Congestion Control for Future Mobile Networks

Simone Mangiante
Vodafone Group R&D
simone.mangiante@vodafone.com

Michael Schapira
Hebrew University of Jerusalem
schapiram@cs.huji.ac.il

Amit Navon
Huawei, Network Technology Lab
amit.navon@huawei.com

Marco Dias Silva
Vodafone Group R&D
marco.silva1@vodafone.com

Brighten Godfrey
University of Illinois (UIUC)
pbg@illinois.edu

Weiguang Wang
Huawei, Network Technology Lab
weiguang.wang@huawei.com

Kevin Smith
Vodafone Group R&D
kevin.smith@vodafone.com

Itzcak Pechtalt
Huawei, Network Technology Lab
itzcak.pechtalt@huawei.com

ABSTRACT

The complexity and volatility of emerging mobile networks, which are intended to support extremely demanding applications such as high-resolution live video and AR/VR, pose immense challenges for congestion control. We present MORC, a novel rate-control protocol for mobile networks. MORC builds on the PCC protocol design framework to strike a balance between low latency and high throughput.

Lab trials and early field tests show that MORC outperforms traditional TCP congestion control and the recent BBR protocol, achieving faster file download times, higher resiliency to network changes, better bandwidth utilization, and improved quality of experience for video clients. We discuss deployment scenarios and future research.

KEYWORDS

congestion control, cellular network, transport protocol, video

ACM Reference Format:
Simone Mangiante, Michael Schapira, Amit Navon, Marco Dias Silva, Brighten Godfrey, Weiguang Wang, Kevin Smith, and Itzcak Pechtalt. 2018. Congestion Control for Future Mobile Networks. In *13th Workshop on Challenged Networks (CHANTS '18), October 29, 2018, New Delhi, India.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3264844.3264850

CHANTS '18, October 29, 2018, New Delhi, India

ACM ISBN 978-1-4503-5926-9/18/10…$15.00
https://doi.org/10.1145/3264844.3264850

1 INTRODUCTION

As internet access via mobile networks rapidly grows, and mobile apps and services quickly increase in popularity, optimizing user experience in this context is becoming ever more important. We argue that the challenging characteristics of mobile networks, as well as the demanding performance requirements of mobile applications (e.g., live video streaming, AR/VR) which 5G networks will likely boost, call for revisiting congestion control in mobile networks.

Challenge I: the complexity and volatility of mobile networks. The performance of data delivery over TCP is challenged by the unique characteristics of cellular communications. Specifically, the high variability in spectrum share assigned to a cellular user over time, as well as handovers due to device mobility, often lead to packet loss. Interpreting such losses as indicating network congestion results in poor choices of sending rates. In addition, TCP is often too slow to react to network changes, resulting in low network utilization and bad user experience.

Challenge II: meeting the demands of mobile applications. Delivery of Virtual Reality (VR) applications, and of 4K, 8K and 360-degree live and on-demand video content is of increasing importance. In the near future, the performance demands of such applications will be beyond the reach of today's TCP stacks [3].

Our research is targeted at developing a next-generation rate-control protocol for mobile applications, with an emphasis on future video and VR/AR. We present MORC (Mobile-Oriented Rate Control), which addresses the specific challenges of high throughput video delivery over mobile networks. MORC is designed within the recently proposed Performance-oriented Congestion Control (PCC) framework [6, 7], and can be viewed as a mobile-network-oriented embodiment of PCC's high-level architecture.

We evaluate MORC via experiments in a private LTE network emulation lab (a full network environment composed of a base station and packet-switched network) and field tests on a real-world LTE mobile network. Our preliminary evaluation results show that MORC outperforms both traditional TCP congestion control (TCP Cubic) and the recently proposed BBR [5] protocol. Specifically, (1) MORC achieves much faster file download times (often 1.5× or more) than Cubic even under static network conditions, (2) MORC exhibits much higher resiliency to changes in packet loss rate and jitter in round-trip time (RTT) than Cubic and BBR, which translates to consistently higher bandwidth utilization (13% and 25% more than Cubic and BBR, respectively), and (3) MORC consistently improves over Cubic in terms of end-user quality of experience, quantified in terms of video Mean Opinion Score (vMOS) [2], by over 16%. Such an improvement can give the mobile network operator a significant competitive advantage in video delivery by increasing customer satisfaction (and consequently retention).

2 CHALLENGES IN MOBILE NETWORKS

On mobile networks. Mobile networks differ from the wired Internet infrastructure that TCP was originally primarily designed for, with varying link rates, burst scheduling, deep packet buffers (which allow the possibility of bufferbloat), unpredictability due to user mobility, and more [11, 13].

LTE (Long Term Evolution) is a network of base stations, called evolved Node B (eNodeB), which serves as the access network to the IP-based Evolved Packet System (EPS) (EPS) [1]. Every 1 ms (the "Transmission Time Interval") an eNodeB needs to: consider the shared physical radio environment among connected User Equipments (UEs); prioritize UEs' QoS requirements; allocate radio resource blocks (the minimum allocation unit for radio resources) to each UE; and carry user data both downlink and uplink accordingly.

The Evolved Packet Core (EPC), the core network of the EPS, has a flat architecture consisting of few network nodes. Figure 1 depicts the user-plane components relevant to the remainder of this paper: (1) **the Serving Gateway (S-GW)**: the point of interconnect to the radio access, (2) **the PDN Gateway (P-GW)**: the point of interconnect to external IP networks, and (3) **the SGi Local Area Network (SGi-LAN):** a network infrastructure connecting the EPC to the Internet and providing services such as NAT and firewall.

Challenges. Based on our experience operating LTE networks, we have identified five main challenges facing congestion control in mobile networks:

(C1) *Signal strength variability.* The signal strength at a cellular device is affected by noise, interference, distance from the transmitting cell, weather conditions, and physical obstacles. These factors may change rapidly, and the received signal strength may consequently change by an order of magnitude in under a second [12]. Signal strength is constantly signalled by devices back to the cell tower, which uses this information to allocate spectrum to devices, assigning more radio resource blocks to devices with stronger signals. Only the radio scheduler knows the maximum throughput available to a device at any given instant.

(C2) *Handover loss.* While LTE's Packet Data Convergence Protocol handles the transfer of transmitted packets during device mobility between cells, packets may be lost at handover [8].

(C3) *Spectral efficiency.* Packets must be queued in under a microsecond to populate radio transport blocks when allocated. This cannot be accomplished by a remote server [8].

(C4) *Link-layer recovery.* LTE networks use two lower-layer mechanisms, ARQ and HARQ, to recover from packet loss and corruption on the radio link. Since the lower layers deliver packets in order to layer 3, this can cause significant delays. These mechanisms are utilized without the knowledge of the transport protocol.

(C5) *Application-layer goals.* The answer to whether a packet should be retransmitted if lost varies across applications. A web page may require all packets, whereas VoIP can recover via human correction. Without this knowledge, the transport protocol might needlessly transmit packets into an already overloaded network.

Unsuitability of traditional TCP. Today's TCP fails to meet the above requirements for the following reasons:

(1) *Conflation of loss and congestion.* Traditional TCP congestion avoidance wrongly assumes loss indicates congestion. However, per (C1) and (C2), any perceived

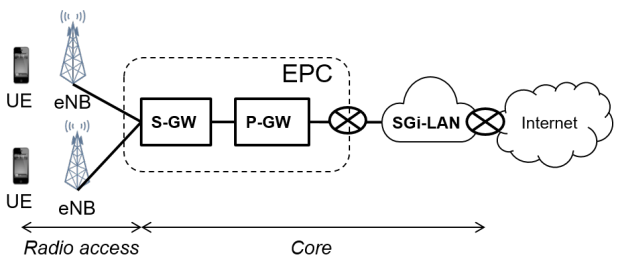


Figure 1: Architecture of an LTE network

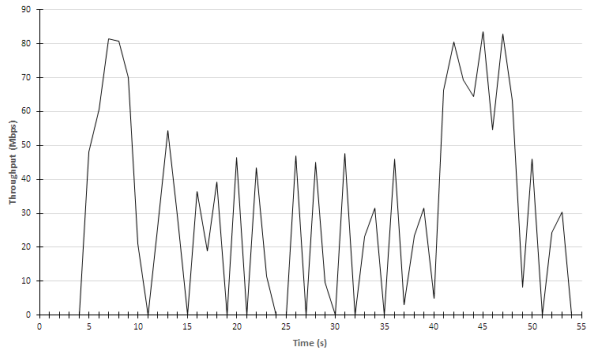


Figure 2: Example 4K video streaming traffic pattern

loss may be temporary, and per (C4), may be handled by lower layers.

(2) *Slow reaction.* The end-to-end nature of TCP congestion avoidance means that reaction to lower-layer dynamics can occur at the farthest point from the actual event. This is also true of fixed networks but is much more problematic on mobile. TCP's sawtooth pattern is inefficient at adapting to signal strength variability (C1), results in jitter, and prevents short flows from being transmitted quickly. The even faster reaction required to attain high spectral efficiency (C3) is infeasible for a remote sender [8].

(3) *Cross-layer obliviousness.* TCP is unaware of higher-layer goals (C5) and of lower-layer recovery mechanisms (C4). This can result in needless retransmissions, burdening an already busy network.

On video applications. Future video applications will further exacerbate these problems. TCP overlooks the unique characteristics of video streaming, compromising overall network performance and end-user quality of experience. Figure 2 illustrates the bursty traffic pattern when streaming 4K video using adaptive bitrate streaming (ABR) over TCP. Congestion control for video delivery over mobile networks should attain high throughput and low latency in the face of rapidly changing network conditions, and accommodate the bursts characterizing video traffic. To meet these challenges, a new congestion control framework is needed.

3 PCC MORC: OVERVIEW AND DESIGN

3.1 Introducing PCC MORC

MORC is designed within the recently introduced PCC framework [6, 7]. Under PCC, time is divided into successive *monitor intervals* (MIs). In each MI, the PCC-sender "tests" a specific sending rate and, after receiving feedback from the receiver in the form of selective acknowledgements (SACKs), aggregates performance-related statistics (e.g., achieved goodput, packet loss rate, average latency) into a numerical *utility value* reflecting the "performance benefit" resulting from sending at that rate. An *online learning* algorithm is employed to adapt sending rates in a manner (direction, extent) that is *empirically* associated with higher utility.

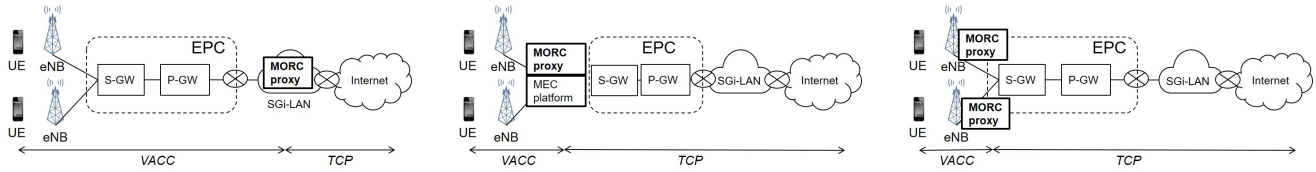
We chose the PCC framework for several reasons: (1) given the high volatility and complexity of mobile networks, generating (even approximately) accurate network models (whether offline [10] or online [5]) seems infeasible, motivating PCC's "black box" approach [6]. (2) PCC's online learning approach significantly improves over TCP in terms of contending with non-congestion loss and reaction speed [6, 7]. (3) PCC's utility framework provides a powerful means for reasoning about performance.

Any specific realization of the PCC architecture reflects two fundamental design choices: (1) the utility function (which statistics are collected? how are these aggregated?), and (2) the online learning algorithm to apply for rate selection (how to choose the next rate to send at?). As established in [6, 7], proper choices of utility functions and online learning algorithm yield both provable *local* performance benefits for the sender and *global* benefits such as convergence to a stable and fair rate-configuration for multiple competing senders.

MORC is an adaptation of PCC Allegro [6] to mobile networks. We next discuss high-level points regarding how MORC's utility function and learning algorithm contend with the challenges posed by mobile networks and applications.

Take into account burstiness of video traffic when measuring utility. Video traffic can exhibit bursty transmission periods followed by long silences. To evaluate the utility from sending at rate r in a MI, MORC awaits a time period in which traffic is continuously transmitted for "long enough" to yield meaningful statistics (1 RTT in our implementation) and bases its utility-value derivation only on that time period (after which the MI is concluded).

Incorporate latency into the utility function to keep in-network buffers non-empty. Implicit in PCC Allegro [6] and PCC Vivace [7] is the desire to approximate the lowest achievable RTT ("RTT_{min}"). Importantly, however, in the context of mobile networking, operating at RTT_{min} is not necessarily optimal as this might imply that packets are not buffered in in-network queues. Recall that mobile networks can exhibit drastic changes in available bandwidth even within a single RTT, and so too quickly for the sender to react to in a timely manner. To contend with this, MORC strives to keep in-network buffers non-empty, so as to quickly utilize spare bandwidth, yet not "too full", so as to not induce



(a) At the SGi-LAN (b) At a MEC location (c) At the eNodeB

Figure 3: MORC deployment options in mobile networks

undesirable delays. Specifically, MORC tracks RTT_{min}, sets a slightly higher *target RTT*, and penalizes deviations of the experienced RTT from this target RTT (in both directions).

Learn more quickly by avoiding unnecessary "rate experiments". The variability of available bandwidth in mobile networks requires online learning to be highly responsive to network changes. This, in turn, requires not lingering "too long" on rate exploration. To better react to network changes, MORC avoids probing rates that are unlikely to result in higher utility. We will elaborate on this further below.

We next discuss MORC's utility function and online learning algorithm in more detail.

MORC's utility function. MORC's utility function incorporates throughput, loss rate, and latency, as follows:

$$u = T(1 - p_L - p_{RTT})$$

where T is the measured received throughput in Mbps, and p_L and p_{RTT} are penalties for packet loss rate and latency, respectively.

Our experimental results indicate that good choices of p_L and p_{RTT} are as follows. Let L be the measured loss rate, i.e., $L \in [0, 1]$ is the fraction of packets sent in the MI considered lost (for which SACKs have not been received). We set $p_L = 10L$.[1] Let $RTT_{target} = RTT_{min} + 60$, where both RTT_{min} and RTT_{target} are measured in milliseconds. We set p_{RTT} to be $p_{RTT} = 5 \times 10^{-4} \times |RTT - RTT_{target}|$. This applies a utility penalty equal to 0.05% of the measured throughput for each 1ms deviation (in any direction) from the target RTT.

MORC's online learning algorithm. MORC builds on PCC Allegro's rate adaptation scheme [6]. MIs in MORC are 1 RTT long. When the connection is started, MORC enters a slow-start phase, doubling its rate every MI. MORC remains in slow-start so long as the utility value keeps increasing as the rate is increased (even if there is some loss, unlike TCP). This enables MORC to quickly utilize spare bandwidth upon initialization.

[1]Observe that this implies that the utility value can also be negative for high loss rates.

After exiting slow-start, the sender enters a "decision mode", probing rates higher and lower than the current rate to determine the utility-maximizing direction (upwards or downwards) and adjusts its rate accordingly. The sender keeps adjusting the rate in the chosen direction so long as the resulting utility values keep increasing, otherwise entering the decision mode again, and so on.

As discussed above, MORC avoids probing rates that are not likely to result in higher utility. Consider, e.g., the scenario that the sending rate is r and the experienced RTT is RTT_{min}. In this situation, probing a rate lower than r is unlikely to result in higher utility, as an RTT of RTT_{min} suggests that in-network buffers are empty and so decreasing the rate will not result in better latency or loss rate, but merely lower the achieved throughput. Consider also the scenario that the received rate is significantly lower than the sending rate (say, 30Mbps vs. 100Mbps), or the experienced RTT is significantly higher than RTT_{target}. This might indicate that in-network queues are quite full and thus sending at higher rates is unlikely to improve utility.

3.2 Where/How to Deploy MORC?

MORC can be deployed on the sender side only, either directly on content/web servers or as a proxy (and without requiring any changes to servers' TCP stacks).

We see three deployment scenarios for MORC in mobile networks, shown in Figure 3, which reflect different tradeoffs between performance benefits and deployment cost.

At the SGi-LAN: deploy the MORC proxy in the core network, just before the traffic from a remote server enters the mobile network. The key advantage is that a single MORC proxy can serve the entire mobile network (thousands of base stations). However, this comes at the cost of not optimizing local radio conditions, as the MORC segment unnecessarily spans both the access and the core network, yet the latter does not pose the challenges discussed in Section 2.

At a Multi-access Edge Computing (MEC) location: the MORC proxy is co-located with a MEC node [9] or implemented as a service provided by the MEC platform. Now,

the MORC segment does not cover the core network, being at an MEC node located at an aggregation point for the radio access network, still serves as many as hundreds of base stations. This represents a reasonable tradeoff between complexity/cost and effectiveness: MORC can contend with the radio access network conditions with but a few nodes deployed in the entire network.

At the eNodeB: the MORC proxy is embedded into a base station. Now, MORC can improve the radio access network the most as each base station employs a dedicated MORC instance. However, this is the most expensive option, deployment-wise, because thousands of base stations need to be upgraded. Also, proxying traffic at the eNodeB also requires handling handover scenarios where a UE moves from one eNodeB's coverage zone to another's. One possibility is adding a new ACK to the TCP options indicating that data from the proxy was received by the client, and not deleting data at the server that is not acknowledged thus by the proxy. This, however, requires server-side changes.

4 PRELIMINARY EVALUATION

We evaluate a kernel module implementation of MORC in both lab-simulated and real LTE networks, for file transfer workloads and video traffic, respectively.

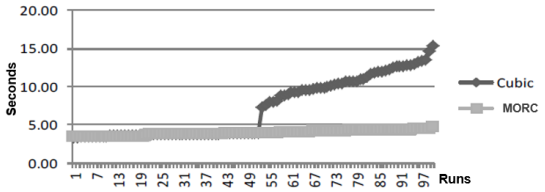
4.1 Lab Tests

In our private LTE lab, a full network environment including base station and EPC, we compared MORC with the most popular congestion control protocol, TCP Cubic, and the recent BBR, under two different file download scenarios: (1) over the Internet and (2) from a local server in the operator's network.

We emulated static and rapidly changing network conditions using the Linux tool `netem` [2] to test different congestion control protocols under the exact same network conditions. Focusing on bandwidth, RTT and packet loss rate (PLR), our experimental framework is able to simulate a number of static and dynamic *test profiles*, reflecting both typical and extreme conditions we observed in a live LTE network. We used, for each test scenario, a single client: a laptop equipped with a USB LTE dongle with the Chrome browser enforcing TCP sessions (QUIC disabled).

File download over the Internet. In this test scenario, the client downloads a 50 MB file from a server hosted in Amazon Web Services supporting Cubic, and the total time needed to complete the transfer is measured. We used static *test profiles* with varied values for bandwidth (in Mbps), RTT (in ms), and PLR (in percentage): 20Mbps/60Mbps/120Mbps, 12ms/30ms, and 0.036%/0.05%/0.5%, respectively. We deployed MORC at the SGi-LAN as depicted in Figure 3a and made 100 runs, comparing MORC to Cubic under all test profiles (i.e., all possible combinations of values for bandwidth, RTT, and PLR).

[2]https://wiki.linuxfoundation.org/networking/netem



Algorithm	Min	Median	Max	Std. dev.	**Avg.**
Cubic	3.43	3.92	15.38	3.79	**7.15**
MORC	3.45	3.92	4.76	0.33	**3.97**

Figure 4: Results from file downloads over the internet: 100 runs plotted in ascending order (top) and statistical summary (bottom)

Figure 4 plots download completion times for runs sorted in ascending order for the following static *test profile*: 120 Mbps bandwidth; 12 ms RTT; 0.036% PLR. Observe that even in this test profile, with high available bandwidth, low latency and low packet loss, Cubic achieves higher average download time and standard deviation. MORC outperforms Cubic by a factor of 1.8 in terms of average download time, exhibiting more consistent and predictable behaviour. The average improvement factor reaches 2.41 when PLR is 0.05% and RTT is 30ms for the same bandwidth.

File download from local server under variable network conditions. We now focus on bandwidth utilization. We set up a web server at the SGi-LAN that supports MORC, Cubic, and BBR. We compared the three algorithms under dynamic test profiles, where the bandwidth and RTT are changed every second and 100 ms, respectively. PLR is fixed throughout each run, and network buffers are dimensioned to queue packets for 300 ms.

See Table 1 for results for a meaningful subset of the test profiles. MORC and Cubic show similar performance in ideal conditions (short bandwidth range, very low PLR and low jitter), but Cubic's bandwidth utilization drops drastically when PLR increases or the bandwidth range is larger. BBR's bandwidth utilization drops as jitter increases; more generally, BBR's evaluated implementation is unable to react to the fast-changing conditions of cellular networks. MORC performs best across the entire set of experiments, achieving average bandwidth utilization of 89% compared to Cubic's 79%.

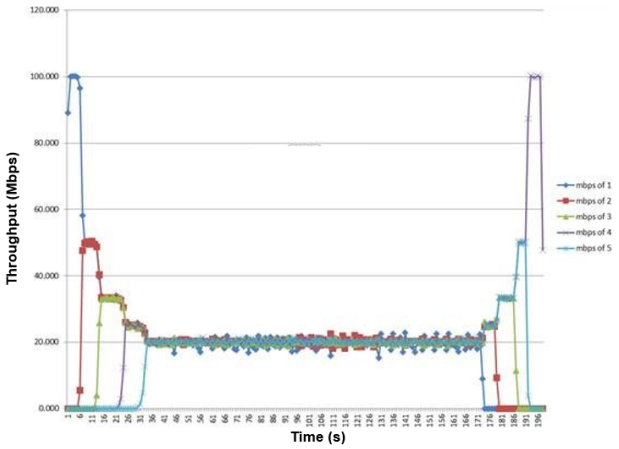


Figure 5: Convergence of 5 concurrent MORC flows

Convergence of MORC flows. We tested how multiple MORC flows converge by running 5 concurrent file downloads using a test profile with 100 Mbps bandwidth, 20 ms RTT, and no packet loss. Flows join sequentially in 5-second intervals and then leave sequentially. Figure 5 shows that MORC flows converge to the fair throughput allocation.

4.2 Field Tests

We also evaluated MORC's behaviour for adaptive video streaming by performing field tests in a real LTE mobile network. We used video Mean Opinion Score (vMOS) to assess

Table 1: Bandwidth utilization of MORC, Cubic and BBR for a selected subset of file download test profiles

Bandwidth range (Mbps)	RTT range (ms)	PLR (%)	Bandwidth Utilization (%)		
			Cubic	MORC	BBR
[20-120]	[12-28]	0.01	64	90	70
	[12-60]	0.01	45	84	40
	[12-28]	0.05	36	84	64
	[12-60]	0.05	25	76	34
[10-60]	[12-28]	0.01	93	93	84
	[12-60]	0.01	84	89	55
	[12-28]	0.05	73	92	82
	[12-60]	0.05	52	89	54
[3-20]	[12-38]	0.01	97	97	89
	[12-90]	0.01	95	77	52
	[12-38]	0.05	96	93	91
	[12-90]	0.05	80	80	51
	[12-38]	0.1	90	97	90
	[12-90]	0.1	70	85	53
Total average across all *test profiles*			**79**	**89**	**71**

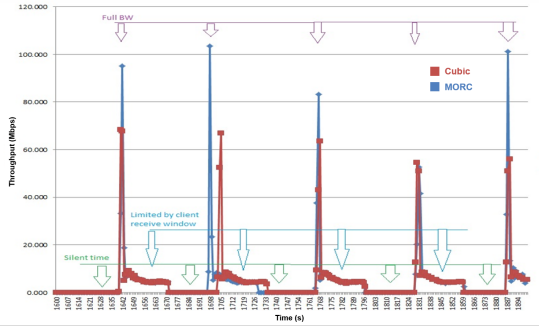


Figure 6: Streaming server throughput in field tests

quality of experience perceived by end users, using Speed-Video Pro[3] for the automatic collection of user experience metrics.

Clients in our experiments were 16 Huawei P9 smartphones. Two streaming servers, one using Cubic and one using MORC, were located in a datacenter in the same city as the clients. The clients were randomly partitioned into two groups: 8 accessed the MORC-enabled server and 8 accessed the CUBIC-enabled server. We performed several runs across different times of day and radio interface conditions.

MORC consistently achieved vMOS above or close to 4 even in congested cells. (vMOS scores are within the range 1-5, with a score of 4 translating to "good" QoE and a score of 3 representing "fair" QoE [2].) Cubic and MORC's average vMOS were 3.37 and 3.74, respectively (an improvement of 16% after subtracting the minimum vMOS score of 1.0). MORC also kept vMOS more stable across clients and runs, with 25% lower variance than Cubic's. MORC allowed the streaming server to send video chunks at a higher rate and sometimes at higher resolutions.

Figure 6 shows the higher throughput achieved by the server with MORC enabled. We observed similar results for YouTube video playback in our lab with MORC deployment at SGi-LAN. MORC generated shorter time to start and fewer buffering events compared to both Cubic and BBR. Packet captures revealed that MORC achieved a higher initial throughput, rendering transmission less bursty thereafter.

5 RELATED WORK

Recent years have witnessed a surge of renewed interest in congestion control. MORC was designed within the recently introduced PCC framework [6, 7], and was contrasted with BBR [5]. Remy [10] is an *offline* optimization approach for rate control. We did not compare with Remy or other PCC variants in part since kernel modules were not available. Copa [4] is another recent entrant that would also offer a valuable comparison.

[3]https://play.google.com/store/apps/details?id=mlab.android.speedvideo.pro

Table 2: Comparison of MORC, Cubic and BBR

Algorithm	Resilience to			ABR video
	packet loss	RTT jitter	varying BW	QoE
MORC	✓	✓	✓	✓
Cubic	✘	✓	✘	✘
BBR	✓	✘	✘	✘

Recent work that targets cellular networks is of special interest. Sprout [11] uses an inference procedure in a probabilistic model of a wireless channel to make short-term forecasts of channel rates. Verus [13] continuously builds an end-to-end delay profile on which it tunes the sending rate. Both protocols [11, 13] are incompatible with TCP, requiring both sender and receiver changes. MORC can be implemented at the sender side only and its customizable utility function provides more flexibility and is more broadly applicable.

6 CONCLUSION AND FUTURE DIRECTIONS

We presented MORC, a member of the PCC family of protocols designed for emerging mobile networks and applications. As summarized in Table 2, our lab trials and early field tests suggest that MORC outperforms both TCP and BBR. We discussed different scenarios for deploying MORC. While we view MORC as an important first step, MORC is still far from realizing the full potential of next-generation mobile networks. We next discuss some of our ongoing and planned efforts in this direction.

Cross-layer optimization. Congestion control in mobile networks is oblivious to both link-layer mechanisms (for recovering from packet loss, in-order delivery) and application-layer information (e.g., regarding the importance of retransmitting a packet). While this is true for rate-control in general, the unique characteristics of mobile networks (see Section 2), and the advent of 5G with its faster and more complex air interface, suggest that cross-layer optimization is especially promising in this context.

Better utility functions and online learning. Incorporating ideas from PCC Vivace [7] to identify utility functions and online learning algorithms that are better suited than MORC's to mobile networking is a promising direction.

Extend our evaluation. We intend to evaluate MORC at larger scale and with 5G radio technologies, and explore other aspects, including convergence behaviour, fairness, friendliness to legacy TCP, other traffic workloads, quality of experience (QoE) for multiple competing video streams, video upload to the cellular network, other QoE metrics (e.g., buffering ratio), and more.

REFERENCES

[1] *3GPP Evolved Universal Terrestrial Radio Access (EUTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN) - Overall Description, Stage 2 (2016).* Technical Report.

[2] *Mobile Video Service Performance Study.* Technical Report. http://www.ctiforum.com/uploadfile/2015/0701/20150701091255294.pdf

[3] *Next Generation Protocols - Market Drivers and Key Scenarios.* Technical Report. http://www.etsi.org/images/files/ETSIWhitePapers/etsi_wp17_Next_Generation_Protocols_v01.pdf

[4] Venkat Arun and Hari Balakrishnan. Copa: Practical Delay-Based Congestion Control for the Internet. In *NSDI 2018.*

[5] N. Cardwell, Y. Cheng, C.S. Gunn, S.H. Yeganeh, and Van Jacobson. 2016. BBR: Congestion-Based Congestion Control. *Queue* 14, 5 (2016).

[6] M. Dong, Qingxi Li, Doron Zarchy, Philip Brighten Godfrey, and Michael Schapira. PCC: Re-architecting Congestion Control for Consistent High Performance. In *NSDI 2015.*

[7] M. Dong, T. Meng, D. Zarchy, E. Arslan, Y. Gilad, B. Godfrey, and M. Schapira. PCC Vivace: Online-Learning Congestion Control. In *NSDI 2018.*

[8] I. Johansson. 2015. Congestion control for 4G and 5G access. Internet-Draft draft-johansson-cc-for-4g-5g-02. (2015).

[9] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief. Mobile edge computing: Survey and research outlook. In *arXiv preprint arXiv:1701.01090 (2017).*

[10] K. Winstein and H. Balakrishnan. TCP ex Machina: Computer-Generated Congestion Control. In *SIGCOMM 2013.*

[11] K. Winstein, A. Sivaraman, and H. Balakrishnan. Stochastic Forecasts Achieve High Throughput and Low Delay over Cellular Networks. In *NSDI 2013.*

[12] C. Perkins Z. Sarker, V. Singh. An Evaluation of RTP Circuit Breaker Performance on LTE Networks. In *Infocom CNCTV 2014.*

[13] Y. Zaki, T. Pötsch, J. Chen, L. Subramanian, and C. Görg. Adaptive Congestion Control for Unpredictable Cellular Networks. In *SIGCOMM 2015.*