IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received December 28, 2021, accepted January 9, 2022, date of publication February 4, 2022, date of current version February 22, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3148998

RBBR: A Receiver-Driven BBR in QUIC for Low-Latency in Cellular Networks

HABTEGEBREIL HAILE, KARL-JOHAN GRINNEMO, (Senior Member, IEEE), PER HURTIG, AND ANNA BRUNSTROM, (Member, IEEE)
Department of Mathematics and Computer Science, Karlstad University, 65188 Karlstad, Sweden
Corresponding author: Habtegebreil Haile (habtegebreil.haile@kau.se)

ABSTRACT BBR is a promising new congestion control algorithm (CCA) that has been shown to result in significantly lower latency compared to conventional loss-based CCAs. However, in cellular networks, where there is a high variability in the available rate, BBR does not perform as well as expected. In such scenarios, BBR tends to overestimate the available capacity and create queues that cause longer packet delays. In this work, we propose Receiver-driven BBR (RBBR), a modified version of BBR that uses rate estimates made at the receiver side rather than at the sender side. We employ a Kalman filter to make a more accurate estimate of the available bandwidth, and we implement the algorithm in QUIC. An evaluation of the proposed CCA is done through extensive 4G trace-based emulations, real 4G network tests and mmWave trace-based emulations representing a 5G scenario. The results show that RBBR is able to achieve an RTT reduction of up to 80% with a worst-case throughput loss of about 30%. The results also show that in real 4G networks, RBBR flows experience a more predictable and consistent RTT than what BBR flows do.

INDEX TERMS 4G, 5G, BBR, congestion control, Kalman filter, QUIC, RTT, throughput.

I. INTRODUCTION

Transport layer protocols and congestion control algorithms (CCAs) are important components of data communication over the Internet. These protocols and CCAs can enable or limit an application from meeting its throughput and delay requirements. In fact, an ill-suited protocol or algorithm can prevent an application from meeting its performance goals despite the underlying network being able to support the application's requirements. QUIC [1] and BBR [2] (Bottleneck Bandwidth and Round-trip propagation time) are promising transport solutions for overcoming some of the limitations of TCP [3] and loss-based CCAs. QUIC is a transport protocol implemented in user-space that supports multiple features not present in TCP, for example, fast connection setup and support for multi-streaming. Since QUIC is implemented at the user level, it is possible to update QUIC when updating the application using it, making it an interesting option for incorporating new features.

BBR is a CCA proposed by Google that operates by building a model of the network path to control its sending behavior. BBR builds the network path model by estimating the bottleneck bandwidth and the minimum round trip propagation time. However, the bottleneck bandwidth estimation mechanism of BBR can be prone to overestimation and cause packets to experience longer delays. This overestimation is noticeable in current 4G cellular networks, where the available rate is often highly variable. The variability is also likely to increase in 5G networks.

In this work we present a novel, latency-aware CCA for cellular networks – Receiver-driven BBR (RBBR). RBBR is a modification to BBR that is implemented in QUIC, and that uses a Kalman-filtered rate feedback loop from the receiver to more accurately estimate the bottleneck bandwidth. In our previous work [4], we have illustrated that it is possible to improve BBR's delay response in cellular networks with long-term variations in the available bandwidth. The long-term variations in the available bandwidth can be due to signal quality deterioration, extended blockage, mobility, or steady changes in network load. In this work, we demonstrate that BBR can also be susceptible to variations occurring over small time scales. We provide a complete design for RBBR that incorporates a measured rate filtering mechanism that makes BBR less vulnerable to short-time scale rate bursts and dips. The detailed contributions of this work are:

The associate editor coordinating the review of this manuscript and approving it for publication was Francisco Rafael Marques Lima.

- An extension to a selected QUIC framework for reducing the impact of short-term rate variations that uses a Kalman filter for available bandwidth estimation from delivery rate samples at the RBBR receiver.
- A signaling mechanism to notify the RBBR receiver to suspend the rate estimation when the RBBR sender reduces its sending rate to probe for the minimum RTT. The signaling mechanism prevents the use of erroneous samples in the available bandwidth estimation.
- Extended 4G and mmWave trace emulations, as well as real 4G network evaluations with results showing that RBBR achieves a noticeable reduction in RTT compared to BBR in cellular 4G and 5G scenarios. The evaluations over real 4G networks also show that RBBR has a more consistent RTT performance that is less susceptible to temporal changes.

The remainder of this paper is organized as follows. Section II gives a brief overview of QUIC, BBR and rate variability in cellular networks. Next, we give a description of the components of RBBR in Section III. The evaluation setup is given in Section IV. In Section V, the results of the evaluation are presented and discussed. Finally, we conclude the paper in Section VII.

II. BACKGROUND

In this section, we give a brief background on QUIC and BBR, which are the transport technologies that this work is based on. We also present the challenges of dealing with the high variability in data transfer rate that is present in cellular networks and that motivated this work.

A. QUIC

QUIC [1] is a transport protocol that is intended to overcome several of the shortcomings of the Transmission Control Protocol (TCP) [3]. One of TCP's main shortcomings, addressed by QUIC, is Head-of-Line (HOL) blocking. In TCP, HOL blocking can happen when multiple HTTP/2 [5] streams are being multiplexed over a single TCP connection, and a loss of a packet from one stream delays the delivery of packets from a different stream to the application layer. QUIC solves this problem by creating multiple independent streams. In addition, QUIC facilitates a fast connection startup by saving and reusing configurations of frequent connections and also use transport-layer encryption for secure communication. Figure 1 shows that QUIC is a user-level protocol that works on top of the User Datagram Protocol (UDP) [6]. The fact that QUIC is an encrypted transport implemented in user-space makes it relatively easy to upgrade the protocol as new versions of the protocol appear. It also enables the addition of new features that are not part of the mainstream specification. Therefore, fast updates of the QUIC implementation can be made as part of application updates. This can be particularly advantageous to self-contained services that are delivered through third-party applications.

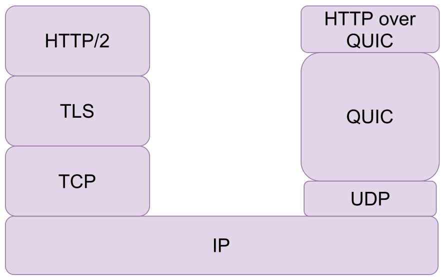


FIGURE 1. Comparison of the TCP and QUIC stacks.

QUIC payload is carried inside predefined frames, each containing a frame-type identifier as well as other fields that vary depending on the frame type. QUIC has gone through many iterations of development, with features continuously being added to the protocol. At the time of writing, QUIC has become an IETF Request for Comment (RFC) with a proposed standard status. During its evolution, the number of defined frames in the QUIC RFC has more than doubled and nowadays the RFC includes frame types that transport a lot of different types of connection-related data between the communicating peers. QUIC's version negotiation phase allows peers to identify supported features and agree on a communication format. The frame-based structure of QUIC packets provides a convenient way for extending QUIC and to integrate new features that can enable improved performance. In this work, we take advantage of the frame-based structure of QUIC and employ it as a way to exchange delivery rate information between the peers that is relevant for congestion control.

B. BBR

Loss-based CCAs have been shown to be responsible for large queuing delays and bufferbloat [7]. The bufferbloat phenomenon is likely to occur in networks with large buffers, unless some in-network mechanism is available to complement the CCA [8]. On the other hand, delay-based CCAs find it difficult to compete with loss based dittos, and might fail to fully utilize available capacity. The shortcomings of loss-based and delay-based CCAs have driven researchers to look into other types of congestion-control approaches. One such approach is model-based congestion control.

BBR [2] is arguably the most popular model-based CCA. The algorithm has managed to achieve wide-scale implementation and deployment for Google's services. It has gained popularity because of it's ability to achieve equal or higher throughput than loss-based CCAs with lower self-inflicted delay [19]. BBR operates by controlling the pacing rate and the congestion window by building a model of the network path using estimates of the bottleneck bandwidth (BtlBW) and the round-trip propagation time (RTT) at the sender. Figure 2 shows the dynamics of the RTT and delivery rate as the amount of in flight data increases and the optimal point that BBR tries to acquire as the model of the network path.

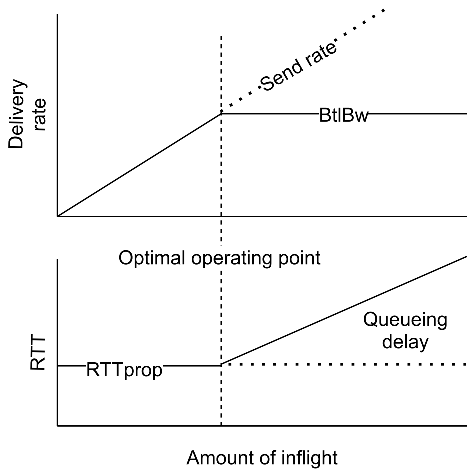


FIGURE 2. **Optimal network path model.**

BBR calculates the receiver's delivery rate at the sender for every arriving acknowledgement (ACK), and uses that in the BtlBW estimation. The algorithm employs a new delivery-rate estimation mechanism that is proposed together with the CCA [9]. BBR removes noisy delivery measurements that could lead to an underestimation of the BtlBW by passing them through a max filter over a prescribed number of RTTs (10 RTTs). The final BtlBW estimate is computed as the minimum of the max-filtered delivery rates and the sending rate. Since the BtlBW and RTT of a connection can vary throughout the lifetime of the connection, BBR contains mechanisms that continuously probe for changes in the path as represented by the BBR model. The original BBR algorithm probes for available bandwidth in a bandwidth probing cycle, which is referred as *ProbeBW*, by increasing the pacing rate by 25% every 8th RTT. To probe for the minimum RTT, the algorithm significantly reduces the amount of in-flight data every *RTprop*, which is set to 10 seconds, provided a minimum RTT estimate that is lower than the current estimate has not been detected within this 10-second interval. The CCA is said to be in the *ProbeRTT* state when it reduces the in-flight data to detect changes in the minimum RTT. Another important component of BBR is the mechanism to counter the effect of ACK aggregation. This mechanism tries to estimate the level of aggregation by using the difference between the amount of ACKed data and the expected amount of data to be ACKed. In a similar way as in the BtlBw computation, the max filter is applied over multiple RTTs to give an extra amount of data to be added to the computed congestion window (CWND). Therefore, this mechanism is closely tied to the BtlBw estimation, and it could result in longer packet delays as it implies injecting possibly excessive amounts of data into the network.

With the increased use of the more volatile mmWave bands in cellular networks, and the deployment of a high number of devices that communicate sporadically (IoT) in those networks, the scale of short-term variability observed by a BBR

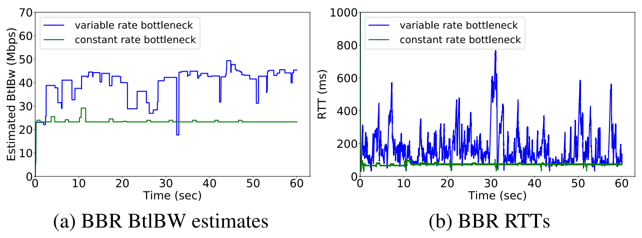


FIGURE 3. **BtlBW estimates and RTT of BBR in a constant rate scenario vs. a variable rate scenario with a mean rate equal to the constant rate scenario (minRTT = 30 ms).**

CCA used for applications over cellular networks is likely to increase. In such cases, BBR's maximum-filter-based, sender-side rate estimation mechanism could significantly overestimate the available bottleneck bandwidth. Figure 3 gives the performance of BBR in emulated networks, based on rate traces collected from a 4G network. The figure gives the RTT experienced by packets in an emulated network configured with the highly variable rate trace and with a constant rate that has the same average rate as the variable trace. The RTT in the highly variable setup is notably larger than the RTT in the constant rate configuration. The plot in Figure 3a shows that the BtlBw estimates of BBR fails to closely track the highly variable capacity. On the other hand, BBR does a good job of estimating the available capacity of the constant rate capacity. This allows BBR to maintain a lower RTT in such scenario. But Figure 3b shows that the RTT can increase and become variable when the available capacity varies. A more detailed look into the challenge rate variability in cellular networks presents to making a more practical rate measurement is given in the next subsection.

C. RATE VARIABILITY IN CELLULAR NETWORKS

The majority of the technological advances introduced in current cellular networks (4G/5G) are intended to improve the capability of the access network to accommodate a large number of users with different throughput and delay requirements.

However, the need for an increased and efficient use of radio frequency resources has resulted in cellular networks becoming more complex and dynamic in their behavior. The complex mechanisms of 4G and 5G technologies can introduce some difficulties that can become challenging to handle in higher layers of the Internet protocol stack. Some of these complications are variability due to sudden resource reallocations to accommodate services with higher priorities and/or latency requirements, increased retransmissions from using sensitive frequency ranges, and variable delays caused by reconfigurations from frequent handovers between small cells.

These types of issues can be particularly difficult to handle for transport-layer CCAs that are used by applications that require high throughput and low delay. As exemplified by BBR, there has been an increased interest in CCAs that rely on estimating the state of the network and that use these estimates as a basis for a model of the path. The increased bandwidth variability experienced in cellular networks makes the task of network-state estimation and modeling in these networks a more difficult task, something which could negatively impact the performance of model-based CCAs. Therefore, it is important to keep track of the possible performance pitfalls of CCAs in the increasingly variable cellular networks, and propose solutions that could help the CCAs mitigate the issues arising from the highly variable available bandwidth.

To ensure that a rate-based CCA responds quickly to changes in the network state, the rate measurements need to be made at a reasonably high frequency to make a swift response to changes in the network capacity possible. In cellular networks, the rate achieved by a user primarily depends on the signal quality between the user and the cellular access point, the amount of load in the network, and the type of scheduler used at the base station. Therefore, it is reasonable to think that if those conditions are stable then the rate achieved by a user should remain the same as well. However, unless we measure the rate over very long time intervals, it is highly unlikely that we observe the same rate over successive time intervals. The variations in the measured rate could happen for a number of reasons, e.g., channel variations, scheduling, loss, retransmissions, or aggregation. As a result, two intervals with temporal proximity could experience vastly different rates without actual, significant changes in the state of the network.

Since a short-interval rate sample might greatly differ from the long-term rate of a user, using the unprocessed rate measurements can lead to an undesired performance, such as higher RTT or underutilization. This is especially true for connections with longer RTTs. In such cases, it could take a while until a rate sample is received by the sender, and in the mean time the transient rate might have changed a lot. Using unprocessed, transient rate samples could also affect the rate samples we get in future measurements. For example, using a very low transient rate sample as the sending rate at the sender could limit the rate that can be detected over the following time intervals. To this end, we need a mechanism that can identify and remove the transient variations, while keeping track of the long-term variations of the rate.

III. RBBR DESIGN

The core idea behind RBBR is to replace the sender-side max-filter based bottleneck bandwidth estimate used in BBR by a Kalman-filter based rate estimate made at the receiver. The receiver-side rate estimate is then sent back to the sender inside a newly defined QUIC feedback frame (*fbk_frame*). RBBR builds on the initial proposal for a receiver and feedback-based BBR in [4], and aims to make RBBR more robust to small time-scale rate variations. These variations can occur in cellular networks due to a number of lower-layer mechanisms. In this section, we describe the three main components that work together to enable RBBR. These components are a rate-estimation mechanism

Algorithm 1 Rate Estimation and Feedback at the QUIC Receiver

```
Set rate sampling duration $\Delta t$
while connection is open do
    if Ack scheduled then
        Attach rate to packet
        Send packet to sender
    if Sender limited then
        Suspend estimation
    else
        if interval $> \Delta t$ then
            Reset timer
            Sample rate
            Filter rate
```

based on an adaptive Kalman filter, a QUIC extension for rate feedback, and a reverse (sender-to-receiver) signaling to inform the receiver to suspend rate estimation when the BBR sender is in the ProbeRTT state. The adaptive Kalman filter rate estimation mechanism extracts a viable rate from the receiver-side rate measurements. The rate feedback through a newly defined frame extension transports the rate estimation back to the sender. Algorithm 1 gives a simplified view of the rate estimation and feedback process at the QUIC receiver. The reverse signaling helps the receiver avoid using invalid rate samples in the rate estimation process.

A. ADAPTIVE KALMAN FILTER

The Kalman filter is an optimal-state estimator for a linear process with Gaussian noise. Given an unreliable measurement that is corrupted by Gaussian noise with known co-variance, the basic Kalman filter is able to extract the correct state of a system that is characterized by a linear state-space model given in Equation 1. In the equation, x_k is the actual value of the state to be estimated at time k, x_{k-1} is the previous value of the state at time $k-1$, and A is the state transition model. The term Bu_k represents the external input, and w_k is the process noise. In addition to the state-space model, the Kalman filter also requires a linear measurement model. The measurement model can be represented as given in Equation 2, where y_k is the measured quantity, x_k is the state of the system, C is the measurement gain, and v_k is the measurement noise.

$$x_k = Ax_{k-1} + Bu_k + w_k \tag{1}$$
$$y_k = Cx_k + v_k \tag{2}$$

The standard Kalman filter goes through two stages to acquire the final estimate. The first is the prediction stage that uses the state-space model to make an a priory state estimate x_k^- (Equation 3), and update the co-variance, p^-, that determines the uncertainty in the estimate (Equation 4). The co-variance of the process noise Q should be known to make the prediction. The second stage updates the prediction

TABLE 1. The predict and update stages of a standard Kalman filter.

Predict stage	Update stage
$x_k^- = Ax_{k-1} + Bu_k$ (3)	$K = p_k^- C(Cp_k^- C^T + R)^{-1}$ (5)
$p_k^- = Ap_{k-1}A^T + Q$ (4)	$x_k = Ax_k^- + K(y_k - CAx_k^-)$ (6)
	$p_k = p_k^- + KCp_k^-$ (7)

using the measurement value. This is done by first calculating the Kalman gain K, which requires the co-variance of the measurement noise R as shown in Equation 5. The Kalman gain then determines the contribution of the measurement to the final estimate (Equation 6) and the co-variance of the final estimate (Equation 7). Table 1 summarizes the predict and update stages of the standard Kalman filter.

To be able to use the Kalman filter, it is necessary to have knowledge of the co-variances of the process and measurement noise. However, when applying the Kalman filter for the purpose of estimating the rate of a cellular network, these values are hard to acquire. In addition, characteristics of the noise experienced during measurement and prediction might change with time, as explained in [10]. To overcome this problem, we apply the adaptive measurement and process noise co-variance computation method proposed in [11]. Equations 8 and 9 show the computation of the process and measurement noise co-variances, respectively. In the equations, ϵ_k is the residual and d_k is the innovation at time k, and α determines how fast the co-variances are updated.

$$R_k = \alpha R_{k-1} + (1-\alpha)(\epsilon_k \epsilon_k^T + Cp_k^- C^T) \tag{8}$$
$$Q_k = \alpha Q_{k-1} + (1-\alpha)(K_k d_k d_k^T K^T) \tag{9}$$

In our system, we use a simple state transition $x_k = x_{k-1} + w_k$ and a simple measurement model $y_k = x_k + v_k$. The measured sample, y_k, is obtained as shown in Equation 10, where S_k and S_{k-1} are the total amount of data received at successive measurement time instants t_k and t_{k-1}, respectively. The final estimate, x_k, of each interval is then sent in the feedback frame to be used as an estimate of the bottleneck bandwidth by the RBBR sender.

$$y_k = \frac{S_k - S_{k-1}}{\Delta t} \tag{10}$$
$$\Delta t = t_k - t_{k-1} \tag{11}$$

B. RATE FEEDBACK

The rate estimated in the above steps is sent back to the sender. To be able to send the estimated rate as feedback from the receiver to the sender, we extend QUIC by defining a new frame as described in [4]. We implement the new frame in the *mvfst* QUIC implementation from Facebook [12] by adding the necessary code to encode and decode the frame into a QUIC packet. According to the specification of a standard QUIC frame, the frame contains an 8-bit frame identifier and 8 bytes type-dependent field. The rate feedback is located in the type-dependent field of the feedback frame as shown in Figure 4. The rate feedback frame is scheduled by default whenever an ACK is scheduled, thus having the same frequency as the ACKs. However, it is possible to schedule the feedback frame to be sent every few ACKs, or periodically with a specified time interval.

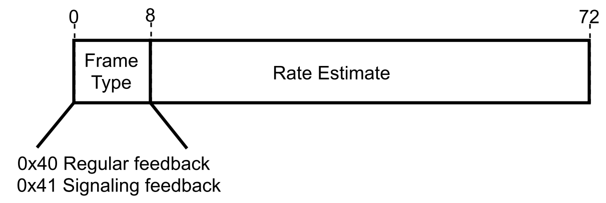


FIGURE 4. The QUIC feedback frame.

C. SIGNALING FOR RATE LIMITED CASES

One of the core parts of BBR is the ProbeRTT phase. In this phase, the BBR sender reduces its congestion window to a very low value to allow the sender to get a new estimate of the minimum RTT. This poses a problem for the receiver-side delivery rate estimation mechanism of RBBR since the rate measured in the ProbeRTT (or any other sender-limited case) is not determined by the available capacity over the bottleneck. If the receiver does not know about the reduced sending rate at the sender, then it might take the sender-limited rate sample as a bandwidth-limited sample and feed it back to the sender. To prevent that from happening, we define a signaling frame that is sent from the sender when the sender is in the ProbeRTT phase.

Every time the CCA goes into the probeRTT phase, we set a value, *suspendRemote*, in *mvfst* that is checked during feedback frame construction. If *suspendRemote* is set, the feedback frame will also serve as a signaling frame, otherwise the feedback frame will be a standard feedback frame. The process for signaling the remote receiver to suspend rate sampling is given in Algorithm 2. At the other side, when a feedback frame is received, it is checked whether it is a regular feedback frame or a feedback frame that also signals a suspension of rate estimation. If it is the latter, then the rate estimation process is suspended until a regular feedback frame is received. The process for suspending the rate sampling and estimation at the receiver is given in Algorithm 3. By allowing rate feedback in the signaling frame, we are able to avoid having two different frames for rate feedback and signaling the sender limited case in the same packet. This is particularly important for a receiver that only has ACKs to send in a unidirectional downlink data transfer, and therefore will never reach the full bottleneck bandwidth. Therefore, in this case, the sender will avoid making the rate estimation calculations.

IV. EXPERIMENTAL SETUP

We evaluate the proposed modified CCA through emulation of 4G and 5G link scenarios as well as through real 4G

Algorithm 2 Procedure for Notifying Remote Node to Suspend Rate Estimation

```
while connection is open do
    if RBBR transits to probeRTT or Startup then
        set suspendRemote
    if RBBR transits to probeBW then
        unset suspendRemote
    if suspendRemote then
        send suspend_fbk_frame
    else
        send regular_fbk_frame
```

Algorithm 3 Procedure for Suspending and Resuming Rate Estimation

```
while connection is open do
    if fbk_frame received then
        if fbk_frame_type == suspend_fbk_frame then
            set isSuspended
        else
            unset isSuspended
        if isSuspended then
            Stop rate sampling
            Freeze rate estimation
        else
            Resume rate sampling and estimation
```

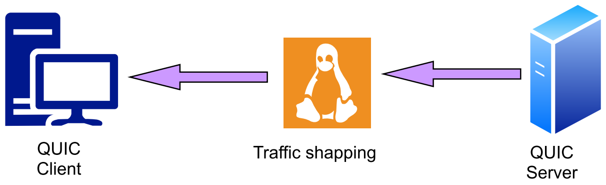


FIGURE 5. Emulation setup.

network tests. We use the *tperf* tool that is available in *mvfst* with only minor changes to allow for external configuration of some CCA parameters. The tool provides a client/server setup for transferring data for a period of time and to report the throughput. We use the tool by creating a single stream that continuously sends data from a QUIC server to a QUIC client using a selected CCA. The *tperf* tool and the CCAs used in our experiments are from the May 2020 commit of mvfst.

A. EMULATION ENVIRONMENT

The emulation environment is composed of two Linux end nodes that serve as sender and receiver, and a middle node that connects the two end nodes. We set up the emulation configuration in CloudLab [13]. Figure 5 illustrates the emulation environment. The middle node serves as a traffic shaper to vary the rate available to send data between the end nodes. The middle node is also used to control the RTT

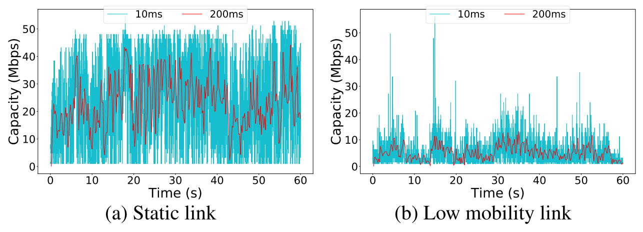


FIGURE 6. Fine-grained 4G traces.

and bottleneck buffer size. The RTT is set to a minimum of 30 ms for all experiments, and we allocate a very large buffer for the experiments where the impact of the buffer size is not being tested. We vary the available rate based on cellular network traces from commercial 4G networks and mmWave links. We use a hierarchical token bucket (HTB) traffic shaper to configure the downlink rate, and *netem* [14] to configure the delay and buffer size at the appropriate interfaces of the middle node.

The 4G traces that are used to configure the traffic shaper are acquired by sending constant bit rate (CBR) UDP traffic through commercial 4G networks and logging the reception of each UDP datagram. We collect two types of traces to evaluate the CCA. The first trace is collected from a test setup with a static mobile node stationed at an office in the university, while the second trace is collected by walking around in the university buildings. To ensure that the bottleneck is fully utilized when collecting the trace data, we configure the CBR sender to send at a very high rate. The UDP receiver will also acknowledge the reception of each datagram, which we use to compute an estimate of the RTT. We check the RTT data to confirm that it is kept at a much higher value throughout the duration of the CBR transfer than the lowest recorded RTT. From the datagram reception log, we calculate the rate samples at $\Delta t = 10\,ms$ intervals, and use this rate trace to configure the traffic shaper to exhibit a similar rate pattern. We believe that by configuring the HTB traffic shaper in this way, we obtain a realistic emulation setup that captures the bursty nature of cellular access links, while at the same time minimize any possible adverse effects from the frequent rate reconfigurations taking place in these tests. In Figure 6, we show two representative 4G rate traces from among the ones used in our evaluations. The traces are for a static device and a mobility scenario that is created by walking around the university building. For each scenario, we plot a 10 ms-interval trace and a 200 ms-interval trace. We use the 10 ms-interval trace in our experiments since that trace better emulates the burstiness of the channel, however, we show the 200 ms-interval trace for better visibility of the general capacity variation.

In addition to the fine-grained (10 ms-interval) 4G traces, we also provide initial evaluations over coarse-grained mmWave traces that sample a rate about every 250 ms. The coarse-grained mmWave traces are shown in Figure 7. The mmWave traces are based on the traces from [15], and were also used in [4]. The coarse-grained mmWave traces might

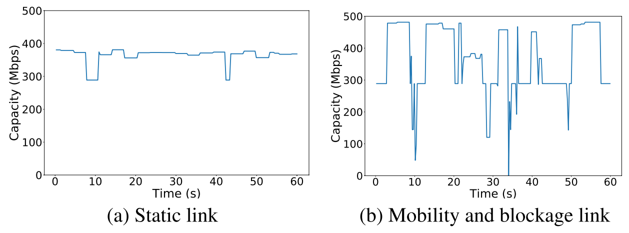


FIGURE 7. Coarse-grained mmWave traces.

not give an altogether realistic representation of the amount of short-term burstiness experienced in real 5G access networks, but we still think they properly show that the modified CCA is capable of performing well at the high rates of 5G. We use the given mmWave traces because we do not have access to a commercial 5G network for trace collection. Similar to the 4G case, we use both static and mobility mmWave traces in our evaluations.

B. 4G TEST SETUP

To evaluate the performance of RBBR in 4G networks, we send QUIC traffic that employ different CCAs to receivers connected to three different mobile network operators. The receivers used in the experiments are MONROE nodes [16] that use a Debian-based Linux operating system. The *tperf* client is deployed as a container in the MONROE node, while the *tperf* server is located in an office at the university, and accesses the Internet through a high-speed Ethernet connection at the university. The 4G test setup is illustrated in Figure 8. One of the MONROE nodes is connected to two different operator networks. However, when we perform tests on one network, we disconnect the node from the other network.

V. RESULTS

In this section, we present the results from our emulations and real 4G network tests. We provide throughput and RTT results comparing RBBR to BBR and CUBIC [17] from emulation experiments using fine-grained 4G and coarse-grained 5G traces. We also give results from tests that evaluate the performance of RBBR by varying the configuration of some relevant network and algorithm parameters. We also present real network measurement results from 4G networks of three different Swedish operators.

A. 4G TRACE RESULTS

The emulation results given here compare the performance of RBBR with BBR and CUBIC using identical traces and conditions to test each CCA. We include CUBIC in some of the experiments to add some perspective for the performance of a widely used loss-based CCA that is also the default in Linux [17]. We also use emulations to evaluate the performance of RBBR for different variables: filter parameter, sampling interval, buffer size, and loss rate. We use a middle value of 0.6 for the adaptive Kalman filter

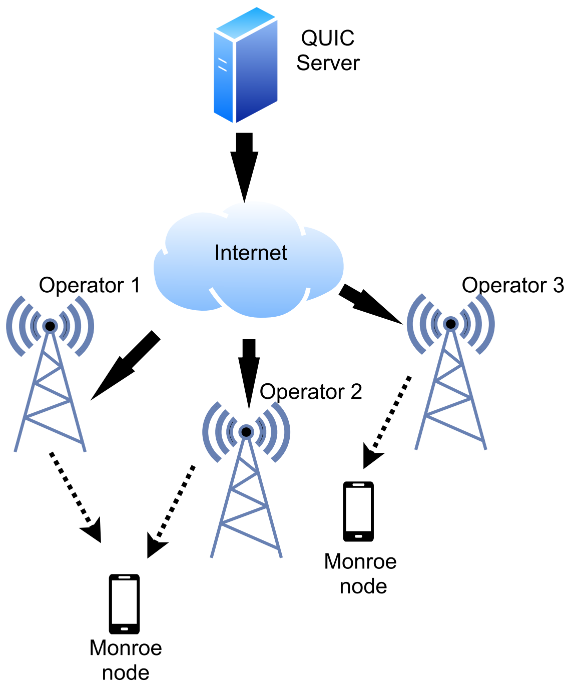


FIGURE 8. 4G test setup.

parameter α for most experiments unless specified otherwise. The RTT is configured to 30 ms to make it comparable to the minimum RTT we observed in the 4G networks on which we performed live tests. For each given combination of CCA, buffer size, and loss rate, we repeat the experiment at least 10 times on every trace. For RBBR, in addition to the above, we also perform at least eight iterations for each filter parameter and sampling interval.

1) THROUGHPUT AND RTT

The throughput and RTT of RBBR, BBR, and CUBIC from the 4G trace based emulation experiments on a bottleneck with a large buffer ($> 4\times$BDP) are given in Tables 2 and 3. Example traces from the tables are also plotted in Figure 9. In Figure 9b and 9d, it can be seen that RBBR achieves a significantly lower RTT than standard BBR and CUBIC without losing much throughput in Figures 9a and 9c. The longer RTT of BBR is caused by the rate overestimation of the sender-side mechanism and the extra packets injected by the aggregation compensation mechanism. From the RTT results, it is evident that BBR greatly exceeds the $1.5\times$BDP buffer that it is expected to maintain [18]. With the minimum RTT configured to be 30 ms, BBR's mean RTT values of mostly more than 100 ms is likely the result of an average queue that is at least $2\times$BDP of buffer. On the other hand, RBBR's mean RTTs of mostly below 77 ms indicates an average queue that is around or below the target buffer size.

The figure also gives the throughput and RTT for BBR with the aggregation compensation mechanism turned off

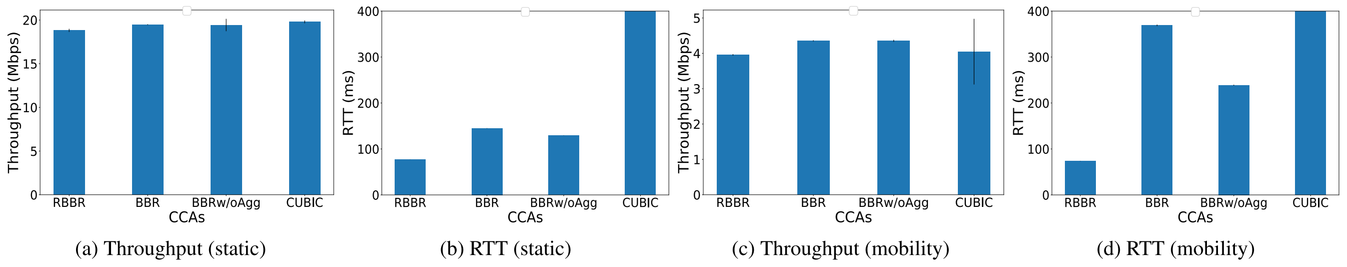


FIGURE 9. Throughput and mean RTT of RBBR, BBR, BBRw/oAgg, and CUBIC for 4G static and 4G mobility traces.

TABLE 2. Throughput and RTT of CCAs for different traces collected in static scenario from three different operators.

Operator	Trace	Throughput (Mbps) / RTT (ms)			
		RBBR	BBR	BBRw/oAgg	CUBIC
Op1	1	18.8 / 77.4	19.5 / 145.1	19.4 / 129.6	19.8 / 4377.7
	2	21.7 / 74.8	22.0 / 140.3	22.1 / 123.5	22.1 / 4133.4
	3	27.5 / 77.2	27.6 / 111.0	27.6 / 102.5	27.5 / 3961.1
	4	34.6 / 68.7	34.5 / 92.1	34.5 / 84.5	33.8 / 3276.2
Op2	1	26.3 / 68.1	26.5 / 114.8	26.6 / 100.3	27.0 /3826.8
	2	9.7 / 75.7	10.2 / 192.6	10.1 / 147.9	10.1 / 5356.5
	3	11.2 / 69.9	11.6 / 160.2	11.7 /127.6	11.5 /6691.8
Op3	1	10.4 / 75.8	11.3 / 237.5	11.3 / 197.5	10.8 / 6351.9
	2	3.96 / 73.99	4.6 / 282.7	4.5 / 199.3	4.6 / 9225.5
	3	5.9 / 72.2	6.6 / 315.2	6.6 / 203.3	6.2 / 6864.1

TABLE 3. Throughput and RTT of CCAs for different mobility traces from a walking pace.

Trace	Throughput (Mbps) / RTT (ms)			
	RBBR	BBR	BBRw/oAgg	CUBIC
Indoor 1	4.0 / 74.0	4.4 / 369.2	4.4 / 239.0	4.1 / 8196.0
Indoor 2	3.0 / 113.9	4.3 / 575.3	4.3 / 325.7	4.2 / 10671.9
Outdoor	26.2 / 67.1	26.3 / 122.4	26.3 / 105.0	26.5 / 4179.1

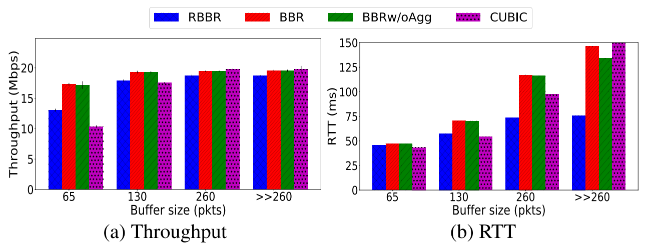


FIGURE 10. Throughput and mean RTT of RBBR, BBR and BBRw/oAgg in a bottleneck with different buffer sizes (around 1BDP to >> 4BDP).

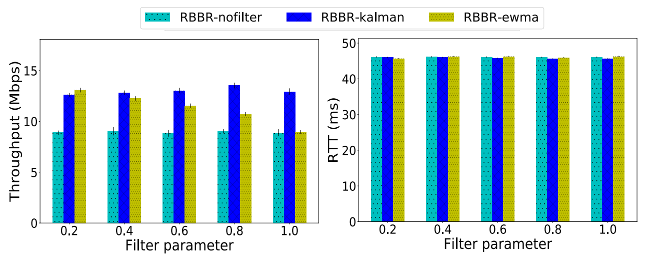
(BBRw/oAgg). We can see the advantage offered by the estimation mechanism of RBBR by making comparisons with BBRw/oAgg, where the overestimation of BBR's sender side mechanism is the main culprit for the observed longer RTTs. It should be stated that BBR's aggregation compensation mechanism plays an important role in some networks like WiFi, where the level of ACK aggregation is very high. The aggregation compensation mechanism of BBR is closely tied to the sender-side estimation mechanism, therefore it would have behaved unpredictably if it had been used in RBBR. To use RBBR in networks with high level of ACK aggregation, a different aggregation compensation mechanism must be applied.

In the figures, BBR and BBRw/oAgg experience a noticeably higher RTT in the mobility scenario compared to the static scenario. From the trace in Figure 6b, it can be seen that the rate in the mobility scenario is most of the times lower than in the static scenario, however, it can also be seen that it exhibits large increases from time to time. These sudden rate increases are interpreted as the long-term, available rate by BBR's max filter, which causes BBR to oversend resulting in larger queues and longer RTTs. In contrast, RBBR avoids the increase in RTT in the mobility scenario that is observed for BBR. Particularly, RBBR is able to filter out the transient rate bursts and follow a more accurate progression of the steady-state rate.

The extra queue accumulated from the overestimation and aggregation compensation mechanism could explain the slightly higher throughput of BBR as shown in Figure 9. The accumulated packets could be available for instances where the capacity increases instantly and significantly, allowing BBR to achieve a slightly better throughput. We use multiple traces to evaluate RBBR. As can be seen from Tables 2 and 3, the sometimes slightly lower throughput obtained with RBBR is more than compensated for by a significantly lower RTT than BBR. The 95 percent confidence intervals for the mean values given in the tables are omitted as they are small and similar to the values shown in Figure 9.

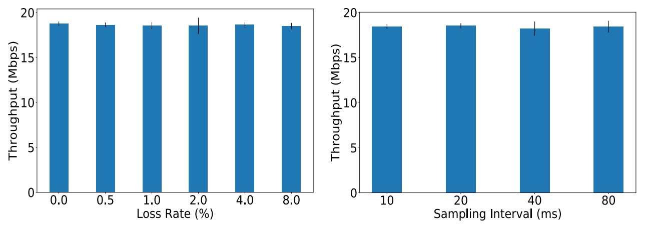
2) LOW BUFFER SCENARIO

RBBR is intended to reduce the delay caused by BBR in cellular networks with a highly variable rate and large bottleneck buffers. Still, we also need to know the performance of RBRR outside of its target scenarios. In Figure 10b, we show the evaluation results of RBBR, BBR and CUBIC over a



(a) Throughput (b) RTT

FIGURE 11. Throughput and mean RTT of RBBR without filter (RBBR-nofilter), RBBR with EWMA filter (RBBR-ewma), and RBBR with Kalman filter (RBBR-kalman) in a small buffer (1 BDP).



(a) Throughput vs. uplink loss rate (b) Throughput vs. sampling intervals

FIGURE 12. RRBR throughput for different ACK loss rates and rate sampling intervals.

variable rate bottleneck with different buffer sizes. It can be seen that the reduction in RTT compared to BBR experienced with RBBR disappears as the buffer size becomes 65 packets (around $1\times$BDP) or lower. The throughput obtained with the evaluated CCAs for different buffer sizes is given in Figure 10a. As shown, a decrease of the buffer size below $2\times$BDP results in RBBR's throughput becoming much lower than that of BBR. However, it also follows that with such small buffers, BBR's throughput falls below the available capacity.

The low throughput of RBBR when the buffers are small is likely due to the limit the buffer puts on the range of rates that could be sampled. The rate estimation mechanism will only get proper rate samples that reflect the available rate when the available rate is low. When the available rate increases beyond what is supported by the buffer, the receiver will not be able to get the proper rate samples, thus biasing the estimate to a lower rate than the actual available capacity. As a solution to this problem, a Kalman filter that is adjusted for biased samples can be used or one can compensate for the underestimation of the available capacity with additional packets when the buffer size is low. In general, obtaining an accurate sample of the available rate is a challenging task since a small buffer will bias the rate sampling towards lower samples. This is not just a problem experienced by RBBR, but by any system that relies on rate measurements in variable networks such as BBR, which evidently also underutilizes the variable bottleneck when the buffer is small.

3) ALTERNATIVE FILTERING

In this part, we compare the performance of the Kalman filter-based estimation mechanism with unfiltered rates and an exponentially weighted moving average (EWMA), an alternative filtering mechanism that is routinely used to remove the effect of transient variations. Figure 11 compares the throughput and RTT of the three receiver-side rate estimation mechanisms.

In our previous work [4], it was shown that replacing BBR's sender-side rate estimation with unfiltered rate feedback from the receiver can result in an improved RTT performance. The evaluations were done on traces that only emulate the long-term variation experienced in 5G networks. However, through follow-up experiments, we have confirmed that in addition to worsening the RTT performance of BBR, short-term variation also tend to cause a decrease in the throughput achieved by the unfiltered rate feedback. This can be caused by the system being locked in to a low-sending-rate interval until the next probing occurs after detecting a few successive low rate samples. Figure 11 shows that this happens in small-buffer scenarios, with the unfiltered mechanism being particularly more sensitive in small buffer scenarios. This throughput performance degradation is the main motivation behind applying filtering. It should be noted that the filter parameter has no role in the results for the unfiltered rate feedback. In the figure, the results for the unfiltered rate feedback are repeated for different filter parameters along with results of methods that employ filtering to allow convenient comparison.

One method that can be used to remove transient rate variations is the EWMA filter, which is given in Equation 12, where r_t is the rate measured at time t, α is the EWMA filtering parameter, and R_t and R_{t-1} are the rate estimates at time t and $t-1$ respectively. EWMA is a simple and widely used method for smoothing out a range of variable quantities. Therefore, it is reasonable to ask why the more complicated adaptive Kalman filter is to be preferred over EWMA. The figure shows that for lower values of α, the EWMA mechanism results in a performance that is similar to the Kalman-filter-based system. However, as the α value increases the performance of EWMA becomes more similar to the low throughput results observed with the unfiltered mechanism. This is to be expected since as α increases the contribution of the previous estimate to the next estimate disappears while the contribution of the current measurement is increased. The Kalman-filter, on the other hand, produces a more consistent and predictable performance as the α value changes. This can make filter parameter selection easier with lower chance of misconfiguration.

$$R_t = \alpha \times r_t + (1-\alpha) \times R_{t-1} \tag{12}$$

Based on the results given in Figure 11, one might opt to choose to use a more efficient EWMA filter with a lower α instead of the adaptive Kalman-filter, which requires more computations per estimate. However, a lower EWMA parameter value would mean a higher level of filtering, which might be good for a highly variable network but might make

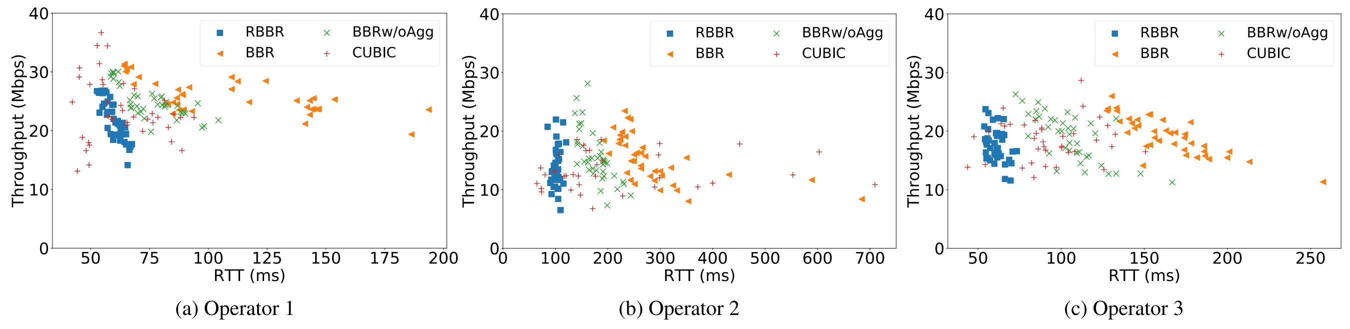


FIGURE 13. Throughput and mean RTT of RBBR, BBR, BBRw/oAgg and CUBIC in three different operators.

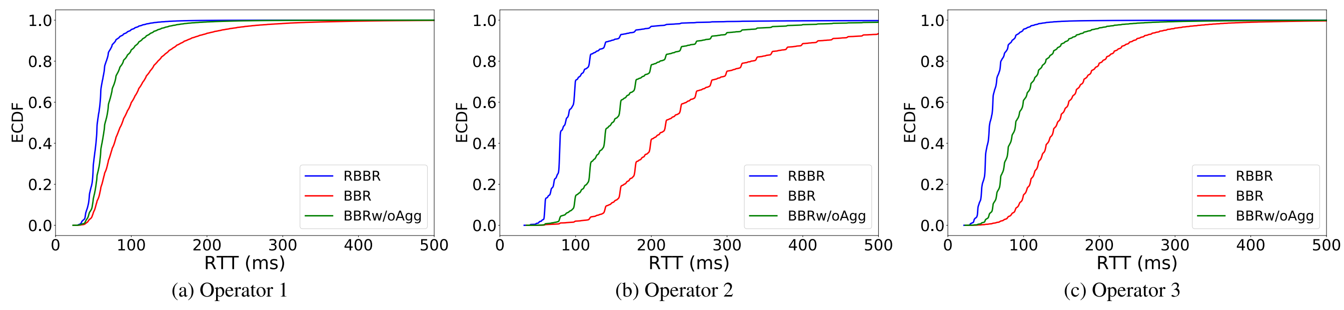


FIGURE 14. RTT CDFs for RBBR, BBR and BBRw/oAgg in three drifferent operators.

the system slow to adapt if the bottleneck should change to less variable links.

4) ACK LOSS AND SAMPLING INTERVAL

Beyond the buffer size and filter parameter α, there are other variables that could potentially affect the performance of RBBR. In this part, we explore the impact of variables like uplink loss rate and rate sampling interval. The default implementation of RBBR sends the rate feedback frame whenever an ACK is scheduled. Therefore, it would be good to know how robust the algorithm is to ACK losses, i.e., losses of the rate feedback. We evaluate this scenario and present the results in Figure 12a. From the figure, it can be seen that RBBR's performance does not show much difference between different uplink loss rates. In our evaluations, we also evaluate the robustness of the algorithm for different sampling rates and present the results in Figure 12b. The results show that the CCA achieves consistent throughput performance for different rate sampling intervals. However, using smaller sampling intervals could be a better option in access networks with a more variable available rate than the one given in the trace used in this experiment.

B. 4G NETWORK RESULTS

Here we present results from our experiments done in the 4G networks of three mobile network operators. In Figure 13, we present the throughput and RTT achieved by RBBR, BBR and BBR without aggregation compensation. We performed 40 iterations with each CCA through four experiment campaigns carried out on different days and at different times. We also added a few CUBIC results for comparison. Each data transfer over a specific operator's network with one of the evaluated CCAs is done for a period of 60 seconds. We perform an RBBR data transfer between a data transfer using BBR without aggregation compensation and a data transfer using standard BBR.

The results of the 4G experiments show that RBBR is able to maintain a much lower and consistent average RTT than BBR and BBR without aggregation compensation. The impact of the RBBR rate estimation mechanism can be observed from the RTT difference between RBBR and BBR without aggregation compensation. An interesting general trend is that the difference in RTT seems to increase as the throughput decreases. Most of the experiments that resulted in the highest throughput for BBR and RBBR were performed at night time when there is less competition and a user is most likely to be given consistently the maximum amount of resources that it could get for its signal quality. At other times of the day, the throughput was lower as the number of users was larger, and the scheduling was likely to be more bursty, which increased the risk of BBR doing an overestimation of the bandwidth.

In most cases, BBR and BBR without aggregation compensation achieve a slightly better throughput, while RBBR's throughput can be seen to be close to the BBR throughput and better than the throughput achieved by CUBIC in all operators. In all three operators, it can be seen that there is a considerable difference in the RTT of BBR and BBR without aggregation compensation without much gain in throughput. This indicates that BBR's aggregation compensation mechanism could be further improved. The

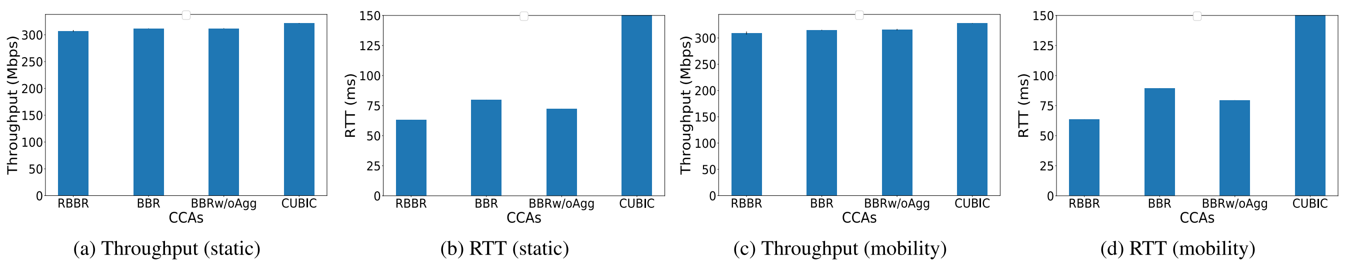


FIGURE 15. **Throughput and mean RTT of RBBR, BBR and BBR w/o agg and CUBIC for a coarse-grained mmWave static and mmWave mobility and blocking traces.**

CDF of the RTT of the CCAs is given in Figure 14 for each operator. From the CDF figures it can be seen that the RTT improvement for operator two and three is noticeably higher than the improvement in operator one. Based on the results of the CUBIC transfers over operator one, we deduce that the bottleneck for the connections through operator one is likely on a link with a lower buffer than the other operators. This in turn reduces the amount of accumulated queue and the RTT experienced by the CCAs. Figure 14 also shows that the slope of the CDF for RBBR is steeper than the slope of the CDF of BBR and BBR without aggregation compensation, suggesting flows with RBBR could experience less jitter.

C. mmWave TRACE RESULTS

Millimeter wave (mmWave) is a crucial part of 5G that uses the high frequency spectrum (30 GHz and 300 GHz) for wireless data transfer. mmWave is one of the core technologies for achieving the high data rates delivered by 5G. The use of mmWave, however, introduces complications, such as signal attenuation and increased rate variability. Thus, we also tested the system on the coarse-grained mmWave traces. Figure 15 gives the results of the mmWave trace evaluations. The figure shows that there is a noticeable improvement in the RTT and the throughput loss is negligible. It can also be seen that the RTT improvement is bigger for the mobility case (Figure 15d). RBBR achieves a more consistent RTT between the static and the mobility scenarios (63.2 ms for static and 63.53 ms for mobility). That means that the improvement in RTT for the mobility case comes from the slight increase in the RTT of BBR in the mobility scenario (89.37 ms for BBR and 79.57 ms for BBRw/oAgg) when compared to the static scenario (79.74 ms for BBR and 72.4 ms for BBRw/oAgg).

The RTT and throughput difference between RBBR and BBR in the coarse-grained mmWave traces is, however, not as big as the difference observed in the fine-grained 4G traces and live network results. When using coarse-grained traces, BBR is less likely to experience frequent sporadic bursts of rate, while RBBR will be better at converging to a new steady state rate. Thus, we anticipate bigger differences in RTT and throughput in future experiments with more fine-grained mmWave traces.

VI. RELATED WORK

At the time of writing, the first version of BBR has been around for about six years. Following it's initial release, several performance studies have been performed and a number of evaluations and proposals have also discovered scenarios where the algorithm seems to underperform. The performance of BBR was compared with CUBIC's performance on the highway in [19]. The paper found that BBR can achieve comparable throughput to CUBIC with lower self-inflicted delay. Wang *et al.* [20] evaluate the performance of TCP in high-speed rails (HSR) and found out that BBR is more handover-agnostic and carrier-agnostic. The paper also revealed that BBR can experience suboptimal throughput performance due to an inflexible and conservative round-trip propagation time estimation mechanism. In addition to evaluating BBR, a number of works have also proposed modifications to improve throughput [20], [21]. Grazia *et al.* [21] target to improve the throughput performance of BBR in high-aggregation WLAN networks by increasing the 25% probing pacing gain of BBR to a higher value. The BBR modification for HSR in [20] alters the pacing gain values for the BBR ProbeBW cycle for more frequent probing and uses a dynamic RTprop interval. There have also been a few proposals focusing on BBR's fairness [22], [23]. BBRv2 [24], a second version of BBR has also been released by Google. However, the second version is mainly concerned with BBR's fairness and response to high loss, and keeps the bottleneck bandwidth estimation method used in the first version of BBR largely unchanged.

On the other hand, queuing delay resulting from rate overestimation is the main focus of a few recent papers [4], [25]–[27]. Su *et al.* [26] suggest a method of avoiding spurious rate bursts by delaying the rate update until the rate is found to be consistent. Their method was evaluated over a simulated link with a constant rate. Vargas *et al.* [25] found that router burstiness can cause BBR to overestimate the bottleneck rate and cause poor QoE for DASH by building long queues. A history-based solution, BBR-sampled (BBR-S), that collects samples over a period of time and takes the rate at the 85^{th} percentile is proposed. This approach is tested on both LANs and WANs, however, it's performance in a variable and bursty cellular environment is yet to be tested. Another BBR modification, which incidentally is also called BBR-S, has been proposed in [27]. This BBR-S modification replaces the max filter of standard BBR with an adaptive Kalman filter. However, in contrast to our work, for BBR-S, the rate samples are calculated at the sender from RTT information and the performance of this modification is evaluated through ns-3 simulations. The work in [4] is a precursor to this work showing the advantage of using receiver side rate feedback for BBR in QUIC. Our work builds on that to produce a more robust CCA by integrating a Kalman filter for rate estimation and signaling for sender limited cased. This work also provides a more extended evaluation with multiple traces and in real 4G networks.

VII. CONCLUSION

In this work, we propose RBBR, a modified version of BBR for QUIC that uses filtered, receiver-side rate measurements as estimates of the bottleneck bandwidth, and which let its sending rate and congestion window be governed by these measurements. The receiver-side estimates replace BBR's sender-side estimates that tend to overestimate the available rate in cellular networks. The rate estimates are made at the receiver by applying a Kalman filter to raw delivery rate measurements to remove temporary fluctuations. The receiver-side rate estimates are transmitted to the sender through a feedback frame that is an extension to standard QUIC.

Our evaluations of RBBR on fine-grained 4G and course-grained mmWave traces show a notable improvement in RTT with a limited loss in throughput. We also test RBBR on static nodes in multiple 4G networks, and our observations show that RBBR is robust and consistent in 4G networks. Our observations also show that the delay experienced by BBR can vary depending on the trace used for emulation, time of real 4G network test, and location and mobility of the mobile device. In general, the RTT performance of BBR becomes worse when the overall rate becomes low, and in indoor mobility scenarios. Our performance results show that RBBR is able to maintain a lower and more consistent RTT in real 4G networks, with only a very small penalty in throughput compared to standard BBR, a penalty that is eclipsed by the reduction in RTT.

There still exists multiple avenues for future work. The Kalman filter employed by RBBR operates on the assumption of system linearity. We intend to extensively test the performance of RBBR in situations where the linearity assumption does not hold. The results from the coarse-grained mmWave traces are promising. But we also intend to perform more evaluations using fine-grained 5G traces, as well as live measurement campaigns in actual 5G networks. Intra and inter CCA fairness analysis and evaluations can also be done to assess RBBR's friendliness. RBBR can also be vulnerable to greedy-receiver attacks. Future works can explore for ways to prevent these types of attacks for receiver-driven algorithms.

REFERENCES

[1] J. Iyengar and M. Thomson, *QUIC: A UDP-Based Multiplexed and Secure Transport*, document RFC 9000, May 2021.

[2] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh, and V. Jacobson, "BBR: Congestion-based congestion control," *Queue*, vol. 14, no. 5, pp. 20–53, Oct. 2016.

[3] *Transmission Control Protocol*, document RFC 793, Sep. 1981.

[4] H. Haile, K.-J. Grinnemo, S. Ferlin, P. Hurtig, and A. Brunstrom, "WIP: Leveraging QUIC for a receiver-driven BBR for cellular networks," in *Proc. IEEE 22nd Int. Symp. World Wireless, Mobile Multimedia Netw. (WoWMoM)*, Jun. 2021, pp. 252–255.

[5] M. Belshe, R. Peon, and M. Thomson *Hypertext Transfer Protocol Version 2 (HTTP/2)*, document RFC 7540, May 2015.

[6] *User Datagram Protocol*, document RFC 768, Aug. 1980.

[7] J. Gettys, "Bufferbloat: Dark buffers in the internet," *ACM Queue-Virtualization*, vol. 9, pp. 40–54, Nov. 2011.

[8] K. Nichols and V. Jacobson, "Controlling queue delay: A modern AQM is just one piece of the solution to bufferbloat," *Queue*, vol. 10, no. 5, pp. 20–34, May 2012.

[9] Y. Cheng. (Jul. 2017). *Delivery Rate Estimation Internet-Draft Draft-Cheng-ICCRG-Delivery-Rate-Estimation*. Accessed: Oct. 7, 2021. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-cheng-iccrg-delivery-rate-estimation-00

[10] H. Haile, P. Hurtig, and K.-J. Grinnemo, "Rate measurement over short time scales in stationary cellular receivers," in *Proc. Netw. Traffic Meas. Anal. Conf.*, 2019, pp. 203–208.

[11] S. Akhlaghi, N. Zhou, and Z. Huang, "Adaptive adjustment of noise covariance in Kalman filter for dynamic state estimation," in *Proc. IEEE Power Energy Soc. Gen. Meeting*, Jul. 2017, pp. 1–5.

[12] Facebookincubator. *Mvfst*. Accessed: Oct. 7, 2021. [Online]. Available: https://github.com/facebookincubatormvfst

[13] D. Duplyakin, R. Ricci, A. Maricq, and G. Wong, "The design and operation of CloudLab," in *Proc. Annu. Tech. Conf. (ATC)*, Jul. 2019, pp. 1–14.

[14] *Tcnetem(8) Linux Manual Page*. Accessed: Oct. 7, 2021. [Online]. Available: https://man7.org/linux/man-pages/man8/tc-netem.8.html

[15] A. Srivastava, F. Fund, and S. S. Panwar, "An experimental evaluation of low latency congestion control for mmWave links," in *Proc. IEEE Conf. Comput. Commun. Workshops*, Jul. 2020, pp. 352–357.

[16] O. Alay, "MONROE: Measuring mobile broadband networks in Europe," in *Proc. IRTF ISOC Workshop Res. Appl. Internet Meas. (RAIM)*, 2015, pp. 1–34.

[17] 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, Jul. 2008.

[18] D. Scholz, B. Jaeger, L. Schwaighofer, D. Raumer, F. Geyer, and G. Carle, "Towards a deeper understanding of TCP BBR congestion control," in *Proc. IFIP Netw. Conf. (IFIP Networking) Workshops*, May 2018, pp. 1–9.

[19] F. Li, J. W. Chung, X. Jiang, and M. Claypool, "TCP CUBIC versus BBR on the highway," in *Passive and Active Measurement* Cham, Switzerland: Springer, 2018, pp. 269–280.

[20] J. Wang, Y. Zheng, Y. Ni, C. Xu, F. Qian, W. Li, W. Jiang, Y. Cheng, Z. Cheng, Y. Li, X. Xie, Y. Sun, and Z. Wang, "An active-passive measurement study of TCP performance over LTE on high-speed rails," in *Proc. 25th Annu. Int. Conf. Mobile Comput. Netw.*, Aug. 2019, pp. 1–9.

[21] C. A. Grazia, M. Klapez, and M. Casoni, "BBRp: Improving TCP BBR performance over WLAN," *IEEE Access*, vol. 8, pp. 43344–43354, 2020.

[22] Y. Zhang, L. Cui, and F. P. Tso, "Modest BBR: Enabling better fairness for BBR congestion control," in *Proc. IEEE Symp. Comput. Commun. (ISCC)*, Jun. 2018, pp. 646–00651.

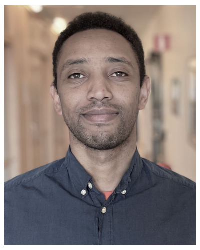
[23] G.-H. Kim and Y.-Z. Cho, "Delay-aware BBR congestion control algorithm for RTT fairness improvement," *IEEE Access*, vol. 8, pp. 4099–4109, 2020.

[24] N. Cardwell. *BBR V2: A Model-Based Congestion Control*. Accessed: Oct. 7, 2021. [Online]. Available: https://datatracker.ietf.org/meeting/104/materials/slides-104-iccrg-an-update-on-bbr-00

[25] S. Vargas, R. Drucker, A. Renganathan, A. Balasubramanian, and A. A. Gandhi, *BBR Bufferbloat in DASighway Video*. New York, NY, USA: Association for Computing Machinery, 2021, pp. 329–341.

[26] B. Su, X. Jiang, G. Jin, and H. Chen, "Rethinking the rate estimation of BBR congestion control," *Electron. Lett.*, vol. 56, no. 5, pp. 239–241, Mar. 2020.

[27] F. Chiariotti, A. Zanella, S. Kucera, and H. Claussen, "BBR-S: A low-latency BBR modification for fast-varying connections," *IEEE Access*, vol. 9, pp. 76364–76378, 2021.



HABTEGEBREIL HAILE received the M.Sc. degree in telecommunications engineering from the University of Trento, Italy, in 2014, and in embedded computing from Åbo Akademi University, Finland, in 2016. He is currently pursuing the Ph.D. degree with Karlstad University. His current research interests include fulfilling the requirements of a diverse set of applications and services in existing and future cellular networks.

KARL-JOHAN GRINNEMO (Senior Member, IEEE) received the Ph.D. degree in computer science from Karlstad University, in 2006. He has worked almost 15 years as an Engineer in the telecom industry, first at Ericsson and then as a Consultant at Tieto. A large part of his work has been related to Ericsson's signaling system in the mobile core and radio access network. Since Fall 2014, he has been a Senior Lecturer with Karlstad University. He has authored and coauthored around 70 conferences and journal articles. His research interests targets application- and transport-level service quality. In recent years, his research has to a large degree focused on the use of multipath transport protocols, such as multipath TCP to increase reliability and throughput and decrease latency in IP networks.

PER HURTIG received the M.Sc. and Ph.D. degrees in computer science from Karlstad University, Sweden, in 2006 and 2012, respectively. He is currently an Associate Professor with the Department of Computer Science, Karlstad University. His research interests include transport protocols, lowlatency internet communication, multi-path transport, and network emulation. He has participated in several international research projects and is also involved in internet standardization within the IETF.

ANNA BRUNSTROM (Member, IEEE) received the B.Sc. degree in computer science and mathematics from Pepperdine University, CA, USA, in 1991, and the M.Sc. and Ph.D. degrees in computer science from the College of William & Mary, VA, USA, in 1993 and 1996, respectively. She joined the Department of Computer Science, Karlstad University (KAU), Sweden, in 1996, where she is currently a Full Professor and the Research Manager of the Distributed Systems and Communications Research Group. Her research interests include internet architectures and protocols, techniques for low latency internet communication, multi-path communication, and performance evaluation of mobile broadband systems, including 5G. She is currently the KAU Principal Investigator within the EU H2020 Project 5GENESIS. She has authored/coauthored over 200 international journals and conference papers. She is the Co-Chair of the RTP Media Congestion Avoidance Techniques (RMCAT) Working Group within the IETF.