Available online at www.sciencedirect.com
ScienceDirect
Pervasive and Mobile Computing 00 (2018) 1–21
Pervasive and Mobile Computing

Fine-grained LTE Radio Link Estimation for Mobile Phones

Nicola Bui[a,*], Foivos Michelinakis[b,c,d], Joerg Widmer[b]

[a]Northeastern University, Boston
[b]IMDEA Networks Institute
[c]Universidad Carlos III de Madrid
[d]Simula Research Laboratory

Abstract

Recently, spectrum optimization solutions require mobile phones to obtain precise, accurate and fine-grained estimates of the radio link data rate. In particular, the effectiveness of anticipatory schemes depends on the granularity of these measurements. In this paper we use a reliable LTE control channel sniffer (OWL) to extensively compare mobile phone measurements against exact LTE radio link data rates. We also provide a detailed study of latencies measured on mobile phones, the sniffer, and a server to which the phone is connected. In this study, we show that mobile phones can accurately (if slightly biased) estimate the physical radio link data rate. We highlight the differences among measurements obtained using different mobile phones, communication technologies and protocols. We also provide detailed instructions on how to replicate our measurements and describe alternative measurement setups and their tradeoffs.

Keywords: LTE, mobile phone measurements, sniffer, radio link

1. Introduction

Can we trust mobile phone data rate measurements? This apparently trivial question is key to evaluate the feasibility of the anticipatory networking [1] paradigm and the related future network solutions [2, 3]. For instance, exploiting achievable rate prediction to optimize mobile applications [4, 5, 6] requires some information exchange between mobiles and base stations so that current decisions (e.g. scheduling, admission control) can be made taking into account the future states of the system. However, while prediction errors have been studied [7, 8], the capability of mobile phones to obtain accurate measurements has never been investigated in mobile networks.

In addition to that, many recent studies [9, 10, 11, 12, 13] rely on crowd-sourced datasets to derive their conclusions without questioning mobile phone measurements accuracy and whether it is possible to aggregate them. Although reliable mobile phone applications to measure the network bandwidth exist [14, 15, 16], they focus on end-to-end measurements that do not provide the required level of granularity

*Corresponding author
Email addresses: n.bui@northeastern.edu (Nicola Bui), foivos.michelinakis@imdea.org (Foivos Michelinakis), joerg.widmer@imdea.org (Joerg Widmer)

to enable anticipatory optimization. In fact, while end-to-end data rate is ideal to optimize TCP performance, the resource allocation optimization would rather benefit from the actual radio link data rate between eNodeB and user equipment (UE).

In this paper, we study whether mobile phones can accurately measure LTE radio link data rate and with which granularity (i.e. sampling frequency). To achieve this, we compare the data rate estimates computed at the physical layer of the radio link through a sniffer, at the mobile phone kernel through tcpdump and by a mobile application.

Our study is divided into two measurement campaigns: the first and largest set of experiments consists of burst transmissions, where a small amount of data is sent back-to-back to collect data rate estimates computed by the different entities (i.e., phone, sniffer and server), while in the second set, we evaluate latencies between single data packet transmissions and their corresponding acknowledgements (ACKs). These latencies allow us to study the root-causes of differences among the behaviors of different phones. In all the tests, we compared three mobile phones by different vendors and equipped with different chipsets, first performing the test from the server to the phone and, then, in the opposite direction.

The main findings of our study are the following:

1. Mobile phones achieve accurate (> 85%) and precise (> 82%) data rate measurements with as few as 20 KB in the downlink, where accuracy and precision are related to how close the measurement are to the sniffer ground-truth readings.
2. Uplink measurements are less accurate and less precise (65% and 60% respectively in the worst case), because LTE uplink scheduling delay causes a higher variability in the results.
3. Different chipsets exhibit variable biases and performance, thus requiring dedicated calibration to optimize accuracy.
4. Downlink accuracy and precision are linked to the latency measured on the phone: chipsets providing shorter and more deterministic latencies obtain better estimates.

Further, we provide a thorough guide on how to perform similar measurements. We present the state of the art of tools that can be used to assess the performance of LTE components, such as eNodeBs and UEs, as well as discuss the tradeoffs of each approach.

The rest of the paper provides a survey of related work in Section 2, specifies the measurement setup and the devices involved in Section 3, and discusses the two measurement campaigns in Sections 4 and 5. Section 6 summarizes the main findings, while Section 7 specifies how to repeat our measurements and discusses a few setup alternatives. Finally, Section 8 concludes the paper.

2. Related work

A considerable number of recent papers focus on LTE measurements and measurement techniques, but, to the best of our knowledge, none of them rely on accurate LTE scheduling information to validate their findings. Among them, Huang et al. [10] studied LTE performance measured from mobile phone data. In order to obtain a known reference for the results, the authors performed experiments using controlled traffic patterns to validate their findings.

The fraction of LTE resources used for communication is detected in [17] by means of power measurements. The goal of the authors is to evaluate the performance of M2M communications using experimental data. Similarly, RMon [18] is a solution to assess which resource blocks are used by comparing the average power measured over the resource bandwidth with that of the closest LTE reference signals. RMon achieves good performance and robustness, but it can only assess the average fraction of used resources. Hence, it cannot be used to capture the actual base station data rate.

LTEye [19] was the first attempt to decode the LTE control channel to access scheduling information. However the authors found in their later work [18] that LTEye could not provide sufficient reliability and a significant fraction of control messages remain undecoded. To overcome this limitation, we developed a reliable LTE control channel sniffer, called Online Watcher for LTE (OWL) [20]. In our tests, OWL successfully decoded 99.85% of the LTE control messages, thus obtaining a complete log of the eNodeB scheduling. MobileInsight [21] is a mobile phone application capable of accessing LTE control messages directly from the radio chipset and could also be seen as an alternative to OWL. Moreover, a few complete open-source platforms exist to realize an LTE system. Among them, the most advanced solutions are offered by OpenAirInterface [22] and srsLTE [23].

A few papers [24, 25, 26] use commercial tools and/or operator network information to evaluate LTE performance. For instance, direct access to network logs is used in [24] to provide a detailed comparison between LTE and UMTS, but since network logs are usually always anonymized (if they are released at all), it is impossible to identify a given device under test among the set of traces. A commercial LTE modem from Teldat is used in [25] to measure application level performance, but this only provides information related to the modem itself. It does not provide resource allocation information about other mobile devices in the same cell. Similarly, a network maintenance tool is used in [26] to analyze quality of service, but such tools only provide compound information averaged over periods of time, which do not achieve the required granularity for our measurements. The vast majority of papers however, just rely on measurement performed using mobile phones or replicated in laboratory experiments. Phone traces are used in [12] to evaluate network performance. The same authors developed a framework [13] to manage mobile phone measurements and a similar project was developed in [9]. In [11], LTE performance predictors are evaluated in laboratory setups. In addition, [27] uses TCPdump traces to perform energy efficiency evaluation of smartphones and [28] studies LTE shared access in a trial environment.

Finally, although not specifically developed for LTE, the following contributions discuss mobile measurements in general terms. The most popular approach is Ookla's Speedtest [14], which can provide a very accurate evaluation of the steady-state rate achievable by long-lived TCP connections. However, Speedtest is both data intensive (with fast connections, one test can consume more than a few tens of megabytes) and cannot provide estimates at the granularity required in this study. A few recent papers [15, 16] studied end-to-end achievable throughput, also accounting for inter-arrival times and passive monitoring techniques, but without comparing their findings with ground-truth readings. The accuracy of WiFi measurements performed by mobile phone is studied in [29] based on a timing analysis. However, their results cannot be applied to our scenario for two reasons: WiFi and LTE differs significantly in terms of scheduling and MAC protocols, and tcpdump traces do not provide a reliable ground truth for the physical radio link.

This study improves over the current state of the art by, first, evaluating data rate estimates on the radio link, instead of end-to-end throughput and, second, by relying on an accurate LTE sniffer to obtain a ground truth of the measurements.

3. Setup and Definitions

Figure 1 illustrates our experimental setup, which consists of five entities. The **target UE** is the mobile device under test which is connected to the **target eNodeB**. The **sniffer** is a BladeRF x40 software defined radio [30] that samples and records the LTE signal to be decoded by OWL. The sniffer is shown as connected to the eNodeB-UE link only, but it actually records and decodes all control messages sent by the eNodeB and, thus, it is aware of all of the traffic exchanged in the cell. The **server** is a PC in our local network configured with a public IP address in order to be reachable by the target UE. The Internet cloud in the figure groups all the links that form the backhaul of our setup including the operator network. Finally, the

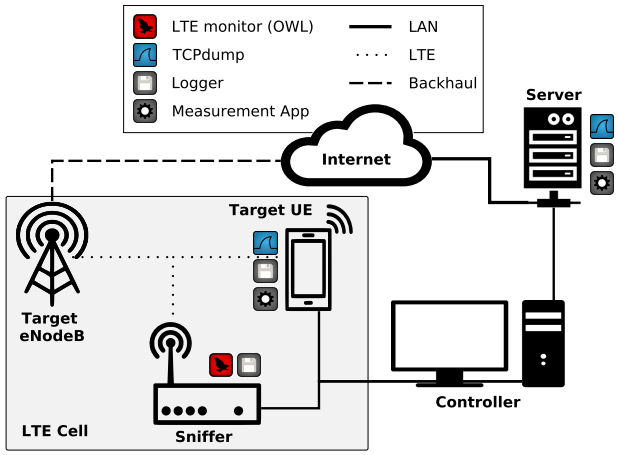


Figure 1. Experiment setup showing devices, connections and software.

controller is a second PC in our local network which is directly connected to the target UE and the sniffer via USB and to the server via Ethernet.

In order to assess the impact of different hardware, we choose three mobile phones from different vendors with comparable technical specifications but equipped with different chipsets. In particular, we opt for a Motorola MotoG LTE [31], a Huawei P8 Lite [32] and a ZTE Blade A452 [33] equipped with Qualcomm, Huawei and MediaTek chipsets, respectively. The following list summarizes the features relevant for this study (the short names used in the rest of the paper are written in bold face):

- Motorola **MotoG** 4G (2014) – Chipset: Qualcomm Snapdragon 400 MSM8926; CPU: ARM Cortex-A7, 1200 MHz (4 cores); Android: 4.4.2 KitKat; RAM: 1 GB.
- **Huawei** P8 lite (2015) – Chipset: Huawei HiSilicon KIRIN 620; CPU: ARM Cortex-A53, 1200 MHz (8 cores); Android: 5.0.2 Lollipop; RAM: 2 GB.
- **ZTE** Blade A452 (2015) – Chipset: MediaTek MT6735P; CPU: ARM Cortex-A53, 1000 MHz (4 cores); Android: 5.1 Lollipop; RAM: 1GB.

We have four different software modules in our setup. The gear-shaped icon refers to the **Measurement App**, which controls the communication between the target UE and the server. For every successful socket call (either "send" or "receive"), it logs the time and the amount of data exchanged. This application is implemented in Python to obtain the same behavior both on the phone and the server. The shark-fin-shaped icon refers to **TCPdump** [34], which we use both on the UE and the server to obtain transmission timestamps at the kernel level as well as the payload size. The floppy-disk shaped icon illustrates the **Logger** application that formats the output of the other tools for later analysis.

The **LTE monitor** (owl-shaped icon) implements our Online Watcher for LTE (OWL [20]) control channel measurements. OWL is built starting from srs-LTE [23], an open-source implementation of LTE, and extends its functionalities to provide a reliable decoder of the physical control channel. From LTE control messages, OWL computes the transport block size assigned to each downlink and uplink communication. In this way, we can measure the actual LTE radio link data rate in every transmission time interval (TTI), i.e. 1 ms. This data rate differs from the usual notion of end-to-end throughput and it is the main metric needed for anticipatory networking.

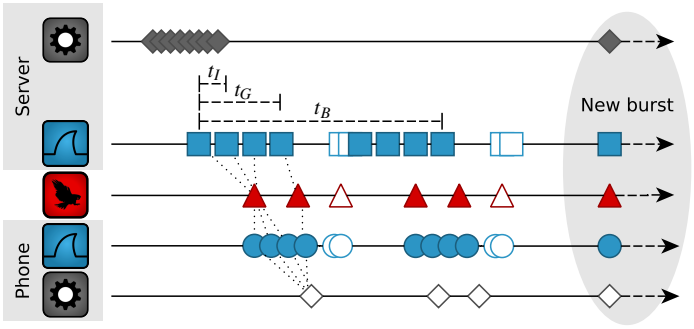


Figure 2. Communication diagram for downlink burst transmissions.

The LTE cell used during the tests belongs to Yoigo, a Spanish mobile network operator, and operates in LTE band 3 (1800 MHz) using a bandwidth of 10 MHz. The cell is chosen due to the relatively low load and the very good signal quality from the test location. Since it is not possible to retrieve the radio network temporary identifier RNTI from the mobile phone itself, we design our tests to have a highly distinct pattern, such as fixed burst size and periodicity. Also, we make sure that the logger application on the three devices captures the first event of each test for an easier mapping of events related to the same transmission. Our measurements do not require the different measurement points to be synchronized, since we just monitor event durations and not absolute time.

Our setup is characterized by three physical and five logical measurement points: we monitor the communications at the target UE, at the sniffer and at the server. Both the UE and the server collect information by means of TCPdump and at the application to capture the difference between application and kernel measurements by means of a data rate estimation technique using packet train dispersion [35].

As introduced above, we perform two measurement campaigns, the first dealing with burst transmission (see Section 4.1 and 4.2 for the test description and the results respectively) and the second with periodic isolated transmissions (Section 5.1 and 5.2). In the first campaign our goal is to evaluate the accuracy and the precision of fine-grained measurements, while in the second we study latencies in the different devices. Both campaigns consider both downlink (from the server to the UE) and uplink communication.

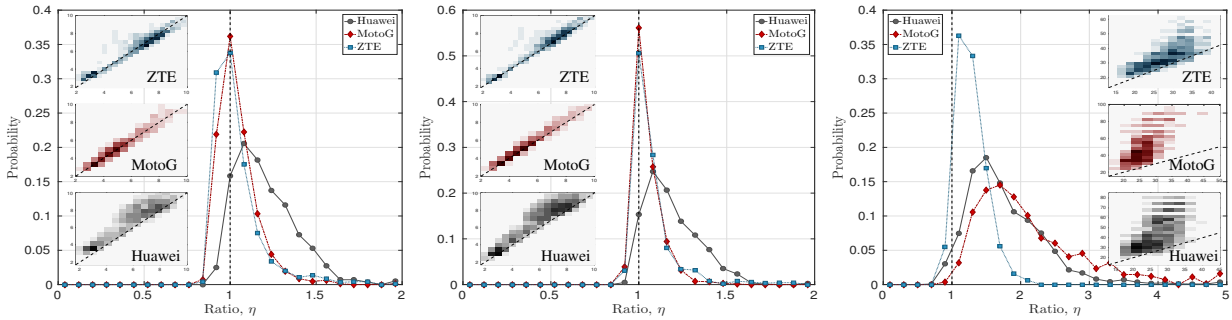
4. Burst Transmissions

The first measurement campaign has the main objectives of evaluating the accuracy and the precision of data rate estimates obtained by mobile applications, and to analyze the differences in performance obtained by the three phones.

We use the following symbols: t, s, n and r denote durations, transmission sizes, number of packets, and the data rates. All these quantities are easy to compute from the information available in our tests and they do not require complex filtering. In fact, we just evaluate the data rate $r = s/t$ as the ratio between the amount of data s transmitted in a given time and the time t itself.

4.1. Experiment Description

We focus on packet trains (burst) from when they are first sent back-to-back from an application to their reception at the other endpoint. In particular, we are interested in comparing transmissions in the LTE radio link and the events tracked by a mobile phone at the application and the kernel level. We use Figure 2 as an



(a) Application–bursts (b) Kernel–bursts (c) Application–groups

Figure 3. Comparison of the estimator ratios computed on burst by the application (a) and the kernel (b) and on groups (c). The small plots on the left show estimator densities: the x-axis is the cell ground truth and the y-axis the estimate.

example of a downlink test. The packets are generated by the application almost at the same time. As they are sent through a TCP socket they become spaced according to TCP dynamics and delays. For all layers, empty markers represent ACKs, except for the phone application layer where they mark packet receptions.

For the analysis, we define interarrival time t_I as the interval between two consecutive arrivals on the same layer and burst time t_B as the time between the first and the last packet of a train. LTE may impose a further grouping of packets when large transport blocks can fit more than a single TCP packet; this is observed at the phone as a group of packets arriving almost at the same time and as a single event at the sniffer. We define group time t_G as the time elapsed between the first and the last packet of a series of continuous arrivals. The data rate computed on groups is the measure that approaches the most the physical rate. In what follows groups are identified by those packets whose interarrival times are shorter than a threshold $t_I \leq \tau \leq \tau_C$, where $\tau_C = 1$ ms is the TTI of LTE.

In Figure 2 we see a burst of packets sent by the application, which is split into two groups in the eNodeB and the phone. Also, passing from the server to the eNodeB, multiple packets can fit the same LTE transmission, the size of which depends on the signal quality and the number of resource blocks assigned to the user. After being decapsulated by the phone kernel, packets are delivered to the phone application as streams of data of variable sizes.

To compare LTE with phone and server traces, we fix the burst size to 100 and 30 KB for downlink and uplink experiments, respectively, to obtain at least 10 transmissions per burst: in our setup with a 10 MHz channel, the maximum LTE transport block size is 73392 and 28336 [36] bits in downlink and uplink, respectively.

4.2. Experiment Results

In this section we compare data rates measurements by means of an estimator ratio defined as $\eta = r/r_0$, where r_0 is the reference data rate, which, if not otherwise specified, is measured by the LTE sniffer. The estimators' accuracy is highest when the the ratio is $\eta = 1$ and degrades if it is either higher (overestimation) or lower (underestimation). Moreover, the standard deviation of the ratio is proportional to the estimator precision. Thus, we show the distribution of the estimators' ratios and we provide accuracy $\alpha = [1 - |1 - \overline{\eta}|]_0$ and precision $\rho = 1 - \sigma(\eta)$, where $\overline{x}$, $|x|$, $\sigma(x)$ are the empirical average, the absolute value and the standard deviation of x and $[x]_0$ is x if $x > 0$ and 0 otherwise. In what follow the overheads between the application and the kernel (about 3.95%) and between the kernel and the sniffer (about 0.8%) are compensated.

The first and foremost results of our study are illustrated by Figure 3, which shows the empirical probability density function (epdf) of the estimator ratios obtained using burst by the three different phones

computed by the application (on the left) and by the kernel (on the center) and using groups computed by the kernel (on the right). The small plots on the left of the figures show the density of the estimators in a reference system where the x-axis reports the cell ground truth and the y-axis the estimate: the darker the color the more estimators are in that area. The black dashed lines show what an ideal estimator would achieve: the ratio distribution would be a single spike in 1 and the densities would be on the line $x = y$.

R1 – Phone applications can obtain accurate and precise downlink data rates measurements: Figure 3(a) demonstrates that the peaks of the epdfs are very close to 1. The three phones achieves accuracy of $\alpha = 85\%$ (Huawei), $\alpha = 96\%$ (MotoG), and $\alpha = 95\%$ (ZTE). The width of the edpfs is related to the estimators' precision, in particular, the three phones have $\rho = 89\%$ (MotoG), $\rho = 85\%$ (ZTE), and $\rho = 82\%$ (Huawei). The precision is also related to the width of the estimator clouds in the small plots – wider clouds corresponds to the lower precision.

R2 – Different phones have different biases: the slightly lower score of the Huawei phone means that it tends to overestimate the data rate by about a 10%, which can be easily compensated. The same results can be verified in the small plots: the MotoG's and ZTE's densities are centered on the $x = y$ line, while the Huawei's is slightly above. Since the estimators are obtained as size-over-time ratios and the bursts have fixed size, the root cause for accuracy and precision has to be looked for in the variability of the burst duration. In particular, if a phone consistently measures shorter burst times, it will overestimate the rate and, if the time measurements are variable (e.g., random delays due to different loads on the CPU) the corresponding precision will be lower. Thus, systematic errors impact the accuracy, while random errors affect the precision of the estimates. As a consequence, it is important to compensate for the biases of the different phones when dealing with crowd-sourced measurements, to avoid unpredictable errors.

R3 – Accuracy and precision are independent of the actual data rates: examining the small plots in Figure 3(a) the estimators span the whole x and y axes between 2 and 10 Mbps. This means that, during the experiments, the network load varied so that the actual data rate achievable by our target phones (all of them show similar behavior) was changing. In addition, the actual data rate of the experiment does not affect the estimator quality. The slightly larger cloud at higher rates is expected since the same percentage error causes a larger absolute error at higher rates.

R4 – Kernel measurements are slightly more precise: Figure 3(b) shows the estimator ratios' epdfs when the measurements are performed by tcpdump. We expect these measurements to show better performance than those obtained from the application, since they are collected by tcpdump, they are time stamped when the kernel receives packets (through an interrupt from the chipset). However, not only are the precision improvements very small (i.e., 3%, 1% and 1% for MotoG, Huawei and ZTE respectively), but the accuracy scores are almost unchanged.

Figure 3(c) is obtained using groups instead of the longer burst. Since the transmissions within a group are expected to belong to consecutive radio link layer transmissions, the data rate estimate is likely to capture the exact rate used by the eNodeB. However, these measurements are more sensitive to timing precision. Since the group threshold for the cell is 1 ms, groups are characterized by transmissions in every TTI and, as soon as a single TTI is skipped, the group ends. Thus, if timing is not precise, two or more separate groups in the cell can be detected as a single group at the kernel or by the phone application. As a consequence, while it is possible to easily separate bursts and have a unique mapping between bursts in the different layers, this is not true for groups, whose composition is device and layer dependent.

R5 – Group-based data rate estimators are imprecise and biased: our measurements show that the accuracy (α) and the precision (ρ) of data rate estimators computed on groups are low. In particular, MotoG achieves $\alpha = 16\%$ and $\rho = 46\%$, Huawei $\alpha = 25\%$ and $\rho = 48\%$ and ZTE $\alpha = 75\%$ and $\rho = 79\%$. Due to the timing precision sensitivity of group-based measurements, the results obtained are in line with our

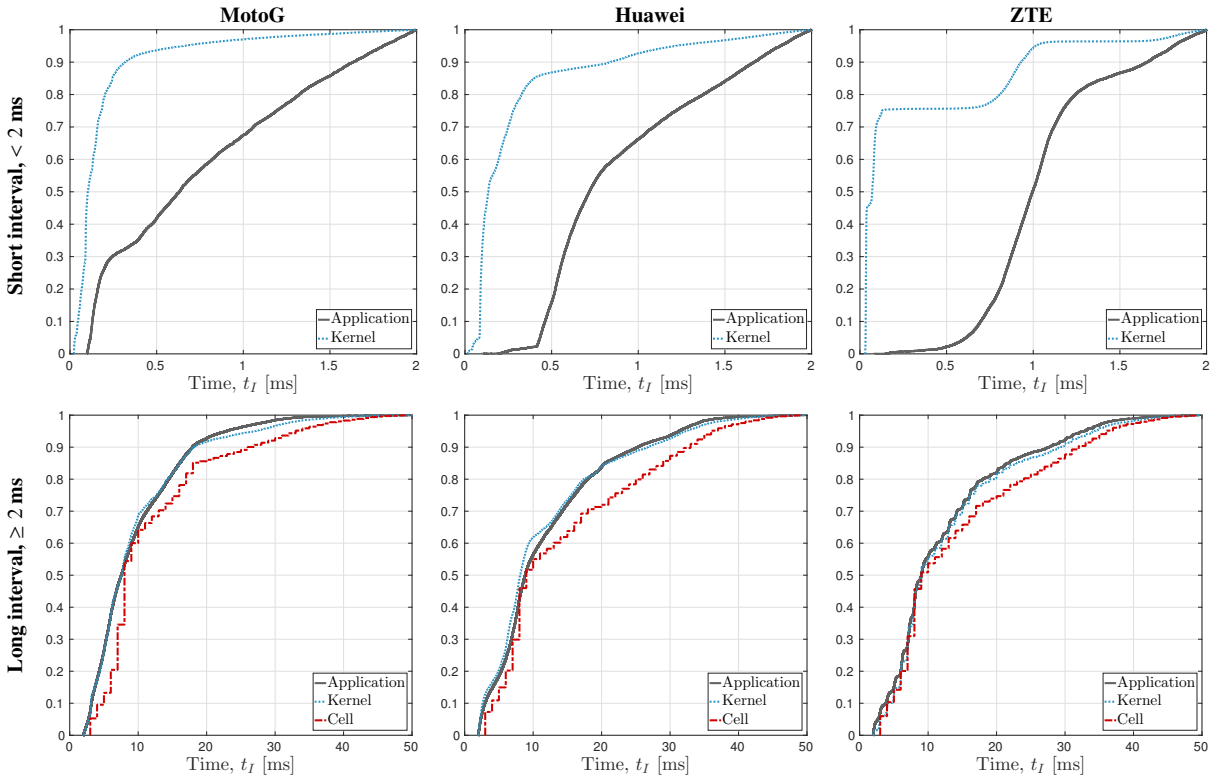


Figure 4. Interarrival time CDFs for short (left) and long (right) intervals and the three phones.

expectations. Instead, the ZTE phone measurements, even though overestimating by circa 25%, maintains a quite acceptable precision, close to 80%.

A close examination of the density plots reveal that group-based measurements have to estimate higher and more variable data rates, because they are not averaged over the longer duration of bursts. The ground truth data rate (x-axis) extends up to 45 Mbps, which is close to the maximum throughput in our setup. Moreover, all the phones overestimate the actual readings, reaching estimates even higher than the maximum reachable of our setup. This is a further proof of the importance of precise timing and deterministic latency in the phones to enable the most fine-grained estimation.

Before moving to uplink results, a few considerations on the packet interarrival times are in order. Figure 4 provides a set of graphs showing the Cumulative Distribution Function (CDF) of the interarrival times. To emphasizes the difference between the interarrival times of packets related to the same LTE transmission from those related to intervals separating continuous arrivals, we plot on the left column the CDFs for the interarrival times shorter than 2 ms (2 TTIs) and, those longer or equal to 2 ms and shorter than 50 ms on the right column. We don't show interarrival times longer than 50 ms, since those are almost always related to inter-burst rather than intra-burst arrivals. The matrix rows show from the top to the bottom, results for MotoG, Huawei and ZTE. We omitted the cell CDFs in the plots on the left, since it would have been a single spike at 1 ms.

Focusing on the left column, we can see that the CDFs of different phones and those obtained at the kernel and at the application are very different. For instance, the MotoG plot shows that the majority (90%) of interarrival times measured at the kernel are shorter than 0.3 ms, but only 30% of those measured by the application are shorter than 0.3 ms. This means that multiple packets that are distinguishable at the kernel

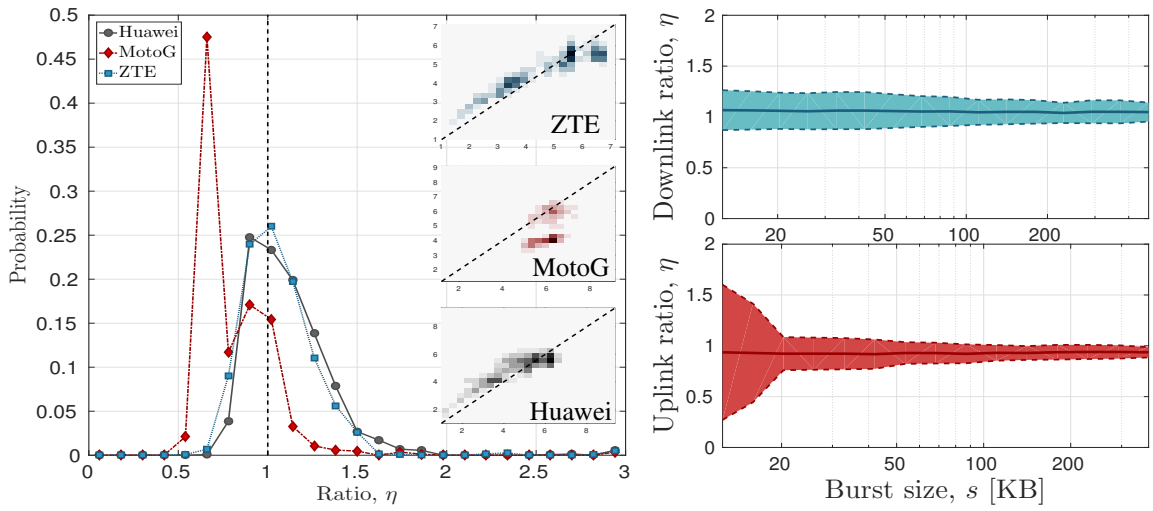


Figure 5. Estimator ratios computed on burst in the uplink (left). Estimator ratios against different burst size (right).

are seen by the application as a single stream. Fixing the threshold $\tau = 0.3$ ms allows to identify packets belonging to the same LTE transmission.

Instead, the CDF of ZTE interarrival times at the kernel shows two flat regions, one before 0.15 ms and the second after 1 ms: this is caused by intra-transmission arrivals (the former) and inter-transmissions arrivals (the latter). Conversely, this noticeable distinction is not found in the application trace. Accordingly, we fix the grouping thresholds to different values: $\tau = 0.2$ ms for the kernel and $\tau = 1$ ms for the application.

Finally, the Huawei CDFs only show slight inflections at 0.4 ms (kernel) and 0.7 ms (application), but both are less marked than those of the other phones. We set the two thresholds accordingly. As will be more evident hereafter, a more skewed CDF with distinguishable intra- and inter-transmission thresholds corresponds to more deterministic latencies in the phone and, in turn, to better group data rate estimates.

Analyzing the plots on the right, we can compare the interarrival time CDFs measured by the application, the kernel and the sniffer. Here we observe that the Huawei phone that has a slightly lower accuracy in terms of data rate estimation, and also shows a larger gap between the cell CDF and the other two, in particular between 10 and 30 ms. This confirms that data rate measurements are influenced by timing precision.

One final observation related to the ZTE phone is that both the application and the kernel CDFs show the same stair-shaped trend as in the cell CDF. Again, this is due to a lower variability of the ZTE latency, which will become more evident in the following second set of experiments. Comparing this to the data rate estimation above, we conclude that a deterministic delay between the radio chipset and the kernel (i.e., very similar CDFs in Figure 4) can achieve more accurate data rate measurements on shorter data bursts.

Figure 5 (left) shows the uplink data rate estimator ratios of the three phones. Again, we compare kernel measurements on the phone against the sniffer's ground truth for the cell. Note that uplink application measurements would require a dedicated application that could intercept ACKs or especially designed to monitor the sending socket. The normal socket behavior is to accept send requests from the application until the transmission buffer is full and, since this buffer is usually larger than our data burst, the application can send to the socket a whole burst at once making it impossible to measure the data rate at the phone application.

R6 – Mobile phones can obtain accurate and precise uplink data rates measurements: although the MotoG underestimates the rate by about 30%, the other two phones have the peaks of their epdfs very

close to 1. In particular, they achieve an accuracy α of 93% (Huawei) and 97% (ZTE), while MotoG stops at 65%. The edpfs are also wider than those related to the downlink. This is even more evident from the density plots on the right of the figure, which highlight that the precision of uplink measurements is lower than that obtained in the downlink: 73% for the Huawei and 74% for the ZTE. Not only does the MotoG have only 60% precision, but also its density plot shows two regions where the densities accumulate. This exhibits a binary behavior of the device that will become more evident in the next section where we analyze the phone latencies.

In addition, we compare the phone kernel to the server data rate measurements. Since the results obtained are very similar to those shown in Figure 5 (left) we omit the graphics. However, it is interesting that in our experiments, uplink burst can be measured both on the phone and the server achieving similar results. Since phone to cell measurements are taken before traversing the backhaul while phone to server results include it, we can conclude that the backhaul plays a minor role in our setup, because of the favorable location of the measurement server.

Figure 5 (right) reports the results obtained by varying the burst size from 10 KB to 1 MB for the downlink, and from 6 to 300 KB for the uplink. All the figures plot the average estimator ratio in the center of a shaded area that extends one standard deviation on each side. The figures are obtained by mixing together the results for all the phones.

R7 – Bursts of 20 KB provide high accuracy and high precision: the figures show that the estimator accuracy is independent of the burst size and the precision slowly improves with increasing size. Uplink proves to be more sensitive to very small burst (i.e., the shaded area is larger in the uplink plots for small bursts) as it is subject to more network randomness and it requires slightly longer transmissions. In contrast, in the downlink communication as few as two LTE transmissions are sufficient to obtain an accurate estimate. In our test we choose the minimum burst size to cause at least two transmissions at the maximum reachable data rate. In a larger bandwidth setup and when the next LTE releases will be deployed, the minimum burst size to achieve this results has to be increased proportionally to the maximum data rate.

R8 – UDP tests obtain the same results: the network provider used in our campaigns does not allow us to make reliable UDP tests, because of firewall and traffic shaping policies. To overcome this limitation, we repeated all the tests by emulating UDP by sending its packets with a TCP header through a raw socket. All the repetitions result in performance almost identical to that obtained by their TCP counterparts. The reason is that the measurement characteristics are dictated by the intra-burst timing, which, in turn, depend on the radio link technology, and not by the inter-burst timing, which, instead, depends on the protocol. Thus, radio link measurements only need for clearly separated burst in order for mobile phones to precisely estimate the data rate.

R9 – WiFi measurements are consistent, but different: we repeated the main tests on WiFi (IEEE 802.11g) by replacing OWL with a Warp Software Defined Radio [37] using the 802.11 reference design. We consistently observe that the performance obtained on WiFi are on the same order of magnitude of those obtained on LTE, but they are not identical in terms of bias and precision. Figure 6 shows how the WiFi estimators obtained in our tests exhibit a higher bias and a lower precision. The figure shows the estimates obtained with the MotoG phone. The results for the other two phones (not shown here) are very similar. While we agree on the main claims of [29] about overheads, we believe that different technologies require specific tests to evaluate their performance. Thus, each interface should be calibrated separately.

Before moving to the next set of experiments, we discuss a few more results for which we do not provide dedicated figures. We test our data rate measurements under three other conditions: 1) we stress the phone CPU to full load during the experiments; 2) we inject additional traffic in the cell under test up for to 95% load. Although, we expect the CPU load to add some delay to our measurement, we find that the phone

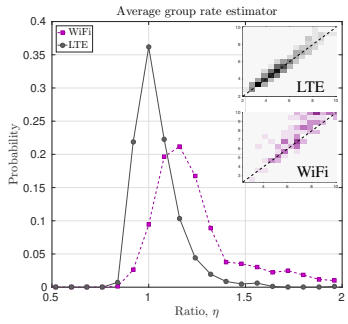


Figure 6. Comparison between estimator ratios obtained on LTE and WiFi.

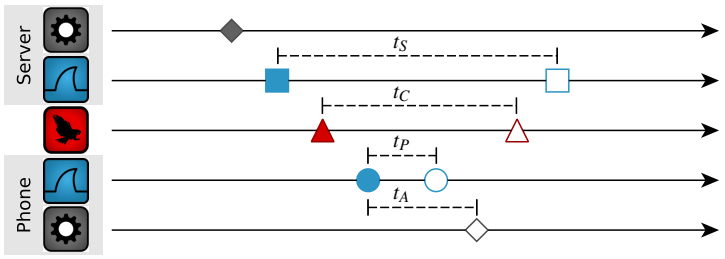


Figure 7. Communication diagram for the downlink isolated transmission. Dimension lines illustrate data-to-ack latency.

kernel copes well with this load and we did not notice any significant change in the estimator performance. Similarly, the additional traffic injected in the cell only changed the actual measured data rates (i.e., lowering them), but did not decrease the estimator's accuracy.

5. Isolated Transmissions

This section details the second set of measurements. The objective of this campaign is to measure phone communication latencies to justify the differences in their behavior. As above, we first illustrate the experiment on a diagram (Figure 7) and on some trace examples and then we discuss the results.

5.1. Experiment Description

For the analysis of latencies measured at each communication layer we take particular care to link homologous events in the different measurement devices. While this is trivial in the phone and the server where we can access all packet header fields, identifying which LTE transmission contains a given packet in the scheduling log poses several problems. First of all, we need to find the correct RNTI of the target UE among the rest of the traffic, but while for burst transmission we could both rely on fixed burst size and periodicity, in isolated transmission tests a single packet is sent from the application, the payload of which should be large enough to differentiate it from LTE control messages and small enough to fit in a single transmit unit in the phone and the server interface. We fix the packet size to 500 bytes with a periodicity of 400 ms to leave enough time between subsequent repetitions not to confuse them with possible retransmissions. While other UEs scheduled together with our target may have similar periodicity, our UE could always be correctly detected.

Figure 7 shows the ideal communication diagram for the downlink isolated transmission test. Here, we monitor the time elapsed between each packet and the corresponding ACK. In what follows, we refer to this data-to-ack time as latency.

Although we only monitor relative times and we do not need a perfect synchronization between the measurement layers, in order to correctly couple events we make sure to capture the first event of each test in all the layers (to have a common reference) and, then, we run a causality check on each trace to compensate possible violations. Since each subsequent layer events have to occur after the events of the upper layer, we realign traces to follow causality.

Dimension lines illustrate latencies in the different layers with the only exception of t_A which refers to the time between a packet being captured by the kernel (phone for the downlink or server for the uplink)

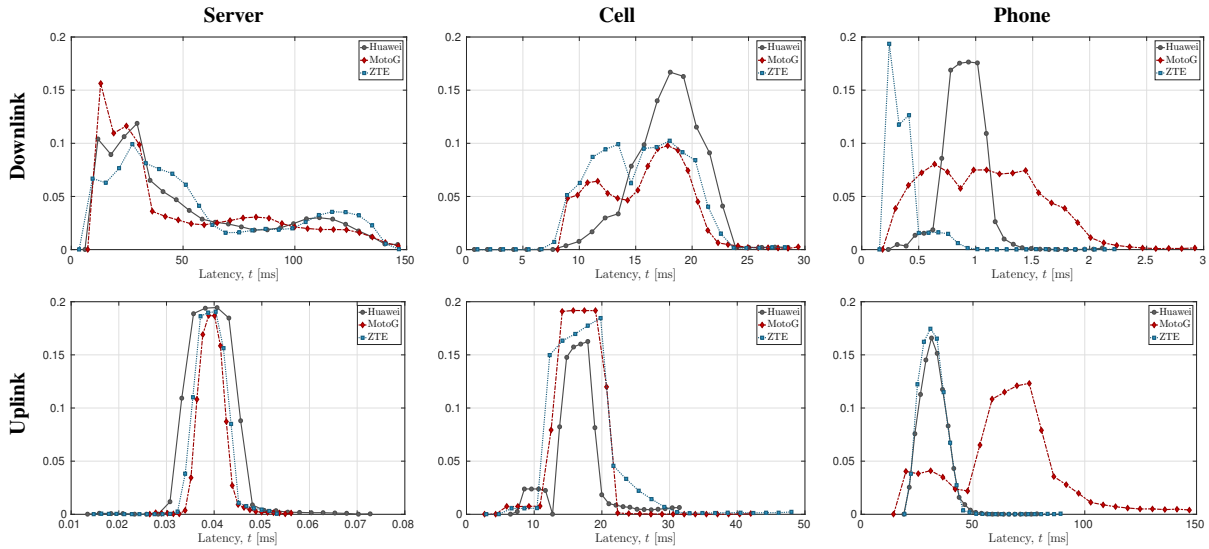


Figure 8. Empirical probability density functions of the latencies.

and when it is delivered to the application. Note that applications cannot measure their own latency without intercepting the communication ACKs at the kernel level.

5.2. *Experiment Results*

Figure 8 summarizes the main results of the latency measurements from which we draw conclusions about differences in the three phones' behaviors. All plots show the epdfs of the latency measured at the three measuring devices. All latencies are measured at the kernel level, since the application is not automatically notified of ACK receptions.

The plots in the figures are grouped vertically by communication direction and horizontally by layers and they are best read from the top left in clockwise order to follow the communication sequence.

The latency measured at the server in the downlink tests (top left) is the sum of the delays caused by two Internet traversals, two LTE scheduling delays (downlink first and then uplink) and phone processing (chipset time plus protocol stack traversal in the kernel). The latency at the cell downlink (top center) starts when the downlink LTE transmission is already scheduled and, as a consequence, it only contains the phone processing and the LTE uplink scheduling delays. The latency at the phone downlink (top right) starts from when the kernel receives the reception interrupt from the chipset to when the ACK transmission to the communication interface.

R10 – Chipsets with short and deterministic latency achieve more accurate and precise data rate estimation: in downlink tests, the latencies are similar in all the layers, except on the phone. The ZTE latencies exhibit a single peak before 0.5 ms, the Huawei a single, slightly wider peak at about 0.9 ms, while the MotoG shows a wider distribution of its latencies ranging from 0.3 to almost 2 ms. Recalling from Figure 3(a) that MotoG and ZTE achieve higher accuracy and precision in their data rate estimates, we can conclude that chipsets with a shorter latency are more accurate in estimating the data rate. Instead, the Huawei latency is closer to the length of the LTE TTI and is the cause of the overestimation of the data rate. To explain the difference in performance between the ZTE and MotoG, we need to consider the CDFs of their short interarrival time (recall Figure 4 top and bottom graphs on the left). While the MotoG application captures intra-group events (shorter than 0.5 ms), the ZTE application distribution starts only after 0.5 ms,

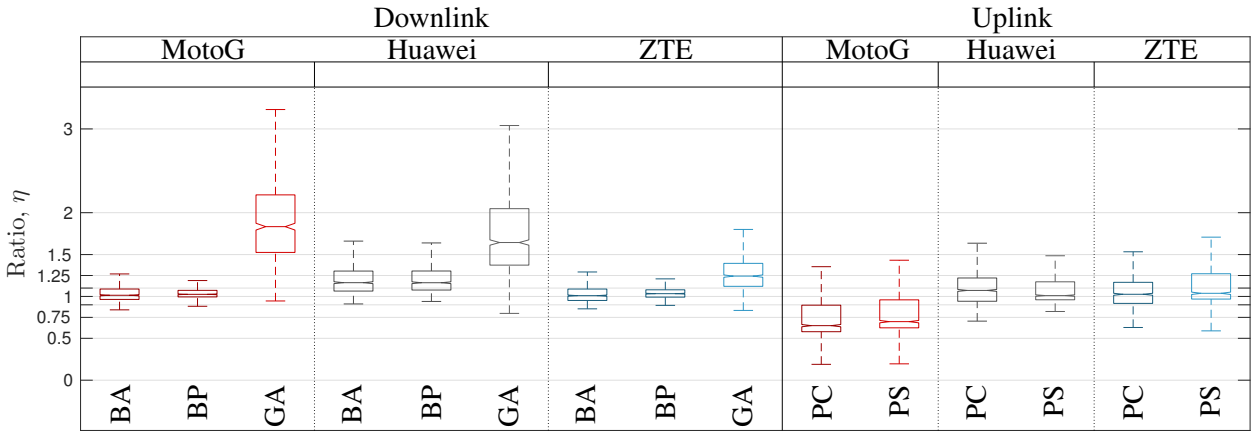


Figure 9. Overall comparison among the data rate estimators. All boxplots show the distribution of the estimator ratios from the 25th and 75th percentiles.

but it is very precise at the kernel. Thus, the MotoG application data rate estimate fares better than ZTE, which, instead, is more precise at the kernel level only.

The low variability of ZTE latencies explains why the ZTE long interarrival time distribution has the same stair-shaped trend as the cell distribution. As a consequence of this higher precision at the kernel, the ZTE phone can better discriminate between LTE transmissions, which, instead, are smoothed in the other two phones, and is able to obtain more accurate group-based data rate estimates (recall Figure 3(c)). Also, since the three phones show similar times for server and cell latencies we can exclude that network traversals and LTE scheduling impact data rate estimates between phone and cell in our setup.

R11 – LTE discontinuous reception configuration [38] influences uplink data rate estimates: the bottom row of Figure 8 illustrates the epdfs of uplink experiments. Latencies measured on the phone kernel include both uplink and downlink LTE scheduling, two Internet traversals and the server processing; latencies at the cell include the Internet traversals and the server processing delays, while the server latencies only include processing delay. The server processing is negligible, since it is shorter than 50 µs. Similarly, we can exclude that the network delays play an important role in the uplink data rate estimates, since the three phones show almost identical latencies when measured at the cell. Conversely it is the LTE uplink scheduling delay that influences the estimator the most. This delay is expected to be about 20 ms for a connected device starting a new transmission, while in our measurements the epdfs are centered at about 50 ms (Huawei and ZTE) and 85 ms (MotoG) with a smaller peak at 40 ms. All these longer uplink delays are due to LTE discontinuous reception (DRX), which is an energy saving feature that allows mobile phones to duty-cycle between sleep and wake phases. Since to discriminate among different transmissions we separate them by 400 ms (1 s for bursts), all the transmissions start with the devices in DRX mode. The actual duration of the sleep time depends on agreements between UE capabilities and eNodeB requirements. Thus, MotoG uses a more conservative DRX setup (with a longer sleeping period), most likely due to the fact that it is an earlier (2014) model than the other phones (2015). The overall effect of this latency is that uplink data rate estimators are less precise than downlink estimates by circa 10% due to the wider distributions of the latencies.

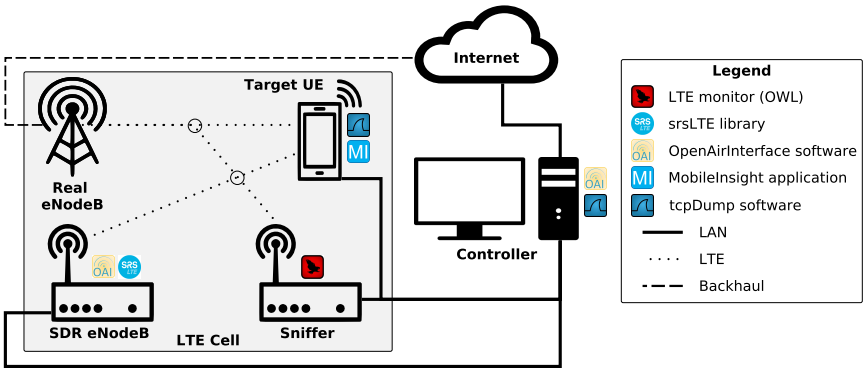


Figure 10. Measurement setup showing all the options described in the text.

6. Summary

Figure 9 provides a visual summary of the results discussed in the paper. In the figure, one boxplot is shown for each of the main experiments, highlighting the median (central mark of the boxes) and the 25th and 75th percentiles (box edges) of the estimator ratios η. At the bottom of the figure we specify the type of η used. BA are ratios between data rate measurements performed on bursts (B) at the application (A) and cell estimate. BP are the same but computed by the phone kernel (P), while GA are the same as BA, but computed on groups (G). All downlink ratios use the sniffer as a reference. On the uplink side, we show PC, which compares phone kernel (P) estimates with cell (C) and PS which use the server (S) application as a reference. All uplink results are computed on bursts. Note that we do not show graphs for PS in the previous results, since PC and PS are quite similar. This shows that in our setup uplink data rate estimates on the server and the phone obtain comparable results.

Looking at all the results side-by-side, it is evident that on the one hand side, all the phones are capable of accurate and precise data rate estimation, but on the other hand they have significantly different biases and precisions, as seen for instance, with Huawei and ZTE for BA in the downlink and for PC in the uplink. Similarly noticeable is that group-based estimators, GA, achieve reasonable accuracy on the ZTE phone only and the MotoG uplink measurements are heavily impaired by the different LTE latency.

To conclude our study on LTE radio link estimation, we can affirm that the precision and the accuracy achieved by the three devices are sufficiently high to enable anticipatory networking optimization up to a time granularity of about 50-100 ms and after having compensated for the device bias. This is true for both uplink and downlink estimates either obtained by the kernel or the application. Conversely, to increase the measurement granularity and, in turn, the optimization potential, direct readings of the actual physical rate are needed.

7. Measurement Setup Alternatives

This section describes how to setup a mobile phone measurement environment by providing a few software alternatives and discussing their benefits and their drawbacks. The main objective of the measurements we are going to describe is to provide a flexible environment that allows testing both mobile phones and base stations.

Figure 10 shows a complete environment exploiting all four software solutions, namely OWL [20], MobileInsight [21], OpenAirInterface [22] and srsLTE [23]. However, in the following we describe three

simpler scenarios that are subsets of the complete setup and offer different advantages. In particular, these three scenarios will be named Sniffer-based, Phone-based, and eNodeB-based setup respectively.

1. **Sniffer-based setup:** an SDR is used to record and decode the traffic exchanged in a given cell.
2. **Phone-based setup:** the main measurements are performed on the target UE itself.
3. **eNodeB-based setup:** a controlled eNodeB is used to measure the performance of the served UE(s).

In the figure, hardware components are shown as black silhouettes, while software components are represented as icons. Starting from the left hand side of the figure, the shown hardware components are: a real eNodeB (top left part drawn as a network antenna), a SDR-based eNodeB (bottom left box with small antenna), the target UE (top center), sniffer node (bottom center box with small antenna) and a controller PC on the right hand side of the figure. The real eNodeB can be either a basestation of a commercial operator or a private small cell. Both the sniffer and the SDR-based eNodeB require a software defined radio capable of generating an LTE compliant clock. For instance, we had good results with both USRPs [39] and BladeRFs [30]. Finally, the controller must have enough resources to run the software for both the eNodeB and the backhaul services.

The complete setup allows for a broad range of tests and measurements, but in what follows we will focus on five main tests:

1. **UE measurements:** the target UE performance is monitored to verify its capabilities and/or to check the accuracy of the readings carried out on different devices.
2. **UE debug:** the target UE functionalities are verified against their expected behavior in a controlled environment.
3. **eNodeB measurements:** the performance of a target eNodeB is measured to verify its specifications and capabilities.
4. **eNodeB debug:** the functionalities of a target eNodeB are tested against their expected behavior.
5. **Cell measurements:** all the devices connected to the same eNodeB are monitored to obtain cell-wide information.

These tests provide a broad set of tools for the control and management of LTE networks.

Before analyzing the details of the three deployment scenarios, we discuss the main features of the software components:

- **OWL [20]** is our software solution to decode LTE control channel. The decoded information provides the complete scheduling for all devices connected to a given eNodeB in both the uplink and downlink channels.

- **MobileInsight [21]** is a software solution that exposes protocol messages from off-the-shelf 3G/4G chipsets. As such, it provides the exact logging of all the operations performed by the target UE.

- **OpenAirInterface [22]** is a platform designed to build complete LTE ecosystems. It provides an open-source implementation spanning the full protocol stack of 3GPP standard both in the radio access and in the backhaul (Enhanced Packet Core (EPC)).

- **srsLTE [23]** started as an open-source LTE library providing standard compliant functions to build an LTE system. Subsequently, stand-alone eNodeB and UE solutions were added to the platform and, recently, a simple EPC implementation has been released.

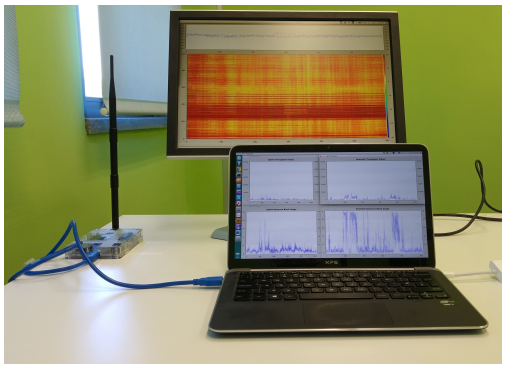


Figure 11. Sniffer-based deployment setup.

```
0449 0 65535 1 3 8 104 ...  0 3 1 100
0449 3 12529 1 0 3 56 ...  8 3 2 100
0449 4 12529 0 29 4 0 ...  8 2 2 100
0450 0 65535 1 14 37 552 ...  0 3 1 99
0450 5 65535 1 6 3 176 ...  0 3 1 100
0450 9 65534 1 4 9 120 ...  0 3 1 100
0451 0 65535 1 26 50 1480 ...  0 3 1 99
0451 5 65535 1 26 43 1480 ...  0 3 1 99
0451 9 65534 1 2 6 72 ...  0 3 1 99
0452 5 65535 1 6 3 176 ...  0 3 1 100
0454 5 65535 1 6 3 176 ...  0 3 1 99
0456 0 65535 1 9 19 296 ...  0 3 1 100
```

Figure 12. Example of a trace collected with OWL.

7.1. Sniffer-based Scenario

The sniffer-based deployment scenario is the one described earlier in Section 3 and that, in addition to the results shown here, allowed us to investigate prediction-based optimization solutions applied to LTE networks [40] and carrier aggregation performance [41].

This scenario exploits OWL to obtain the complete scheduling information of a target eNodeB. Figure 11 illustrates our basic configuration, where an SDR is controlled by a laptop. In the figure, the software is running in graphic mode and shows a spectrogram of the received power on the monitor behind the laptop and four graphics of the aggregate performance of the monitored cell.

While the graphic mode provides an intuitive visualization of the traffic on the monitored cell, Figure 12 shows a short sample of a trace extract by OWL. Each line in the sample is related to a single transmission and, starting from the left, the columns provide the System Frame Number (SFN), the subframe index, the RNTI of the device involved in the transmission, the direction of the communication (1 = downlink, 0 = uplink), the modulation and coding scheme (MCS) used in the transmission, the number of resource blocks used and the number of bits of information transferred. The rest of the columns provide other protocol data that can be used to distinguish retransmissions from regular transmissions, among other details.

In this setup, the decoded information pertains to all the devices connected to the monitored eNodeB. Even though this allows for simple aggregate measurements, it requires some additional effort to identify specific devices within the trace. In fact, given that RNTIs are assigned randomly, it is not possible to know the RNTI used by a given device without accessing its internals. To overcome this limitation and to enable UE measurements, we force the target UE to show a unique communication pattern (e.g., ping every 7 seconds) before starting the actual measurements. In such a way, it is possible to identify the target UE by looking in the trace for an RNTI that shows the imposed communication pattern.

By comparing the measurements obtained using OWL with tcpdump readings carried out on the server and on the target UE, as we did for this paper, it is possible to calibrate the bias of any phone. Then this phone may be later used for, e.g., crowd-based measurements.

The main advantages of this setup are its versatility, as it allows UE, eNodeB and cell measurements while being independent of the hardware used. In fact, since the measurements are performed by decoding the network control channel, they neither depend on the UE nor on the eNodeB involved in the communication. Conversely, this setup requires some post-processing and, since it depends on the received signal quality, it is prone to small information loss [20].

```
{"Subframe Number": 0, "type_id": "LTE_PHY_PDSCH_Packet", "Version": 23, "Frequency Selective
PMI": "WideBand", "Number of Rx Antennas(N)": 2, "MCS 0": "64QAM", "MCS 1": "64QAM", "RB
Allocation Slot 1[0]": "0xffffffffffffffff", "RB Allocation Slot 0[0]": "0xffffffffffffffff",
"Transmission Scheme": "Open-loop spatial multiplexing", "PDSCH RNTIl ID": 29692, "Serving
Cell ID": 6 ...}
{"Subframe Number": 1, "type_id": "LTE_PHY_PDSCH_Packet", "Version": 23, "Frequency Selective
PMI": "WideBand", "Number of Rx Antennas(N)": 2, "MCS 0": "64QAM", "MCS 1": "64QAM", "RB
Allocation Slot 1[0]": "0xffffffffffffffff", "RB Allocation Slot 0[0]": "0xffffffffffffffff",
"Transmission Scheme": "Open-loop spatial multiplexing", "PDSCH RNTIl ID": 29692, "Serving
Cell ID": 6 ...}
{"Subframe Number": 2, "type_id": "LTE_PHY_PDSCH_Packet", "Version": 23, "Frequency Selective
PMI": "WideBand", "Number of Rx Antennas(N)": 2, "MCS 0": "64QAM", "MCS 1": "64QAM", "RB
Allocation Slot 1[0]": "0xffffffffffffffff", "RB Allocation Slot 0[0]": "0xffffffffffffffff",
"Transmission Scheme": "Open-loop spatial multiplexing", "PDSCH RNTIl ID": 29692, "Serving
Cell ID": 6 ...}
```


Figure 13. Example of a trace collected with Mobileinsight.

7.2. Phone-based Scenario

The second scenario is the simplest from a deployment point of view. In fact, the phone-based setup only requires to install MobileInsight [21] on a target rooted phone or on a computer connected to the phone. Since MobileInsight obtains information directly from the radio chipset, it provides exact logs of all the operations performed by the phone independently of the radio technology used.

Figure 13 shows a sample of a trace of the physical layer collected during LTE communications. Every transmission is logged within curly brackets and each parameter is printed together with its name. In the shown example, the first transmission is performed at SFN 618 subframe 0 using a 64 QAM modulation with spatial multiplexing sending two transport blocks of 36696 bits each. In addition, the sample reveals that the device uses RNTI 29692 and the bitmask of the resource blocks used for the transmission is also specified.

The abundance of information provided by MobileInsight makes it the perfect choice for UE debugging and single UE measurements. However, not every phone is supported by the software as it can only communicate with specific chipsets. Moreover, performing measurements on a mobile phone has two main drawbacks: first, it only provides information about the target UE and can only perform single-user tests on the eNodeB, and, second, the mobile phone resources might limit the tests due to physical memory limitations and/or computational power constraints.

7.3. eNodeB-based Scenario

The last deployment scenario involves the use of a controlled eNodeB. To this extent, while commercial solutions might be easier to deploy, both OpenAirInterface (OAI) [22] and srsLTE [23] provide fully featured open-source implementation of LTE systems. They include both radio access (eNodeB) and backhaul (EPC) functionalities. These software frameworks allow to connect both commercial mobile phones and virtual UEs implemented on SDRs.

The simplest deployment for this scenario requires a SDR and a PC capable of running both the EPC and the eNodeB software components in real time. This setup allows for a broad range of experiment scenarios measuring both the eNodeB and the UE. It also provides the most accurate measurements on either side. Figure 14 provides an example of the eNodeB log recorded using srsLTE (OpenAirInterface logs are not

```
17:44:31.209237 [RRC ] Info SRB1 - Rx RRC Connection Reconfiguration Complete
   0000:  10 00
17:44:31.209247 [RLC ] Info SRB2 Rx SDU
   0000:  00 48 01 a4 fc ee db 52 40 20 e8 60 00 6a 40 18
   0010:  40 ea a0 e6 c8
17:44:31.209264 [PDCP] Info RX SRB2 SDU SN: 0
   0000:  48 01 a4 fc ee db 52 40 20 e8 60 00 6a 40 18 40
17:44:31.209287 [RRC ] Info Rx SRB2 PDU
   0000:  48 01 a4 fc ee db 52 40 20 e8 60 00 6a 40 18 40
17:44:31.209289 [MAC ] Info [00877] CE: Received Short BSR rnti=0x46, lcg=0, value=0
17:44:31.209312 [RRC ] Info SRB2 - Rx UL Information Transfer
   0000:  48 01 a4 fc ee db 52 40 20 e8 60 00 6a 40 18 40
17:44:31.209326 [S1AP] Info Received RRC SDU
   0000:  27 e7 76 da 92 01 07 43 00 03 52 00 c2
17:44:31.209352 [S1AP] Info Sending UplinkNASTransport for RNTI:0x46
   0000:  00 0d 40 37 00 00 05 00 00 00 02 00 01 00 08 00
   0010:  02 00 01 00 1a 00 0e 0d 27 e7 76 da 92 01 07 43
17:44:31.209852 [MAC ] Info [00874] SCHED: DL tx rnti=0x46, pid=0, mask=0x1800, dci=1,8,
n_rtx=0, tbs=11, buffer=0
17:44:31.209881 [RLC ] Info SRB1 Tx status PDU - ACK_SN = 7, N_nack = 0
```

Figure 14. Example of a trace collected with srsLTE.

shown as they provide similar information, but are even more verbose). The log, illustrates the messages exchanged with a commercial phone just after the end of the first attachment phases. In addition to the information provided in the previous scenarios, here all the layers of the protocol stack are monitored and can be analyzed.

Furthermore, both frameworks being open-source, they allow for the deployment and test of custom solutions on all the layers of the protocol stack and monitoring all the entities involved in the communication. However, this setup is the most expensive of the three and it doesn't allow for field measurements as it is not portable. Finally, running an eNodeB requires the use of ISM frequencies or a research license.

In this section, we described a complete setup for LTE measurements and tests and we provided three simple deployment scenarios that allow the interested reader to repeat our measurements (as well as to deploy an LTE test and development setup). Table 1 provides a comparison among the features offered by the three deployment scenarios.

8. Conclusions

In this paper we presented the first experimental evaluation of the accuracy and the precision of LTE data rate measurements performed by mobile phones. To summarize the main finding of the study:

Mobile phones can achieve accurate and precise data rate measurements: we showed that downlink application measurements are both accurate and precise (R1), downlink kernel-level measurements improve the precision, but only slightly (R4), and uplink kernel estimates are accurate and precise, but less than those obtained on the downlink (R6).

Mobile phones have different biases: R2 and R6 show that mobile phone estimates vary depending on many factors. R8 highlights chipset latency as the main cause for downlink differences, while R11 identifies the LTE uplink delay as the main source of variability. As a consequence, when combining the results

Table 1. Comparison among the three setups for mobile phone measurement.

	Sniffer	**Phone**	**eNodeB**
Accuracy	99.85%	99.95%	100%
Setup Complexity	Low	Low	High
Test Complexity	High	Medium	Low
HW agnostic	✓	✗	✓[a]
UE measurements	✓	✓	✓
UE debug	✓[b]	✓	✓
eNB measurements	✓	✓[c]	✗[d]
eNB debug	✓	✓[c]	✗
Cell measurements	✓	✗	✗

[a] Some devices are not supported by OAI.
[b] Limited to unencrypted control messages.
[c] Single UE only.
[d] However, OAI provides an UE implementation which allows to test eNodeBs.

obtained using different phones, it is good practice to evaluate the bias of each of them in order to compensate systematic errors. Moreover, R9 shows that measurements performed on different communication technologies with the same mobile phones show similar, but not identical results.

Small data bursts are sufficient for 95% accuracy at 80% precision: we studied how the estimator quality varies with burst size (R7) and we found that 20 KB (or 50 ms) are sufficient to obtain accurate and precise estimators. Moreover, in our tests the burst estimator quality showed very little dependence of the measured data rate (R3). Instead, we found that estimating the data rate from groups of packets arriving back-to-back at the measurement point leads to low quality (high bias, low precision) estimates (R5). This shows that anticipatory networking solutions can rely on precise information computed over 50 ms windows, but the actual radio link data rate has to be obtained to increase this resolution.

Burst measurements are protocol independent: by comparing TCP and UDP tests, R8 shows that precision and accuracy depend on the capability of detecting burst precisely.

Finally, we provided a complete description about how to reproduce our measurement setup together with a few alternative deployment options and we discussed their advantages and drawbacks comparing their respective features. Moreover, we believe that this study offered a new perspective about crowdsourced measurements and that it will help improve the reliability of future campaigns.

Acknowledgments

This work has been supported by the European Union H2020-ICT grant 644399 (MONROE), by the Madrid Regional Government through the TIGRE5-CM program (S2013/ICE-2919), the Ramon y Cajal grant from the Spanish Ministry of Economy and Competitiveness RYC-2012-10788 and grant TEC2014-55713-R.

References

[1] N. Bui, M. Cesana, A. Hosseini, Q. Liao, I. Malanchini, J. Widmer, A survey of anticipatory mobile networking: Context-based classification, prediction methodologies, and optimization techniques, Submitted to IEEE Communications Surveys and Tutorials.

[2] NGMN, Next Generation Mobile Networks.
URL {http://www.ngmn.de/publications/all-downloads/article/ngmn-5g-white-paper.html}
[3] G. P. Fettweis, The tactile internet: applications and challenges, Vehicular Technology Magazine, IEEE 9 (1) (2014) 64–70.
[4] N. Bui, I. Malanchini, J. Widmer, Anticipatory admission control and resource allocation for media streaming in mobile networks, in: ACM MSWiM, 2015, pp. 255–262. doi:10.1145/2811587.2811604.
URL http://doi.acm.org/10.1145/2811587.2811604
[5] X. Yin, A. Jindal, V. Sekar, B. Sinopoli, A control-theoretic approach for dynamic adaptive video streaming over HTTP, ACM SIGCOMM Computer Communication Review 45 (4) (2015) 325–338.
[6] Y. Sun, X. Yin, J. Jiang, V. Sekar, F. Lin, N. Wang, T. Liu, B. Sinopoli, CS2P: Improving video bitrate selection and adaptation with data-driven throughput prediction, in: ACM SIGCOMM, 2016, pp. 272–285.
[7] N. Bui, J. Widmer, Mobile network resource optimization under imperfect prediction, in: IEEE WoWMoM, 2015.
[8] T. Mangla, N. Theera-Ampornpunt, M. Ammar, E. Zegura, S. Bagchi, Video through a crystal ball: effect of bandwidth prediction quality on adaptive streaming in mobile environments, in: ACM International Workshop on Mobile Video (MoVid), 2016, p. 1.
[9] J. Huang, F. Qian, A. Gerber, Z. M. Mao, S. Sen, O. Spatscheck, A close examination of performance and power characteristics of 4g lte networks, in: ACM MobiSys, 2012, pp. 225–238. doi:10.1145/2307636.2307658.
[10] J. Huang, F. Qian, Y. Guo, Y. Zhou, Q. Xu, Z. M. Mao, S. Sen, O. Spatscheck, An in-depth study of lte: Effect of network protocol and application behavior on performance, in: ACM SIGCOMM, 2013, pp. 363–374. doi:10.1145/2486001.2486006.
[11] J. B. Landre, Z. E. Rawas, R. Visoz, Lte performance assessment prediction versus field measurements, in: IEEE PIMRC, 2013, pp. 2866–2870. doi:10.1109/PIMRC.2013.6666636.
[12] A. Nikravesh, D. R. Choffnes, E. Katz-Bassett, Z. M. Mao, M. Welsh, Mobile network performance from user devices: A longitudinal, multidimensional analysis, in: Springer PAM, 2014, pp. 12–22.
[13] A. Nikravesh, H. Yao, S. Xu, D. Choffnes, Z. M. Mao, Mobilyzer: An open platform for controllable mobile network measurements, in: ACM MobiSys, 2015, pp. 389–404. doi:10.1145/2742647.2742670.
[14] Ookla, Speedtest mobile apps, http://www.speedtest.net/mobile/ (last accessed November 2016).
[15] Q. Xu, S. Mehrotra, Z. Mao, J. Li, PROTEUS: network performance forecast for real-time, interactive mobile applications, in: ACM MobiSys, Taipei, Taiwan, 2013, pp. 347–360.
[16] Y. Xu, Z. Wang, W. K. Leong, B. Leong, An end-to-end measurement study of modern cellular data networks, in: Springer PAM, 2014, pp. 34–45.
[17] C. Ide, B. Dusza, C. Wietfeld, Performance of channel-aware M2M communications based on LTE network measurements., in: IEEE PIMRC, 2013, pp. 1614–1618.
[18] X. Xie, X. Zhang, S. Kumar, L. E. Li, pistream: Physical layer informed adaptive video streaming over lte, in: ACM MobiCom, 2015, pp. 413–425.
[19] S. Kumar, E. Hamed, D. Katabi, L. Erran Li, LTE radio analytics made easy and accessible, in: ACM SIGCOMM, Vol. 44, 2014, pp. 211–222.
[20] N. Bui, J. Widmer, OWL: a Reliable Online Watcher for LTE Control Channel Measurements, in: ACM All Things Cellular, 2016.
[21] Y. Li, C. Peng, Z. Yuan, J. Li, H. Deng, T. Wang, Mobileinsight: Extracting and analyzing cellular network information on smartphones, in: ACM Mobicom, 2016.
[22] N. Nikaein, R. Knopp, F. Kaltenberger, L. Gauthier, C. Bonnet, D. Nussbaum, R. Ghaddab, OpenAirInterface 4G: an open LTE network in a PC, in: ACM MobiCom, 2014.
[23] I. Gomez-Miguelez, A. Garcia-Saavedra, P. D. Sutton, P. Serrano, C. Cano, D. J. Leith, srsLTE: An open-source platform for LTE evolution and experimentation, in: ACM WinTECH, 2016.
[24] A. Elnashar, M. A. El-Saidny, Looking at lte in practice: A performance analysis of the lte system based on field test results, IEEE Vehicular Technology Magazine 8 (3) (2013) 81–92. doi:10.1109/MVT.2013.2268334.
[25] N. Becker, A. Rizk, M. Fidler, A measurement study on the application-level performance of lte, in: IFIP Networking Conference, 2014, pp. 1–9. doi:10.1109/IFIPNetworking.2014.6857113.
[26] M. Jovanovic, M. K. Karray, B. Blaszczyszyn, QoS and network performance estimation in heterogeneous cellular networks validated by real-field measurements, in: ACM PE-WASUN, 2014, pp. 25–32. doi:10.1145/2653481.2655236.
[27] M. Siekkinen, M. A. Hoque, J. K. Nurminen, M. Aalto, Streaming over 3g and lte: How to save smartphone energy in radio access network-friendly way, in: ACM MoVid, 2013, pp. 13–18.
[28] M. Palola, M. Matinmikko, J. Prokkola, M. Mustonen, M. Heikkil, T. Kippola, S. Yrjl, V. Hartikainen, L. Tudose, A. Kivinen, J. Paavola, K. Heiska, Live field trial of licensed shared access (LSA) concept using LTE network in 2.3 GHz band, in: IEEE DYSPAN, 2014, pp. 38–47. doi:10.1109/DySPAN.2014.6817778.
[29] W. Li, R. K. P. Mok, D. Wu, R. K. C. Chang, On the accuracy of smartphone-based mobile network measurement, in: IEEE INFOCOM, 2015, pp. 370–378.

[30] Nuand, Bladerf software defined radio, https://www.nuand.com/blog/product/bladerf-x40/ (last accessed November 2016).
[31] Motorola, MotoG LTE, http://www.devicespecifications.com/en/model/10da2ca3 (last accessed November 2016).
[32] Huawei, P8 Lite, http://www.devicespecifications.com/en/model/b95e33c3 (last accessed November 2016).
[33] ZTE, Blade A452, http://www.devicespecifications.com/en/model/a44f38ba (last accessed November 2016).
[34] MartinGarcia, Luis, TCPdump, http://www.tcpdump.org/ (last accessed November 2016).
[35] F. Michelinakis, N. Bui, G. Fioravantti, F. Kaup, D. Hausheer, J. Widmer, Lightweight mobile bandwidth availability measurement, Elsevier Computer Communications 84 (2016) 73–83.
[36] ETSI, E-UTRA; Physical layer procedures, 3GPP TS 36.213 (2016) V13.
[37] Wireless Open Access Research Platform, 802.11 reference design for WARP v3, http://warpproject.org/trac/wiki/802.11 (last accessed November 2016).
[38] ETSI, E-UTRA; Medium Access Control (MAC) protocol specification, 3GPP TS 36.321 (2016) V13.
[39] Ettus Research, Usrp-x-series, https://www.ettus.com/product/details/X310-KIT.
[40] N. Bui, J. Widmer, Data-driven Evaluation of Anticipatory Networking Optimization on LTE Networks, in: IEEE Internet Teletraffic Conference (ITC29), 2017.
[41] N. Ludant, N. Bui, A. García Armada, J. Widmer, Data-driven performance evaluation of carrier aggregation in lte-advanced.