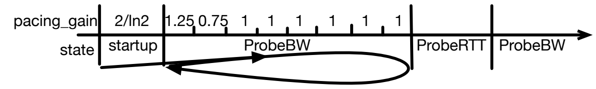

Fig. 1. BBR's periodic behavior

never drops a packet. We also assume that in the `Steady State` phase, the total inflight of the two flows is bounded by the two times of the network's *bandwidth-delay product*, which is the amount of packets the network can hold without a buffer. The flow RTTs are assumed to be between 5-500 ms, a typical range in the Internet.

Without loss of generality, our analysis mainly focuses on one `ProbeBW` phase. This is because the dynamics of the bandwidth share of the two BBR flows exhibits a periodic pattern as shown in prior experiments [4], [5]. A period consists of one `ProbeBW` phase and one `ProbeRTT` process. Once a flow enters the `ProbeRTT` mode, the two flows are synchronized, as can be shown by the following argument.

When flow 0's `MinRTT` time stamp[1] expires, flow 0 enters the `ProbeRTT` mode and defers its `MinRTT` time stamp for 10s. In this way, the queue in the bottleneck dips quickly, and flow 1's `MinRTT` expires. As a result, the `MinRTT` time stamp of flow 1 is also put off by 10s based on current time instance. This implies at the time instance flow 0 exits `ProbeRTT`, these two flows have the same time stamp value for `MinRTT`.

IV. Analysis of BBR Bandwidth Dynamics

In this section, we analytically characterize the bandwidth shares of the two BBR flows through three steps. *First*, we model the BBR's sending behaviors. *Second*, we give an iterative formulation to describe the relationship between the sending rates and other parameters, such as RTT ratio, network capacity and initial rate. *Third*, we derive a lower bound of the bandwidth share for the short-RTT flow. *Fourth*, we extend the analysis to the multiple-flows scenario.

A. Modeling BBR

Consider a slotted system with two competing BBR flows in a system with capacity c as shown in Fig. 2. Each flow adjusts its sending rate based on the measurements from the previous time slot. The RTTs of the two flows are denoted by RTT_0 and RTT_1, where $\text{RTT}_0 \leq \text{RTT}_1$. Let a be the RTT ratio of the long and short flows, i.e., $\text{RTT}_1 = a * \text{RTT}_0$. At the end of each time slot $[j, j+1]$, let $\text{inflight}_i(j)$ be the ith flow's inflight, $\text{eBW}_i(j)$ the estimated `BtlBW`, and $S_i(j)$ the sending rate in the next slot $[j+1, j+2]$. Table I summarizes the notations used in the paper.

Then we start formulating the BBR flow's sending rate. *First*, we formulate the sending rate based on the feedback at the end of each time slot. In BBR, a flow updates its sending rate based on the maximum estimated bandwidth (`MaxBW`) upon each arrival. We consider the moment at which a flow exits `ProbeRTT` and enters the `ProbeBW`. If the flow comes

[1]The `minRTT` time stamp records when a BBR flow should enter the ProbeRTT mode.

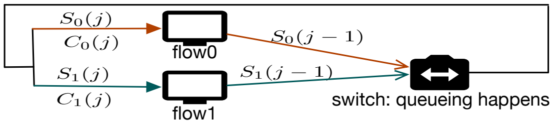

Fig. 2. The feedback model for the two-flow scenario

TABLE I
NOTATIONS IN THE MODEL

Notation	Meaning
c	network capacity
r_i	The ith flow's initial rate
a	RTT ratio
RTT_i	The ith flow's RTT
$S_i(j)$	The ith flow's sending rate for the time slot $[j+1, j+2]$
$\text{inflight}_i(j)$	The ith flow's inflight during the time slot $[j, j+1]$
$\text{eBW}_i(j)$	The ith flow's estimated bandwidth after the time slot$[j, j+1]$

into the 3rd phase among the 8-phase cycle,[2] `pacing_gain` is 1.25 during the $8m+7$th RTT, and is 0.75 during the $8m+8$th RTT. According to the BBR behavior, we have

$$\texttt{MaxBW} = \max\{\texttt{eBW}(j), \texttt{MaxBW}\},\ \texttt{eBW}(j) = \texttt{MaxBW},$$
$$S(j) = \begin{cases} 1.25 * \text{eBW}_(j), & (j-6)\ \%\ 8 < 1; \\ 0.75 * \text{eBW}_(j), & (j-7)\ \%\ 8 < 1; \\ \text{eBW}_i(j), & \text{otherwise.} \end{cases} \quad (1)$$

Second, we formulate the inflight of the two flows based on the flow's sending rate S_i. Note that when flow 0 gets new estimated bandwidth at the end of time slot $[n-1, n]$, flow 1 may still use the sending rate from the last time slot $[aj - a, aj]$. Therefore, the inflight of each flow at the end of time slot $[n, n+1]$ is

$$j = \lfloor (n-1)/a \rfloor,\ \text{inflight}_0(n) = S_0(n-1) * \text{RTT}_0, \quad (2)$$
$$\text{inflight}_1(n) = S_1(j) * (n\%a) * \text{RTT}_0 + S_1(j-1) * (a - n\%a) * \text{RTT}_0. \quad (3)$$

The inflight of each flow at the end of $[n, n+a]$ is

$$\text{inflight}_0(n) = ((an\text{RTT}_0)\%1) * S_0(\lfloor a(n-1) \rfloor) + (\text{RTT}_0 - (an\text{RTT}_0)\%\text{RTT}_0) * S_0(\lfloor a(n-1) \rfloor - 1), \quad (4)$$
$$\text{inflight}_1(n) = a\text{RTT}_0 * S_1(n-1) \quad (5)$$

Third, we formulate the estimated bandwidth of the two flows through the queueing theory. According to the queueing theory, the bandwidth share of competing flows is determined by their backlog, which means a flow's inflight determines its estimated bandwidth. Combining (2), (3), (5), and (4):

$$\text{eBW}_i(n) = c * \frac{\text{inflight}_i(n)}{\sum_{i=0}^{1} \text{infight}_i(n)}. \quad (6)$$

From the model, flows' sending rate determines throughput and thus bandwidth share. Therefore, analyzing $S_i(j)$ alone is sufficient to quantify BBR's bandwidth dynamics.

[2]BBR randomly selects from the six phases with `pacing_gain=1`, but such random indexing has no effect on our analysis.

B. Bandwidth Dynamics vs. Parameters

We further analyze the relationship between the sending rate $S_i(j)$ and the RTT based on the model in IV-A through four steps. *First*, we derive two cases by whether the RTT ratio is greater than 8. *Second*, we formulate $S_1(j)$ when $a < 8$. *Third*, we analyze BBR bandwidth dynamics when $a < 8$. *Fourth*, we formulate and analyze $S_0(j)$ when $a > 8$.

1) Step 1: Two cases for different RTT ratio: If the time slot J_a [3] of flow 1 contains a time slot I_1 of flow 0, then S_1 during I_1 is a constant, further to inflight_1. We then define **the two time slots J_a and I_1 as overlapping here**. Flow 0 updates `MaxBW` according to eBW_0 every 8 RTT_0s, indicating flow 0's sending rate may not change.

If we take the 8 RTT_0s as a large time slot I_8, when $a < 8$, I_8 may contain a flow 1's time slot J_a. If so, S_0 and inflight_0 in the time slot J_a are constants. These two time slots overlap.

If $a > 8$, after flow 0 updates its `MaxBW` according to eBW_0, it is possible that flow 1 has not finished one round trip. This suggests that flow 1's time slot J_a contains more than eight flow 0's time slots I_8. Therefore, S_1 and inflight_1 in the time slots I_8 are constants. These two time slots hence overlap.

Therefore, **we divide our discussions into two cases depending on whether RTT ratio a is larger than 8**.

2) Step 2: Formulate BBR bandwidth dynamics when $a < 8$: From (1), (5), (4), (6):

$$t = na, \quad n_0 = \lfloor (n-1)a \rfloor, \quad (7)$$
$$S_1(n) = \frac{acS_1(n-1)}{aS_1(n-1) + (t\%1)S_0(n_0) + (1 - t\%1)S_0(n_0 - 1)}. \quad (8)$$

Recall that inflight_0 is the inflight of flow 0. From (4), (8):

$$\frac{1}{S_1(n)} = \frac{1}{c} + \frac{1}{ac} * \frac{\text{inflight}_0}{S_1(n-1)}. \quad (9)$$

3) Step 3: How RTT ratio, network capacity and initial rate contribute to BBR unfairness: **When the time slots of the two flows overlap**, S_0 and inflight_0 are constants. Therefore, $1/S_1(n)$ follows **an exponential function**:

$$x = \frac{a}{\text{inflight}_0 - ac}, \quad y = \frac{\text{inflight}_0}{ac},$$
$$\frac{1}{S_1(n+1)} = y^n \left(\frac{1}{r_1} + x \right) - x. \quad (10)$$

First, the bandwidth share is exponential to RTT ratio a. Since $\text{inflight}_0 < c$ always exists and $a > 1$, we see that $y < 1$ always holds. Meaning, S_1 increases exponentially in terms of RTT ratio a. *Second*, network capacity helps RTT ratio to accelerate achieving final bandwidth share. In y, c appears in denominator together with a, which means higher network capacity contributes to more extreme overwhelming towards flow 0. Specifically, network capacity is not a flow's property, which means that network capacity has nothing to do with

[3]J_a means the time slot from j to $j+a$, which is $[j, j+a]$. This simplified writing format always holds in this paper.

RTT unfairness. *Third*, initial rate does not affect bandwidth allocation because r_0 does not appear in y.

If the time slots of the two flows don't overlap, S_0 is not a constant, where S_1 still increases. Because from (9):

$$\frac{1}{S_1(n)} = \frac{1}{c} + \frac{1}{ac} * \frac{\text{inflight}_0}{S_1(n-1)} < \frac{1}{S_1(n-1)} \quad (11)$$

Then we analyze the increasing speed of S_1. The function in (10) iterates 1 time if S_0 is not a constant. The only way to control the rising speed of S_1 is increasing inflight_0. However, higher S_1 results in higher eBW_1, indicating flow 1 keeps preempting bandwidth from flow 0. Therefore, y decreases as time goes by. S_1 increases even faster than exponential.

In conclusion, when $1 < a < 8$, flow updates throughput **exponentially** towards RTT ratio, meaning, BBR suffers a severe RTT fairness problem.

4) Step 4: Model BBR bandwidth dynamics when $a > 8$: We use the same method to analyze flow 0's behavior when $a > 8$. According to (2), (3), (6), S_0 decreases exponentially when the time slots of the two flows overlap:

$$\frac{1}{S_0(n+1)} = \frac{S_0(n) + \text{inflight}_1}{S_0(n) * c}. \quad (12)$$

When the time slots of the two flows do not overlap, S_0 quickly decreases, and the previous analysis remains applied. In other words, the bandwidth shares that flow 0 can obtain deteriorates exponentially fast.

C. Bandwidth Share Final State vs. RTT Ratio

In the last section, it has been showed that, during the `ProbeBW` mode, S_1 keeps increasing while S_0 keeps decreasing. The experimental data in [4] also shows that even though a 10 ms-RTT flow gets about 6 Mbps bandwidth on average when competing with a 50 ms-RTT flow on a 100Mbps bottleneck link, its `cwnd` is only 4 most of the time, corresponding to less than 0.1% bandwidth share.

However, in the `ProbeRTT` mode, the short-RTT flow could increase bandwidth share because the inflight of long-RTT flow is only 4. The experiment in [4] also shows that the 10 ms RTT flow can increase the bandwidth share in `ProbeRTT`.

In this section, we capture possibilities on bandwidth share of the short-RTT flow except being drained out. We discuss this question in three moves. *First*, based on (10), we analyze $S_0(j)$'s final state possibilities when $a > 8$. *Second*, we discuss the final bandwidth share based on (12) when $a < 8$. *Third*, we derive a lower bound on bandwidth share of flow 0 if the lower bound exists. Here, if flow 0 gets at most 0.1% bandwidth share, it is considered as being drained out.

1) Step 1: When $a > 8$, flow 0 gets at most 0.1% bandwidth share finally: If flow 0 drains out at the end of the time slot D when $a > 8$, it suffices to show the inequalities below:

$$\|S_0(D)\| < \varepsilon, \varepsilon \to 0, \text{and} \quad (13)$$

$$D * 8 * \text{RTT}_0 < 10s - a * \text{RTT}_0, \quad (14)$$

$$10s - a * \text{RTT}_0 > a * \text{RTT}_0. \quad (15)$$

ε means the bandwidth share threshold towards draining out, which is close unlimitedly to 0. Therefore, (13) means at the end of time slot D, flow 0 is drained out. (14) and (15) means when D ends, `ProbeRTT` in the next period has not come, indicating flow 0 is drained out during `ProbeBW`. According to (12), (13), we have

$$D \leq \log_a \left(\frac{\frac{1}{c-\varepsilon} - \frac{1}{c}}{\frac{1}{c/2} - \frac{1}{c}} \right). \quad (16)$$

We then apply (16) to (14) and (15), it shows that when $a > 8$, these inequalities above must be satisfied. **This means when $a > 8$, there is no possibility for the short-RTT flow to scratch more than 0.1% bandwidth.**

2) Step 2: When $2 < a < 8$, flow 0 is drained out finally: If flow 0 is drained out at the time slot X, it suffices to show these inequalities:

$$\|S_1(X) - c\| < \varepsilon, \varepsilon \to 0, \text{and} \quad (17)$$

$$10\text{s} - 200\text{ms} > 8\text{RTT}_0 \quad (18)$$

$$a * X * \text{RTT}_0 < 10s - a * \text{RTT}_0 \quad (19)$$

(17) means at the end of the time slot X, flow 1 drains out flow 0. (18) and (19) means when X ends, `ProbeRTT` in the next period has not come, indicating flow 0 is drained out during `ProbeBW`[4]. Now according to (10) and (17) we have,

$$X \leq \log_{\frac{1}{a}} \frac{\frac{1}{c-\varepsilon} - \frac{a}{c*(a-1)}}{\frac{1}{c/2} - \frac{a}{c*(a-1)}}. \quad (20)$$

Applying (20) to (18) and (19), we find that when $a > 2$, all of these equalities must hold, meaning flow 0 is drained out. **When $a > 2$, there is no possibility for the short-RTT flow to get more than 0.1% bandwidth share.**

3) Step 3: When $a < 2$, the final bandwidth share of flow 0 has a lower bound L, related to RTT_0 and a: Whether flow 0 is drained out depends on whether flow 0 losts all the bandwidth share before the next period's `ProbeRTT`, because the bandwidth share of flow 0 keeps decreasing in the `ProbeBW` mode. Therefore, the final bandwidth share of flow 0 is the flow 0's bandwidth share at the point when the current period ends.

The closed form iteration is described as follows,

$$N = \lfloor \frac{10s - 200\text{ms}}{a * \text{RTT}_0} \rfloor, \text{and} \quad (21)$$

$$L >= c - S_1(N) \quad (22)$$

N means the number of RTTs of flow 1 in one `ProbeBW`. L is the lower bound of flow 0's final bandwidth share.

This is an iterative closed form description, we conduct numerical analyses using MATLAB and compare the result with simulations in the next section.

[4] 200ms is the lasting time of `ProbeRTT` mode in one period of BBR.

D. Extension to the Multi-Flows Scenario

We first discuss the existence of the unfair bandwidth allocation problem. From the analysis above, the BBR unfairness rises from the imbalance of the buffering in the network bottleneck. Since longer RTT time always means more inflight in the network, long RTT flows always grab more bandwidth.

Then we discuss the severity of the unfair dynamics qualitatively. The multi-flows scenario can be divided to several two-flows scenario. Based on the above bound analysis, during a `ProbeBW`, if the longest flow can grab all the bandwidth whatever its initial rate is, the draining out finally happens in multiple-flows scenario. In other words, if there is a flow whose difference with the longest flow in RTT is less than 2, the draining never happens.

Therefore, to improve the BBR unfair situation, the core issue is to decrease or increase the buffering amount of long or short flows respectively. Several changes are likely to made. For example, if a flows detects continuous bandwidth decreasing, it selfishly paces $x\times$ for T RTTs, where $x \geq 2/ln2$, to validate whether it is drained out by others. Further, long flows may conduct `ProbeBW` slowly when detecting continuous estimated bandwidth increasing.

V. Evaluation

In this section, we simulate BBR flow dynamics in a two-flow scenario to verify our theoretical results.

Three findings are derived through our evaluation. First, the number of RTTs to drain out the 10 ms RTT flow decreases exponentially to RTT ratio with the base of 2.7. Second, when a is greater than 2, the short-RTT flow must be drained out. Third, the bandwidth share is determined solely by the RTT difference.

Unless otherwise specified, we set the total bandwidth to 100 Mbps, and the RTT ratio is set to 5.

A. Validation of Simulation

To verify that our simulation is able to reflect BBR flow's behavior, we simulated the throughput of a 10-ms RTT flow and a 50-ms RTT flow when they compete on a 100 Mbps bottleneck link. In [4], the same competition is conducted in an in-house server. We compare the results of the simulation and the experiment in Fig. 3. The bandwidth share of the two flows in the simulation is close to the one measured in the real-world experiment. Moreover, the 50 ms RTT flow spends about 1 second to reach the final steady state in both simulation and experiment. The result shows that our simulation is consistent with the real-world experiment.

B. Bandwidth Dynamics

First, we verify the relationship between the bandwidth dynamics and the initial rate. At first, $r_0 = 10$ Mpbs, $r_1 = 10$ Mbps, after 10s, we change the initial rate to $r_0 = 40$ Mbps, $r_1 = 10$ Mbps. The final bandwidth share keeps unchanged as shown in Fig. 4.

Second, we verify the relationship between the bandwidth dynamics and the network capacity. At first, the bottleneck link is 100 Mbps, after 10s, we change the bottleneck link to 200 Mbps as shown in Fig. 5. The final bandwidth share doesn't change with different network capacity. Moreover, higher network capacity corresponds to higher increasing rate of flow 1's bandwidth share. This verifies that higher network capacity helps to accelerate the draining out process.

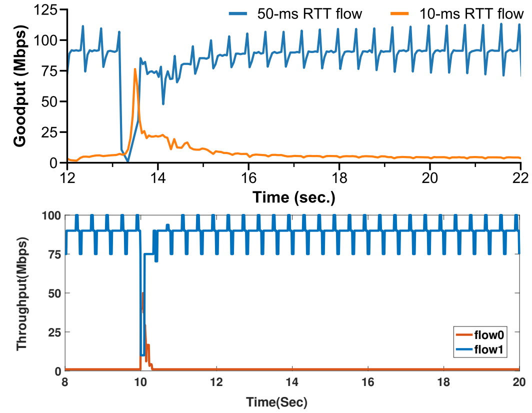

Fig. 3. Both the final bandwidth share and the increasing trend in the simulation are similar to experimental data.

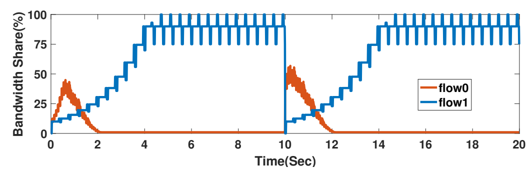

Fig. 4. At time = 10s, we increase the initial rate of flow 0 by 4 times, but the final bandwidth share of the two flows remains the same.

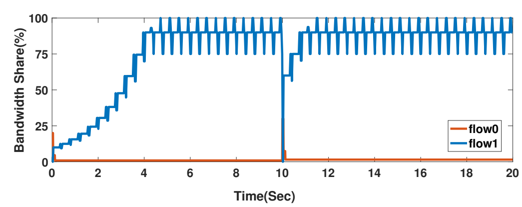

Fig. 5. At time = 10s, we increase the network capacity by two times, and the final bandwidth share of the two flows remains the same.

Third, we verify the exponential function between the bandwidth share and the RTT ratio. From Fig. 4 and 5, we can find a long-RTT flow increases its bandwidth share sharply. To make this conclusion more quantitive, we incorporate a 10 ms RTT flow to compete with a long-RTT flow with RTT ratio ranging from 2 to 9.5. We record the number of RTTs for the long-RTT flow to drain out the 10 ms flow. The result is shown in Fig. 6.

We fit this curve to an exponentially decreasing function as $\text{RTT} = 4 + 2.7^{6-a}$.

C. Bandwidth Bound Existence of Short BBR Flows

We verify the lower bound on bandwidth share of the short-RTT flow here. The verification is in 3 steps.

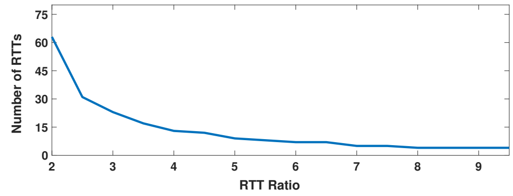

Fig. 6. RTT ratio is from 2-9.5, the number of RTTs of long-RTT flow to drain out the 10 ms RTT flow is from 60 to 6.

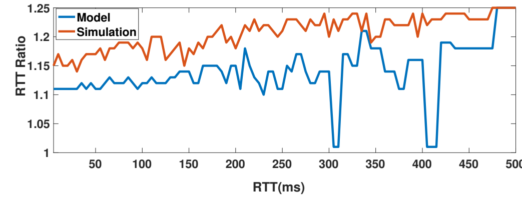

Fig. 9. X: Dirferent RTT_0, Y: The largest RTT ratio to drain out the short flow.

First, we draw a bandwidth share curve of flow 0 under the configuration of $a = 1.1, \text{RTT}_0 = 10$ ms in Fig. 7. Flow 0 gets a 5% bandwidth share finally from Fig. 7. And the final state of flow 0's bandwidth share appears between 10s to 12s, when the current period terminates.

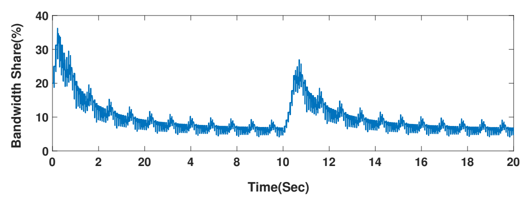

Fig. 7. RTT ratio is from 1.1, the bandwidth share of flow 0 with time

Second, we verify if flow 0's lower bound on bandwidth share by our model is reasonable. We calculate flow 0's lower bound on bandwidth share when competing with a long-RTT flow with different RTT ratios, then compare the calculated lower bound with the simulation result. The configuration is: $\text{RTT}_0 = 10\ ms, a \in [1.01, 1.11]$. The result is shown in Fig. 8.

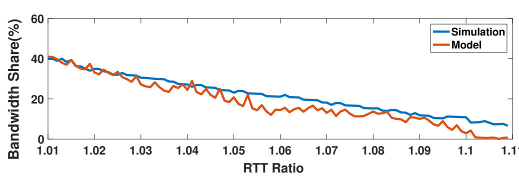

Fig. 8. RTT ratio is from 1.01-1.11, flow 0's lower bound on bandwidth share from simulation and model.

At last, note that the final bandwidth share is only related to RTT_0 and RTT ratio from $(18), (19)$and(20). To figure out how RTT_0 affects final bandwidth share together with RTT ratio, we get the upper bound of RTT ratio where flow 0 is not drained out under different RTT_0 through the model and the simulation. RTT_0 ranges from 5 ms to 500 ms, which is common in Internet. The result is shown in Fig. 9.

Although both of the lines fluctuate, larger RTT_0 still corresponds to higher RTT ratio to be drained out. This indicates that RTT_0 actually takes minor effect on final bandwidth share. Moreover, if a is large than 1.25, flow 0 must be drained out if RTT_0 is less than 500 ms.

VI. Conclusion

In this paper, we build a theoretic model to characterize BBR bandwidth dynamics. We demonstrate that the bandwidth share between different flows exhibits an exponential behavior dictated by the RTT ratio, irrelevant with the initial sending rate and link capacity. In addition, we show there exists a threshold on the RTT ratio, beyond which the short RTT flow will be drained out. Simulations are carried out to verify our model and evaluate the results, which we believe present new insights in the understanding of intrinsic behavior of BBR and searching for improvement in its fairness.

References

[1] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh *et al.*, "BBR: Congestion-Based Congestion Control," *Commun. ACM*, vol. 60, no. 2, pp. 58–66, 2017.

[2] S. Ha, I. Rhee, and L. Xu, "CUBIC: A New TCP-Friendly High-Speed TCP Variant," *ACM SIGOPS operating systems review*, vol. 42, no. 5, pp. 64–74, 2008.

[3] S. Jain, A. Kumar, S. Mandal, J. Ong, L. Poutievski, A. Singh, S. Venkata, J. Wanderer, J. Zhou, M. Zhu *et al.*, "B4: Experience with A Globally-Deployed Software Defined WAN," in *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4. ACM, 2013, pp. 3–14.

[4] S. Ma, J. Jiang, W. Wang, and B. Li, "Towards RTT Fairness of Congestion-Based Congestion Control," *arXiv preprint arXiv:1706.09115*, 2017.

[5] M. Hock, R. Bless, and M. Zitterbart, "Experimental Evaluation of BBR Congestion Control," in *IEEE ICNP*, 2017, pp. 1–10.

[6] V. Jacobson, "Congestion Avoidance and Control," in *ACM SIGCOMM*, 1988, pp. 314–329.

[7] E. Altman, C. Barakat, E. Laborde, P. Brown, and D. Collange, "Fairness Analysis of TCP/IP," in *Proceedings of IEEE Conference on Decision and Control*, vol. 1, 2000, pp. 61–66.

[8] S. Poojary and V. Sharma, "Analytical Model for Congestion Control and Throughput with TCP CUBIC Connections," in *IEEE GLOBECOM*, 2011, pp. 1–6.

[9] S. Ha, I. Rhee, and L. Xu, "CUBIC: A new tcp-friendly high-speed tcp variant," *ACM SIGOPS Oper. Syst. Rev.*, vol. 42, no. 5, pp. 64–74, 2008.

[10] V. Jacobson, "Berkeley TCP Evolution from 4.3-tahoe to 4.3-reno," in *IETF*, 1990.

[11] M. Alizadeh, A. Javanmard, and B. Prabhakar, "Analysis of DCTCP: Stability, Convergence, and Fairness," in *ACM SIGMETRICS*. ACM, 2011, pp. 73–84.

[12] M. Alizadeh, A. Greenberg, D. A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, and M. Sridharan, "Data Center TCP(DCTCP)," in *ACM SIGCOMM*, vol. 40, no. 4. ACM, 2010, pp. 63–74.

[13] L. S. Brakmo and L. L. Peterson, "TCP Vegas: End to End Congestion Avoidance on a Global Internet," *IEEE J. Selected Areas in Commun.*, vol. 13, no. 8, pp. 1465–1480, 1995.

[14] C. Boutremans and J.-Y. Le Boudec, "A Note on the Fairness of TCP Vegas," in *2000 IEEE International Zurich Seminar on Broadband Communications*, pp. 163–170.