2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS) | 978-1-6654-1494-4/21/$31.00 ©2021 IEEE | DOI: 10.1109/IWQOS52092.2021.9521270

# *CUP*: Cellular Ultra-light Probe-based Available Bandwidth Estimation

Lixing Song*, Emir Halepovic†, Alamin Mohammed‡, Aaron Striegel‡
*Rose-Hulman Institute of Technology, IN, USA,
†AT&T Labs – Research, NJ, USA
‡University of Notre Dame, IN, USA,
Email: *song3@rose-hulman.edu, †emir@research.att.com, ‡{amohamm2, striegel}@nd.edu

***Abstract*—Cellular networks provide an essential connectivity foundation for a sizable number of mobile devices and applications, making it compelling to measure their performance in regard to user experience. Although cellular infrastructure provides low-level mechanisms for network-specific performance measurements, there is still a distinct gap in discerning the actual application-level or user-perceivable performance from such methods. Put simply, there is little substitute for direct sampling and testing to measure end-to-end performance. Unfortunately, most existing technologies often fall quite short. *Achievable Throughput* tests use bulk TCP downloads to provide an accurate but costly (time, bandwidth, energy) view of network performance. Conversely, *Available Bandwidth* techniques offer improved speed and low cost but are woefully inaccurate when faced with the typical dynamics of cellular networks. In this paper, we propose *CUP*, a novel approach for Cellular Ultra-light Probe-based available bandwidth estimation that seeks to operate at the cost point of *Available Bandwidth* techniques while correcting accuracy issues by leveraging the intrinsic aggregation properties of cellular scheduling, coupled with intelligent packet timing trains and the application of Bayesian probabilistic analysis. By keeping the costs low with reasonable accuracy, our approach enables scaling both with respect to time (longitude) and space (user device density). We construct a *CUP* prototype to evaluate our approach under various demanding real-world cellular environments (longitudinal, driving, multiple vendors) to demonstrate the efficacy of our approach.**

## I. Introduction

Cellular connectivity has advanced tremendously over the past decade. While early smartphones labored heavily under the umbrella of early 3G connectivity, the LTE networks today can easily offer speeds in excess of tens of megabits per second and beyond enabling high-quality video streaming and increasingly diverse and immersive applications. Network designs for 5G aim to radically expand bandwidth and connectivity options with special consideration for next-generation applications and the significant implications posed by the Internet of Things (IoT)[1].

However, while peak speeds have increased dramatically, the converse is that network dynamics have also increased as well. Users have the potential to experience ultra-fast connectivity in one moment only to experience painfully slow speeds in the next due in part to mobility, competing clients, or ill-advised WiFi connectivity. From the perspective of both the network operator and the end user, these *dynamic experiences* are extremely frustrating. Moreover, such challenging scenarios tend neither to be consistent nor persistent making troubleshooting especially problematic. Although LTE offers numerous mechanisms for understanding lower level wireless dynamics, the gold standard is still to directly test network performance at the application layer. Unfortunately, the majority of existing techniques are ill-suited to enable testing at the scale (user density) and timescales (longitudinally) required for proper network instrumentation.

In one group, techniques that fall into the category of *Achievable Throughput* (*AT*) tests are considered accurate but expensive. *AT* tests attempt to download volumes of data over a period of time giving a precise measurement of what the mobile device (User Equipment - UE) would have experienced at that point in time. Notable examples include SpeedTest.net [2], *iperf3* [3], and Mobiperf [4]. The perceived accuracy of the tests stems from moving actual data and competing with existing flows over a period of time. However, large bulk downloads can crowd out other useful traffic making it difficult to conduct overlapping tests in the same area [5]. Most importantly, bulk downloads are extremely expensive from the perspective of the UE, taking up to tens of seconds to complete and consuming significant amounts of energy, making it difficult to run tests longitudinally.

In contrast, Available Bandwidth (`AB`) techniques are cheap and potentially quick but are considered less accurate. Available Bandwidth techniques leverage precisely constructed packet sequences and observe the resulting packet dispersion amongst said sequences. Thus, rather than continuously exerting pressure over time as with *AT* tests, `AB` tests selectively nudge against the bottleneck link capacity through a series of fixed and/or adaptive probing sequences. Canonical examples from the literature include PathChirp [6], IGI [7], and others [8, 9]. Critically, `AB` techniques are exceptionally vulnerable to the effects of aggregation present in cellular, which can nullify the expected dispersion effects. Although some work has been able to overcome aggregation effects in WiFi networks [10, 11], the multiple levels of control and increased complexity of UE dynamics (mobility, roaming) make `AB` characterization incredibly challenging. However, it is precisely because of those increased dynamics that makes a better instrumentation tool at scale tremendously compelling.

Thus, the focus of this paper is to try to address this

978-0-7381-3207-5/21/$31.00 ©2021 IEEE

978-1-6654-1494-4/21/$31.00 ©2021 IEEE

fundamental juxtaposition: (1) *AT* techniques are considered accurate but too expensive to run longitudinally due to UE energy cost and network bandwidth cost, while (2) AB techniques allow for longitudinal and large scale observation due to their low cost but are minimally useful due to their high inaccuracy in cellular networks. In this paper, we propose a novel mechanism – *CUP*, as a Cellular Ultra-light Probe-based available bandwidth estimation approach that solves this very juxtaposition. We posit that *CUP* provides sufficient accuracy to gain useful insights while operating in an exceptionally efficient manner to enable scaling across time and space (user density). The key contributions of our paper are as follows:

- *TBS - Effective Congestion Indicator:* We show how Transport Block Size (TBS) fluctuation can be a useful indicator to determine when network capacity has been exceeded. Through the careful construction of a packet probing train, we demonstrate how fluctuations in TBS and the resulting timing dispersion can be used to derive AB quickly and efficiently.
- *Bayesian Change-Point Estimation Algorithm*: We focus on utility from a trending standpoint, which allows us to introduce a probabilistic model that translates the problem of estimating AB into a change-point detection problem over the observed measurements. Our Bayesian approach carefully considers the large uncertainties involved in the estimation process. By utilizing *Gibbs sampling*, we solve the problem and further manipulate the sampled posterior distributions to assess the estimated result.
- *Real World Experimental Evaluation:* We present real-world experiments to validate the effectiveness of *CUP* by creating a working prototype to conduct both laboratory and real-world experiments. Our prototype operates with typical HTTP, requiring zero modifications to the client itself, and with all measurement provided by a Linux-based web server. We conduct several longitudinal experiments in different cellular environments, e.g., mobile versus static, indoor versus outdoor. Moreover, our prototype demonstrates that *CUP* can be deployed across both time and space (node density) offering an important new tool for measurement at scale.

## II. Related Work and Motivation

In this section, we start with an overview of existing work on bandwidth measurement in LTE. We then discuss the challenges of estimating AB and present motivating real-world experiments to demonstrate the shortcomings of existing AB solutions in LTE.

### A. Cellular Network Measurement

Broadly speaking, existing work on cellular network measurement and characterization can be classified into two approaches: *passive* and *active*. Passive works focus on monitoring bandwidth by observing network activities surreptitiously [12–17]. While some works [15, 17–19] attempt to capture physical layer information to infer bandwidth conditions, most make traffic observation based on upper layer information (e.g., the transport layer). For example, an empirical study on LTE [13] deployed a large-scale monitoring system to examine instantaneous AB through packet timing information at the transport layer. Unencrypted nature of control packets between UE and eNodeB was leveraged to determine overload in LTE networks [12]. Another recent work tries to improve AB estimation accuracy by considering special packet patterns in LTE [16].

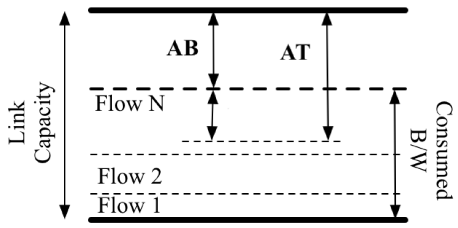


Figure 1: Illustration of the concepts of *AB* and *AT*.

Unlike passive approaches, active approaches require injection of probe traffic [2–4, 20–24]. Bandwidth can be measured using different metrics. The most common metric, *AT*, measures the maximum throughput a user can achieve by launching large TCP flow(s). For example, popular tools `iperf3` [3] and *SpeedTest* [2] fall into the category of *AT* estimation. Critically, *AT* estimation is expensive in terms of time, bandwidth, and client energy. Moreover, given the limited bandwidth resources of the cellular network, the aggressive traffic from an *AT* test can potentially degrade the performance of other users [5]. To avoid these shortcomings, we focus on another metric – AB [6, 20–22].

**AB Estimation:** The metric of *AB* specifically refers to the *available* or *residual* capacity of a link. Unlike an *AT* test that can suppress existing flows (as illustrated in Fig 1), AB measures the unoccupied portion of the bandwidth. Critically, AB estimation tries not to compete with existing traffic making the AB test less intrusive and potentially less costly.

To estimate AB, existing solutions adopt the idea of packet pairs or sequence trains. The general approach is to send a series of probe packets from a server to a receiver with particular variations on packet timing and sizes. AB solutions can be largely divided into two categories: the Packet Gap Model (PGM) and the Packet Rate Model (PRM). PGM approaches (e.g., IGI [7], Spruce [25] and Abing [26]) send packet pairs with specific gaps and analyze the packet dispersion to infer AB. In contrast, PRM approaches (e.g., PathLoad [27], PathChirp [6], and TOPP [9]) send packet sequences with different rates and estimate AB by discerning when the probe traffic induces network congestion. Fundamentally, AB estimation relies on an expected correlation between network conditions and probe traffic observations.

**Estimating AB in Wireless Networks:** The relationship between network condition and probe traffic observations can vary under different types of networks, e.g., Ethernet, WiFi,

and LTE. As many known tools are designed for wired networks, they often fail in modern wireless networks due to different transmission schemes (such as frame aggregation in WiFi and Transport Block batching in LTE). In the past few years, many attempts [10, 11, 20, 21] have been made to improve AB estimation on wireless networks. In WiFi, WBest+ [11] improves on a previous work, WBest [8] (a PRM approach on WiFi), to mitigate the detrimental impact of frame aggregation on estimation accuracy. Another effort, AIWC [10], leverages the embodied information in aggregated frames to improve AB estimation accuracy in modern WiFi networks.

In cellular networks, several recent works [20, 21, 28] try to revamp *PathChirp* [6] for LTE. The improvements mostly emphasized two aspects: 1) re-designing the probe packet pattern to cope with the high dynamics of LTE [20, 21], and 2) adopting curve fitting approaches to handle the packet batching effects [20, 28, 29]. Although the estimation accuracy was improved, those methods inevitably involve increasing the duration of the test to alleviate the impact of packet aggregation. To better understand exactly how aggregation impacts existing AB solutions, we conduct further exploration in Section II-B.

**Beyond AB:** To improve TCP performance in cellular networks, several works (e.g., Sprout [30], Verus [31]) exploit ongoing TCP packet delays for bandwidth inference to facilitate better congestion control. These approaches require lengthy packet exchanges and kernel space modifications at both clients and servers while *CUP* offers a light-weight solution with server-only modifications. As a PGM-based user-space tool, *QProbe* [32] attempts to detect if the bottleneck is on the cellular radio link or not (a binary decision), but does not measure the bandwidth. *CUP*, on the other hand, provides a numeric AB estimation.

### B. Challenges of Estimating AB in LTE

**LTE Downlink Scheduling:** LTE is designed to be highly flexible in radio resource allocation among multiple devices. The LTE downlink uses OFDMA (Orthogonal Frequency Division Multiple Access) to enable fine-grained channel resource allocation in the time and frequency domains. As shown in Figure 2, the channel resource is divided into a grid structure. The frequency domain (vertical) is divided into 180-kHz sub-channels. The time domain (horizontal) is divided into continuous 0.5 *ms time slots*. Thus, a minimal resource unit (i.e., one cell in the grid) is called *Physical Resource Block* (PRB). For every Transmission Time Interval (TTI) of 1 $ms$, the base station (eNodeB) will make a scheduling decision to assign PRBs to different UEs. As Figure 2 shows, different UEs can obtain different shares of the channel over time.

When assigning PRBs, eNodeB will also decide the proper *Modulation and Coding Scheme* (MCS) for transmitting the

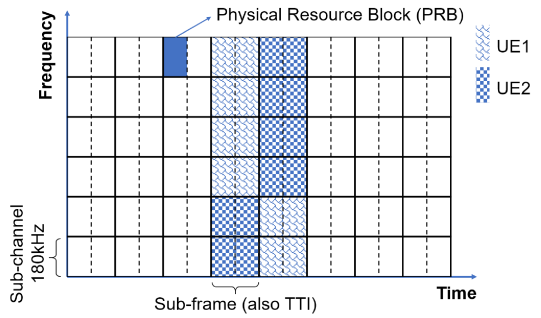


Figure 2: Channel resource allocation on LTE downlink.

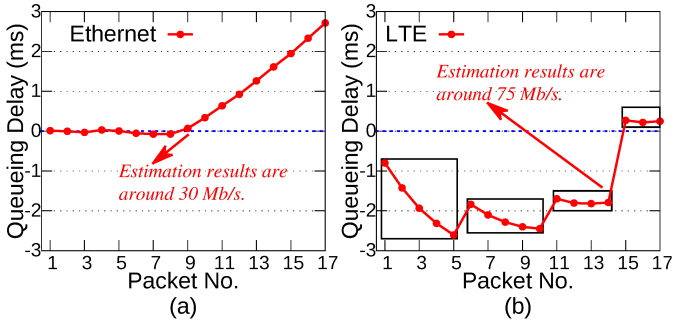


Figure 3: AB estimation (PathChirp) on Ethernet v.s. LTE under similar network setting.

data[1]. Combining PRBs and MCS, the total data that is scheduled to transmit to a UE during a TTI is called a Transport Block (TB), and its size is referred to as Transport Block Size (TBS). From the perspective of upper layers (e.g., transport layer), since a TBS (up to 9 KB) is usually larger than the packet (e.g., $\leq$ 1500 B), the packets are often transmitted in batches. This makes TBS-based transmission in LTE bursty.

**Experimental Study of AB Estimation on LTE:** Through real-world experiments, we will show how the batched transmission can harm traditional AB estimation methods. In this experiment, we run *PathChirp* on a smartphone (Huawei Nexus 6P) over LTE. *PathChirp* sends packets with exponentially decreasing packet gap, and estimates AB by searching for a turning point where the packet queuing delay (i.e., packet sending gap minus the receiving gap) starts to rise. For comparison, we set up a local wired network with a wired link capacity of 70 Mb/s (similar to the typical capacity of a 10-MHz carrier with 2x2 MIMO under ideal RF conditions). To emulate traffic load, we first measured the throughput on LTE via `iperf3` (i.e., 40 Mb/s). Next, we created similar throughput conditions ($AT$ = 40 Mb/s) on the wired network by generating 30 Mb/s of cross traffic (using `iperf3`) from another wired client. In this case, we know the ground truth of AB should be less than or close to $AT$, according to their definitions.

In Fig 3, we plot the packet queuing delay in a packet sequence as output from *PathChirp*. The test probes vary from 10 Mb/s to 200 Mb/s. The rectangles indicate the packets that

[1]The scheduling policy of how to assign PRBs and MCS is vendor-specific and can adopt different principles [33].

arrived at the same time. We start by looking at the Ethernet network. As Fig 3a shows, the queuing delays gradually increase when the probe rate reaches the AB. The estimated AB is around 30 Mb/s which is close to *AT* (40 Mb/s). In contrast, in Fig 3b, the queuing delay pattern on LTE is dramatically distorted and shows bursty patterns. As a result, the estimated results are erroneously high (75 Mb/s) which is considerably above *AT*.

**Implication:** Unsurprisingly, the batching in LTE breaks the expected correlation between network conditions and probe observations (e.g., consistently increasing delay once capacity is exceeded). It is the need for revisiting this relationship and designing a new AB estimation solution for LTE that motivates this paper.

## III. System Design

In this section, we introduce the design of our proposed AB estimation solution for LTE networks. The designed solution adopts the traditional methodology of probe packet-based measurement. However, in contrast to existing methods, we devise a new physical layer metric – *TBS Fluctuation* – to capture network congestion conditions when induced through probe packet observations. The designed metric serves as an effective network congestion indicator (Section III-A). Based on this metric, we design new probe packet train pattern to cope with the aggregation effect of LTE (Section III-B). The proposed design is implemented on an HTTP-based packet probing system [34] which requires no modification on the client side (Section III-C). In particular, we discuss how we can leverage the packet observations to infer the designed network metric. Along the discussions in this section, we conduct real-world experiments to validate the various design principles.

### A. TBS Fluctuation as the Network Congestion Indicator

Conceptually, the goal of estimating AB is to measure the portion of bandwidth that a UE can obtain without inducing congestion to existing traffic. Thus the key problem of measuring (AB) can be reduced to the following: *Can an end host detect if it is experiencing congestion?* If the problem can be solved, the task of AB estimation can then be reduced to searching for the maximum traffic rate to the UE that does not cause congestion. To discern congestion, we design a network congestion indicator – *TBS Fluctuation*. As we know that the TBS is adaptively determined for UEs according to their *channel quality* and *traffic competition* and that channel quality for a UE is likely to be relatively consistent within a small time window (e.g., 1 second[2]), variations in TBS should be highly correlated with traffic competition and congestion.

Namely, the TBS received at the UE is supposed to be relatively consistent when the bandwidth resource is ample and requires no or light competition. Otherwise, if a UE is

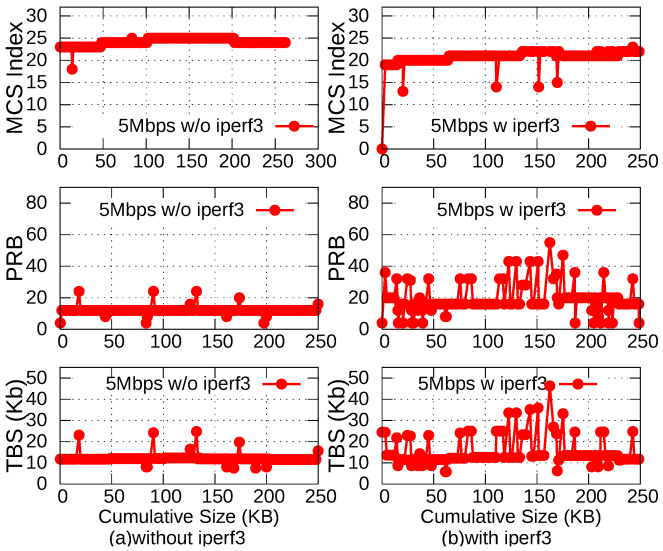


Figure 4: The PHY layer observation under different network competition conditions.

experiencing serious bandwidth competition with other UEs (e.g., network congestion), the TBS assigned will likely vary. This is because the eNodeB trying to maintain a proportional fair schedule[3] among UEs will inevitably prioritize the needs of different UEs over multiple TTIs. As a consequence, a UE will receive varying TBS assignment over time. Technically, by observing the variation of TBS for a UE over time (i.e., TTIs), we are able to detect whether or not a UE is experiencing congestion. Therefore, we define *TBS Fluctuation* ($TF$) to describe the degree of TBS variation. For a series of TBS assigned to a UE, $TBS_i, i = 1, ..., N$, we have:

$$TF_i = |TBS_{i+1} - TBS_i|, i = 1, ..., N-1 \quad (1)$$

**Experimental Validation:** To demonstrate how *TBS Fluctuation* can discern congestion (traffic competition), we employ two LTE phones (Huawei Nexus 6P and Google Pixel, both running Android 7.1.1) on a production LTE cellular network in the U.S. with both phones connected to the same radio cell. We use one phone as a test UE to generate CBR traffic on the downlink. The second phone generates cross traffic via `iperf3`. We first send 200 packets at a CBR of 5 Mb/s to the test UE without competing traffic. Then, we launch `iperf3` traffic to the competing UE and repeat the same CBR traffic to the test UE. The PHY layer trace was collected from the test UE using *MobileInsight* [35].

In Fig 4, we plot the PHY trace collected both without (Fig 4a) and with (Fig 4b) competing traffic, in a cell with the 15 MHz carrier on the 1900 MHz band equipped with the outdoor Distributed Antenna System (DAS). Similar characteristics were observed during the test in the conventional macro cell using the same carrier and bandwidth. Since the

[2]As observed in our real-world experiment, the MCS (which is proportional to CQI–Channel Quality Index) is largely consistent over 1-2 seconds.

[3]According to the survey [33], a practical scheduler should guarantee fair throughput distribution among users.

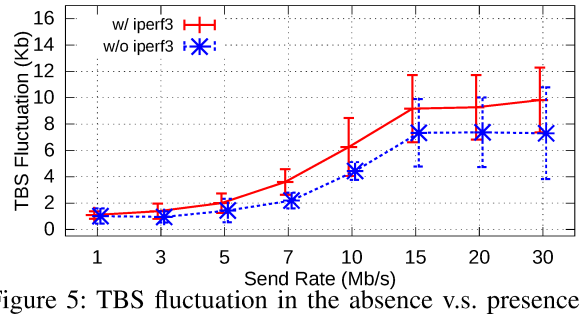


Figure 5: TBS fluctuation in the absence v.s. presence of *iperf3* traffic w.r.t. different send rates.

two runs were conducted within several seconds of each other, the overall bandwidth and channel conditions are expected to be consistent over this period of time [36]. The observed difference should be largely attributable to the generated competing traffic. As shown, the MCS under both cases is similar as the channel quality was relatively consistent. When there is no competing traffic (left-hand plots), the patterns of assigned PRBs and TBS show as smooth horizontal lines with few spikes. However, in the presence of competition (right-hand plots), the PRB and TBS patterns fluctuate with a large number of spikes. The different observations of TBS clearly show promise for revealing traffic competition.

**Analytic Evaluation:** To further evaluate *TBS Fluctuation*, we extend the previous experiment to vary the send rate of the CBR traffic to the test UE from 1 Mb/s to 30 Mb/s. Following the same pattern for each send rate, we send ten 200-packet sequences in the absence and presence of the `iperf3` competing traffic. From the captured PHY layer traces, we calculate the *TBS Fluctuation* over all TBS values and take the aggregated result of each 200-packet sequence. In Fig 5, we plot the comparison of the measured *TBS Fluctuation* in the absence and presence of competing traffic. The points represent the mean and the bars indicate the standard deviation. We see that the *TBS Fluctuation* gradually increases with the sending rate. As expected, the `iperf3` competing traffic causes higher *TBS Fluctuation* compared to the case without `iperf3`. It is noteworthy that the *TBS Fluctuation* remains quite small in the low send rate range (i.e., 1 to 5 Mb/s) and then quickly increases past 7 Mb/s. This potentially reveals that the `AB` under that experimental environment was roughly 7 Mb/s. We arrive at this conclusion as the traffic experienced marginal competition when the send rate was less than 7 Mb/s while the competition indicated by *TBS Fluctuation* became more severe above 7 Mb/s. This observation informs our *self-induced congestion* probe packet approach, which will be discussed in the following subsection.

### *B. Probe Train Design*

Following the intuition of leveraging the *TBS Fluctuation* as a congestion indicator, we adopt the concept of *self-induced congestion* to form the probe packet sequence (a.k.a train). The general idea is to send a train with monotonically increasing probe rate to search for a "tipping point" where the network condition changes from uncongested to congested. Thus the `AB` can be approximated with the probe rate at the "tipping point". To minimize the probe traffic cost, we design the probe train as a "one-shot" test that requires only injecting one packet train to complete the test, rather than doing iterative packet streams such as *PathChirp* and *IGI* do.

To generate different probe rates, we vary the packet gaps and fix packet sizes at the Maximum Transmission Unit (MTU) size as different packet sizes may experience significantly different delays over LTE [37]. For our baseline, we employ a linear pattern to our probe rate. However, given the batched transmission in LTE, when the probe rate is high, a large number of packets may be aggregated together, thus generating only a few timing measurements at the receiver. To cope with this aggregation effect, we want the probe rate to grow more slowly at higher probe rates. This means that we should assign more packets to the higher rates to increase the number of measurements. Following this intuition, the probe rate can be designed as a two-stage model that 1) starts with linear increase, and 2) from a certain point, changes to grow in a logarithmic manner.

Technically, the probe train can be designed to search for `AB` from zero[4] to a pre-defined maximum rate $\mathcal{R}_{max}$. This maximum rate is an adjustable parameter that we can use to set a "target range". Namely, the proposed `AB` test only gives a numerical bandwidth estimation below $\mathcal{R}_{max}$. We denote the probe rate at the $i$-th packet in the train as $\mathcal{R}_i$, ($i = 1, 2, ..., \mathcal{L}$, where $\mathcal{L}$ is total length of the probe packet train). Then the packet gap between the $i$-th and $(i+1)$-th packet can be computed as $\mathcal{G}_i = \frac{\mathcal{P}}{\mathcal{R}_i}$, where $\mathcal{P}$ is the packet size which is set to MTU. Assuming the probe rate increasing step is $\Delta_i = \mathcal{R}_{i+1} - \mathcal{R}_i$, we have

$$\Delta_i = \delta \cdot min(\frac{\mathcal{G}_i}{\mathcal{G}_c}, 1) \tag{2}$$

We define a cut-off gap $\mathcal{G}_c$ as the point where the probe rate increase switches from linear to logarithmic. When the probe rate is small such that the packet gap is greater than this value ($\frac{\mathcal{G}_i}{\mathcal{G}_c} > 1$), the probe rate grows linearly with the slope $\delta$. When the probe rate reaches a certain value that makes the packet gap less than the cut-off gap ($\frac{\mathcal{G}_i}{\mathcal{G}_c} < 1$), the increasing step $\Delta$ proportionally decreases along with the packet gap $\delta \cdot \frac{\mathcal{G}_i}{\mathcal{G}_c}$. By doing this, we ensure that the increasing step becomes small when the probe rate increases. In addition, this design forces the probe rate to increase by $\delta$ for each time duration of $\mathcal{G}_c$. Overall, once $\mathcal{G}_c$, $\mathcal{L}$ and $\mathcal{R}_{max}$ are set, the $\delta$ can be computed using a search algorithm (e.g., binary search).

**Relief Gaps:** In LTE, the short-term channel fading (e.g., fast fading due to movement) may result in transient congestion effects. This short-term congestion can confound the actual

[4]Since it is impractical to generate a zero probe rate, we define the lowest probe rate as a small value (i.e., 250 Kb/s).

Table I: Packet Train Format Parameters

| | | |
|---|---|---|
| System | $\mathcal{G}_c$ | The cut-off packet gap for switching the probe rate increase from linear to logarithmic. |
| | $\mathcal{N}$ | The packet frequency to insert one relief gap. |
| | $\mathcal{G}_r$ | The width of a relief gap. |
| Custom | $\mathcal{L}$ | The total number of packets in a probe train. |
| | $\mathcal{R}_{max}$ | The maximum probe rate for searching the AB. |

congestion that results from exceeding the actual AB. While the congestion caused by transient channel fading is often resolved fairly quickly, congestion effects due to exceeding the AB should persist as long as the bandwidth is saturated. In order to mitigate the impact of short-term congestion, we intentionally insert fixed large gaps (pauses) into our probe packet train at fixed packet intervals that we call *relief gaps*. These relief gaps alleviate short-term congestion and prevent early transient issues from distorting later higher rates in the probe train.

Although there is a valid concern that adding relief gaps can potentially slow down the overall probing sequence, this is alleviated by *TBS Fluctuation*, which captures the variation of TBS, *not the average of TBS*. As long as the probe traffic creates enough queuing pressure on eNodeB before inserting a gap, the *TBS Fluctuation* will not fundamentally change while the mean of TBS may drop by adding the gaps. The setting of relief gap can be defined by $\mathcal{N}$ and $\mathcal{G}_r$. Namely, we insert a gap of $\mathcal{G}_r$ for every $\mathcal{N}$ packets.

**Packet Train Parameters:** Overall, a packet train can be described with the following parameters: $\mathcal{G}_c$, $\mathcal{L}$, $\mathcal{R}_{max}$, $\mathcal{N}$ and $\mathcal{G}_r$. We divide the parameters into two groups: *system* and *custom* parameters, as shown in Table I. The *system* parameters are set to yield optimal system-wide performance regardless of bandwidth conditions, including $\mathcal{G}_c$, $\mathcal{N}$ and $\mathcal{G}_r$. The *custom* parameters are adjustable based on the needs, and include $\mathcal{L}$ (i.e., data cost) and $\mathcal{R}_{max}$. In the next subsection, when describing implementation, we also discuss the settings of the system parameters that work the best. The custom parameters settings will be described in the later section of experimental evaluation (Section V).

### C. A Client-Free Probing System

To implement such a probe train design, it would normally be expected to require dedicated applications at both the server (sender) and client (receiver) to coordinate the packet sequences. This kind of implementation can significantly impede the ability to deploy a solution. To that end, we leverage an HTTP-based packet transmission system derived from the shared source code for [34] to achieve a simplified client implementation with sufficient measurement samples. Furthermore, we leverage a *TCP Timestamp*-based solution that in turn allows us to infer TBS statistics of the client which we describe shortly.

**System Overview:** The *CUP* system poses as a regular web server that handles regular HTTP transactions (HTTP GET) to enable simple traversal of the network. However, the server implements a customized TCP-like stack based on *libpcap* which enables network and transport layer control. The result is traffic that appears to be TCP-like but allows for the short bursts of data in the packet probes to be sent in any desired sequence and timing.

The workflow for *CUP* begins by the client initiating a test by sending an HTTP GET request to the server (i.e., visiting a URL) following a normal TCP 3-way handshake. The client can also send the probe train parameters in the HTTP GET request or tie the parameters to a particular URL (object). Upon receiving the request, the server incorporates the timing sequence in the returned HTTP response. With full control over the network and transport layers, the data packets are sent in the form of the desired probe train.

Notably, TCP ACKs, as provided by the client, are used to infer packet reception statistics at the client side. Critically, we increase the number of available ACK samples by utilizing reordering from TCP Sting [38] and expanded upon in RIPPS [34]. The simple but clever idea from TCP Sting was to swap the first and the last packet in a certain window (e.g., every 5 packets). The reordering will disable delayed ACKs by the client (typically a 2:1 data to ACK ratio), thus causing the client to send a single ACK for every received probe packet (1:1 ratio).

**Inferring TBS using TCP Timestamps:** With increased acknowledgements traversing the generally uncongested cellular uplink, we utilize the *TSval* carried in the TCP timestamp option to infer packet arrival timing at the client. As the timestamp option is enabled by default for almost all mobile phones (Android and iOS devices), timestamp-based measurement works have become quite popular [39, 40]. The modern mobile devices can provide at least a 3.3 *ms/tick* resolution[5] which reveals the packet arrival timing information at a reasonable granularity. As TBS describes the overall data size delivered in a TTI (1 $ms$), the packets in a TB usually appear to arrive at the same time (recall the rectangles indicated in Fig 3). Intuitively, we can infer the TBS using the total size of the packets whose returned TCP ACKs have the same *TSval*. Since the time span of a *TSval* is not necessarily equal to a TTI, the calculated result needs to be further divided by the timestamp resolution (e.g., 3.3 $ms/tick$ for Android phones) to compute the average transmitted data size per TTI.

To estimate the timestamp resolution of a client, we exploit the first pair of packets in the probe train to measure the value.

[5]Android (after Android 6.0.0) and iOS devices have 3.3 *ms/tick* and 1 *ms/tick* resolution, respectively.

By assigning a constant large time gap[6] (i.e., 40 *ms*) between the first sent two packets, the receiving gap is largely equal to the sending gap due to the low data rate. Thus the resolution can be computed from the difference of the *TSval* values of the first two TCP ACKs.

We conducted real-world experiments to evaluate the TBS inference accuracy from the timestamps. Overall, the inferred value tends to overestimate especially when the actual TBS is high as the coarse resolution of the timestamp can amplify the impact of aggregation. To overcome this issue, we apply a moving minimum filter (of window size 2) over the inferred series. The filtered result significantly improves the inference accuracy which helps control the estimation error within 2 Kb for all tested TBS values (ranging from 1 Kb to 30 Kb).

**Probe Packet Train System-Defined Parameter Setting:** We set $\mathcal{G}_c$ to 3.33 $ms$ as the maximum timestamp resolution for most smartphones. The setting guarantees that we can obtain at least one unique *TSVal* value for every time probe rate increases by $\delta$. For the relief gap setting, we set the gap width $\mathcal{G}_r$ to 20 $ms$ (which yields merely 500 Kb/s data rate). Under modern LTE networks with typical downlink rates greater than 10 Mb/s [37], this interval is sufficient to empty the UE's buffer at the eNodeB. For $\mathcal{N}$, we have to ensure that the amount of probe traffic is sufficient to generate at least one TBS inference value. Thus, the size of the probe traffic before each relief gap should be at least $max$(TBS) $\times$ $max$(Timestamp Resolution). If we assume that the packet size is equal to the MTU, we set $\mathcal{N} = 20$ ($\approx$ $max$(TBS) $\times$ $max$(Timestamp Resolution)$/MTU$).

## IV. A Bayesian Change Point-based Estimation Algorithm

In this section, we introduce the estimation algorithm to calculate the AB. Recalling the intuition behind the probe train design, as the sending rate increases along the train, we leverage TBS observations to capture the network state changing from uncongested to congested when we observe the increase in TBS fluctuation. The AB can then be estimated as the probe sending rate corresponding to that change point. However, due to the high variation in LTE channel characteristics and signal conditions, the TBS observations can be so noisy that a naïve threshold-based approach is inadequate to pick the right change point. To cope with the diversity and noise involved in the observations, we design a probabilistic Bayesian change-point detection algorithm (Wu et al. [41]).

**Pre-processing:** As discussed earlier, TBS fluctuations can indicate whether or not packets experienced network congestion. Therefore, given a sequence of TBS fluctuations $TF$, we can mark each $TF$ as uncongested (0) or congested (1) by comparing it with a threshold value $TH_\theta$. We define this binary expression as $BTF$. If the $i$-th TBS fluctuation $TF_i$ is less than the threshold ($TF_i < TF_\theta$) we set $BTF_i = 0$. Otherwise, $BTF_i = 1$, indicating that congestion is observed. In order to set the proper $TF_\theta$, we inspect the empirical distributions of $TF$ under uncongested vs. congested conditions. We find that $TF$ is usually below 3 $Kb$ when uncongested and above 3 $Kb$ when congestion is experienced. Thus, we set the cutoff $TH_\theta$ as 3 $Kb$.

**Model Design:** With the self-induced congestion design of probes trains, it is expected to see that the probe traffic would experience little congestion at the beginning of the train at the low sending rate. After a point where the sending rate exceeds AB, the probe traffic would have to compete against existing traffic thus leading to congestion. Given the congestion indicators $BTF$, we can search for the change point $\tau$ where $TF$ starts to change from uncongested to congested. Technically, we divide the sequence of $\{BTF_i\}$ ($i \in [1, N]$) into two segments: the first segment includes primarily zeros and the second segment contains mostly ones. To formulate this problem, we assume $BTF$ follows the *Bernoulli* distribution with parameter $p$. Particularly, $p$ takes different values in the two segments: $p_0$ in the first segment, and $p_1$ in the second segment. The change point is then located at $\tau$ ($1 \leq \tau \leq N$). Overall, the $BTF$ can be expressed as:

$$BTF \sim Bernoulli(p), where\ p = \begin{cases} p_0 & : 0 \leq i \leq \tau \\ p_1 & : \tau < i \leq N \end{cases} \tag{3}$$

By adopting the Bayesian approach, the distribution of $p_0$ and $p_1$ can be generated by the conjugated prior distribution $Beta$ with $a_0, b_0$ and $a_1, b_1$ as hyper-parameters.

$$p_0 \sim Beta(a_0, b_0),\ p_1 \sim Beta(a_1, b_1) \tag{4}$$

By assigning the proper hyper-parameters, we can apply the prior knowledge of $p_0 < p_1$ to imply that the first segment usually has the smaller mean than the second segment because it contains more zeros. For example, we can set $a_0 = 9$, $b_0 = 1$ and $a_1 = 1$, $b_1 = 9$ to convey the presumption that only 1 ($b_0$) out of 10 ($a_0 + b_0$) TBs can experience size fluctuation with no network congestion, while 9 ($b_1$) out of 10 ($a_1 + b_1$) TBs will do so under network congestion.

The goal of the model is to find a proper $\tau$ to maximize the joint probability, which is *maximum a posteriori* estimation:

$$\hat{\tau} = \underset{\tau}{\operatorname{argmax}} \iint p(BTF, \tau, p_0, p_1) dp_0 dp_1 \tag{5}$$

**Estimating $\hat{\tau}$ via Gibbs Sampling:** To solve this problem, we adopt *Gibbs sampling* to estimate the latent parameters. One of the merits of using a sampling method is that it reveals the complete posterior distributions of all the latent parameters: $p_0, p_1$ and $\hat{\tau}$. These distributions can help further assess the quality of estimation (as will be discussed soon). In Section V, we will further experimentally evaluate the performance of the inference approach.

**Interpreting Estimation Results:** Once the sampling process converges, we can look into the estimated parameters

[6]The large gap forces the *TSval* to increase by more than one.

to tease out AB. Ideally, the probe rate at the change point $\hat{\tau}$ indicates AB. However, we need to further assess the quality and confidence of such estimation by checking $p_0$ and $p_1$. We define a congestion threshold $p_\theta$ to assess if the $BTF$ segment (i.e., the first or the second) experienced network congestion. We will explore setting $p_\theta$ in Section V. Based on the values of $p_0$, $p_1$, and $p_\theta$, we have the following cases:

(i) $\boldsymbol{p_0 < p_\theta < p_1}$: This is the desired case where AB is indicated by $\hat{\tau}$.

(ii) $\boldsymbol{p_0 > p_\theta}$: This means that the change point is selected in a position where even the first segment involves congestion. So we discard the result as it portends overestimation.

(iii) $\boldsymbol{p_0 < p_\theta}$ **and** $\boldsymbol{p_1 < p_\theta}$: This means the entire probe packet train does not incur network congestion. It indicates that AB is above the pre-defined upper bound $\mathcal{R}_{max}$.

**Incorporating Loss:** To integrate another important performance impact factor – *loss*, we include the probe packet loss into estimation. Intuitively, given the estimate returned from the change point algorithm, we proportionally discount the result to the actual probe traffic delivery rate. Therefore, the final AB estimation can be expressed as *change_point_result* $\times \frac{received}{sent}$.

## V. Performance Evaluation

In this section, we conduct extensive real-world experiments to evaluate the performance of the proposed method under various network conditions. Generally, the evaluation requires the knowledge of the ground truth AB on the cellular channel at the time of the test. However, it is impractical to obtain this information, since it requires access to the internal state of base stations. As a workaround, we leverage *Achievable Throughput* (*AT*) as a reference and evaluate AB by analyzing the relationship (e.g., difference, correlation) between *AT* and AB.

We start by exploring a valid TCP variant for the reference measurement, so we can verify the effectiveness of AB estimation (Section V-A). Next, we investigate the performance of *CUP* by examining the correlation between *AT* and AB under a longitudinal test run (Section V-B). We continue with a drive test across many radio cells and locations to study performance under different mobility profiles. Finally, we conclude with an evaluation of *CUP* across varying network infrastructure equipment and examine the accuracy of *CUP* from an operational perspective.

### A. Measurement Efficacy

Since *AT* measurements can vary with different congestion control algorithms (i.e., TCP variants), a proper variant is needed to reveal bandwidth variations in LTE. In this case, we tested three TCP variants as the potential *AT* reference choices, including *Vegas*, *Westwood* and *CUBIC*. In order to validate the effectiveness of different *AT* measurements as well as

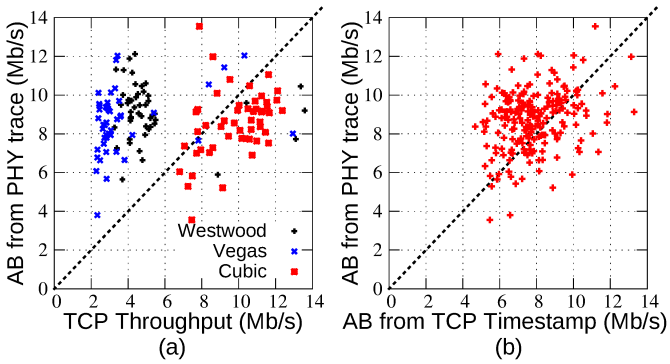


Figure 6: (a) AB (PHY Trace) vs. *AT*; (b) AB (PHY Trace) vs. AB (TCP Timestamp)

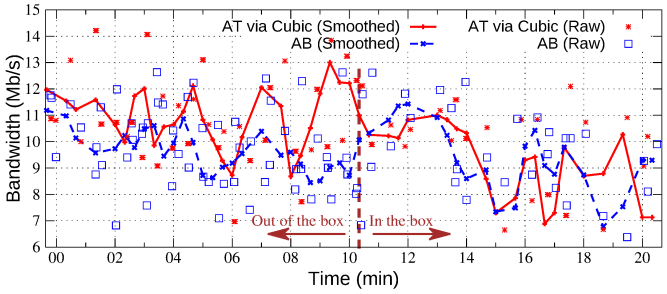


Figure 7: Time series plot of *AT* (CUBIC) and AB measurement.

our AB estimation method, we intentionally cause bandwidth variation by putting the test phone inside and outside of a metal box. The phone registered the Reference Signal Receive Power (RSRP) of $-97\ dBm$ (53%) and $-107\ dBm$ (30%) when inside an outside of the box, respectively. We run *AT* tests by downloading a 5 MB[7] file using TCP, followed by an AB test using our *CUP* approach, periodically every 10 seconds. For each TCP variant, the experiment lasts 20 minutes, with the phone spends 10 minutes each outside and inside the box. The AB probe train parameters were set to $\mathcal{L} = 400, \mathcal{R}_{max} = 30\ Mb/s$ and $p_\theta = 0.2$. We use *Mobile Insight* to capture the PHY layer trace for each AB test.

**Results and Analysis:** We first study the relationship between measured AB versus different TCP throughputs. In Fig 6, we compare the AB estimation to various TCP variants measurement. In Fig 6a, TCP throughput is on the *x-axis* and the estimated AB from the PHY trace on the *y-axis*. Each point is the average of two measurements. We see that the *AT* measured from *Westwood* and *Vegas* are consistently low. This is because the two algorithms utilize delay observations to adjust the sending rate (i.e., congestion window), which is ill-suited under dynamic LTE network conditions. Therefore, the resulting *AT* is constantly suppressed and barely reacts to the channel variation when the phone was inside vs. outside

[7]While popular "speed tests" tend to use much larger downloads, we opt for a more representative size of the actual user traffic, such as 8-10 mega-pixel photos or segments of HD adaptive video streams.

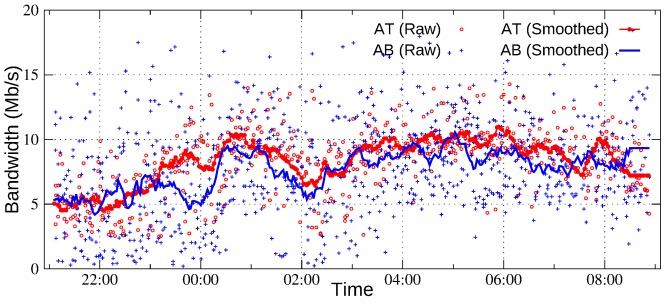


Figure 8: Time-series plot of *AT* and AB from the longitudinal run.

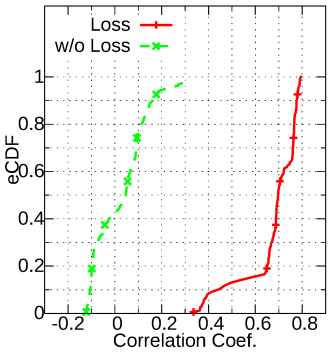


Figure 9: The empirical cumulative distribution function (eCDF) of Pearson correlation coefficient.

of the box. Fortunately, *CUBIC* shows evident variation in throughput. In Fig 7, we clearly see that *CUBIC* throughput starts to drop after the phone was put into the box. In addition, the overall *CUBIC* throughput is greater than our AB estimation, which matches our expectation according to the definitions of *AT* versus AB. Combining both Fig 6a and Fig 7, we select *CUBIC* throughput as the bandwidth reference. In further experiments, the *AT* result refers to the throughput measured using *CUBIC*.

On the other hand, in Fig 6b, we also compare the AB from PHY trace (*y-axis*) versus AB from TCP Timestamps (*x-axis*). The result shows clustered dots around the dashed line (which indicates the ideal case). It means that the TCP timestamp inference approach can yield very close estimate of what PHY trace provides, hence alleviating the need for the PHY trace collection, which usually requires extra effort to obtain such low-level information.

### B. Longitudinal Evaluation

To evaluate how well *CUP* captures bandwidth variation over the long term, we conduct longitudinal overnight experiment in a macro cell with 15 MHz carrier on the 1900 MHz band. During this experiment, we periodically run the *AT* test followed by two AB tests. Thus the throughput test serves as a reference for the following two AB tests. To explore the optimal probe train parameters, we conducted guided empirical experiments to evaluate the performance of

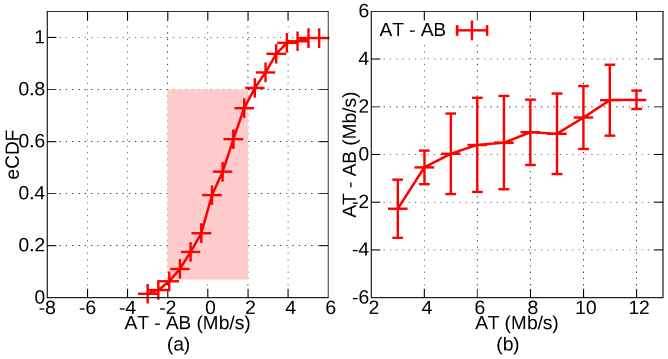


Figure 10: Absolute estimation error.

difference parameter settings, i.e., $\mathcal{R}_{max}$, $\mathcal{L}$, and $p_\theta$ (for inference algorithm). Due to the space constraints, the experiment results for searching the optimal parameters are not included. As a result, the best set up under the given LTE environment is that $\mathcal{R}_{max} = 30\ Mb/s$, $\mathcal{L} = 400$, and $p_\theta = 0.2$. The experiment lasted for 12 hours starting from 9:00 PM to 9:00 AM. In total, we perform 702 tests with 88% of them being valid.

In Fig 8, we plot the time series of results. The dots represent individual measurements and the lines are smoothed measurements over the 20-minute moving average window. We start the analysis by noting the noisiness of the individual *AT* measurements, which is a reflection of the typical cellular dynamics on this metric. We see that the curves of the smoothed *AT* and AB show a good match in the overall tendency. To further characterize the degree of the match, we calculate their *Pearson* correlation coefficient over smoothed result in a 5-hour window in Fig 9. Notably, the inclusion of probe loss dramatically improves the estimation result correlation with a median coefficient of 0.7 across per-window values. Critically, without loss, the median coefficient is only 0.03. This means that probe loss is one of the key factors to reconcile AB and *AT* measurements. Further work is likely needed to improve performance though as we will see in the next test and later in the paper, we believe that the results from an operational standpoint are more than sufficient, particularly given the extremely low cost of *CUP*.

Another aspect to consider in the estimation performance is the absolute estimation error between *AT* and AB. As shown in Fig 10(a), about 80% of AB estimates are smaller than the corresponding *AT* result, which matches our expectation of an *AT* test typically being more aggressive than an AB test. Overall, about 75% of AB tests stay within $\pm 2\ Mb/s$ of *AT* tests (the shaded area). To show how the difference is distributed across different bandwidth conditions, Fig 10(b) plots the estimation difference (*AT*- AB) vs. *AT*. As the figure shows, most of the overestimation from AB tests occurs when *AT* is very low (e.g., 2 Mb/s). This overestimation is mostly due to the different sensitivities of the two tests reacting to the loss. As TCP (especially *CUBIC*) is highly sensitive to loss, any loss can acutely dampen the *AT* test result. In addition, the

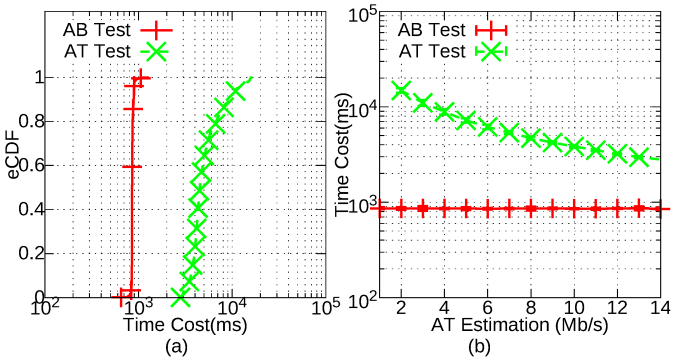


Figure 11: AB vs. *AT* time cost analysis.

*AT* test takes longer time to finish, making it more likely to encounter, potentially multiple, loss events, exacerbating the impact of loss upon *AT* test. In contrast, our AB estimation merely takes linear discount of the loss rate. However, when the *AT* test result is outside the low region, AB estimates gradually fall under the *AT* test as the impact of loss gets diluted.

**Test Cost Analysis:** We examine the test cost using two metrics, the data volume and test duration, based on the above experiments. For data cost, *AT* and AB tests in our experiments have static costs each: an *AT* test costs 5 MB and an AB test costs 540 KB ($\mathcal{L} = 400$). When using popular *AT* test tools (e.g., iperf3), the traffic cost can be much higher and proportional to the target bandwidth.

In terms of duration, the test time can vary with bandwidth conditions, especially for *AT* test. In Fig 11(a), we plot the distribution of time cost of both types of tests. We can see that *CUP*'s AB estimation generally takes less than 1 second to finish (with the median of 800 $ms$). However, the *AT* test costs at least several seconds and goes into the tens of seconds to finish. As shown in Fig 11(b), the duration of *AT* tests grows with the target bandwidth decreasing. In contrast, the duration of the proposed AB test is consistent over different bandwidth conditions. Benefiting from being lightweight and non-intrusive, the AB test is a great choice for longitudinal network monitoring.

### C. Drive Test Evaluation

To evaluate *CUP* under dynamic conditions, we conducted a drive test through the downtown and suburban areas of the town in the US Midwest. Similar to the longitudinal experiment, the phone was set to run the AB and *AT* tests periodically in a back-to-back manner. The GPS trace along with cell information provided from *Mobile Insight* were recorded. The test covered over 21 km of distance, including 9 stops (each lasted for 20 minutes) to capture measurements while stationary. Overall, the experiment lasted for 5 hours and generated 1,360 test runs across 38 radio cells. Fig 12 plots the drive route and illustrates cell information, including *cell ID*, labeled by color, and Reference Signal Received Quality *(RSRQ)*, indicated by the circle radius. The radio cells belong

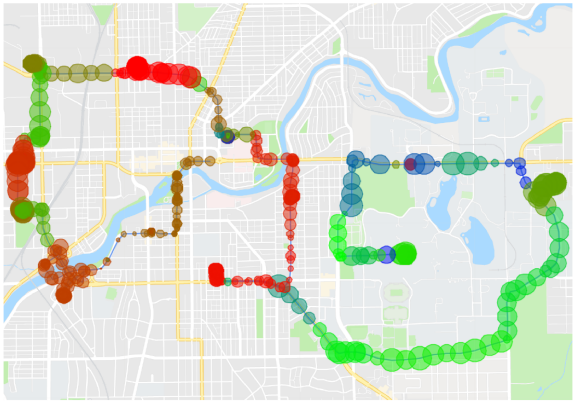

Figure 12: Drive test route map. Colors denote different cells and the radius represents RSRQ.

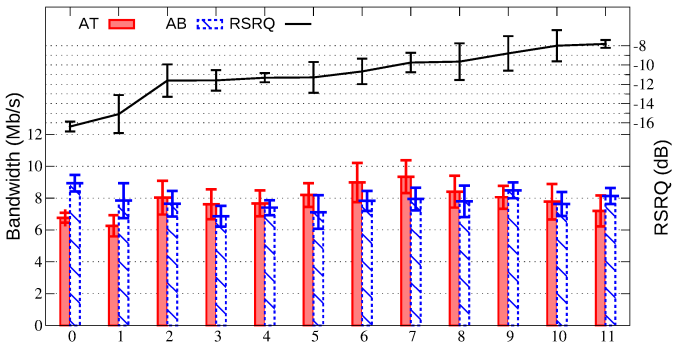


Figure 13: *AT* and AB under different cells. RSRQ of each cell is also plotted.

to 12 eNodeBs, with 8 eNodeBs hosting 26 conventional macro cells, and 4 eNodeBs hosting 12 cells using outdoor DAS. We observed 4 frequency bands across macro cells, and 3 frequency bands across DAS cells.

**Cell-based Analysis:** By categorizing the test results by the cell ID, we plot the measured *AT* and AB in Fig 13. We use cells with the large number of stationary test samples (10 macro and 2 DAS cells), with similar number of two carriers in use, and only cell 11 using a third carrier. In addition, we also plot the average RSRQ observed per cell, and order the cells by RSRQ values. Overall, we see that the estimated AB is close to *AT* and mostly below *AT*. However, when the channel quality is significantly poor (RSRQ $<$ -14 dB), the AB is noticeably higher than *AT* under *Cell 0* and *Cell 1*. The reason is that since *AT* test takes a long time to finish, the poor signal quality can consistently dampen throughput. As the AB test is lightweight, the eNodeB may satisfy the short-term high bandwidth demand even under poor channel conditions [5]. This can lead to occasional over-estimations from the AB test. Continuously performing multiple AB tests is a practical solution to resolve this discrepancy due to the lower cost of AB as shown earlier. Due to space constraints, the impact of speed (stationary, low speed, high speed) is not included.

## VI. Conclusion

In conclusion, we presented a novel mechanism, CUP, which offers the benefits of AB estimation techniques with reasonable accuracy for cellular network measurement. We designed an intelligent packet train that leverages TBS fluctuation as an indicator of congestion, thereby offering a lightweight, accurate mechanism for AB estimation. *CUP* enables more efficient longitudinal characterization by being extremely light-weight, while requiring no changes to the client. We demonstrate several real-world scenarios and believe that *CUP* could prove to be a useful tool for network operators and users to assess ongoing performance. Future work includes exploring UE scaling within a given cell, consistency of monitoring at the edge of a cell, and uplink measurement.

## References


[1] M. Agiwal, A. Roy, and N. Saxena, "Next Generation 5G Wireless Networks: A Comprehensive Survey," *IEEE Communications Surveys Tutorials*, vol. 18, no. 3, pp. 1617–1655, 2016.

[2] Ookla, "Speed test," http://www.speedtest.net.

[3] (2018) iperf3. [Online]. Available: http://software.es.net/iperf

[4] J. Huang, C. Chen, Y. Pei, Z. Wang, Z. Qian, F. Qian, B. Tiwana, Q. Xu, Z. Mao, M. Zhang *et al.*, "MobiPerf: Mobile network measurement system," *Tech. Report. Univ. of Michigan and Microsoft Research*, 2011.

[5] C. E. Andrade, S. D. Byers, V. Gopalakrishnan, E. Halepovic, M. Majmundar, D. J. Poole, L. K. Tran, and C. T. Volinsky, "Managing Massive Firmware-Over-The-Air Updates for Connected Cars in Cellular Networks," in *CarSys '17*, 2017, pp. 65–72.

[6] V. J. Ribeiro, J. Navratil, R. H. Riedi, R. G. Baraniuk, and L. Cottrell, "PathChirp: Efficient available bandwidth estimation for network paths," no. SLAC-PUB-9732, 2003.

[7] P. Steenkiste, "Evaluation and characterization of available bandwidth probing techniques," *IEEE JSAC*, vol. 21, no. 6, pp. 879–894, 2003.

[8] M. Li, M. Claypool, and R. Kinicki, "WBest: A bandwidth estimation tool for IEEE 802.11 wireless networks," in *LCN'08*, 2008, pp. 374–381.

[9] B. Melander, M. Bjorkman, and P. Gunningberg, "A new end-to-end probing and analysis method for estimating bandwidth bottlenecks," in *Globecom '00*, vol. 1. IEEE, 2000, pp. 415–420.

[10] L. Song and A. Striegel, "Leveraging frame aggregation for estimating WiFi available bandwidth," in *SECON'17 .IEEE*, June 2017, pp. 1–9.

[11] A. Farshad, M. Lee, M. K. Marina, and F. Garcia, "On the impact of 802.11n frame aggregation on end-to-end available bandwidth estimation," in *SECON'14*. IEEE, Jun. 2014, pp. 108–116.

[12] V. Adarsh, M. Nekrasov, E. Zegura, and E. Belding, "Packet-Level Overload Estimation in LTE Networks Using Passive Measurements," in *IMC'19*. ACM, 2019.

[13] J. Huang, F. Qian, Y. Guo, Y. Zhou, Q. Xu, Z. M. Mao, S. Sen, and O. Spatscheck, "An in-depth study of LTE," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4, pp. 363–374, 2013.

[14] T. Pögel, J. Lübbe, and L. Wolf, "Passive client-based bandwidth and latency measurements in cellular networks," in *Proceedings IEEE INFOCOM'12 Workshops*, March 2012, pp. 37–42.

[15] A. Chakraborty, V. Navda, V. N. Padmanabhan, and R. Ramjee, "Coordinating cellular background transfers using Loadsense," in *Proc. of ACM MobiCom'13*, 2013, p. 63.

[16] F. Michelinakis, N. Bui, G. Fioravantti, J. Widmer, F. Kaup, and D. Hausheer, "Lightweight mobile bandwidth availability measurement," in *IFIP Networking*. IEEE, 2015, pp. 1–9.

[17] X. Xie, X. Zhang, S. Kumar, and L. E. Li, "Pistream: Physical layer informed adaptive video streaming over LTE," in *Proc. of ACM MobiCom '15*, 2015, pp. 413–425.

[18] X. Xie, X. Zhang, and S. Zhu, "Accelerating mobile web loading using cellular link information," in *Proc. of MobiSys'17*, 2017, pp. 427–439.

[19] D. Perdices, D. Muelas, I. Prieto, L. de Pedro, and J. E. López de Vergara, "On the modeling of multi-point RTT passive measurements for network delay monitoring," *IEEE Transactions on Network and Service Management*, vol. 16, no. 3, pp. 1157–1169, 2019.

[20] T. Oshiba, K. Nogami, K. Nihei, and K. Satoda, "Robust available bandwidth estimation against dynamic behavior of packet scheduler in operational LTE networks," in *Proc. of ISCC*, 2016, pp. 1276–1283.

[21] A. K. Paul, A. Tachibana, and T. Hasegawa, "An Enhanced Available Bandwidth Estimation Technique for an End-to-End Network Path," *IEEE TNSM'16*, vol. 13, no. 4, pp. 768–781, 2016.

[22] M. Rindler, P. Svoboda, and M. Rupp, "FLARP, fast lightweight available rate probing: Benchmarking mobile broadband networks," in *ICC*. IEEE, 2017, pp. 1–7.

[23] C. Midoglu, L. Wimmer, A. Lutu, Ö. Alay, and C. Griwodz, "A closer look into mobile network speed measurements," *ArXiv*, 2017.

[24] P. Torres, P. Marques, H. Marques, R. Dionísio, T. Alves, L. Pereira, and J. Ribeiro, "Data analytics for forecasting cell congestion on LTE networks," in *2017 Network Traffic Measurement and Analysis Conference (TMA)*, 2017, pp. 1–6.

[25] J. Strauss, D. Katabi, and F. Kaashoek, "A measurement study of available bandwidth estimation tools," in *IMC*, 2003, p. 39.

[26] J. Navratil and R. L. Cottrell, "ABwE: A Practical Approach to Available Bandwidth Estimation," in *Proc. of PAM'03*.

[27] M. Jain and C. Dovrolis, "Pathload: A measurement tool for end-to-end available bandwidth," in *In Proc. of PAM*, 2002, pp. 14–25.

[28] A. K. Paul, A. Tachibana, and T. Hasegawa, "NEXT-FIT: Available Bandwidth Measurement over 4G/LTE Networks – A Curve-Fitting Approach," in *Proc. of AINA*. IEEE, 2016, pp. 25–32.

[29] S. Shiobara and T. Okamawari, "A novel available bandwidth estimation method for mobile networks using a train of packet groups," in *Proceedings of the 11th International Conference on Ubiquitous Information Management and Communication*, 2017, pp. 1–7.

[30] K. Winstein, A. Sivaraman, and H. Balakrishnan, "Stochastic forecasts achieve high throughput and low delay over cellular networks," in *Proc. NSDI*. USENIX, 2013, pp. 459–471.

[31] Y. Zaki, T. Pötsch, J. Chen, L. Subramanian, and C. Görg, "Adaptive congestion control for unpredictable cellular networks," in *Proc. of SIGCOMM*, 2015, pp. 509–522.

[32] N. Baranasuriya, V. Navda, V. N. Padmanabhan, and S. Gilbert, "QProbe: Locating the bottleneck in cellular communication," in *Proc. of CoNext*, 2015, pp. 33:1–33:7.

[33] F. Capozzi, G. Piro, L. Grieco, G. Boggia, and P. Camarda, "Downlink Packet Scheduling in LTE Cellular Networks: Key Design Issues and a Survey," *IEEE COMST*, vol. 15, no. 2, pp. 678–700, 2013.

[34] C. D. Mano, A. Blaich, Q. Liao, Y. Jiang, D. A. Cieslak, D. C. Salyers, and A. Striegel, "RIPPS: Rogue identifying packet payload slicer detecting unauthorized wireless hosts through network traffic conditioning," *ACM Trans. Inf. Syst. Secur.*, vol. 11, no. 2, pp. 2:1–2:23, May 2008.

[35] Y. Li, C. Peng, Z. Yuan, J. Li, H. Deng, and T. Wang, "MobileInsight: Extracting and analyzing cellular network information on smartphones," in *Proc. of MobiCom'16*, 2016, pp. 202–215.

[36] Y. Xu, Z. Wang, W. K. Leong, and B. Leong, "An end-to-end measurement study of modern cellular data networks," in *Proc. of PAM 2014*, 2014, pp. 34–45.

[37] 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 *Proc. of ACM MobiSys'*, 2012, pp. 225 – 238.

[38] S. Savage, "Sting: A TCP-based Network Measurement Tool," in *USENIX Symposium on Internet Technologies and Systems*, vol. 2, 1999.

[39] E. Halepovic, J. Pang, and O. Spatscheck, "Can you GET me now?: Estimating the Time-To-First-Byte of HTTP transactions with passive measurements," in *Proc. of IMC'12*, 2012, pp. 115–122.

[40] W. K. Leong, Z. Wang, and B. Leong, "TCP Congestion Control Beyond Bandwidth-Delay Product for Mobile Cellular Networks," in *Proc. of CoNEXT '*, 2017, pp. 167–179.

[41] X. Wu, Y. Dong, C. Huang, J. Xu, D. Wang, and N. V. Chawla, "UAPD: Predicting urban anomalies from spatial-temporal data," in *Machine Learning and Knowledge Discovery in Databases*, 2017, pp. 622–638.