Driving TCP Congestion Control Algorithms on Highway

Feng Li, Jae Won Chung, and Xiaoxiao Jiang
*Verizon Labs
60 Sylvan Road
Waltham, MA, 02145
{feng.li, jae.won.chung, xiaoxiao.jiang}@verizon.com

***Abstract*—Different from IEEE802.11 (wifi) and traditional 3G wireless networks, LTE networks have the features of high link variability and end user mobility. There exists a huge gap in understanding how TCP congestion control algorithms (CCAs) perform under such condition. Mobile carriers waste their efforts to improve the throughput by fine tuning parameters of CUBIC, although CUBIC often fails to ramp up rapidly to satisfy the congestion avoidance needs over LTE links. To have better understanding of TCP performance over LTE networks, we conduct a comprehensive measurement study to compare CUBIC with its latest rival - BBR over a world leading tier-one mobile network in high speed driving condition. To the best of our knowledge, no measurement effort has been done to compare the performances of different TCP flavors over LTE networks on highway. Our in-depth measurement results conclude that 1) CUBIC with Hybrid slow start leads to a low radio resource utilization; 2) BBR yields higher throughputs even when SINR is lower and/or hand-over happens; 3) BBR's bottleneck link bandwidth estimation works well for various conditions (including hand-overs); and 4) BBR is a promising TCP Congestion Control candidate for performance enhancement proxies (PEP) over mobile networks.**

I. Introduction

As several new TCP congestion control algorithms (CCAs) have been added into Linux kernel source tree recently, none of them has been designed or implemented to face the challenges from mobile networks in terms of delay and throughput:

- TCP New Vegas (TCP NV) [4], Data Center TCP (DCTCP) [2] and TCP BBR [5], are all designed originally for data centers or mainframes in wired environment.
- TCP Sprout [22] and TCP Verus [24] are *experimental* CCAs designed for mobile environment, but they are only implemented through UDP tunnels and not ready to be deployed in production.
- The "older" TCP variants (CUBIC [8] and Westwood+ [7]) were observed to have performance degradation over LTE networks [3].

Even though CUBIC was not designed over mobile network and it leads to low link utilization over wireless links [24], mobile carriers still have to spend a large amount of time and efforts to "tune" the parameters of CUBIC on their asymmetric performance enhancement proxies (PEPs) [19] to achieve better performance.

Meanwhile, mobility is one of the most importance features in mobile LTE network, but little is known about the radio characteristics of mobile networks under high speed driving conditions. In fact, neither academic nor industrial has spent many cycles on studying the RF condition in wild, without even mentioning the high speed driving conditions. Such lack of knowledge of RF environment makes the modeling and simulating in LTE networks difficult, which increases the development cost of TCP optimization schemas and new CCAs for mobile networks. Moreover, the high frequency micro cell in 5G would only provide access in urban area, and LTE, especially 700MHz to 800Mhz band, will still provide coverage for rural area and highways. It is important to consider highway driving in new transport protocol design for future mobile networks.

These above motivate us to conduct a measurement study to investigate the performances of different TCP CCAs on highway. The main contributions of the paper include:

- collecting a "realistic" radio network traces from a tier-1 provider's network under high speed driving condition.
- measuring and presenting the PHY and MAC layer results under highway driving condition.
- comparing the performances of TCP CCAs over highway driving conditions. Impacts of Singal-to-Interference plus Noise Ratio (SINR) changes and hand-over between different eNodeBs are also measured.
- discussing the CCA design requirements over mobile network.

The results of this study can be used as "food for thoughts" for designing new CCAs over future mobile networks (5G), while the physical layer (PHY) and media access control layer (MAC) measurement results provide rich inputs for developing simulations over mobile networks. The rest of paper is organized as follows: Section II summarizes the related research work in this area; Section III describes our experiment design and tools used in this study; Section IV presents the PHY and MAC layer measurement results under highway driving condition; Section V compares the performances of three CCAs on highway driving condition; Section VI discusses the possible requirements for designing future CCAs over mobile networks; and Section VII concludes this study.

II. Related Work

Improving the performance of TCP over mobile cellular networks has attracted many researchers in recent years. The study in [3] and [18] conducted by Ericsson is the one of the most related work to our study. They investigated the existing availability of five different CCAs: CUBIC, New Reno, Westwood+, Illinois, and CAIA Delay Dradient (CDG) [9], under

mobility condition with NS-3[1] simulation. Although they choose DCE [20] to have a realistic simulation setup, it is still difficult to judge how close their simulation model matches with the realistic highway driving conditions. Moreover, their simulation based study only focused on the performance of start up phase, moving towards eNodeBs, or away from eNodeBs. They did not investigate the performances of CCAs over realistic long distance highway driving conditions.

There are multiple measurements in moving cars or high speed trains on 3.5/3.75G mobile networks. Li *et al.* [15] measured TCP performances on HSPA+ (3.75G) networks on China's high speed trains. Tso *et al.* [21] conducted extensive measurements on HSDPA(3.5G) network with trains, subways, cars, bus and ferries in Hong Kong. Jang *et al.* analyzed downlink throughput with CDMA-EVDO networks in 300 km/h trains. Yao *et al.* measured bandwidth in mobile vehicles on HSDPA networks. However, 3.5/3.75G networks have different characteristics with LTE networks, for instances, the RTT observed in LTE network is much smaller than 3.5G networks. Thus, the results of above researches may not be valid in current mobile networks. Moreover, carriers often provide Wifi hot-spots on public transportations like trains, subways, and airliners (e.g. United Airlines provides "inFlight Wifi" through satellites). Thus, the value of measuring the LTE performance on high speed public vehicles is kind of trivial.

Huang *et al.* studied the performance of TCP over LTE through packet traces collected from a carrier's core network [11]. Although their results confirm the expected shorter latency over LTE links compared to a 3G network, no PHY or MAC layer information has been provided. Xiao *et al.* measured TCP throughput and RTTs over the stationary, highway driving and high speed railway scenarios [23] in LTE network. Their results show TCP throughput degradation in high-speed moving condition. However, their measured throughput is much less than the theoretic throughput over LTE networks. Mertz *et al.* conducted a measurement study focusing on the performance of the LTE PHY layer in high-velocity conditions [16], but their measurements did not include Transport Layer or Application Layer throughput.

All above measurements in driving or high-speed moving scenarios mainly show the statistics of throughputs, RTT, packet loss rate and etc, without quantitatively analyzing the PHY and MAC metrics on TCP throughputs. Besides, these studies only evaluated one TCP flavor (CUBIC) – it is well known that TCP CUBIC experiences degraded performance over wireless links [5]. In summary, the main difference between our work and others lies not only in the highway driving scenarios, but also in our contribution towards understanding the RF condition as well as a performance evaluation of CUBIC and its latest rival — BBR.

III. Methodology

A. Congestion Control Algorithms

As Figure 1 shows, we focus on three TCP CCAs: BBR, CUBIC(3.19), CUBIC(4.8) in this study. On each of the server,

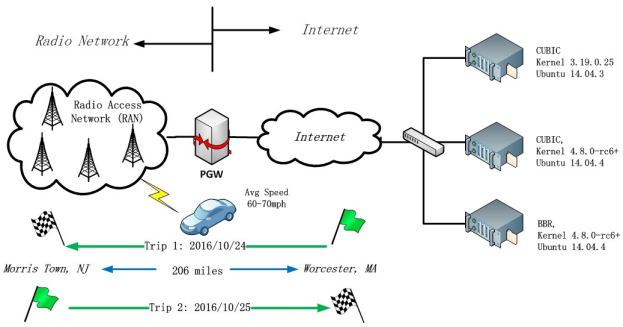


Fig. 1: Measurement Setup

one of the TCP CCAs is set as the default TCP congestion control algorithm.

BBR [5] is a novel congestion control algorithm developed by Google, which calculates the congestion window (CWND) size by measuring the bottleneck bandwidth and round trip propagation time.

CUBIC(k3.19) and CUBIC(k4.8) BBR was released with Linux *net-next* kernel[2] as a patch for 4.8-rc6 kernel, while CUBIC used in production environment is based on 3 series kernels. Therefore, this study uses three servers: BBR and CUBIC(4.8) running with 4.8-rc6 kernel, and CUBIC(3.19) running with 3.19.0-25-generic kernel. From implementation perspective, CUBIC with 4.8 kernel is slight different from the one in 3.19 kernel. Thus, this study treats them as two different CCAs.

The rest of Linux based CCAs perform poorly over LTE networks. Similar to the conclusion in [3], we also observe performance degradation of Westwood+ over LTE networks in our previous stationary tests. Therefore, we only evaluate CUBIC and BBR in this study.

B. Experiment Setup

We perform measurements on a tier-1 cellular network in South New England in United States on two consecutive days, Oct 24-25, 2016 (Figure 1 shows). Before we start our driving test, we setup three HP Proliant 460c Gen9 blade servers with 128GB RAM and a dual socket 2.60GHz ten-core Inter (R) Xeon (R) ES-2660v3 CPUs from the same chassis. All the three servers are connected to the Internet through the same HPE 6120XG 10Gbps switch.

The three servers are configured with same parameters excepts the kernel version and the default TCP congestion algorithm. All kernel parameters are default values, but we did change two parameters of Ethernet cards to improve the throughputs: i) increase the transmission queue size (*txqueuelen*) of Ethernet interface to 10000 packets; and ii) reduce MTU to 1428 bytes in consideration of GTP header length to avoid unnecessary fragmentation on radio access networks (RAN). However, Cardwell *et al* suggest to enable fair queuing through Linux Traffic Control (tc) utilities [5]. Thus, we use the following command to enable fair queuing and pacing on the only BBR server under Cardwell's suggestions:

```
#tc qdisc replace dev eth0 root fq pacing
```

[1]https://www.nsnam.org

[2]git://git.kernel.org/ pub/scm/linux/kernel/git/davem/net-next.git

All the three hosts are running Apache 2.4.7 Web server with PHP 5.5 support. We write a PHP script to dynamically generate 20MB files with random content for smart phones to download. In this way, we can avoid any possible caching inside network devices along the data path. For validation purpose, *tcpdump* runs on the three servers as background jobs, and it captures all TCP packets up-to 300 bytes, which is long enough to have complete TCP headers. The PHP script and *tcpdump* are light weighted, and our preliminary stationary throughput test shows that only ignorable CPU usage (less than 1%) is observed on all the three servers. Note, these three servers are dedicated to our performance study, and only reachable from phones in a small test device pool.

Tools on Smart Phone In this study, we choose the commercial measurement tool named SwissQual Qualipoc[3] to evaluate the effects of SINR on RTT and throughputs under highway driving condition. Qualipoc tools run from a LG G2 VS980 smart phone with 2GB RAM and a 32-bit Qualcomm Snapdragon(R) S4 Prime Qual Core CPU, with Android 4.3.2 OS.

Qualipoc provides a set of measurement tools, but we only use the following tools in our study:

- **Ping tool**, just as regular ping, which sending 56 byte long ICMP packets to measure the baseline RTTs.
- **TCP throughput measurement tool**: a command line version of Web browser similar to *wget* downloads files from Web Servers and report average throughput (goodput) at the end of downloading.
- **LTE measurement tool**: PHY and MAC layers statistics information such as Reception Signal Strength Indicator (RSSI) and Number of Transport Block (TB) are collected by the background program from LTE device drivers every second. The measurement results are recorded into log files.

Since we have *tcpdump* running on all the three servers, it is unnecessary to do packet capture on low end smart phones. Meanwhile, in our preliminary tests, the phone became extremely hot after 1-hour testing with *tcpdump* enabled.

The carrier we cooperated provides LTE services over two radio spectrum: Band XIII and Advanced Wireless Service (AWS). AWS normally provides more link capacities in city area while Band XIII provides a large coverage over rural area. None of US carriers provide continuous AWS coverage along highways, thus, we locked our testing phone to use only Band XIII in this study to reduce unnecessary service interruption. Table I summarizes the carrier's radio frequency band and spectrum involved with study. Note, the LG G2 VS980 phone does not support "carrier aggregation", and locking the test phone on Band XIII would not cause any possible performance degradation.

Our measurement test suite contains 40 test iterations. In each test iteration we sequentially ping and download a dynamically generated 20MB files from each of the three servers. Between each of the ping and file download test session, we put 5-10 second grace time. One test suite would take around 1 hour, and it gives us an opportunity to take a gas station break on the trip.

[3]http://www.mobile-network-testing.com/en/products/optimization2/qualipoc-android/

TABLE I: Radio Spectrum Involved in this Study

Metric	Value
LTE Band Number	Band XIII (13)
UPLink (MHz)	777-787
Downlink (MHz)	746-756
Width of Channel(MHz)	10
Modulation	QPSK, 16QAM, 64QAM

C. Driving Scenario

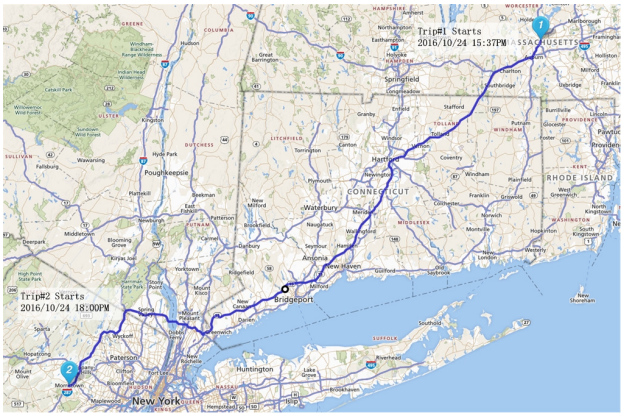


Fig. 2: Driving Route

As Figure 2 shows, we perform our highway driving measurements on a tier-1 carrier's LTE network between Worcester, MA to Morris Town, NJ on two consecutive days: we depart from Worcester MA on 15:37PM Oct 24, 2016 to Morris Town NJ, and returned from Morris Town to Worcester on 18:30PM 2nd day. The total driving distance is 412 miles (600 km) in 8 hours, including rush hours, gas station stops, three accidents or construction detours. On each trip, we repeat three times of our test suite **only** during the highway driving, and we only take gas station breaks in the idle time between tests. The average driving speed is 65-70 mph, around 30 meters per seconds. Note, after locking our device on Band XIII, the device stops reporting GPS location and velocity. Therefore, we could not associate the throughput with car's speed in this study. However, we only run the tests in highway driving condition, and we can eliminate variance in car's speed and assume we travel at our average speed.

Different from high speed railway measurements or any measurements in well-controlled environment, highway driving measurement naturally contains more uncontrolled variables. Thus, we only control the variables under our control (e.g. we chose the same route for our return trip, and fully charged the phone before trips, etc). We leave the uncontrollable variables, such as weather, construction detours, and dead deer on the road etc, alone. Thus, we conduct a "daily life" measurement which provides more "realistic customer experiences" than any other studies.

IV. Radio Network Characteristics

This section analyzes the radio network characteristics such as Signal-to-Interference plus noise ratio (SINR), transmission blocks (TBs) modulation fraction, Transmission Block Error Rate (BLER) and Frame Error Rate (FER), etc. All these characteristics indicate the performance of LTE networks, and also can be used as simulation parameters in future studies. Due to the space limitations, we only present SINR, Modulation, BLER, and FER in this paper.

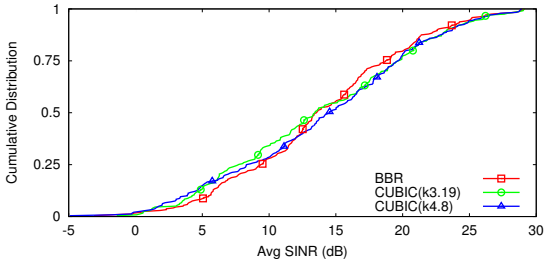


Fig. 3: Distribution of SINRs

Figure 3 describes the distribution of average SINR for different TCP flavors. All the TCP sessions experience similar RF conditions because the average SINRs are in same distribution. Thus, it is unnecessary to distinguish TCP flavors in PHY and MAC layer analysis.

A. SINR vs. Modulation

The modulation (or encoding scheme) selection in LTE networks depends on the SINR measured by both UE and eNodeBs. LTE devices are able to choose several modulation techniques to modulate data and control information. These modulation techniques include: QPSK (2 bits per symbol), 16QAM (4 bits per symbol), and 64QAM (6 bits per symbol). Clearly, the modulation or enoding scheme significantly impacts the throughput over LTE networks. All of these three modulation are supported in the downlink direction (PDSCH), while 64QAM is optional in uplink. Note, this study mainly focuses on downlink direction (PDSCH) because we only have small amount traffic (ping packets and TCP ACK) in uplink direction.

Figure 4 shows the histogram of modulations used under different SINR bins. The x-axis is the recorded SINR (in dB), and the y-axis describes the faction of modulation of Transmission Blocks (TB) over the total number of transmission blocks. When SINR is greater than 20dB, more than 90% of TBs are transmitted in 64QAM modulation; when RF condition becomes worse (SINR less than 5dB), most of TBs are in QPSK modulation. When SINR is between 5dB and 15dB, the situation is complicated: eNodeBs conduct frequent rate adaptation and try to use different modulations.

B. SINR vs. Error Rate

Two error rate metrics are reported from the device drivers on the LG smart phone:

- A Block Error Ratio (BLER) is defined as the ratio of the number of erroneous blocks received to the total number of transmission blocks received. An erroneous block is defined as a Transport Block failed to pass the cyclic redundancy check (CRC).
- Frame Error Rate (FER) is defined as the ratio of the number of erroneous frames over the total number of frames received. A frame is considered to be in error if at least one of the bits in it is detected as incorrect.

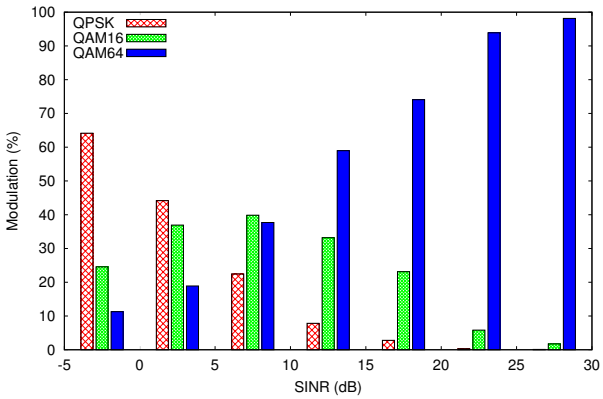


Fig. 4: Modulation under different SINR on Downlink (PDSCH)

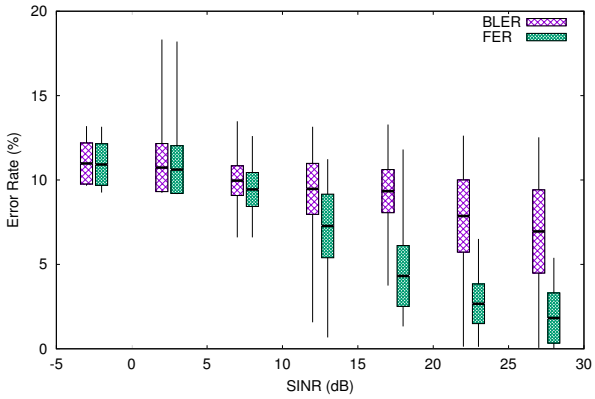


Fig. 5: Transmission Block Error Rate and Frame Error Rate under different SINR

Figure 5 shows BLER and FER under different SINR bins. The x-axis is the recorded SINR (in dB), while the y-axis is the error rate in percentage. The data presents in *boxplot*, with the rectangles depicting the standard deviation, the middle line inside the box shows the mean of throughput, and the whiskers represents the maximum and minimum values observed.

As Figure 5 shows, FER decreases dramatically from 10% to 2% as SINR increases from 0dB to 30dB. On the other hand, BLER drops slightly as SINR increases and it is still around 8% even when SINR is greater than 25dB. FER is lower than BLER is expected because each of the LTE frame consists of multiple blocks and erroneous transmission blocks may be retransmitted multiple times. Note, BLER impacts eNodeB's modulation selection also. When an eNodeB noticed UE experience high BLER through HARQ, it may switch to more robust encoding scheme (QPSK) for the next block transmission. This rate adaptation would be a challenge for

bandwidth estimation algorithms used by end hosts, such as BBR.

C. SINR vs Retransmission

LTE implements Hybrid Automatic Repeat Request (HARQ) to improve the data rate over noisy wireless channels. With HARQ retransmission scheme, when an UE receives transmission blocks with an error, it will buffer the damaged transmission block while requests a retransmission from eNodeB. Then the UE will combine the retransmitted data with its buffered data before checking CRCs. Thus, the UE has the ability to reconstruct the whole transmission block through two or more damaged blocks. In this way, HARQ will increase the efficiency of retransmission over noisy wireless links.

Figure 6 summaries the transmission characteristics over highway driving condition. Figure 6(a) compares the fraction of transmission blocks (TBs) passed CRC check and the fraction of transmission blockes which passed CRC check on its first transmission attempt. Note, the faction of TBs passed CRC check is the complementary of BLER. The x-axis is the recorded SINR (in dB), and the y-axis describes the faction of transmission blocks pass CRC checks. As Figure 6(a) shows 90% of the TBs passed CRC check, and 80% TBs passed CRC check with their first transmission attempts even when SINR is lower than 10dB. The forward error correct scheme of LTE is effective. As the SINR grows, the probability that TBs got delivered on their first transmission attempts increases to 87%.

Figure 6(b) compares the number of TB retransmissions under different SINRs. The x-axis is the SINR (in dB), and the y-axis shows the faction of retransmitted TBs. The three comparative boxes represent retransmission attempts respectively: 1 retransmission, 2 retransmissions and 3 or more retransmissions. Most of the damaged TBs are *fixed* during its first retransmission. Only a few fraction of TBs needs more than two retransmissions even when the SINR is lower than 5dB. Under *good* RF condition where SINR is greater than 20dB, there is almost ignorable numbers of 2nd and 3rd retransmissions. Therefore, the HARQ over LTE networks is efficient enough on noisy channels. Because IEEE 802.11 b/g/n does not support HARQ, the maximum throughput of a single TCP flow is only 24Mbps on a 20MHz channel [14].

V. Comparative Performance between CCAs

Nowadays, many applications are built upon TCP which is the dominant transport protocol for Internet. There is no common accepted TCP performance metrics for applications with different QoS requirements. While throughput is a good performance metric for bulk downloading or file transfer applications, delay is an important metric for interactive applications such as Skype. In this study, we choose throughput (goodput) and Round Trip Time (RTT) as the main performance metrics to compare the performances of the three TCP congestion control algorithms (CCAs) on highway.

A. Round Trip Time

In our study, two methods are used to estimate the round trip time between the testing phone and servers: i) the ICMP *ping* before each HTTP file download test, and ii) the TCP connection setup time measured through the three-way handshakes. Before we start our driving test, we confirm that our test phone is "pinned" to the PDN-gateway located in central Massachusetts, 30 miles away from our servers. Thus, we can assume the latency between our servers and PGW is lower than 10ms.

Figure 7 plots the average *ping* RTTs under different SINRs in our driving test. As the SINR decreases, the average RTT slightly increases. But the average RTT is still under 100ms, which is much less than RTT observed on HPSA+ networks [15]. Meanwhile, we observe high RTT variances in Figure 8(a). The wide RTT variance can introduce various problems for TCP CCAs: it may cause high spurious RTO rate, and make RTT based bandwidth estimation more difficult [5].

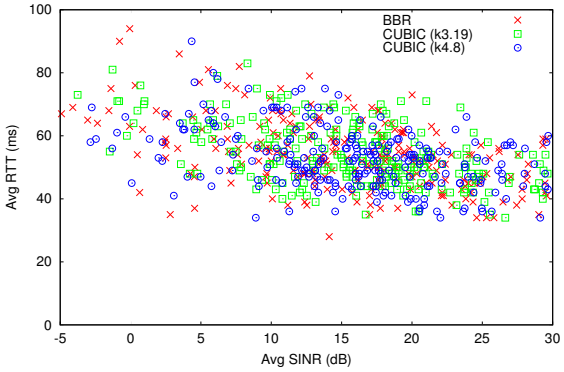


Fig. 7: Average ICMP RTT vs SINRs

Figure 8 compares the distribution of *ping* RTT and TCP three-way handshake RTT. Although *ping* RTT and three-way handshake RTT show different distribution, they are in the same range between 30ms to 100ms. Note, ICMP message are usually processed by network devices as a special case[4], and eNodeB may give higher priority to control packets (e.g ICMP) than regular IP packets. Thus, the TCP three-way handshake RTT may be slightly affected by eNodeBs' scheduling policy but it still can be used to calculate the optimal maximum CWND size [24].

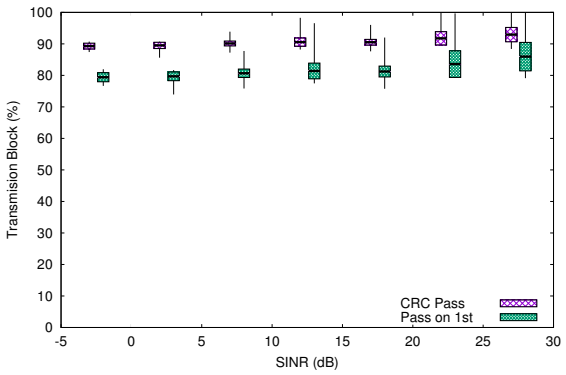
B. TCP Throughputs and SINRs

Because SINR is the key performance metric of cellular networks [16], it significantly affects the modulation selection. Therefore, higher SINR may result in high TCP throughput[5]. Figure 9 compares the throughputs of the three CCAs on highway, and Table II summarizes the mean, standard deviation, median and 95% confidence interval (CI) of mean, of the three CCAs. As Figure 9 shows, all the three CCAs are able to achieve more than 30 Mbps throughput even at high speed driving condition.

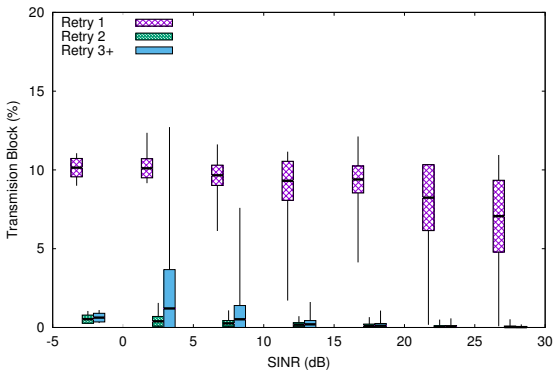
The maximum throughput of BBR reaches more than 44Mbps, which is close to the theoretical maximum downlink bit rate (45Mbps) on 10 MHz channel with 64QAM modulation over LTE networks [13]. The median and mean

[4]https://en.wikipedia.org/wiki/Internet_Control_Message_Protocol

[5]In this study, we interchangeably use throughput and goodput.

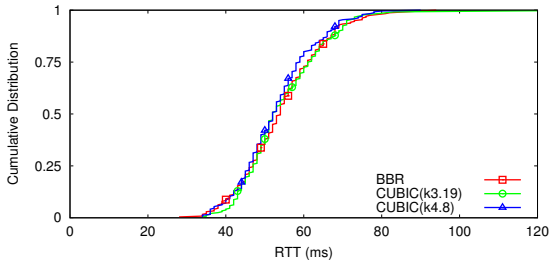


(a) Transmission Blocks Passed CRC Check

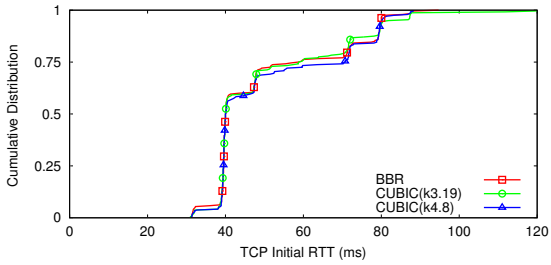


(b) Transmssion Blocks with Retransmission Attempts

Fig. 6: Retransmission Analysis of Transmission Blocks



(a) Distribution of RTT Measured through ICMP Ping



(b) Distribution of RTT Measured through TCP 3-way Handshakes

Fig. 8: Distribution of Ping RTT and TCP Connection Established Time

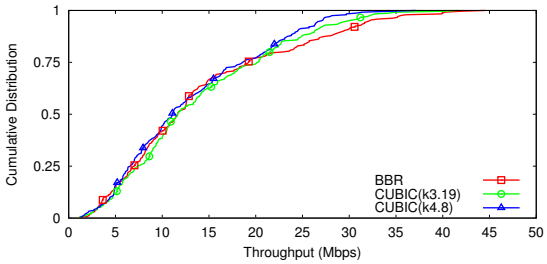


Fig. 9: Distribution of Throughputs

of throughput of BBR are similar to CUBICs. Thus, no performance degradation of BBR is apparent compared with CUBIC on LTE, even under high speed driving conditions.

TABLE II: Throughputs of Different TCP CCAs on Highway

CCAs	Mean (Mbps)	Median (Mbps)	95% CI of Mean	
			Left	Right
BBR	14.1 ± 9.5	11.6	13.1	15.2
CUBIC(k3.19)	14.0 ± 8.4	11.6	13.2	14.8
CUBIC(k4.8)	13.0 ± 7.8	11.1	12.2	13.8

Figure 10 compares the TCP throughputs for different SINRs. As Figure 10 shows, BBR yields higher throughput than the two CUBIC CCAs when SINR is greater than 10dB, especially when SINR is greater than 20dB. When SINR is below 10dB, the throughputs of the three CCAs become

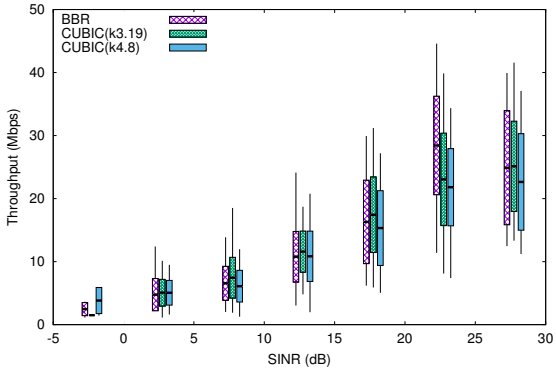


Fig. 10: Compare TCP throughputs under different SINRs

similar because these sessions may involve hand-overs when RF condition is bad (e.g. SINR is lower than 10dB).

C. TCP Throughputs and Hand-overs

Based on 3GPP standard [1], the X2 interface handles the UE hand-overs between eNodeBs: the current serving eNodeB is able to forward packets (PDUs) to next serving eNodeB through X2 interface to avoid possible service interruptions especially during highway driving. However, the serving diameter of a Band XIII (700MHz) cell tower could be up to 8000 meters, and it would take 90+ seconds for Ferrari 488 (top speed 300km/h) to leave its serving zone. In consideration

of LTE's downlink speed (40Mbps), we did not expect to observe a large fraction of 20MB file downloading sessions experiencing multiple "hand-overs", even with high speed driving scenarios.

Meanwhile, TCP flows on LTE cellular networks are small in volume, and usually have short duration [10], [11]. 90% of flows carry no more than 35.9 KB downlink payload [11] and a large portion of TCP flows can be transmitted within the initial burst if *init cwnd* is large enough. Thus, the effect of hand-over on these short live small TCP flows is negligible. But for long live flows (e.g. video flows), frequent hand-over between eNodeBs would impair the performance.

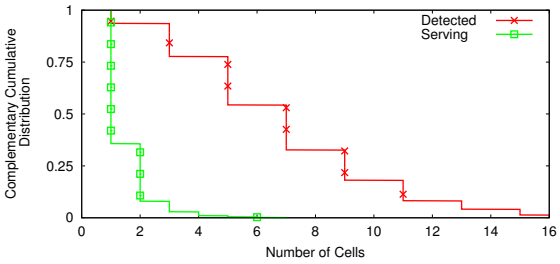


Fig. 11: Distribution of Serving and Observed Cell Sectors

Figure 11 depicts the complementary cumulative distribution (CCDF) of number of serving cell sectors and detected cells for all TCP download sessions. The detected cells include observed neighbor cells and current serving cells. As Figure 11 shows, only 35% TCP downloading sessions suffer at least one time hand-over, while less than 4% TCP sessions experience more than 2 hand-overs. On the *rural* highway (highway between major cities), we did not observe frequently hand-over.

Thanks to the existence of X2, only one of the 720 TCP sessions experience connection failure in our highway driving test. Although X2 yields *excellent* service success rate, hand-over between eNodeBs may still have negative impacts on performance, especially, confusing the bottleneck link capacity estimation algorithm used in BBR [5]. Besides, the hand-overs can cause a large idle gap in TCP connections. It may be longer than the TCP senders' RTO and introduce spurious retransmission eventually. Thus, how to improve TCP performance during hand-overs is a good research topic for future studies.

There are two reasons that might trigger hand-over between cells: i) the current serving eNodeB thinks the UE is leaving its serving zone, ii) UE discovered another eNodeB with better RF condition (stronger SINR). In either case, UE would have worse SINR measurement with the current serving cell than the neighbor cells. In our preliminary stationary test in Waltham MA, an UE experiences an extremely low throughput because it *swings* between two serving cells (both have similar bad SINR), when the tester stand on the edge of two cells.

Figure 12 shows the throughputs of three TCP flavors under hand-overs. Since only 4% TCP sessions experience more than 2 hand-overs, we combine the results with more than 2 (inclusive) hand-overs into one. As Figure 12 shows, when hand-over happens, all three TCP yields lower throughputs.

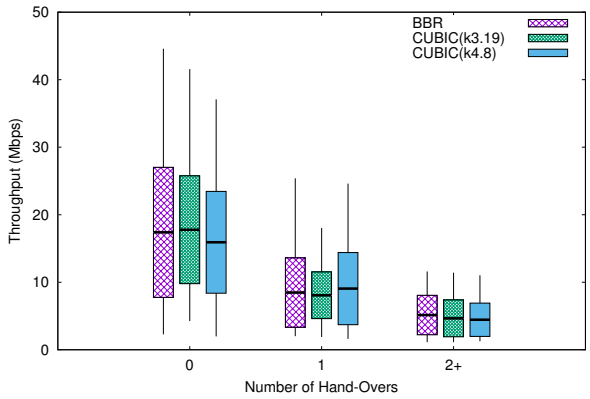


Fig. 12: Compare TCP Throughputs under Hand-overs

When hand-over happens, BBR and CUBICs still perform similarly.

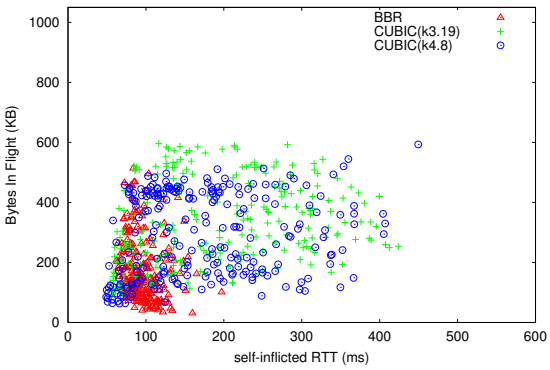
However, it is difficult to reproduce hand-over cases in Lab environment or stationary test; and there are too many uncontrolled variables on highway driving test for hand-over measurement studies. We believe that it may be easier to study hand-over cases through network simulation.

D. Self-Inflicted RTT vs Bytes Inflight and Throughputs

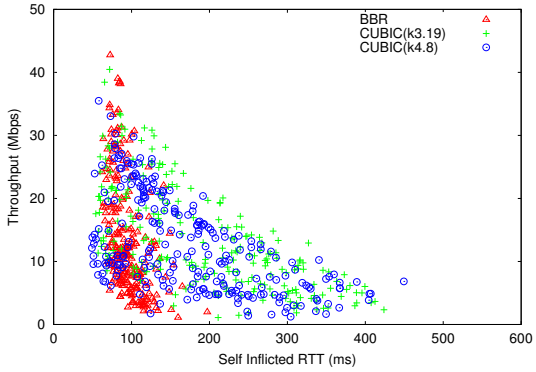
We use the "bytes in flight" and self-inflicted RTT to study the behaviors of different TCP CCAs over LTE networks under high speed driving conditions. Bytes in flight are the bytes which have been sent by servers, but have not been acknowledged yet by ACKs from the phone. It is the effective sending window which is the minimum of CWND and advertised window (RWND) of the receiver (phone). The self-inflicted RTT is calculated as the time span between a data packet and an ACK packet as a response to the data packet excluding duplicated ACKs. The self-inflicted RTT includes the round trip propagation delay and queuing delay introduced by network devices along the data path.

Figure 13(a) compares the self-inflicted delay and bytes in flight of the three CCAs. As we mentioned earlier, BBR and CUBICs have different design principle: BBR attempts to keep the self-inflicted delay low, and CUBIC detects congestion only on possible packet loss. Thus BBR is observed having low self-inflicted RTT as well as bytes in flight.

On the arrival of each ACK, BBR estimates the bottleneck bandwidth by feeding the latest delivery rate into a max-filter, and uses a windowed min-filter to estimate the recent propagation delay. BBR calculates its optimal BDP by using the max-filtered bandwidth and the min-filter RTT [5], [6]. Although BBR's min-filtered RTT measurement would be close to the round propagation delay, it could not fully utilize the device queue inside eNodeBs to improve the throughput. On the other hand, measured RTTs over LTE links show a large variance, especially on noisy radio link. The radio link modulation may change frequently when SINR fluctuates (as shown in Section IV-A). The frequent rate adaptation would impact the estimation of the wireless link capacity. Therefore, only 10% BBR sessions yield more than 30Mbps throughput.



(a) Self-Inflicted RTT vs Bytes In Flight



(b) Self-Inflicted RTT vs Throughput

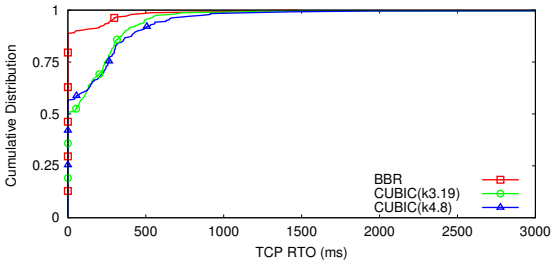
Fig. 13: Self-inflicted RTT vs Bytes In Flight

E. RTOs and Retransmission

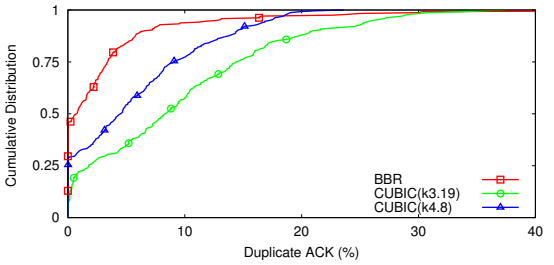
TCP retransmission is another important performance metric over wireless links: when the TCP sender does not receive the expected ACKs before retransmission timer expires, or receives three duplicated ACKs or SACK, the retransmission will be triggered by RTO [15]. TCP RTO is estimated by the TCP senders with smoothed RTT and RTT variance [17], which is initially for wired links where RTT variance is low. Therefore, the accuracy of TCP RTO estimation may decrease when RTT variance is high, especially under highway driving condition which experiences hand-overs between eNodeBs.

Figure 14(a) shows the distribution of RTOs of the three CCAs under highway driving condition. Note, BBR has the smallest RTO values because it controls RTT at a low level. However, the maximum RTO value observed with BBR is still up to 1 second. Such a long RTO indicates the current RTO estimation algorithm is not able to adapt fast enough to the high variance of RTT over LTE networks. There exists a research opportunity to develop new RTO estimation algorithms over high RTT variance wireless environment.

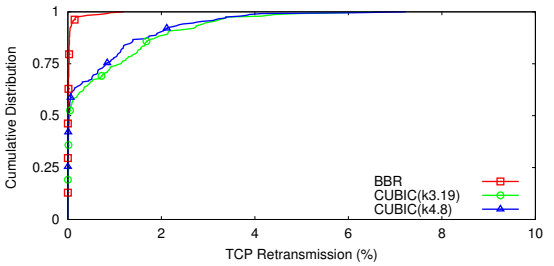
TCP does not use duplicated ACKs to update its RTT and RTO calculation [17]. For wired connection or stationary test cases over wireless environment, the percentage of duplicated ACK would be low [15]. Figure 14(b) shows the distribution of the fraction of duplicated ACKs which is calculated as duplicated ACKs over the total number of ACKs from the UE. Figure 14(c) compares the retransmission fractions of the three CCAs. In terms of all the three metrics: RTO, the fraction of dupACKs and retransmission fractions, BBR outperforms CUBICs. Since BBR attempts to maintain a low RTT with a smaller maximum CWND, BBR successfully avoid the possible "buffer bloat" inside eNodeBs, even under highway driving conditions.



(a) Distribution of TCP RTO



(b) Distribution of Duplicated ACKs



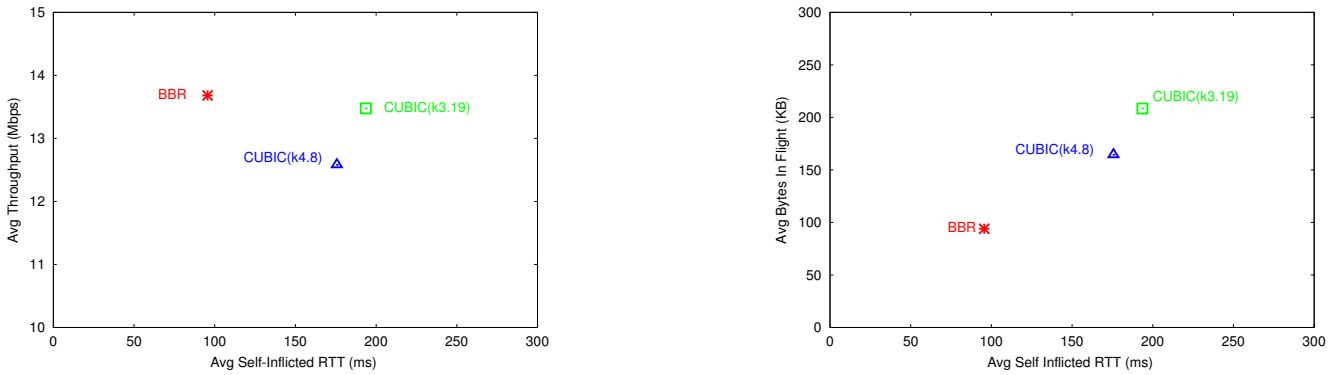
(c) Distribution of TCP Retransmission

Fig. 14: TCP RTO, Duplicated ACK Fraction and Retransmission

F. Summary

Figure 15 summarizes the results of all the three CCAs under the highway driving condition. In each figure, we plot only one point per CCA, corresponding to its average measured value. Figure 15(a) shows that BBR is the winner in terms of RTT, while yields comparable throughput with CUBICs on highway. As we noticed, BBR has the lowest self-inflicted delay among all CCAs, and it may potentially improve the QoE of delay sensitive applications on wireless networks [14].

Figure 15(b) demonstrates the design principle behind BBR.



(a) Average Self-Inflicted RTT vs Throughput

(b) Average Self-Inflicted RTT vs Bytes In Flight

Fig. 15: Compare TCP Congestion Control Algorithms

BBR's control function calculates BDP as maximum delivery rate multiplying minimum RTT observed in its probing phase, and then sets its CWND as small multiple (gain factor 1.25) of the estimated BDP [6]. However, the gain factor selection is a challenge. A small gain factor (e.g. 1.0) would result in small queues inside eNodeBs and low throughput. However, since long self-inflicted latencies may cause unnecessary packet drops and additional retransmissions, keep RTT low may not be a bad design choice.

VI. CCA Design over Mobile Networks

Different from wired environment, the bottleneck devices – eNodeBs, contain large per-device queues to subscribers. Jiang *et al.* pinpointed that "bufferbloat" is one of the most critical reasons behind the performance degradation on LTE networks [12] and there is no practical solution has been widely deployed. The hardware vendors attempt to solve this problem by limiting maximum of TCP receive buffer size to be a small value [12]. However, this trick will lead to sub-optimal performance in many scenarios especially when bandwidth delay product (BDP) is smaller than the pre-selected maximum receive buffer size. Moreover, this approach requires adjustment or modification on every UE and it is not practical to deploy onto devices from different vendors.

Enlarging the buffer spaces may be an effective design to absorb occasionally burst traffic, but may mislead loss-based TCP CCAs (e.g CUBIC) to overshoot packets into the pipe. Consequently, the end-to-end latency may be increased. On the other hand, large queuing size inside eNodeBs would increase the BDP, and may increase throughputs for bulk file transfer applications. Therefore, the future CCA should take advantage of large buffer size inside eNodeBs, and fill the "fat" pipe with optimized amount of data to achieve the balance between delay and throughputs.

Although fairness is usually an important metric to evaluate the congestion algorithm, it is not a big concern for LTE networks. The bottleneck devices in LTE network maintain large per-device queues and there are not many concurrent flows per device [11]. Meanwhile, 90% of flows from LTE network carry less than 35.9KB downlink payload, and 48.1% of flows are less than 5 seconds [11]. Therefore, fairness should not be listed as a top CCA design requirement over mobile network.

In addition to throughput, low self-inflicted RTT should be an important metric to evaluate the performance of CCAs over LTE networks. BBR actually moves in the right direction, and it is able to keep a high throughput without over-saturate wireless links. Throughput should not be the main performance metric over LTE networks, and low RTT should be an important metric when designing CCAs for future mobile networks.

VII. Conclusions

In this paper, we present a comprehensive measurement study of the performances of three TCP flavors, including the latest BBR, under highway driving condition. In total 8-hour highway driving tests, we complete 720 20MB file downloads on a tier-1 carriers' LTE network. More than 13.5GB of data has been collected which is 15% of the "a large scale of 8-month study" conducted on HSPA+ [15]. We analyze the TCP performance on metrics as RTT, throughput, retransmission rate as well as SINR, modulation scheme, MAC Layer retransmission ratio under high speed mobility scenarios, which is important for cellular networks.

We find that TCP CUBIC with hystart enabled may not perform well over LTE networks on highway. Similar to stationary test, it takes more than 4 seconds for CUBIC to "ramp up" to its maximum CWND when SINR is good. Thus, CUBIC may result in low link utilization. Unexpectedly, although BBR is designed for mainframes, it achieves comparable throughputs of its rival – CUBIC, with much lower self-inflicted RTTs. In short, BBR balances throughput and RTT gracefully on highway driving condition: high throughput and lower self-inflicted RTT have been observed. However, in several cases, BBR would only create very small queues in eNodeBs, and may not fully utilize the large queue capacity inside eNodeBs to maximize the throughput. BBR would be a good congestion control algorithm choice for mobile network providers. Last but not least, BBR is a good starting point to design a new congestion control algorithm over future cellular networks.

ACKNOWLEDGEMENTS

We thank our colleagues, Vijay Najundan, Damascene Joachimpillai, Haim Ner, Eduard Rubinshtein, Vikram Siwach, Zhao Li, and Anil Vayaal, who provided insight and expertise that greatly assisted this project. We would also like to show our gratitude to our friends Jamal Salim and Rupinder Makkar for sharing their pearls of wisdom with us to understand "inside of TCP".

REFERENCES

[1] 3GPP TS 36.423 Evolved Universal Terrestrial Radio Access Network (E-UTRAN); X2 Application Protocol (X2AP) (Release 12), September 2014.

[2] M. Alizadeh, A. Greenberg, D. A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, and M. Sridharan. Data Center TCP (dctcp). In *ACM SIGCOMM Computer Communication Review*, volume 40, pages 63–74. ACM, 2010.

[3] E. Atxutegi, F. Liberal, K.-J. Grinnemo, A. Brunstrom, A. Arvidsson, and R. Robert. TCP Behaviour in LTE: Impact of Flow Start-up and Mobility. In *2016 9th IFIP Wireless and Mobile Networking Conference (WMNC)*, pages 73–80, July 2016.

[4] L. Brakmo.

[5] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh, and V. Jacobson. BBR: Congestion-Based Congestion Control. *ACM Queue*, 14, September-Octobor, 2016.

[6] Y. Cheng and N. Cardwell. Making Linux TCP Fast. In *The Technical Conference on Linux Networking (NETDEV 1.2)*, pages 73–80, Octber.

[7] L. A. Grieco and S. Mascolo. Performance Evaluation and Comparison of Westwood+, New Reno, and Vegas TCP Congestion Control. *ACM SIGCOMM Computer Communication Review*, 34(2):25–38, 2004.

[8] S. Ha, I. Rhee, and L. Xu. CUBIC: a New TCP-Friendly High-Speed TCP Variant. *ACM SIGOPS Operating Systems Review*, 42(5):64–74, 2008.

[9] D. A. Hayes and G. Armitage. Revisiting TCP Congestion Control using Delay Dradients. In *the 10th International IFIP TC 6 Conference on Research in Networking*, pages 328–341. Springer, May 2011.

[10] J. Huang, F. Qian, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck. A Close Examination of Performance and Power Characteristics of 4G LTE Networks. In *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services*, MobiSys '12, 2012.

[11] J. Huang, F. Qian, Y. Guo, Y. Zhou, Q. Xu, Z. M. Mao, S. Sen, and O. Spatscheck. An In-depth Study of LTE: Effect of Network Protocol and Application Behavior on Performance. In *ACM SIGCOMM Computer Communication Review*, SIGCOMM '13, 2013.

[12] H. Jiang, Z. Liu, Y. Wang, K. Lee, and I. Rhee. Understanding Bufferbloat in Cellular Networks. In *Proceedings of the 2012 ACM SIGCOMM Workshop on Cellular Networks: Operations, Challenges, and Future Design*, CellNet '12, 2012.

[13] C. Johnson. *Long Term Evolution in Bullets, 2nd Edition*. CreateSpace Independent Publishing Platform, Northampton, UK, 2 edition, July 2010.

[14] F. Li, M. Claypool, and R. Kinicki. Treatment-based traffic classification for residential wireless networks. In *2015 International Conference on Computing, Networking and Communications (ICNC)*, February 2015.

[15] L. Li, K. Xu, D. Wang, C. Peng, Q. Xiao, and R. Mijumbi. A Measurement Study on TCP Behaviors in HSPA+ Networks on High-Speed Rails. In *2015 IEEE Conference on Computer Communications (INFOCOM)*, June 2015.

[16] R. Merz, D. Wenger, D. Scanferla, and S. Mauron. Performance of LTE in a High-velocity Environment: A Measurement Study. In *Proceedings of the 4th Workshop on All Things Cellular: Operations, Applications, & Challenges*, AllThingsCellular '14, pages 47–52, August 2014.

[17] V. Paxson, M. Allman, J. Chu, and M. Sargent. Computing TCP's Retransmission Timer (RFC 6298). Technical report, 2011.

[18] R. Robert, E. Atxutegi, A. Arvidsson, F. Liberal, A. Brunstrom, and K.-J. Grinnemo. Behaviour of Common TCP Variants over LTE. In *the IEEE Global Communications Conference (GLOBECOM)*, pages 1–7, Decemember 2016.

[19] J. Snellman. Mobile TCP Optimization: Lessons Learned in Production. Technical report, Telco. Networks, August 2015.

[20] H. Tazaki, F. Uarbani, E. Mancini, M. Lacage, D. Camara, T. Turletti, and W. Dabbous. Direct Code Execution: Revisiting Library OS Architecture for Reproducible Network Experiments. In *Proceedings of the 9th ACM Conference on Emerging Networking Experiments and Technologies (CoNext'13)*, pages 217–228. ACM, Decemeber 2013.

[21] F. P. Tso, J. Teng, W. Jia, and D. Xuan. Mobility: a Double-Edged Sword for HSPA Networks: a Large-Scale Test on Hong Kong Mobile HSPA Networks. In *Proceedings of the 11th ACM international symposium on Mobile ad hoc networking and computing*, pages 81–90. ACM, September 2010.

[22] K. Winstein, A. Sivaraman, and H. Balakrishnan. Stochastic Forecasts Achieve High Throughput and Low Delay over Cellular Networks. In *Proceedings of the 10th USENIX Conference on Networked Systems Design and Implementation*, nsdi'13, 2013.

[23] Q. Xiao, K. Xu, D. Wang, L. Li, and Y. Zhong. TCP Performance over Mobile Networks in High-Speed Mobility Scenarios. In *Proceedings of the 22nd International Conference on Network Protocols (ICNP 2014)*, pages 281–286. IEEE, October 2014.

[24] Y. Zaki, T. Pötsch, J. Chen, L. Subramanian, and C. Görg. Adaptive Congestion Control for Unpredictable Cellular Networks. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, SIGCOMM '15, 2015.