IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 1, 2021, accepted May 19, 2021, date of publication May 21, 2021, date of current version June 1, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3082715

BBR-S: A Low-Latency BBR Modification for Fast-Varying Connections

FEDERICO CHIARIOTTI[1], (Member, IEEE), ANDREA ZANELLA[2], (Senior Member, IEEE), STEPAN KUCERA[3], (Senior Member, IEEE), AND HOLGER CLAUSSEN[4], (Senior Member, IEEE)

[1]Department of Electronic Systems, Aalborg University, 9100 Aalborg, Denmark
[2]Department of Information Engineering, University of Padova, 35131 Padova, Italy
[3]Nokia Bell Labs, 11481 Munich, Germany
[4]Tyndall National Institute, Dublin 8, D08 WV88 Ireland

Corresponding author: Federico Chiariotti (fchi@es.aau.dk)

ABSTRACT The new possibilities offered by 5G and beyond networks have led to a change in the focus of congestion control from capacity maximization for web browsing and file transfer to latency-sensitive interactive and real-time services, and consequently to a renaissance of research on the subject, whose most well-known result is Google's Bottleneck Bandwidth and Round-trip propagation time (BBR) algorithm. BBR's promise is to operate at the optimal working point of a connection, with the minimum Round Trip Time (RTT) and full capacity utilization, striking the balance between resource use efficiency and latency performance. However, while it provides significant performance improvements over legacy mechanisms such as Cubic, it can significantly overestimate the capacity of fast-varying mobile connections, leading to unreliable service and large potential swings in the RTT. Our BBR-S algorithm replaces the max filter that causes this overestimation issue with an Adaptive Tobit Kalman Filter (ATKF), an innovation on the Kalman filter that can deal with unknown noise statistics and saturated measurements, achieving a 40% reduction in the average RTT over BBR, which increases to 60% when considering worst-case latency, while maintaining over 95% of the throughput in 4G and 5G networks.

INDEX TERMS Congestion control, latency, transport protocols, BBR.

I. INTRODUCTION

Over the next few years, several formerly impossible applications are going to become real, thanks to the novel capabilities of 5G and beyond networks: smooth high-definition video conferencing over mobile networks, remote robotic operations on the factory floor, and wireless Augmented Reality (AR) are just a few well-known examples. All these applications have a high throughput and strict latency constraints, which are expected to be met by the new Radio Access Technologies (RATs), but the existing end-to-end congestion control approaches often do not consider latency, causing self-queuing delay by probing the channel too aggressively [1]. Finding the correct balance between exploiting capacity efficiency and maintaining a low latency is still an open problem in congestion control.

The recent work by Google on this subject has led to the development of BBR [2], a new mechanism that promises to solve this problem by exploiting capacity fully and maintaining a low RTT at the same time. Google has switched to BBR for most of its outgoing traffic and included it in its new QUIC protocol [3], which is poised to replace the Transmission Control Protocol (TCP) over the next few years. BBR works by explicitly estimating the Bandwidth-Delay Product (BDP) of a connection with a simple max filter (i.e., taking the maximum measured value over a window as the "true" capacity), then using it to control the congestion window and pacing rate of the sender.

In this work, we present BBR-S, where the S stands for Sender-side Kalman Inference Procedure (SKIP): our congestion control maintains the basic structure of BBR, but replaces the max filter with the titular SKIP, a better estimation mechanism. The BBR-S mechanism can avoid BBR's pitfalls by improving capacity tracking, significantly reducing the RTT while still achieving fairness with Cubic flows and effectively exploiting capacity. The main contributions in the paper are as follows:

- We present the SKIP filter, which models capacity as the combination of a Gauss-Markov process with rare step-like events and tracks it through an ATKF, which is

The associate editor coordinating the review of this manuscript and approving it for publication was Peng-Yong Kong.

a new theoretical tool that we developed by combining the Tobit Kalman Filter (TKF) [4] and an Adaptive Kalman Filter (AKF) [5];
- We describe the operation modes that the congestion control mechanism uses to keep track of sudden shifts in the capacity, in order to react faster even in cases of blockage or cross-traffic spikes;
- We test the new mechanism in extensive ns-3 simulations over real network traces from modern wireless network, comparing it with standard BBR and other state-of-the-art congestion control mechanisms.

BBR-S manages to stay close to the actual minimum RTT while maintaining more than 90% of standard BBR's throughput in our ns-3 simulations over real network traces. The average RTT reduction over standard BBR was 40%, which increased to approximately 60% when considering the 99th percentile. This result combines the traditional advantages of Vegas, whose RTT is only similar, with the ability to share capacity fairly with more aggressive congestion control mechanisms. In fact, BBR-S can get 90% of its fair share when competing with a Cubic flow, making the protocol extremely versatile. These results show a significant end-to-end performance advantage for latency-sensitive application over existing congestion control mechanisms, including standard BBR, in fast-varying connections like those that use mmWave links.

The rest of the paper is organized as follows: first, the state of the art on congestion control is presented in Sec. II, with a focus on the BBR protocol and its issues. Then, our BBR-S solution is described in detail in Sec. III, defining the SKIP filter and its practical implementation. The simulation scenarios and results are described in Sec. IV, and Sec. V concludes the paper and lists some avenues of future research.

II. RELATED WORK

Congestion control is an essential component of modern networks, since the large number of heavy flows that the infrastructure needs to support can only be delivered if senders limit their rate before flooding connections and generating congestion. This issue is even more pressing in wireless networks, from the first commercial services in the 1990s [6] to modern mmWave links [7]: the nature of the wireless medium makes it prone to losses and fluctuations in the capacity, putting a strain on congestion control. Line of Sight (LoS) to Non-Line of Sight (NLoS) transitions in mmWave [8] are even more damaging, since they cause sharp drops in the capacity which TCP is often too slow to adapt to and recover from, creating a long queue after the drop and taking a long time to get back to full capacity after it is over.

Classic congestion control mechanisms, such as the omnipresent Cubic [9] and the older Tahoe and New Reno, interpret packet losses as a signal of congestion: they gradually increase their sending rate until a loss indicates that the bottleneck buffer is full. Filling the buffer allows Cubic to exploit capacity effectively, but the RTT can grow quickly if the bottleneck buffer is large. This has led to the emergence of the bufferbloat problem [10]: as buffers in the Internet grew, Cubic's RTT did the same, reaching values up to several seconds.

In order to maintain a low RTT, the Vegas congestion control mechanism [11] was developed with a delay-based philosophy: any increase in the RTT is interpreted as congestion, making Vegas suited to low-latency application. Vegas achieves a fair capacity exploitation with no RTT increases when sharing the bottleneck with other Vegas flows, but it has famously never been widely adopted because it cannot coexist with Cubic, since its throughput drops to almost zero when it shares a bottleneck with more aggressive loss-based mechanisms. Compound [12], which is the default mechanism on Windows machines, combines this approach with a loss-based one, attempting to strike a balance between throughput and delay. Verus [13] is a more advanced delay-based congestion control mechanism, which shares Vegas's vulnerability but works very well in fast-varying channels.

Another philosophy that has recently gained traction is learning-driven congestion control: although they are not yet mature for widespread use, the Remy [14] and Performance-oriented Congestion Control (PCC) [15] protocols are two important examples of this recent trend. The main issue that learning-driven congestion control has to face is generalization, since the mechanisms are often tied to knowledge about a specific scenario or a limited training set and cannot be used out of the box on the wider Internet without major performance losses. For a more extensive discussion of congestion control mechanisms, we refer the reader to [16], [17].

A. BBR CONGESTION CONTROL

We now look at BBR [2], the congestion control mechanism our work is based on. We examine its driving mechanism, as well as the flaws that we try to correct with BBR-S. BBR tries to explicitly estimate the BDP and stay close to the connection's optimal operating point, defined as full capacity exploitation with minimum RTT [18]. In other words, the objective of BBR is to transmit data at the highest possible rate without creating a queue. BBR uses a capacity-based philosophy, measuring capacity directly like the older Westwood mechanism [19] and the more recent Sprout [20]. The protocol has four phases, which are represented in Fig. 1:

- In the startup phase, BBR uses a gain of $2/\ln(2)$ to quickly ramp up the sending rate until the actual bandwidth is discovered. This can create a queue of up to twice the BDP, resulting in an RTT increase of twice the minimum RTT of the connection.
- In the drain phase, BBR uses the inverse of the startup gain to reduce the queue before starting normal operation.
- The bandwidth probe phase is BBR's normal mode of operation: in this phase, BBR is driven by its capacity estimates. The estimates of the capacity are passed to a max filter, whose output is the protocol's bandwidth

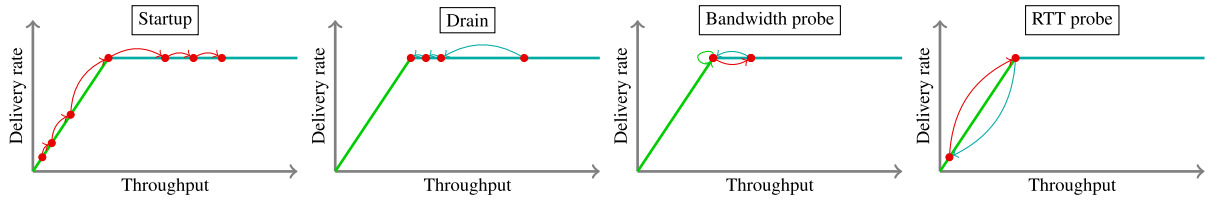


FIGURE 1. Operation of the BBR algorithm.

estimate. This optimistic estimation mechanism is not without issues, as we will discuss later, but it allows BBR to fully exploit the capacity in stable channels. The protocol then sets the pacing rate to the bandwidth and the congestion window to twice the BDP, ensuring that the reaction to capacity drops will not be too slow. Since capacity estimates are limited by the pacing rate, BBR periodically adjusts the pacing rate to 1.25 times the measured bandwidth. In this way, it builds up a queue and gets more accurate estimates of the capacity, identifying large upswings in the capacity. After one RTT, the protocol spends another RTT with a reduced pace of 0.75 times the bandwidth in order to reduce the standing queue.

- The RTT probe phase is repeated periodically, with a default of 10 s. During this phase, BBR updates its estimate of the connection's minimum RTT. In order to do so, it reduces the congestion window to 4 packets for a short period, flushing the queue at the bottleneck and ensuring that the estimate of the minimum RTT is unaffected by self-queuing delay. Naturally, other flows sharing the bottleneck buffer might still bias the estimate. After an RTT probe, operation resumes normally.

Since it does not interpret loss as a signal of congestion, relying on the BDP estimate to avoid buffer overflows, BBR does not reduce its sending rate as a consequence of packet loss. This gives it an advantage over loss-based mechanisms in naturally lossy connections such as wireless and mobile systems.

B. BBR's ISSUES IN MOBILE NETWORKS

BBR is currently on the way to become the standard TCP version for mobile networks: measurements in a highway mobility scenario show that it can achieve a far higher throughput than Cubic in fast-varying scenarios [21]. However, BBR also has some significant shortcomings: aside from being RTT-unfair [22] because of its 2BDP congestion window limit, its fairness to Cubic flow is highly dependent on the bottleneck buffer size, as BBR is very aggressive in shallow-buffer connections, but far more conservative than Cubic in bufferbloat conditions [23]. Another issue is the slow convergence of multiple flows to the fair bandwidth allocation [24], as the synchronization between RTT probing periods can take more than 30 s.

The most important issue of BBR in fast-varying networks, which we aim to solve with BBR-S, is inherent in its bandwidth estimation procedure: since capacity samples are

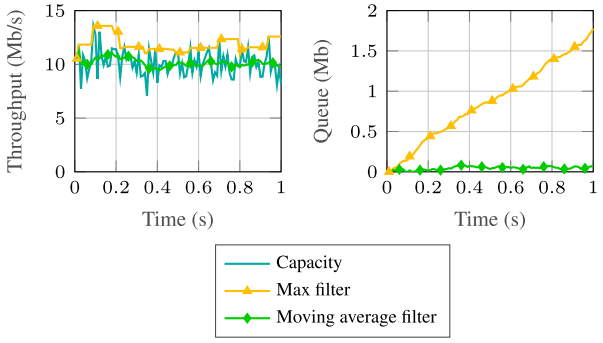


FIGURE 2. Comparison between a sending rate driven by capacity estimates based on the max filter and a simple moving average.

filtered using a simple max filter, any temporary upswing in the path capacity influences BBR's bandwidth estimate until the sample exits the max filter window [25]. The window itself has no fixed length, but expires after 10 subsequent sentinel packets, whose RTT can be much larger than the minimum RTT of the connection. In fact, it is often almost twice as much: if the pacing rate is higher than the actual capacity, packets will be queued at the bottleneck, until the 2BDP limit is reached and the congestion window limits the queuing.

The same behavior has been observed when multiple BBR flows share a bottleneck buffer [26]. In these cases, which are very common for wireless connections with volatile capacity, BBR's operating point will be far from the maximum throughput and minimum RTT objective. Connections whose bottlenecks are mmWave links will experience this phenomenon often, since the capacity of mmWave links can drop suddenly and dramatically after a transition between LoS and NLoS propagation or a beam switching event [27]. In fact, the performance of TCP over mmWave links is already a concern for the research community [7], and cross-layer solutions have already been proposed to overcome the trade-off between underutilization and high latency that traditional end-to-end congestion control faces [28].

Some of these issues are partly addressed by the BBRv2 release [29], which reduces the impact of RTT probes by making the change in the congestion window less drastic: instead of reducing it to 4 packets, the new version reduces it to half of the currently estimated BDP. It also deals with shallow buffers by probing for congestion and maintaining headroom, but quickly resuming standard operation if the buffer is large. Other changes involve an improved mechanism to deal with Explicit Congestion Notification (ECN)

and Active Queue Management (AQM) techniques, improved performance over aggregation-heavy links, and a way to handle lossy networks by setting a maximum acceptable loss rate. However, the max filter structure is unchanged even in BBRv2, so RTT performance in fast-varying networks will be similar.

BBRx [30] is a recent modification to BBR that attempts to correct for the aggressiveness of the standard mechanism. BBRx corrects the max filter output by adding a factor that depends on the difference between the current RTT and the minimum measured RTT of the connection. The parameters of the filter can be learned online.

III. THE BBR-S SOLUTION

The idea behind the BBR-S solution is simple: in order to avoid the queuing problem caused by the max filter, capacity needs to be estimated with an unbiased filter. If the filter is accurate and there are no sudden upswings in the capacity, the throughput loss with respect to standard BBR will be negligible, and the queue at the bottleneck buffer will be much shorter. Fig. 2 shows this in a simple example: we generated a normally distributed capacity trace with a mean of 10 Mb/s and a standard deviation of 1 Mb/s, then ran the max filter alongside a simple moving average (the window for both filters was 0.1 s). The two filters were used to determine the pace of the sender: as the plot on the right side shows, the constant overestimation of capacity by the max filter leads to a very large standing queue, and BBR will soon reach a full congestion window and operate at twice the minimum RTT.

The difference between the simple max filter and the BBR-S SKIP procedure is shown in Fig. 3. While BBR uses a simple rate sampling strategy and filters its outputs directly through a max filter, BBR-S operates in three steps:

1) RTT samples are processed to get a capacity sample. This procedure is more accurate than counting acknowledgments over time, as it can take variations in the pacing rate into account. However, the capacity sample is inaccurate when there is no standing queue at the bottleneck, as the capacity will be censored by the sending pace.
2) We assume that most of the time the capacity of the link is going to be slow-varying, with a certain amount of noise caused by short flows and fading. This can be represented well by a Gauss-Markov model, particularly when the filter is adaptive and can adjust the noise variance. However, all Kalman-based models can take a long time to converge in case of stepwise changes in the system, which are to be expected in wireless systems, e.g., when a heavy cross-traffic flow starts or ends, or when blockage significantly affects the channel quality. For this reason, SKIP has a *drop* mode and a *step* mode, which are triggered when capacity sharply decreases and increases, respectively.
3) If SKIP is in normal mode, the sample is filtered through the ATKF, and the pacing rate is set to the mean of the filter. In drop mode, the pacing rate is still set to the mean of the ATKF, but the filter is not updated with new capacity samples: a counter is increased, and after t_{thr} steps in drop mode, the mean of the ATKF is reset to the average capacity seen during the drop. In this way, the sending rate is not decreased at every temporary drop, but SKIP can still react quickly to persistent changes in the capacity. The requirement to avoid decreasing the sending rate does not hold for step mode: in this case, the ATKF is still evolved with the new samples, but its mean is reset after t_{thr} steps to ensure full capacity utilization.

Thanks to its three modes of operation, SKIP can deal with both small, persistent variations in the capacity, which are well-represented by a Gauss-Markov model, and large stepwise changes: resetting the mean of the filter speeds up the response to the change, giving SKIP a faster response than standard BBR. At the same time, SKIP is not significantly more complex than standard BBR, maintaining the computational cost acceptable in a high-capacity real-time setting.

Additionally, during RTT Probe periods, the congestion window is set to half the BDP instead of just 4 packets in order to estimate the minimum RTT without affecting throughput too heavily. This innovation is also present in BBRv2, but we remark that BBRv2 does not solve the max filter issue, and will suffer from the same overestimation in wireless networks. Finally, BBR-S can exit from the startup phase faster than BBR, basing it on the measured RTT instead of capacity growth: while BBR enters the drain phase only if the measured capacity grows by a factor lower than 1.25, we stop the startup exponential growth if either that condition is met or the RTT reaches 6 times the minimum, indicating that there is a large standing queue.

We would also like to remark that the Gauss-Markov model with steps is not necessarily the optimal model for any given scenario, but it satisfies three extremely important needs in the design of a congestion control algorithm. The model is *flexible*, as it can deal with most practical cases with no stability issues, and has no obvious failure modes that lead to suboptimal performance, unlike the max filter. It is *simple*, as the computational requirements of a one-dimensional ATKF are very limited, and can be executed with little effort on most modern platforms. Finally, it is *extendable*: a more accurate model, considering specific features of a given connection, can be easily integrated in the framework by modifying the underlying model of the ATKF. Furthermore, as our simulation will show, the throughput loss with respect to standard BBR is negligible, while maintaining the RTT very close to the minimum for the connection in different wireless networks. In the following, we will describe the implementation of SKIP in detail.

A. ESTIMATING CAPACITY WITH SKIP

First, we introduce the notations used in the following: given a random variable X, its expected value is denoted by $\mathbb{E}[X]$ and its variance by $\text{Var}[X]$. The conditional expectation of X given the value of Y is denoted by $\mathbb{E}[X|Y]$. Vectors like $\mathbf{x}$ are

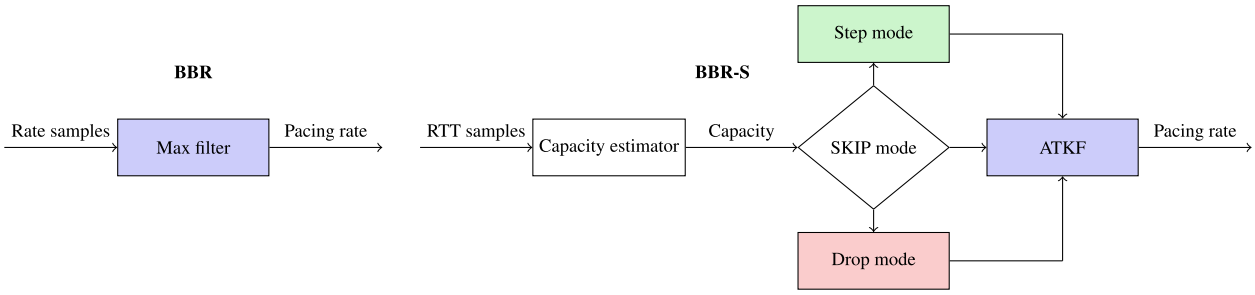


FIGURE 3. Comparison of the pacing rate adaptation in BBR and BBR-S.

TABLE 1. Main notations used in the paper.

Symbol	Meaning
t_i	Send time of the i-th packet
a_i	Arrival time of the ACK for the i-th packet
τ_i	RTT for the i-th packet
$\tau_{\min}$	Minimum RTT
L_i	Length of the i-th packet
C	Channel capacity
$X(t)$	Bytes in flight at time t
$S(t)$	Average sending rate for packets in flight
σ_w^2, Q	Process noise variance
σ_v^2, R	Measurement noise variance
$\hat{x}_{t\|\ell}$	State estimate at time t after the ℓ-th sample
η_t	Distance from censoring threshold
$\phi(x)$	Normal distribution Probability Density Function (PDF)
$\Phi(x)$	Normal distribution Cumulative Distribution Function (CDF)
$\tilde{C}_t$	Kalman error at time t
$\lambda(\alpha)$	Inverse Mills ratio
$\eth(\alpha)$	Variance equivalent Mills ratio
p_t	Censoring probability at time t
$R_{x\tilde{C},t}$	State-error covariance at time t
$R_{\tilde{C}\tilde{C},t}$	Kalman error covariance at time t
$\Psi_{t\|t-1}$	*A priori* state error variance at time t
V_t	Measurement variance at time t
$\hat{Q}_t$	Estimated process variance at time t
$\hat{R}_t$	Estimated measurement variance at time t
$\hat{V}_t$	Estimated censored measurement variance at time t
Γ_t	Estimator weighting factor at time t
γ	Estimator fading parameter
ξ_t	Estimated Kalman error at time t
μ_{drop}	Drop capacity estimate
μ_{step}	Step capacity estimate

written in bold, while matrices like $\mathbf{A}$ are written in bold and indicated with capital letters. The hat symbol indicates that the value is an estimate: $\hat{x}$ is an estimate of x. We refer to the univariate normal PDF as $\phi(\cdot)$, and to the normal CDF as $\Phi(\cdot)$. Accordingly, the estimate of the value of x at time t given the observations up to time ℓ is denoted as $\hat{x}_{t|\ell}$. The main notations used in the following are listed in Table 1.

Let $\mathbf{t} = (t_1, \ldots, t_n)$ be the vector of send times for the last n packets whose ACKs arrived in the last estimation time T_k, and $\mathbf{a} = (a_1, \ldots, a_n)$ be the vector of the corresponding ACK reception instants. Furthermore, let $\mathbf{L} = (L_1, \ldots, L_n)$ be the vector of the corresponding packet sizes. In the following, we assume that $t_i \geq t_j \ \forall i > j$ and $a_i \geq a_j \ \forall i > j$.[1]

[1]This in-order delivery hypothesis is a basic assumption of most TCP congestion control mechanisms, and holds almost universally in modern networks.

The resulting RTT vector is $\boldsymbol{\tau} = \mathbf{a} - \mathbf{t}$, and its i-th element is given by:

$$\tau_i = \tau_{\min} + \frac{L_i}{C} + q_i, \tag{1}$$

where $\tau_{\min}$ is the end-to-end forwarding, propagation and processing time for the packet and its ACK, i.e., the minimum RTT of the path, except for the contribution due to the bottleneck link, C is the bottleneck link's available bitrate, i.e., the path capacity, and q_i is the time required to get rid of the backlog at the bottleneck when the packet is generated, which is defined as:

$$q_i = \begin{cases} \dfrac{X(t_i)}{C} - (t_i - t_1), & \text{if } S(t_i) > C; \\ 0, & \text{otherwise;} \end{cases} \tag{2}$$

where $X(t_i) = \sum_{j=1}^{i} L_j$ is the number of bytes in flight at time t_i, and $S(t_i) = \frac{X(t_i)}{t_i - t_1}$ is the average sending rate between t_1 and t_i. In practice, (2) states that, if the sending rate exceeds the bottleneck capacity, the queuing delay grows linearly in time. By combining (1) and (2), we get:

$$\tau_i = \tau_{\min} + \frac{L_i}{C} + \left[\frac{X(t_i)}{C} - (t_i - t_1)\right]^+, \tag{3}$$

where $[x]^+$ is equal to x if the argument is positive, and 0 if it is negative. Consequently, we have

$$\tau_j - \tau_i = \frac{L_j - L_i}{C} + \left[\frac{X(t_j) - X(t_i)}{C} - (t_j - t_i)\right]^+, \tag{4}$$

where $j > i$. If we assume a quasi-stationary regime, i.e., $S(t_i) \simeq S(t_j)$, we get

$$\tau_n - \tau_1 = \frac{L_n - L_1}{C} + \left[\frac{S(t)(t_n - t_1)}{C} - (t_n - t_1)\right]^+. \tag{5}$$

If we take the derivative over time of the RTT on the path, we get

$$\frac{\partial \tau}{\partial t} = \frac{S(t_n)}{C} - 1, \tag{6}$$

from which we get

$$C = \frac{S(t_n)}{\frac{\partial \tau}{\partial t} + 1}. \tag{7}$$

We can now estimate $\frac{\partial \tau}{\partial t}$ by using linear least squares fitting:

$$\frac{\partial \tau}{\partial t} = \frac{\sum_{i=1}^{n} L_i \sum_{i=1}^{n} \left(t_i - \frac{\sum_{j=1}^{n} t_j}{n}\right)^2 \left(\frac{\tau_i}{L_i} - \frac{\sum_{j=1}^{n} \tau_j}{\sum_{j=1}^{n} L_j}\right)}{n \sum_{i=1}^{n} \left(t_i - \frac{\sum_{j=1}^{n} t_j}{n}\right)^2}. \tag{8}$$

By combining (7) and (8), we can get an estimate of the average capacity, which we can store to get an empirical CDF of the path capacity. The norm of the residual error of the linear fit is useful to gauge the jitter of the path, i.e., capacity fluctuations that are faster than the estimation timestep T_k.

The derivation above assumes that there is queuing, so it will not work when there is no standing queue and the sender's pace $S(t)$ is below the capacity. In that case, the estimation will simply return $S(t)$. Most capacity estimation methods share this limit, as Jaffe proved in 1981 [31]. This kind of saturation effect can be modeled using Tobit Type I censoring [32], in which measurements have an upper or lower threshold, and any value beyond the threshold is clipped.

B. TRACKING CAPACITY WITH THE ATKF

The capacity estimation method we defined above is noisy, both because of imperfections in the estimates and because of the natural fluctuations of the channel. In order to reduce this imprecision, we track the evolution of the capacity using a Kalman Filter (KF) [33], assuming that the noise is Gaussian. The TKF [32] is an extension of the KF that can deal with Tobit Type I censoring, solving the limited measurement issue. The basic system model we use is very simple, and operates on discrete timesteps with interval T_k, which is equivalent to the capacity estimation timestep:

$$x_{t+1} = x_t + w_t; \tag{9}$$

$$C_t = \min(S_t, x_t + v_t). \tag{10}$$

In the model, the measured capacity C_t is a noisy observation of a hidden state x_t, which represents the actual capacity of the channel, whose evolution is modeled by a simple Gauss-Markov process. The measured capacity is limited by the sending rate S_t. The two components w_t and v_t are Gaussian random variables with zero mean and variances $\sigma_w^2 = Q$ and $\sigma_v^2 = R$, respectively. The two variances are required inputs of the filter, which needs to distinguish between temporary variations due to v_t (*estimation noise*) and long-term effects that change the state of the system due to w_t (*process noise*). The KF gain K_t is calculated based on the *a priori* and *a posteriori* estimates of the prediction error variance. The KF is the optimal estimator if its parameters are correct for the tracked system, but it needs the values of Q and R as inputs, and its results can degrade quickly if the settings are wrong [34]. If the variances are unknown or can change over time, as in our case, there are several techniques to estimate them online while tracking the underlying process [35]. These extensions of the KF are called AKFs. In the following, we describe the TKF and extend the recursive AKF implementation from [5] to the Tobit Type I censored case, deriving the Adaptive Tobit Kalman Filter (ATKF). Other AKF techniques such as Autocovariance Least Squares (ALS) [36] can also be used, but the adaptations required to make them work in the Tobit case are more extensive.

We now quickly report the derivation of the TKF, which is adapted from [4]. To simplify the notation in the following steps, we denote the normalized distance from the censoring threshold as η_t:

$$\eta_t = \frac{S_t - \hat{x}_{t|t-1}}{\sigma_v}, \tag{11}$$

where $\hat{x}_{t|t-1}$ is the *a priori* estimate of the hidden state x_t. This parameter is critical in the TKF, as the censoring probability is computed directly from it and, consequently, so are the filter update equations and Kalman error. We also define the inverse Mills ratio, i.e., $\mathbb{E}[X|X < \alpha]$, which denoted as $\lambda(\alpha)$ for the univariate normal case:

$$\lambda(\alpha) = -\frac{\phi(\alpha)}{\Phi(\alpha)}. \tag{12}$$

Its equivalent for the variance is $\eth(\alpha)$:

$$\eth(\alpha) = \lambda(\alpha)\left(\lambda(\alpha) - \alpha\right). \tag{13}$$

Using the notation defined above, the first two moments of C_t are given by:

$$\mathbb{E}[C_t|x_t, \sigma_v] = (1 - \Phi(\eta_t))S_t + \Phi(\eta_t)\left(x_t - \sigma_v \lambda(\eta_t)\right); \tag{14}$$

$$\mathrm{Var}[C_t|x_t, \sigma_v] = \sigma_v\left[1 - \eth\left(\eta_t\right)\right]. \tag{15}$$

The Kalman error $\tilde{C}_t$ is then given by:

$$\tilde{C}_t = C_t - \mathbb{E}[C_t|\hat{x}_{t|t-1}, \sigma_v]. \tag{16}$$

Additionally, the Bernoulli variable p_t represents the censoring of the measurements: the variable is 0 if the measurement is censored and 1 if it is not. It is given by:

$$p_t = I(C_t < S_t), \tag{17}$$

where $I(\cdot)$ is the indicator function. The expected value of the censoring variable is:

$$\mathbb{E}[p_t] = \Phi\left(\eta_t\right). \tag{18}$$

The covariance $R_{x\tilde{C},t} = \mathbb{E}\left[(x_t - x_{t-1})\tilde{C}_t\right]$ is:

$$R_{x\tilde{C},t} = \Psi_{t|t-1} C_t \mathbb{E}[p_t], \tag{19}$$

where $\Psi_{t|t-1} = \mathbb{E}[(x_t - \hat{x}_{t|t-1})^2]$ is the predicted *a priori* state error variance. The variable $\tilde{v}_t = \sigma_v \lambda\left(\eta_t\right)$ represents the bias introduced in the measurement noise by the censoring. As in [4], the Kalman error covariance matrix is $R_{\tilde{C}\tilde{C},t} = \mathbb{E}\left[\tilde{C}_t^2\right]$:

$$R_{\tilde{C}\tilde{C},t} = \mathbb{E}[p_t]^2 \Psi_{t|t-1} + \mathbb{E}[p_t(v_t - \tilde{v}_t)^2]. \tag{20}$$

The second term of the sum corresponds to the measurement variance $V_t = \mathrm{Var}[C_t|\hat{x}_{t|t-1}, \sigma_v]$. The Kalman gain is calculated as:

$$K_t = R_{x\tilde{C},t} R_{\tilde{C}\tilde{C},t}^{-1}. \tag{21}$$

The TKF is then given by:

$$\hat{x}_{t|t-1} = \hat{x}_{t-1|t-1}; \tag{22}$$
$$\Psi_{t|t-1} = \Psi_{t-1|t-1} + Q_{t-1}; \tag{23}$$
$$\hat{x}_{t|t} = \hat{x}_{t|t-1} + R_{x\tilde{C},t} R^{-1}_{\tilde{C}\tilde{C},t} \tilde{C}_t; \tag{24}$$
$$\Psi_{t|t} = \left((1 - \mathbb{E}[p_t]) R_{x\tilde{C},t} R^{-1}_{\tilde{C}\tilde{C},t}\right) \Psi_{t|t-1}. \tag{25}$$

The ATKF combines the TKF from [4] with the recursive AKF implementation from [5]. In the following, we repeat the derivation of the noise covariance estimator, adapting it to the TKF. The *a posteriori* density function of the noise covariance of a standard Kalman filter can be estimated using the Maximum A Posteriori (MAP) coupling form [37]:

$$\hat{Q}_t = \frac{1}{t} \sum_{\ell=1}^{t} \left[\left(\hat{x}_{t|\ell} - \hat{x}_{t|\ell-1} \right)^2 \right]; \tag{26}$$
$$\hat{R}_t = \frac{1}{t} \sum_{\ell=1}^{t} \left[\left(C_\ell - \hat{x}_{t|\ell} \right)^2 \right]. \tag{27}$$

Even in the standard scenario with no censoring, the terms $\hat{x}_{t|\ell-1}$ and $\hat{x}_{t|\ell}$ require multiple prediction steps and make the optimal formulation above impractical, as it cannot be described in recursive form. A practical one-step approximation is presented in [5]. In the TKF case, the derivation for the practical unbiased estimator for $\hat{Q}_t$ follows from (9). The process noise variance is still $Q = \mathbb{E}\left[(x_t - x_{t-1})^2\right]$, and if we assume that the ATKF state estimate is close enough to the real state, i.e., $\hat{x}_{t|t} \simeq x_t$ (an assumption that the standard TKF also requires) we get:

$$Q = \mathbb{E}\left[(x_t - x_{t-1})^2 \right] \tag{28}$$
$$\simeq \mathbb{E}\left[(\hat{x}_{t|t} - \hat{x}_{t-1|t-1})^2 \right] \tag{29}$$
$$\simeq \mathbb{E}\left[(\hat{x}_{t|t} - \hat{x}_{t|t-1})^2 \right] \tag{30}$$
$$\simeq \mathbb{E}\left[\left(K_t \tilde{C}_t \right)^2 \right], \tag{31}$$

where K_t is the Kalman gain of the filter at time t and $\tilde{C}_t = C_t - \mathbb{E}\left[C_t | \hat{x}_{t|t-1}\right]$ is the Kalman error. We can now estimate the value of Q from the available samples:

$$\hat{Q}_t = \frac{1}{t} \sum_{\ell=1}^{t} \left(K_\ell \tilde{C}_\ell \right)^2. \tag{32}$$

The expected value of the process noise covariance is:

$$\mathbb{E}\left[\hat{Q}_t\right] = \mathbb{E}\left[\frac{1}{t} \sum_{\ell=1}^{t} \left(K_\ell \tilde{C}_\ell \right)^2 \right] \tag{33}$$
$$= \frac{1}{k} \mathbb{E}\left[\sum_{\ell=1}^{t} \mathbb{E}[p_\ell] K_\ell \Psi_{\ell|\ell-1} \right] \tag{34}$$
$$= \mathbb{E}\left[\frac{1}{t} \sum_{\ell=1}^{t} \Psi_{\ell|\ell-1} - \Psi_{\ell|\ell} \right]. \tag{35}$$

The *a posteriori* state error variance $\Psi_{\ell|\ell}$ is given in (25). The derivation of (34) follows from the fact that $\mathbb{E}\left[\tilde{C}_t^2\right] = R_{\tilde{C}\tilde{C},t}$. In the following step, we use (25) to remove the Kalman gain from the equation. We can now subtract the expected value in (35) from (32) to write the unbiased estimator for the process noise covariance:

$$\hat{Q}_t = \frac{1}{t} \sum_{\ell=1}^{t} \left(K_\ell \tilde{C}_\ell \right)^2 + \Psi_{\ell|\ell} - \Psi_{\ell-1|\ell-1}. \tag{36}$$

The derivation of (36) is the same as in [5], using the TKF modified equations instead of the standard KF. The estimation noise variance R_t does not have a linear estimator, since the Kalman error covariance matrix $R_{\tilde{C}\tilde{C},t}$, given by (20), contains the censored measurement noise variance $V_t = \mathbf{E}\left[(\max(v_t, S_t - x_t))^2\right]$ instead of R_t. We get the unbiased estimator for $\hat{V}_t$ by substituting the TKF equations into the one-step version of (27):

$$\hat{V}_t = \frac{1}{t} \sum_{\ell=1}^{t} (1 - K_\ell) \left(\tilde{C}_\ell^2 (1 - K_\ell) + R_{x\tilde{C},\ell} \right). \tag{37}$$

In order to estimate $\hat{R}_t$, we can now simply invert the formula for computing the variance of a censored Gaussian random variable, which is given in (15):

$$\hat{R}_t = \frac{\hat{V}_t}{1 - \eth(\hat{\eta}_t)}; \tag{38}$$
$$\hat{\eta}_t = \frac{S_t - \hat{x}_{t|t-1}}{\hat{\sigma}_{v,t-1}}, \tag{39}$$

where $\hat{\sigma}_{v,t-1} = \sqrt{\hat{R}_{t-1}}$ and $\hat{\eta}_t$ is the estimate of the normalized distance of the current state from the censoring threshold. The resulting heuristic estimator is not unbiased, since the function is non-linear, but corresponds to the one-step Maximum Likelihood Estimator (MLE) for the censored Gaussian distribution, as derived by Gupta [38].

The TKF converges to the standard KF when the censoring region is far from the state value. The noise estimator also converges to the unbiased noise estimator used in the AKF. After substituting the terms in (36), the estimator is the one in [5]. The same goes for the measurement noise estimator, since $\lim_{\eta_t \to -\infty} \hat{R}_t = \hat{V}_t$. In order to deal with time-varying noise, the estimation of the noise covariances needs to be performed over a limited window in time. This can be done with a simple lowpass filter, so that the new estimate of the covariance is the linear combination of the old estimate and the latest sample, weighted by a factor Γ_t:

$$\Gamma_t = \frac{1 - \gamma}{1 - \gamma^t}, \tag{40}$$

where $\gamma \in [0, 1)$ is a fading factor. Lower values of the fading factor correspond to a higher weight to new samples, and setting $\gamma = 0$ is equivalent to only considering the latest sample. However, the covariance samples are sensitive to noise outliers, and the estimator might even diverge. For this reason, the estimator can use an Innovation Covariance

Estimator (ICE) that averages the covariance samples over a rectangular sliding window to guarantee that noise covariance matrices are semidefinite positive [39]:

$$\xi_t = \frac{1}{N} \sum_{\ell=t-N+1}^{t} \left[\tilde{C}_\ell^2 \right], \tag{41}$$

where N is the size of the sliding window. The ICE is another lowpass filter, and its length determines the reactiveness of the estimator. If the noise statistics are fast-varying, shorter windows are recommended. It is now possible to rewrite the estimator recursive equations, considering (40) and (41):

$$\hat{Q}_t = (1-\Gamma_t)\hat{Q}_{t-1} + \Gamma_t \left(K_t^2 \xi_t + \Psi_{t|t} - \Psi_{t-1|t-1} \right); \tag{42}$$

$$\hat{V}_t = (1-K_t) \left(K_t \xi_t (1-K_t) + R_{x\tilde{C},t} \right); \tag{43}$$

$$\hat{R}_t = (1-\Gamma_t)\hat{R}_{t-1} + \Gamma_t \left(1 - \eth \left(\hat{\eta}_t \right) \right)^{-1} \hat{V}_t. \tag{44}$$

The full ATKF is given by the full estimator in (42)-(44), along with the standard update formulas in (22)-(25). While it is more complex than a max filter, its complexity is still linear in the number of packets, and an efficient implementation can run in real time with no issues even in mobile processor. The most complex component of the filter is the ICE, which can be implemented in a way that only requires 4 operations at each step by exploiting previous steps. All other components are a simple series of conditions and basic mathematical operations, which are not significantly more complex than mechanisms such as Cubic.

C. SKIP OPERATION

The ATKF can deal well with gradual changes to the capacity, but it does not respond quickly enough to sudden capacity drops or even full outages. Wireless channels are often characterized by sharp drops and temporary outages, and this issue is particularly significant at higher frequencies, where transitions from LoS to NLoS propagation due to blockage from walls, objects or humans are very fast. For this reason, SKIP has three modes of operation, which we will describe below: the pseudocode for the filter mode operation is given in Algorithm 1.

In order to describe the operation mode logic, we will consider time step t, for which the ATKF has a mean $\hat{x}_{t|t-1}$ and a total variance $\Psi_{t|t-1}+Q+R$. First, we examine the condition to trigger the drop mode. If the sender is in app-limited mode, i.e., the application is not providing enough data to fill the available capacity, BBR-S should not go into drop mode: the measured capacity is an artifact of the limited sending rate. We then formulate a first condition:

$$C_t < S_t. \tag{45}$$

This condition is checked in line 2 of the pseudocode. We then state a second condition, considering the full outage case: if the measured capacity is 0 for at least t_{thr} samples, the link is in outage and the sending rate should be reduced drastically:

$$\sum_{k=t}^{t-t_{\text{thr}}} C_k = 0 \wedge \hat{x}_{t|t-1} > 0. \tag{46}$$

If this second condition, implemented in line 7 of the pseudocode, is verified, the mean of the ATKF is reset to the initial minimum value (50 kB/s in our setup) straight away.

We now define another condition, which gives the drop mode its name: a sharp capacity drop. We define a capacity drop as a 3σ event for the ATKF: according to the model, the probability that the capacity is lower than the ATKF's mean by more than 3 times its standard deviation should be about 10^{-3}, but propagation conditions can make drops happen far more often. The condition for a drop is

Algorithm 1 SKIP Mode Operation

```
1:  function SKIPmode(C, S, ATKF, t_drop, t_step, t_out)
2:      if C ≥ S then                                  ▷ Application-limited mode
3:          t_drop, t_step, t_out ← 0;
4:          update(ATKF, C, S);
5:          return;
6:      end if
7:      if C = 0 then                                  ▷ Outage mode
8:          t_out ← t_out + 1;
9:          t_step ← 0;
10:         if t_out = t_thr then                      ▷ Reset filter
11:             t_out ← 0;
12:             reset(ATKF, C_min);
13:             return;
14:         end if
15:     else
16:         t_out ← 0;
17:     end if
18:     if C < ATKF.x − 3ATKF.σ then                   ▷ Drop mode
19:         t_drop ← t_drop + 1;
20:         t_step ← 0;
21:         blindUpdate(ATKF);
22:         if t_drop = t_thr then                     ▷ Reset filter
23:             t_drop ← 0;
24:             reset(ATKF, μ_drop);
25:         end if
26:     else
27:         t_drop ← 0;
28:         if C < ATKF.x + 3ATKF.σ then               ▷ Step mode
29:             t_step ← t_step + 1;
30:             if t_step = t_thr then                 ▷ Reset filter
31:                 t_step ← 0;
32:                 reset(ATKF, μ_step);
33:             end if
34:         else                                       ▷ Normal mode
35:             t_step ← 0;
36:         end if
37:         update(ATKF, C, S);
38:     end if
```

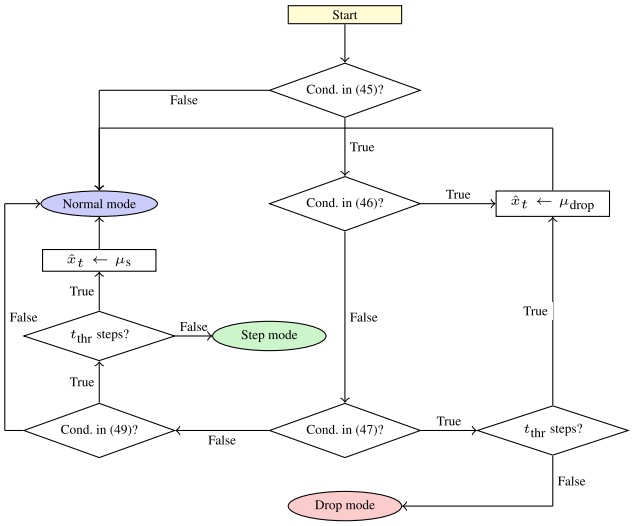


FIGURE 4. Block diagram of the filter operation modes.

then given by:

$$C_t < \hat{x}_{t|t-1} - 3\sqrt{\Psi_{t|t-1} + Q + R}. \tag{47}$$

This condition is implemented in line 18 of the pseudocode. During the drop, we keep evolving the ATKF by running the prediction step, thus gradually increasing its variance, but do not update it with the new capacity samples. We consider the drop over after τ steps if the measured capacity is higher than $\hat{x}_{t|t-1} - 3\sqrt{\Psi_{t|t-1} + (\tau+1)Q + R}$. If the drop lasts for a number of steps $t_{\text{drop}} \geq t_{\text{thr}}$, we consider the capacity shift to be permanent and reset the mean of the ATKF $\hat{x}$ to accelerate the adaptation to the new conditions. The new value of the mean is the average capacity measured during the drop:

$$\mu_{\text{drop}}(t) = \frac{1}{t_{\text{thr}}} \sum_{\tau=t}^{t-t_{\text{thr}}+1} C_t. \tag{48}$$

The drop mode increases the filter's reactiveness to sudden drops in the capacity, making the protocol more resistant to this kind of issues and limiting the build-up in the bottleneck buffer. During the drop mode, the mean of the ATKF is unchanged, so the pacing rate is as well, but the mean reset can allow BBR-S to react faster to sudden capacity drops. The number of necessary samples t_{thr} increases the robustness of the mechanism to random temporary changes in the capacity, but there is a trade-off between stability and responsiveness. We found that setting $t_{\text{thr}} = 3$ guarantees that the mechanism avoids following noise in the capacity measurement, while still strongly outperforming standard BBR in terms of responsiveness to outage event. With a longer t_{thr}, the filter would take longer to react to changes in the capacity, operating with a wrong estimate for a longer time. On the other hand, setting t_{thr} to 1 or 2 would make it more vulnerable to temporary changes due to physical layer effects, such as fast fading: resetting the filter mean every time it receives an outlier sample would reduce its stability and, consequently, its ability to filter out noise from the capacity measurements.

TABLE 2. Main parameters of the simulation scenario.

Parameter	Value
Minimum RTT	50 ms
BBR-S Kalman timestep	10 ms
BBRx gain β	0.2
BBRx RTT parameter K	1.25
Simulation duration	100 s
Bottleneck buffer	20 MB

We also consider a third mode of operation for SKIP, which we named step mode. In this case, the mechanism needs to react to a sharp step upwards in the capacity. The issues of the ATKF are the same as with drops, with the added problem of the censoring, so the unaided ATKF can be very slow to reach the full capacity. For this reason, we turn on step mode if the opposite condition to drop mode is verified:

$$C_t > \hat{x}_{t|t-1} + 3\sqrt{\Psi_{t|t-1} + Q + R}. \tag{49}$$

During step mode operation, the ATKF is evolved normally, but if the condition persists for more than t_{thr} steps, its mean is reset to the average capacity measured during the step:

$$\mu_{\text{step}}(t) = \frac{1}{t_{\text{thr}}} \sum_{\tau=t-t_{\text{thr}}+1}^{t}. \tag{50}$$

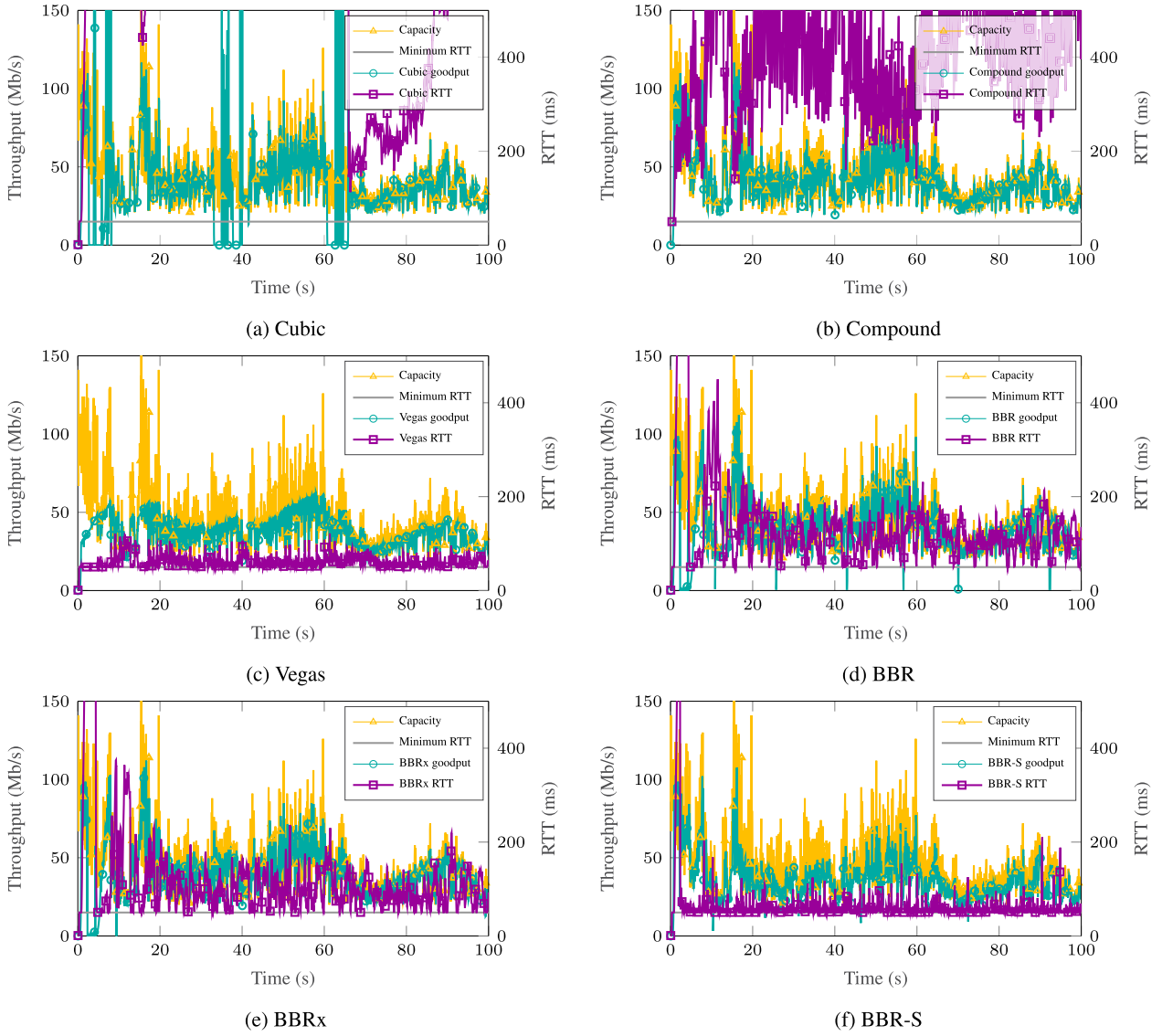
As for the drop mode, the trade-off in setting t_{thr} is between underexploiting capacity if the threshold is too high and reducing stability if it is too low. The step mode also helps the filter converge faster in the startup phase, making its convergence fast enough even in high-capacity scenarios. As soon as BBR-S exits its drain phase, the filter is already operational, with a good estimate of the current capacity. The modes of operation of the filter are also graphically illustrated in the block diagram in Fig. 4. These conditions allow BBR-S to avoid BBR's failure modes, maintaining responsiveness to sudden events while following the trend of the capacity, and are a fundamental part of the mechanism's operation.

IV. TESTING SCENARIO AND RESULTS

In order to test the performance of BBR-S, we simulated the protocol over ns-3 in a controlled scenario. We used two `PointToPointChannel` objects to simulate the paths, varying their capacity according to two real network traces:

- An LTE trace from a driving mobility scenario, with highly variable capacity but no sharp drops.
- A WiGig trace from an experimental setup, with frequent sharp drops due to LoS/NLoS transitions.

In our simulation, a backlogged source sends data to a client using TCP, and we measure performance in terms of achieved throughput and RTT. In order to show the performance of BBR-S, we compare it with legacy BBR and BBRx, along with three traditional congestion control mechanisms: Cubic, Compound, and Vegas. In our simulation, we assumed that



(a) Cubic (b) Compound

(c) Vegas (d) BBR

(e) BBRx (f) BBR-S

FIGURE 5. Goodput and RTT over time in the LTE trace.

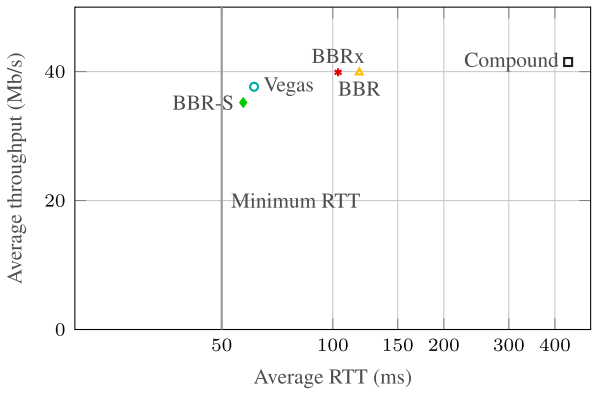


FIGURE 6. Average throughput versus RTT comparison in the LTE scenario.

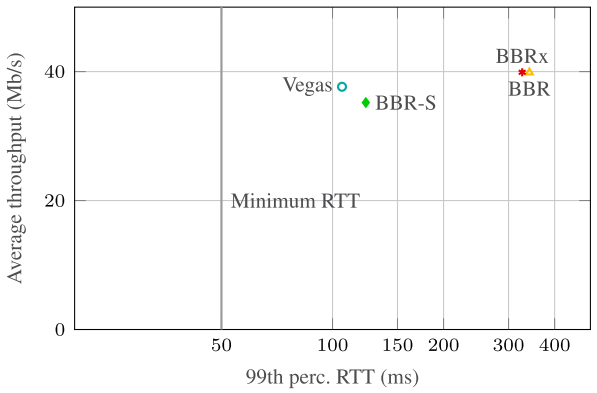


FIGURE 7. Average throughput versus 99th percentile RTT comparison in the LTE scenario.

there are no physical layer losses in order to compare the algorithms fairly: while earlier mechanisms, including Vegas, Cubic, and Compound, always interpret packet loss as a sign of congestion, BBR variants, including BBRx and BBR-S, are relatively insensitive to it and infer congestion from other measurements. This gives all BBR-based algorithms

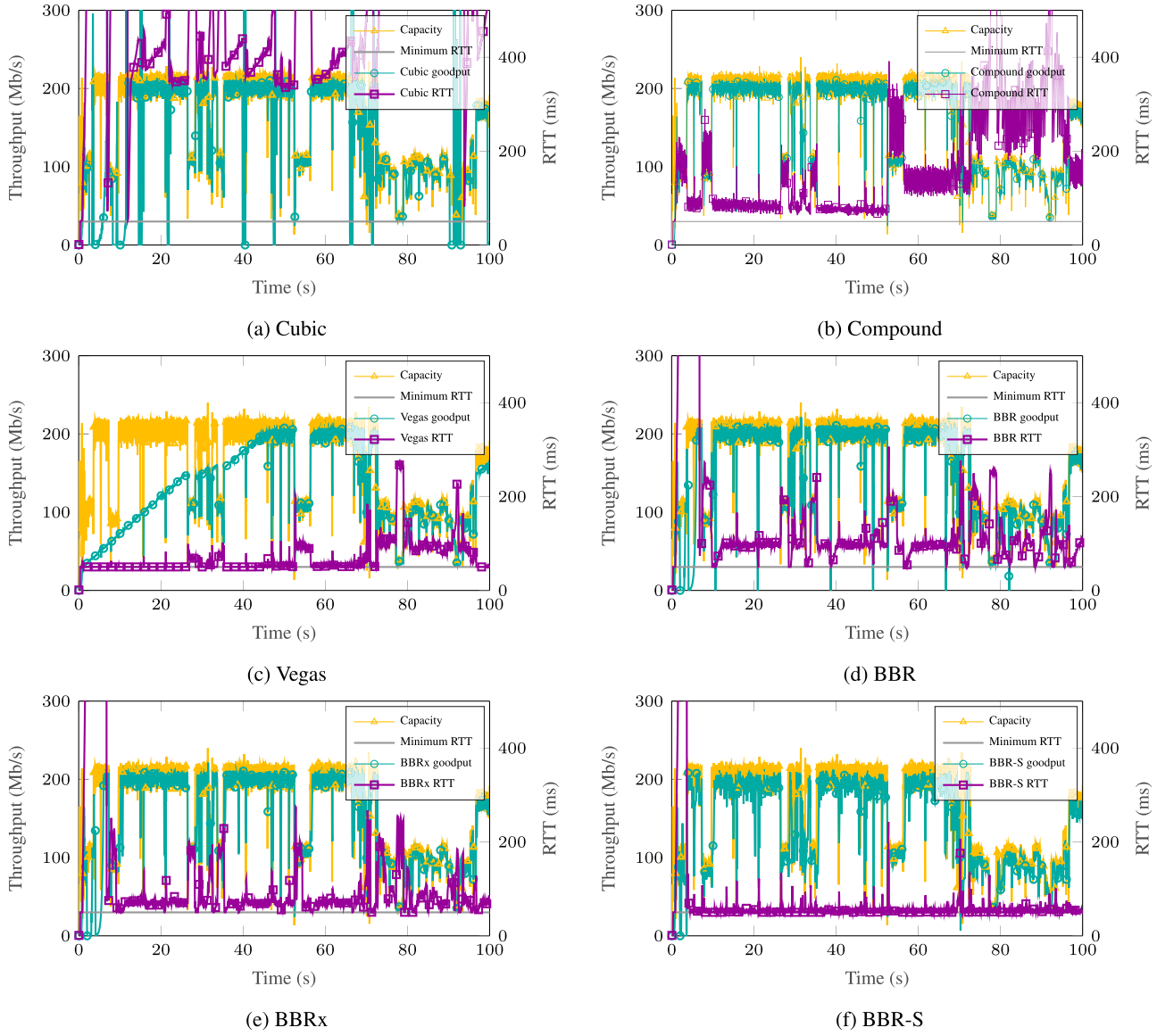


FIGURE 8. Goodput and RTT over time in the WiGig trace.

an advantage over lossy connections, as they can still fully exploit the capacity, without reacting to individual packet losses by halving the congestion window.

In the LTE scenario, BBR-S is able to exploit the capacity almost as well as the more aggressive algorithms, but performs much better in terms of RTT, as Fig. 5 shows. In particular, BBR-S has a faster start than Vegas, and can follow the average throughput, but does not exploit short upward spikes as well as BBR, BBRx, Compound, or Cubic, as it does not maintain a long queue. Note that, in this case, BBRx performs almost identically to BBR in terms of RTT and throughput, while BBR-S can control the latency much better. In this case, Compound and Cubic can generate huge delays, and are not shown in the figure because their average RTT is over 500 ms.

Fig. 6 shows a comparison of the average throughput and RTT obtained by the different congestion control algorithms. While BBR-S and Vegas have a slightly lower throughput, they are able to maintain a far lower RTT, both on average and in the 99th percentile, as Fig. 7 shows. BBR-S stays very close to the minimum RTT, while other versions of BBR perform far worse, and Compound and Cubic can go far over 500 ms (as above, they are not shown in the figure).

The benefits that BBR-S gets from running the SKIP capacity estimation and filtering are even more evident in the WiGig scenario, as Fig. 8 shows. BBR-S is the only algorithm that can follow the transition from LoSto NLoS that happens after the 70 second mark, tracking the capacity while remaining close to the minimum RTT when even Vegas has a long-term increase in the delay. As above, Cubic and

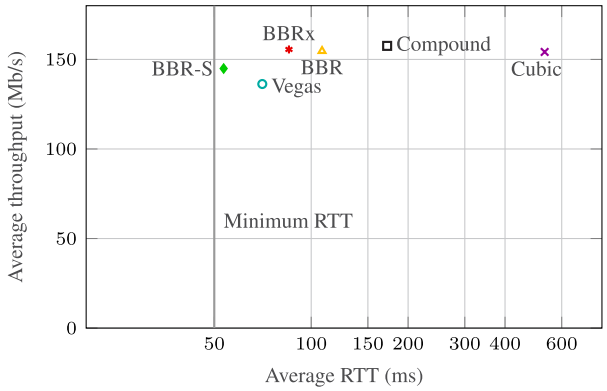


FIGURE 9. Average throughput and RTT comparison in the WiGig scenario.

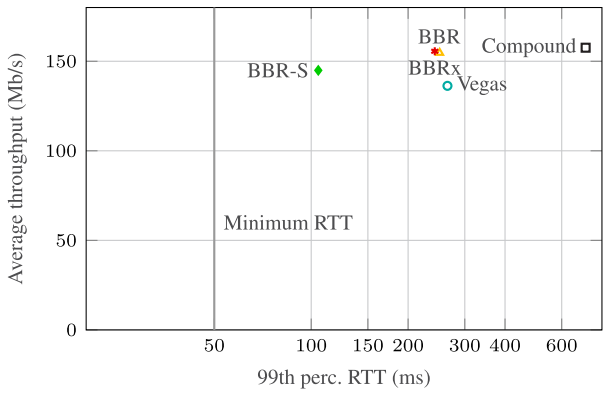


FIGURE 10. Average throughput and 99th percentile RTT comparison in the WiGig scenario.

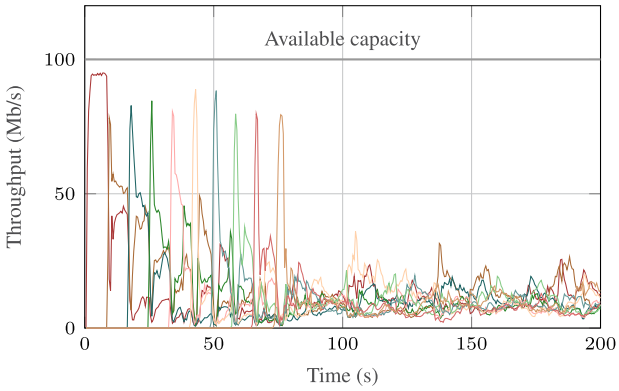


FIGURE 11. Throughput of 10 BBR-S flows over a 100 Mb/s bottleneck.

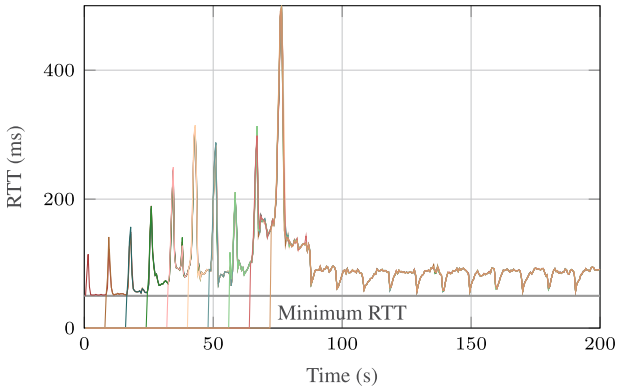


FIGURE 12. 10 BBR-S flows over a 100 Mb/s bottleneck.

Compound perform worse than any of the other mechanisms in terms of delay, with RTTs over 500 ms during the NLoS event. The biggest issue with Vegas is the extremely slow start, as it does not reach the full capacity until the 40 second mark, while the BBR versions and Cubic have an extremely aggressive slow start, bringing the RTT over 500 ms. Thanks to the SKIP filter, BBR-S can reduce the start-up time by about half, starting normal operation after less than 4 seconds (i.e., as soon as the SKIP filter converges to the connection capacity), while BBR and BBRx's performance suffers for almost 10.

We can also observe that BBRx is better than standard BBR at maintaining a low RTT, but still has increased delays when the capacity changes. We can see BBRx as an intermediate solution between our BBR-S, which loses some throughput but maintains a very low RTT, and standard BBR. This is confirmed by Fig. 9, which shows the average throughput and RTT of the different congestion control mechanisms (measured after the 10 second mark, to remove the effect of the slow start): while BBR-S has a 10% loss in the average goodput with respect to the other algorithms (Vegas has 20%), it has an average RTT just above the minimum, doing better than even Vegas. This is even more evident if we look at the 99th percentile of the RTT, which represents the worst-case scenario and is crucial for reliability-oriented applications: as Fig. 10 shows, BBR, BBRx, and Vegas reach an RTT of approximately 250 ms, while BBR-S only arrives at 100 ms. Cubic is not even shown in the figure, as the 99th percentile of the RTT is over 1 second.

A. BEHAVIOR WITH MULTIPLE FLOWS

In order to verify the BBR-S congestion control mechanism's fairness, we test the scenario in which 10 flows compete for a shared bottleneck with a capacity of 100 Mb/s and a minimum RTT of 50 ms. The flows arrive at an 8 second interval, as Fig. 11 shows. The spikes in the throughput are due to the start-up phase of new flows, which probes the capacity like standard BBR. However, after a few seconds, the flows get back to a fair share of capacity, albeit with some oscillations. The Jain Fairness Index (JFI) is a measure of fairness in communication networks that we can use to gauge the fairness of the congestion control mechanism:

$$\mathcal{J}(x_1, \ldots, x_n) = \frac{\left(\sum_{i=1}^{n} x_i\right)^2}{n \sum_{i=1}^{n} (x_i)^2}. \tag{51}$$

An ideal division into exactly equal shares would get a JFI of 1. The JFI of the 10 BBR-S flows is 0.9711, which is very close to the optimum. As Fig. 12 shows, the RTT of the flows is also limited: while the performance is slightly worse than in

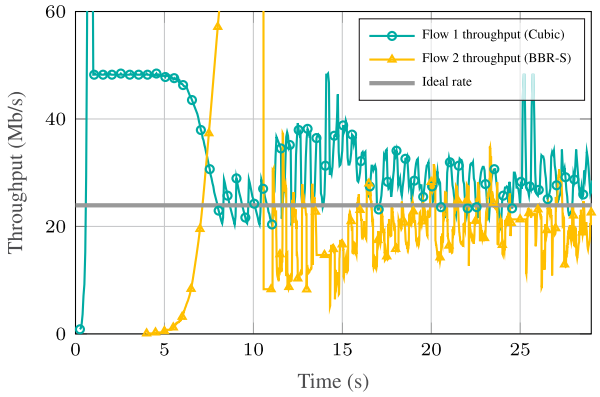


FIGURE 13. Throughput of a Cubic flow and a BBR-S flow over a 50 Mb/s bottleneck.

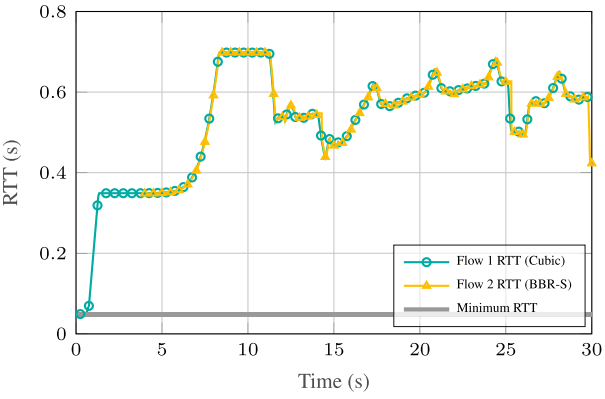


FIGURE 14. RTT of a Cubic flow and a BBR-S flow over a 50 Mb/s bottleneck.

the single-flow case, the average RTT after all the flows have started is just 84 ms, and no flow's RTT goes over 100 ms outside of the startup phases.

In modern cellular networks, data for different pairs of hosts is stored in separate buffers [20], and if the wireless link is the bottleneck of the connection, flows will not interact directly. Cross-traffic will be experienced as a uniform reduction in capacity, since wireless resources are allocated by a scheduler that tries to maintain fairness while each flow is on a separate buffer. In this case, self-interaction is the foremost cause of queuing delay, and BBR-S can keep the RTT close to the minimum thanks to BBR-S's more accurate estimates of the available capacity. If the bottleneck is not the cellular wireless access, but rather an 802.11 link, or in the network core, BBR-S will have no performance loss over BBR, which is here considered as the state-of-the-art protocol. By inheriting the basic structure of BBR, BBR-S also gets its resistance to lossy connections.

BBR-S can also coexist fairly with Cubic, while BBR is arguably too aggressive [24] and highly sensitive to buffer size, even though the next version of the protocol [29] promises to overcome these issues. Fig. 13 shows the throughput over time of a Cubic flow and a BBR-S flow sharing a 50 Mb/s bottleneck, with a minimum RTT of 50 ms for both flows. The BBR-S flow takes up between 80% and 90% of its fair share of capacity. The two RTT Probe phases after approximately 15 and 25 seconds make Cubic quickly occupy the whole space. However, the shallow RTT probing makes the protocol quickly get back its half of the capacity. As Fig. 14, the RTT can grow significantly, as Cubic does not try to maintain a low latency.

V. CONCLUSION

In this work, we presented BBR-S, an improvement to the BBR congestion control scheme that significantly improves its latency with negligible capacity losses. While the max filter used by BBR tends to make it too aggressive in fast-varying connections, making the congestion window the limiting factor on the send rate, and not the pace as BBR was designed for, our SKIP filter maintains the send rate close to the capacity. In order to effectively track capacity from censored observations, which are limited by the pace at which the packets were sent, we developed the ATKF, an adaptation of the TKF that can deal with linear systems with unknown noise covariances. Furthermore, SKIP can overcome the limitations of Kalman-based filters by adapting directly to sudden transitions in the capacity.

The results of our ns-3 simulations show that BBR-S can have more than 90% of BBR's throughput while reducing the average and 99th percentile RTT by more than 50% when operating on capacity traces from real wireless network environments. In particular, its reactiveness to sharp capacity drops makes it ideal for WiGig scenarios. Unlike Vegas, the other common latency-oriented congestion control mechanism, it can also share a link fairly with capacity-oriented flows, albeit without most of its latency advantages. BBR-S is far better at maintaining a low latency than current mechanisms, as well as adaptable to different scenarios and able to deal with sudden changes in the capacity.

Future extensions of the protocol will focus on further refinements on the filter, including more complex statistical tools such as Gaussian Mixture Models (GMMs) to improve the capacity tracking. Other improvements to BBR such as the ones suggested for BBRv2 might be patched in, reducing the impact of bandwidth probing on the protocol's latency and improving fairness.

REFERENCES

[1] M. Polese, R. Jana, and M. Zorzi, "TCP and MP-TCP in 5G mmWave networks," *IEEE Internet Comput.*, vol. 21, no. 5, pp. 12–19, Sep. 2017.
[2] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh, and V. Jacobson, "BBR: Congestion-based congestion control," *ACM Queue*, vol. 14, no. 5, pp. 50-20–50-53, Oct. 2016.
[3] A. Langley, A. Riddoch, A. Wilk, A. Vicente, C. Krasic, D. Zhang, F. Yang, F. Kouranov, I. Swett, J. Iyengar, and J. Bailey, "The QUIC transport protocol: Design and Internet-scale deployment," in *Proc. Conf. ACM Special Interest Group Data Commun. (SIGCOMM)*, Aug. 2017, pp. 183–196.
[4] B. Allik, C. Miller, M. J. Piovoso, and R. Zurakowski, "The tobit Kalman filter: An estimator for censored measurements," *IEEE Trans. Control Syst. Technol.*, vol. 24, no. 1, pp. 365–371, Jan. 2016.
[5] W. Gao, J. Li, G. Zhou, and Q. Li, "Adaptive Kalman filtering with recursive noise estimator for integrated SINS/DVL systems," *J. Navigat.*, vol. 68, no. 1, pp. 142–161, Jan. 2015.

[6] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz, "A comparison of mechanisms for improving TCP performance over wireless links," *IEEE/ACM Trans. Netw.*, vol. 5, no. 6, pp. 756–769, Dec. 1997.
[7] M. Zhang, M. Polese, M. Mezzavilla, J. Zhu, S. Rangan, S. Panwar, and M. Zorzi, "Will TCP work in mmWave 5G cellular networks?" *IEEE Commun. Mag.*, vol. 57, no. 1, pp. 65–71, Jan. 2019.
[8] M. Zhang, M. Mezzavilla, R. Ford, S. Rangan, S. Panwar, E. Mellios, D. Kong, A. Nix, and M. Zorzi, "Transport layer performance in 5G mmWave cellular," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Apr. 2016, pp. 730–735.
[9] S. Ha, I. Rhee, and L. Xu, "CUBIC: A new TCP-friendly high-speed TCP variant," *ACM SIGOPS Operating Syst. Rev.*, vol. 42, no. 5, pp. 64–74, Jul. 2008.
[10] J. Gettys and K. Nichols, "Bufferbloat: Dark buffers in the Internet," *Queue*, vol. 9, no. 11, p. 40, Nov. 2011.
[11] L. S. Brakmo, S. W. O'Malley, and L. L. Peterson, "TCP Vegas: New techniques for congestion detection and avoidance," *ACM Comput. Commun. Rev.*, vol. 24, no. 4, pp. 24–35, Oct. 1994.
[12] K. Tan, J. Song, Q. Zhang, and M. Sridharan, "A compound TCP approach for high-speed and long distance networks," in *Proc. 25th IEEE Int. Conf. Comput. Commun. (IEEE INFOCOM)*, Apr. 2006, pp. 1–12.
[13] Y. Zaki, T. Pötsch, J. Chen, L. Subramanian, and C. Görg, "Adaptive congestion control for unpredictable cellular networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 4, pp. 509–522, Sep. 2015.
[14] K. Winstein and H. Balakrishnan, "TCP ex machina: Computer-generated congestion control," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 4, pp. 123–134, Sep. 2013.
[15] M. Dong, Q. Li, D. Zarchy, P. B. Godfrey, and M. Schapira, "PCC: Re-architecting congestion control for consistent high performance," in *Proc. 12th USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, May 2015, pp. 395–408.
[16] M. Polese, F. Chiariotti, E. Bonetto, F. Rigotto, A. Zanella, and M. Zorzi, "A survey on recent advances in transport layer protocols," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3584–3608, Sep. 2019.
[17] H. Haile, K.-J. Grinnemo, S. Ferlin, P. Hurtig, and A. Brunstrom, "End-to-end congestion control approaches for high throughput and low delay in 4G/5G cellular networks," *Comput. Netw.*, vol. 186, Feb. 2021, Art. no. 107692.
[18] L. Kleinrock, "Power and deterministic rules of thumb for probabilistic problems in computer communications," in *Proc. Conf. Rec., Int. Conf. Commun.*, Boston, MA, USA, Jun. 1979, pp. 43.1.1–43.1.10.
[19] A. Zanella, G. Procissi, M. Gerla, and M. Y. Sanadidi, "TCP westwood: Analytic model and performance evaluation," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, vol. 3, Dec. 2001, pp. 1703–1707.
[20] K. Winstein, A. Sivaraman, and H. Balakrishnan, "Stochastic forecasts achieve high throughput and low delay over cellular networks," in *Proc. 10th Symp. Netw. Syst. Design Implement. (NSDI)*, Lombard, IL, USA, Apr. 2013, pp. 459–471.
[21] F. Li, J. W. Chung, X. Jiang, and M. Claypool, "TCP CUBIC versus BBR on the highway," in *Proc. Int. Conf. Passive Act. Netw. Meas. (PAM)*. Cham, Switzerland: Springer, Mar. 2018, pp. 269–280.
[22] 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 Netw.) Workshops*, May 2018, pp. 1–9.
[23] P. Hurtig, H. Haile, K.-J. Grinnemo, A. Brunstrom, E. Atxutegi, F. Liberal, and Å. Arvidsson, "Impact of TCP BBR on CUBIC traffic: A mixed workload evaluation," in *Proc. 30th Int. Teletraffic Congr. (ITC)*, vol. 1, Sep. 2018, pp. 218–226.
[24] P. Farrow, "Performance analysis of heterogeneous TCP congestion control environments," in *Proc. Int. Conf. Perform. Eval. Modeling Wired Wireless Netw. (PEMWN)*, Nov. 2017, pp. 1–6.
[25] Z. Zhong, I. Hamchaoui, R. Khatoun, and A. Serhrouchni, "Performance evaluation of CQIC and TCP BBR in mobile network," in *Proc. 21st Conf. Innov. Clouds, Internet Netw. Workshops (ICIN)*, Feb. 2018, pp. 1–5.
[26] M. Hock, R. Bless, and M. Zitterbart, "Experimental evaluation of BBR congestion control," in *Proc. IEEE 25th Int. Conf. Netw. Protocols (ICNP)*, Oct. 2017, pp. 1–10.
[27] M. Zhang, M. Mezzavilla, J. Zhu, S. Rangan, and S. Panwar, "TCP dynamics over mmwave links," in *Proc. IEEE 18th Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, Jul. 2017, pp. 1–6.
[28] M. Pieska and A. Kassler, "TCP performance over 5G mmWave links—Tradeoff between capacity and latency," in *Proc. IEEE 13th Int. Conf. Wireless Mobile Comput., Netw. Commun. (WiMob)*, Oct. 2017, pp. 385–394.
[29] N. Cardwell, Y. Cheng, S. H. Yeganeh, I. Swett, V. Vasiliev, P. Jha, Y. Seung, M. Mathis, and V. Jacobson, "BBR v2: A model-based congestion control," in *Proc. IETF Meeting*, Mar. 2019, pp. 1–36.
[30] J. Chung, F. Li, and B. Kim, "BBRx: Extending BBR for customized TCP performance," in *Proc. NetDev 0x12*, Jul. 2018, pp. 262–276.
[31] J. Jaffe, "Flow control power is nondecentralizable," *IEEE Trans. Commun.*, vol. COM-29, no. 9, pp. 1301–1306, Sep. 1981.
[32] J. Tobin, "Estimation of relationships for limited dependent variables," *Econometrica, J. Econ. Soc.*, vol. 26, no. 1, pp. 24–36, Jan. 1958.
[33] R. E. Kalman, "A new approach to linear filtering and prediction problems," *J. Basic Eng.*, vol. 82, no. 1, pp. 35–45, Mar. 1960.
[34] C. Price, "An analysis of the divergence problem in the Kalman filter," *IEEE Trans. Autom. Control*, vol. AC-13, no. 6, pp. 699–702, Dec. 1968.
[35] R. Mehra, "On the identification of variances and adaptive Kalman filtering," *IEEE Trans. Autom. Control*, vol. AC-15, no. 2, pp. 175–184, Apr. 1970.
[36] B. J. Odelson, M. R. Rajamani, and J. B. Rawlings, "A new autocovariance least-squares method for estimating noise covariances," *Automatica*, vol. 42, no. 2, pp. 303–308, Feb. 2006.
[37] A. P. Sage and G. W. Husa, "Adaptive filtering with unknown prior statistics," in *Proc. Joint Autom. Control Conf.*, no. 7, Aug. 1969, pp. 760–769.
[38] A. Gupta, "Estimation of the mean and standard deviation of a normal population from a censored sample," *Biometrika*, vol. 39, nos. 3–4, pp. 260–273, Dec. 1952.
[39] D.-J. Jwo and T.-P. Weng, "An adaptive sensor fusion method with applications in integrated navigation," *J. Navigat.*, vol. 61, no. 4, pp. 705–721, Oct. 2008.

FEDERICO CHIARIOTTI (Member, IEEE) received the bachelor's and master's degrees in telecommunication engineering and the Ph.D. degree in information engineering from the University of Padova, in 2013, 2015, and 2019, respectively. He was a Research Intern with Nokia Bell Labs, Dublin, in 2017 and 2018. He is currently a Postdoctoral Researcher with the Department of Electronic Systems, Aalborg University, Denmark. He has authored over 40 published articles on wireless networks and the use of artificial intelligence techniques to improve their performance. His current research interests include network applications of machine learning, transport layer protocols, smart cities, bike sharing system optimization, age of information, and adaptive video streaming. He was a recipient of the Best Paper Award at several conferences, including the 2020 IEEE INFOCOM WCNEE Workshop.

ANDREA ZANELLA (Senior Member, IEEE) received the Laurea degree in computer engineering and the Ph.D. degree from the University of Padova, Italy, in 1998 and 2001, respectively. He spent nine months with Prof. Mario Gerla's Research Team, University of California, Los Angeles (UCLA), in 2000. He is one of the coordinators of the SIGnals and NETworking (SIGNET) Research Laboratory. He is currently a Full Professor with the Department of Information Engineering (DEI), University of Padova. His long-established research activities are in the fields of protocol design, optimization, and performance evaluation of wired and wireless networks. He has been serving as a Technical Area Editor for the IEEE Internet of Things Journal, and an Associate Editor for the IEEE Transactions on Cognitive Communications and Networking, IEEE Communications Surveys and Tutorials, and *Digital Communications and Networks*.

STEPAN KUCERA (Senior Member, IEEE) received the Ph.D. degree in informatics from the Graduate School of Informatics, Kyoto University, Kyoto, Japan, in 2008. He currently works as an Senior 5G Expert and 3GPP RAN2 delegate with Nokia Munich, Germany, where he is responsible for driving research and development of advanced 5G NR concepts as well as their 3GPP standardization, mainly in areas of the industrial IoT, V2X sidelink, and enhanced positioning. In the past decade, he developed at Nokia innovative networking technologies that became the basis for new commercial products and have been demonstrated to top-tier business customers worldwide and in industry shows. Impactful examples include multi-connectivity with user-defined latency/reliability guarantees, zero-touch machine-designed self-optimization for large-scale networks, light-based Gigabit mobile access, optimal interference management, and medium access for heterogeneous networks. He also served as a work package leader, primary beneficiary/investigator, board member in large-scale research projects totaling 15+ Mil. EUR, such as MINTS, ENLIGHTEM, and ELIOT, as well as founded and managed advanced research projects with six major universities in EU and Japan. He has filed more than 50 patents, published more than 70 book chapters, transactions, and conference papers in peer-reviewed ACM/IEEE venues. He actively serves on technical boards of major ACM/IEEE journals and conferences. He was a recipient of multiple professional awards, among others the Nokia Top Inventor 2018, the Nokia UK&I Top Inventor of All Times, the Hummies Gold Award for Best Human-Competitive Design 2019, and the Irish Laboratory Scientist of the Year Award 2018.

HOLGER CLAUSSEN (Senior Member, IEEE) received the Ph.D. degree in signal processing for digital communications from The University of Edinburgh, U.K., in 2004. He is currently the Head of the Wireless Communications Laboratory, Tyndall National Institute, where he is building up research teams in the area of RF, access, protocols, AI, and quantum systems to invent the future of wireless communication networks. He led the Wireless Communications Department, Nokia Bell Labs, Ireland and USA. In this role, he and his team innovated in all areas related to future evolution, deployment, and operation of wireless networks to enable exponential growth in mobile data traffic and reliable low latency communications. His research in this domain has been commercialized in Nokia's (formerly Alcatel-Lucent's) Small Cell product portfolio and continues to have significant impact. Prior to this, he directed research in the areas of self-managing networks to enable the first large scale femtocell deployments. He joined Bell Labs, in 2004, where he began his research in the areas of network optimization, cellular architectures, and improving energy efficiency of networks. He is author of the book *Small Cell Networks*, more than 130 journal and conference publications, 78 granted patent families, and 46 filed patent applications pending. He is a Fellow of the World Technology Network. He is a member of the IET. He received the 2014 World Technology Award in the individual category Communications Technologies for innovative work of the greatest likely long-term significance.