NETWORKS & DISTRIBUTED SYSTEMS

ANALYSIS AND OPTIMIZATION OF METHODS FOR MEASURING NETWORK PERFORMANCE AND QUALITY - DEVELOPED FOR ANDROID DEVICES

Lars Møller Mikkelsen, Michael Sølvkær Pedersen
& Steffen Riber Thomsen

AALBORG UNIVERSITY
DENMARK

MASTER'S THESIS
SPRING SEMESTER 2013

Frontpage picture:
`http://ww1.prweb.com/prfiles/2012/02/08/9684106/WebsitePhoto3.jpg`

School of Information and Communication Technology
Study board for Electronics and IT
Frederik Bajers Vej 7
Phone: +45 9940 7228
http://sict.aau.dk

Title:

Analysis and Optimization of Methods for Measuring Network Performance and Quality - Developed for Android Devices

Theme:

Networks & Distributed Systems
Master's Thesis

Project period:
Spring Semester 2013

Project group:
13gr1022

Group members:
Lars Møller Mikkelsen
Michael Sølvkær Pedersen
Steffen Riber Thomsen

Supervisors:
Tatiana Kozlova Madsen
Jens Myrup Pedersen

Circulation: 6

Page number: 139 (154 including appendices)

Annex number and –art: 2 appendice excluding CD

Submission date: June 6th, 2013

Abstract:

This project is a continuation of the work on the NetMap system, which is a network measurement system developed for Android devices, aimed at using crowdsourcing for gathering measurements, to be used in creating a Network Performance Map. In this project there are 2 main focus areas: Extended analysis of the measurements gathered with the NetMap system, and optimization of measurement methods.
The extended analysis includes analyzing the impact of the following points on the connection performance: movement speed, handovers between cell antennas, area types, and device models. Furthermore, the currently implemented methods for measuring Round-Trip Times (RTTs) and throughput are analyzed in-depth to get a deeper understanding of the communication taking place during the measurement.
In the optimization of measurement methods, 2 techniques for estimating the available bandwidth, while keeping the data usage at a minimum, are implemented in the system and tested on a real cellular connection. Results from these techniques are compared to results from the current throughput estimation method, and evaluated based on these circumstances.
From the extended analysis it is concluded that when a device moves faster a slight decrease in performance is seen, and that in areas where handovers tend to happen the performance is decreased as well. Furthermore, it is shown that the performance typically is better in small town areas than in urban areas, believed to be due to the combination of fewer users and more network resources. From the analysis of device models, examples of differences in performance between high and low-end devices are presented. Based on the analysis of the current measurement methods, some alternative methods for extracting more detailed results are suggested.
Two additional bandwidth estimation techniques have been implemented and analyzed. It is concluded that one technique delivers satisfactory estimations while keeping the data usage extremely low, where the other technique experienced issues due to the requirement of time precision in transmission of measurement packets.

AALBORG UNIVERSITY

School of Information and Communication Technology
Study board for Electronics and IT
Frederik Bajers Vej 7
Phone: +45 9940 7228
http://sict.aau.dk

Title:

Analyse og Optimering af Metoder til Måling af Netværksydelse og Kvalitet - Udviklet til Android Enheder

Tema:

Netværk og Distribuerede Systemer
Masterafhandling

Projektperiode:
Forårssemestret 2013

Projektgruppe:
13gr1022

Gruppe medlemmer:
Lars Møller Mikkelsen
Michael Sølvkær Pedersen
Steffen Riber Thomsen

Vejledere:
Tatiana Kozlova Madsen
Jens Myrup Pedersen

Cirkulation: 6

Side antal: 139 (154 herunder bilag)

Bilags nummer og tilbehør: 2 bilag, eksklusiv CD

Afleveringsdato: 6. Juni 2013

Synopsis:

Dette projekt er en fortsættelse af arbejdet på NetMap systemet, som er et netværks målingssystem, udviklet til Android enheder, rettet mod at bruge crowdsourcing til at samle målinger, som kan bruges i et netværks ydelseskort.
I dette projekt er der 2 primære fokusområder: Udvidet analyse af målinger indsamlet med NetMap systemet, og optimering af målemetoder.
Den udvidede analyse omfatter en analyse af hvorledes de følgende punkter påvirker ydelsen af forbindelsen: bevægelseshastighed, områdetype og modellen der måles med. Ydermere vil de nuværende implementerede metoder til at måle Round-Trip Times (RTTs) og overførselshastighed blive analyseret dybdegående, for at opnå en bedre forståelse af kommunikationen der sker under målingerne.
I optimeringen af målemetoderne er 2 teknikker, til at estimere den tilgængelige båndbredde imens dataforbruget holdes nede, implementeret i systemet og testet på en rigtig mobil netværksforbindelse. Resultater fra disse teknikker er sammenlignet med resultater fra den nuværende estimeringsmetode for overførselshastighed, og evalueret på grundlag af disse omstændigheder.
Fra den udvidede analyse er det konkluderet, at når en enhed bevæger sig hurtigere, eller befinder sig i et område med tendens til handovers, vil ydelsen af forbindelsen falde lidt. Endvidere er det vist at ydelsen typisk er bedre i småbyer sammenlignet med storbyer, hvilket menes at skyldes en kombinationen af færre brugere og flere tilgængelige netværksressourcer. I analysen af måleenheder vises eksempler på forskelle i ydelsen mellem high- og low-end enheder. Baseret på analysen af de nuværende implementerede målemetoder foreslås alternative metoder til at udtrække mere detaljerede resultater fra disse.
Yderligere to estimeringsmetoder til overførselshastighed er blevet implementeret og analyseret. Det konkluderes at den ene teknik leverer tilfredsstillende estimeringer, imens at dataforbruget holdes meget lavt, hvor den anden teknik oplever nogle problemer grundet kravet til tidspræcision ved afsendelse af målepakker.

Lars Møller Mikkelsen

Michael Sølvkær Pedersen

Steffen Riber Thomsen

Preface

This report is the documentation of a master's thesis done by group 13gr1022 in the spring semester of 2013, by students from Networks & Distributed Systems, at Aalborg University.

Reading Guide This report is divided into chapters and sections numbered by x.y, where x indicates the chapter and y indicates section. References to figures and tables are indicated by x.y, where x indicates chapter, and y the figure number in the given chapter. External literature is indicated by [x], where x is a reference to the bibliography. If an entire text section is based on a single source, the source is indicated at the beginning of the text section. Otherwise the references will be placed immediately after the referenced material.
A list of the acronyms used throughout the report is located just before the appendices on page 144.
The report follows work done on previous semesters, [DMCP12] and [MTP12]. As a result some figures and text parts may stem from previous semester reports, but updated to fit the current state of the project. When previous work is reused in the report references will always be present, and will be done as [xx, p. yy] or [xx, p. yy✎], where xx is the reference to the project in the bibliography, yy is the page number in the corresponding report, and ✎ indicates if the content has been edited.

Prerequisites This report is directed at students of the Master specialization in Networks and Distributed Systems or similar education. Knowledge of the Open Systems Interconnection (OSI) Model, the TCP/IP suite and a general understanding of the Internet's topology is recommended. Furthermore a general understanding of the concepts of probability theory and stochastic processes is assumed.

Appendix This report includes a number of appendices. These consist of material produced in connection with the project, but only considered as complementary.

Attached CD The attached CD contains parts of the literature used, source code of the measurement system and the MATLAB data processing code. When references to the CD is made, it is done as follows: *folder/file* , which indicates the location relative to the root of the CD.

Acknowledgements We would like to thank the following for support during the project: The *Networking and Security group*, Department of Electronic Systems at Aalborg University, with special thanks to the *Collaborating Living Labs*[1] project financed by NordForsk - for providing eight data subscriptions to the cellular network.
The *Mobile Devices group*, Department of Electronic Systems at Aalborg University - for providing Android smartphones.

[1] `http://www.coll-livinglab.org/`

Table of Content

Chapter 1

Introduction

This report describes the continued work on the Network Mapping (NetMap) system, which is a Android based network performance measurement system aimed at using crowdsourcing for collecting measurements. The purpose of the work on this semester is to extend the development, analysis and correctness of the NetMap system. The report presents an analysis of a data set collected between Brovst, Aalborg and Aarhus, analysis and evaluation of current measurement methods, and development and evaluation of alternative methods for measuring the available bandwidth.

The previous work includes [DMCP12] where the initial NetMap system was developed, as a simple system collecting, among other data, Round-Trip Time (RTT) measurements by using crowdsourcing. Previous work also includes [MTP12] where the system, and the concept, was developed further with the goal of collecting different measurements for use in a network performance map, including analysis of influencing factors on the measured performance parameters.

This chapter will serve as a brief presentation to, and overview of, the work done previously, and is mainly directed at people unfamiliar with this. It does however also function as a general introduction to the project, and is recommended for everyone reading the report.

1.1 Background of the Network Mapping System (NetMap)

[MTP12, p.4✎] The NetMap System has its origin in the 8th semester project *"Mismatch Probability Modelling of Central Decision Process with Multiple Information Providers and Empirical Network Models"* [DMCP12], undertaken during the spring of 2012, at Aalborg University. The system was developed to collect data about cellular networks, with the goal of being able to model selected network characteristics. The idea was to use smartphones to perform the measurements and data gathering, which would reduce the costs compared to developing a dedicated hardware platform to perform the task. The NetMap system was developed to gather information on 2G and 3G networks, and models were fitted to the measured data.

The work on the system was continued in the fall of 2012 on the 9th semester, with the project *"Measuring Network Performance & Quality using Android Based Devices"*. For this project the NetMap application was remade to include functions for measuring additional network metrics. Instead of focusing on the crowdsourcing approach as previously, which resulted in a lot of measurements spread across the country, the measurements for this project were performed in a more limited scenario, both regarding number of devices and area size. The collected data was analyzed with the goal of finding the most useful metrics to consider, when trying to classify cellular networks for use in a Network Performance Map (NPM), as well identifying the impact of time of day, location, and cross traffic. This was done by grouping the data in 100x100 meters tiles, and evaluating the difference between tiles.

1.1.1 The Concept of NetMap

NetMap was made with the purpose of capturing network information by utilizing crowdsourcing, as the currently available information about cellular networks is limited to that published by the Internet Service Provider (ISP). In recent years more detailed information about the networks has been requested from different instances, and the goal of NetMap is to provide this information in a useful way to a broad variety of users or clients.

In Figure 1.1 the concept of the system has been simplified to consist of only two points, capturing and displaying data. Capturing can be done by anyone with an Android device and access to a cellular network, by downloading and running the NetMap application. For their effort they will receive information about their measurements directly on their devicec, but as it can be seen in the figure they will also have the possibility to access more detailed information later, through a web interface which displays a Network Performance Map (NPM).

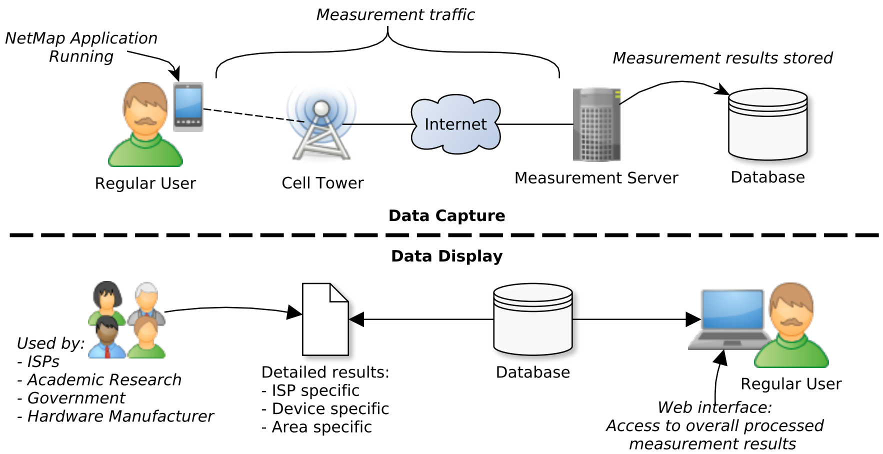


Figure 1.1: The concept of NetMap seen from both a capture and display perspective, showing how data can be presented to different groups of users.

The information available through the web interface should contain only relevant information, which has been processed such that it is easy to understand and use by the public. To further increase the use of the system more detailed information will be available, which is directed at the network providers or companies who require detailed information about the cellular networks. Unlike the publicly available information, the additional level of detail is what potentially makes NetMap profitable, as the information can be sold as a service which will also help keep the system operational. Finally the raw measurements can be made available to studies or research, as empirical information is difficult to obtain, but often required to ensure the correctness of obtained results.

The philosophy behind NetMap Initially the application was made to be simple and easy for the users to have running in the background on their devices, and the data consumption was kept as low as possible such that users would keep the application running.

During the development additional measurements have been added, which has caused data usage to increase. This contradicts the philosophy of keeping the data usage at a minimum, but was deemed necessary to get a more comprehensive set of network metrics to work with.

Also the distributing of the application is very simple, as it can be downloaded through the Google Play store. This is a key feature as well, as it is expected that a lot of potential users will not consider using the application if they need to go through a tedious installation process.

1.1.2 The Future of NetMap

In the previous work with the NetMap system, measurements have been done either at a national scale, or using a predefined route and selected locations for stationary measuring. During the work in this report, it is expected that the scale is increased to accommodate for multiple routes, while still being limited compared to the national wide measurements.

Before the system can be deployed nationally with the new metrics, there are a lot of issues that will need to be considered. Since the initial release a lot of functionalities have been added to the application, causing it to consume more data than it did initially. As the system relies on crowdsourcing, and a higher data usage will make it less likely that people will volunteer, the application should be optimized to reduce the usage to as little as possible. Alternatively some reward could be considered in return for volunteering, e.g. free data corresponding to the amount used by the application.

Another issue is the server scalability. The server capacity quickly became an issue when the system had a lot of concurrent users. To resolve this issue a suitable server structure will have to be developed, and deployed at locations so that no unnecessary bias is created by measuring at different servers.

1.2 Motivation for Continuous Development of the NetMap System

During the development work on the NetMap system, the project has received a great deal of attention from various sides. This interest includes a series of articles by Ingeniøren at `ing.dk` [ing12a] [ing12b] [ing12c] [ing12d] [ing13], radio interview, TV interview [24N12], and an invitation to speak at a conference titled *"Internet Quality - More Than Bandwidth"*[Tel12]. This indicates the need for a solution to properly measure the network performance, in contrast to the theoretical network coverage maps provided by the service providers. Furthermore, based on this publicity, the interest for the system from regular private users increased drastically, indicating that regular users have great interest in testing and measuring the performance of the product they are paying for - access to the network. All this serves as a good motivation for continuing the work on the NetMap system.

Other motivation for continuing the work is that with such a system it is possible to gather an enormous base of information and data about cellular networks. This could then be used by service providers as a diagnostics tool to identify issues with their networks, or by researchers who need data from actual networks in developing models for simulations.

All this is possible with the NetMap system, as it is developed for Android devices, which offer a wide range of information through the Android API, and the fact that Android devices are very wide spread effectively forming a huge potential user base for the system. Similar info used by the NetMap application can also be obtained for Apple products[ios13] or Windows Phones[win13], but is not considered in this report.

Chapter 2

Problem Domain and Description

In this chapter, to start with, the overall goal of the NetMap project will be described, followed by the problem domain undertaken in this report. Based on this the problem description will be given, which will describe the specific goal of this report, along with the focus area of the problem. Finally an outline of the rest of the report will be given.

2.1 The NetMap Project Overall Goal

As described earlier, the goal of the previous project[MTP12] was to evaluate parameters for use when generating a NPM. This was based on the allegation that a network coverage map showing the expected rates in an area, which are based on models describing the possible received signal strength in that area, is not satisfactory with the huge development in cellular network usage and numbers of smartphones. Therefore the goal is to generate a NPM based on parameters measured by the NetMap system, which more directly are describing the performance of data transmission on cellular networks.

The NPM should, as concluded in [MTP12], be generated by the use of tiles. A tile is here meant as a square area bounded by geographical coordinates, in which the measurements performed are grouped together. The size of the tiles should be assigned according to the number of measurements in the area, such that for each tile an overall performance can be seen along with some information regarding the fluctuations of the performance. The overall performance could for instance be the mean maximum throughput and the mean RTT, measured in the area. The information regarding the fluctuations could be information about the rates during the rise time of throughput or the spread of the measured rates, and for RTT the typical spread of the RTTs in a measurement sequence.

By assigning the size of the tiles according to the number of measurements, then, based on the assumption that the amount of users in urban area typically is high, urban areas will be covered by small tiles, and vice versa for rural areas. This will increase the level of detail in the areas where the most users reside, such as urban and small town areas. By doing this the NPM will be able to more accurately show the performance of the network in an area based on the utilization of the network.

Alternative usage of the NetMap system could be as a diagnostics tool, both of the network performance and of the performance of individual device models. Furthermore, the measurements performed with the NetMap system could form a general knowledge base regarding real cellular networks and mobile devices, for use by researchers e.g. when modeling the networks or in simulations of these.

2.2 Problem Domain

Currently the NetMap system is not in a final and release worthy state, but rather in a development and testing state. It currently performs as many measurements as possible and is therefore not concerned with usage of resources such as power and data (see Chapter 3 for system description). This state is however very well suited for testing and optimization of the system and of measurement methods, which is the focus area in the current development state of the system. Overall there are 2 problem areas of the NetMap system that should be attended:

The first area is related to the influence on the measured parameters caused by the context of the device while performing the measurements. In the analysis performed in [MTP12], the study of the context of the device was limited to the impact of local area and time of day. Due to this limitation it was chosen to only gather the measurements in a limited scenario. In a new analysis this should be extended to include other influencing parameters, based on the what scenario the new data is gathered in.

The second area is related to one of the main goals in a final system, which is low data usage while measuring. In the current state of the NetMap system the method consuming the most data is the throughput estimation, which is implemented as a bulk transfer capacity method, transferring as much data as possible for a period of time. This method was chosen as it is simple, in that it uses brute force to estimate the available throughput, why it is the method with greatest potential for improvement in terms of resource consumption.

2.3 Problem Description

Following is the problem description of this project, which is based on the description of the problem domain given earlier. After the problem description the focus areas of the problem will be elaborated on.

> *Based mainly on points noted as possible future work in [MTP12], it should be evaluated what impact the following points have on the performance of the network connection measured with the NetMap system:*
>
> - *Movement speed, i.e. when moving in car while measuring.*
> - *Area type, such as rural and urban.*
> - *Device model, when simultaneously measuring at same location.*
>
> *Furthermore, is it possible to implement a low data usage method on Android devices to accurately, when compared to a bulk transfer capacity method, estimate the available bandwidth.*

Focus Area

The problem description covers 2 problem areas: extended analysis of measurements and optimization of measurement method.

The first area will focus on making the results and conclusions drawn from the measurements performed by the system stronger. This will be done through analysis of the impact caused by the context of the device. This will include analysis of the impact of movement speed on measurements, the impact of the area type that measurements are performed in, and the impact of different device models on the measurements. This will help acquire a deeper understanding of the capabilities of the NetMap system and of the measurements gathered by this.

The second area will focus on developing, implementing and testing alternative low data usage methods to replace the bulk transfer capacity method. In doing this the alternative methods should be evaluated and compared to the already implemented method. This will be done based on measurements performed with both methods implemented on Android devices measuring real life cellular networks. It is chosen to compare actual implementation and measurements done on actual networks, as opposed to network simulations, to get an understanding of the true performance of the methods in a real environment.

2.4 Report Outline

The remaining of the report is structured in 3 parts, which are described in the following.

Part I In this part first a detailed technical description of the NetMap system will be given. Following this is an analysis of measurements gathered by car between Brovst, Aalborg and Aarhus, with main focus of the impact of movement speed on the performance. Next is an analysis of the impact of the area type of where the measurements are gathered in, on the performance experienced. Then an analysis of the impact of different device models on the performance measured. Finally the RTT and Throughput measurement methods will be looked into, in order to describe the process going on while performing the measurements.

Part II In this part the focus will be on the alternative methods to the bulk transfer capacity method. Here the novel methods will be described conceptually, implemented and compared to the bulk transfer capacity method, based on measurements performed on real cellular networks. The methods include a Packet Pair Technique (PPT) and a combined Probe Gap Model (PGM) and Probe Rate Model (PRM) technique.

Part III In this part the conclusions reached throughout the report will be combined and discussed, and the possible future work of the project will be stated.

Part I

Current State Of The NetMap System and Analysis of New Measurements

Overview of Part I

In this part the content will focus on the current implementation of the NetMap (Network Mapping) system.

In Chapter 3 a thorough walk through of the NetMap system will be given, where the design and implementation will be presented in great detail. Each measurement module will be described using message sequence diagrams to give a better overview of their flow. Also the scheduling of the different modules will be described, giving a complete picture of the system flow. Firstly, a quick overview of the system will be given. Also some of the important changes made based on knowledge from [MTP12] will be described, and an introduction to the measured parameters and how they are defined will be given.

Chapters 4, 5 and 6 will present some analysis of measurements captured using the NetMap system where the focus is on movement speed of the devices, the impact of different area types have on the performance metrics and lastly measurement results that different devices can produce.

In Chapter 7 some of the measurement modules are looked deeper into, to provide a more detailed view of what is happening during some of the different measurements.

Chapter 3

Introduction to the NetMap System

Much of the material in this chapter is based on the description of the NetMap system given in the previous project [MTP12]. Therefore, some of the figures, tables and descriptions are the same as in the previous project, while others have been updated according to the current state of the NetMap system.

In Figure 3.1 it is illustrated where the NetMap system conceptually is going to operate in terms of clients, or measurement devices, compared to the measurement server, and the measured network.

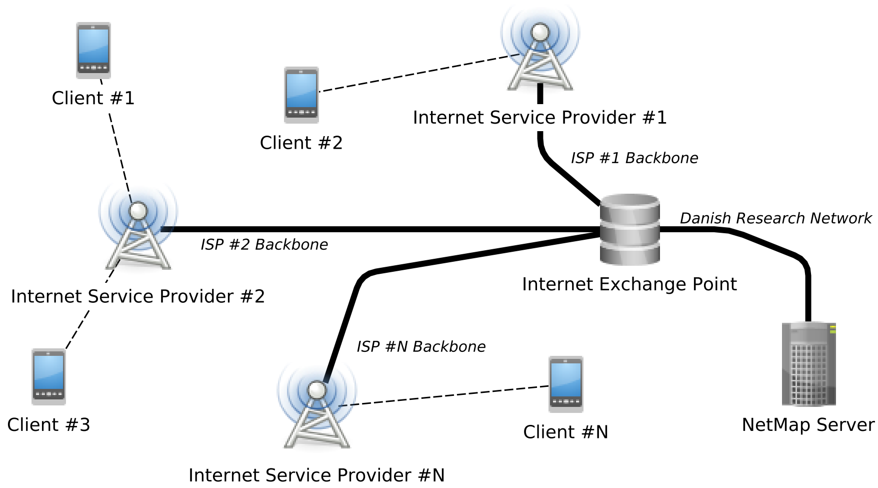


Figure 3.1: [MTP12, p.5]The network location of the devices while they measure the network between them and the NetMap server.

The main task for the system is to gather network measurements, it is therefore important to make the data gathering unbiased such that no ISP is prioritized. This means, as seen in Figure 3.1, that the measurement point must not favor any of the ISPs. The Clients (mobile devices) are scattered around on different ISPs and will be conducing the actual measurement. All the ISPs are connected to a common Internet Exchange (IX) point and then to the rest of the Internet. By choosing a network which does not favor any of the ISPs, the gathered measurements can be compared. Currently the NetMap measurement server is located on the Danish Research Network which does not favor any of the ISPs since no cellular devices can be directly connected, making it an ideal place the measurement server. This is elaborated on further in Section 3.4, page 16.

3.1 Notes and Changes to the NetMap System

In this section the changes performed on the system compared to the state of the system during the previous project [MTP12] will be highlighted.

Timeout values: The timeouts for the different parameters were made more strict, based on the discussion in [MTP12], and in the following table the new and old timeout values can be seen.

Parameter	Previous Timeout	Current Timeout
Connectivity	2 sec	1 sec
TCP RTT	10 sec	1 sec
UDP RTT	10 sec	1 sec
Throughput Download	2 sec	2 sec
Throughput Upload	2 sec	2 sec

The more strict timeouts are based on that when regular users use the network, for instance for web browsing, a response time higher than 1 second will seem very slow. Therefore a 1 second timeout is chosen for the Connectivity, TCP and UDP RTT, while a 2 second timeout is chosen for the Throughput tests.

For the Connectivity the 1 second timeout will mean that when the network is really slow it will be represented as no connection, why as the Connectivity is running continuously it will serve as an always-on network health indicator.

For TCP and UDP RTTs the 1 second timeout will for most of the time being have no impact, as for 3G the RTTs typically are lower than 200ms, while for 2G EDGE the RTTs typically are around 300-400ms. However, some devices are capable of falling back to GPRS as well which may cause even slower RTTs than for EDGE, possibly causing many of the RTTs to time out.

So if many RTT timeouts are observed in an area, this would indicate that the area may lack 3G coverage, why the network naturally will perform poorly.

For the Throughput a slightly higher timeout is chosen because of the higher packet size used, why when on slower connections such as EDGE the connection is much more likely to time out than the case is for RTTs where very small packets are used.

File Transfer: In order to automatically transfer the files containing the measurements to the server a module to handle this is added to the system. For now, this module uses the device ID to check the credibility of the devices before receiving files. This can easily be extended to utilize stronger security, but for now this is sufficient.

Application layout and output: The layout of the application has been changed from including only a few buttons (Figure 3.2), to showing an overview of the most recent results (Figure 3.3), and showing a console with various output from the application (Figure 3.4).

Figure 3.2: The layout of the application during the previous project [MTP12], where focus was getting the system up and running.

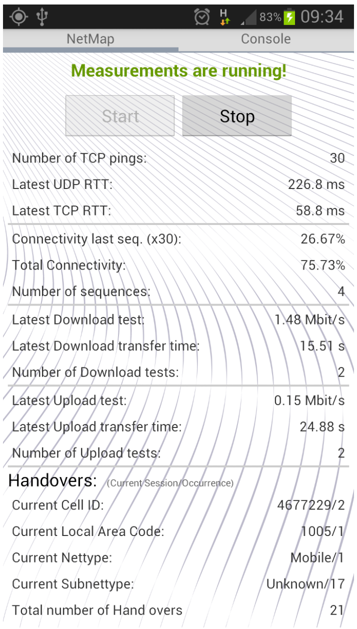


Figure 3.3: The main screen in the new layout of the application.

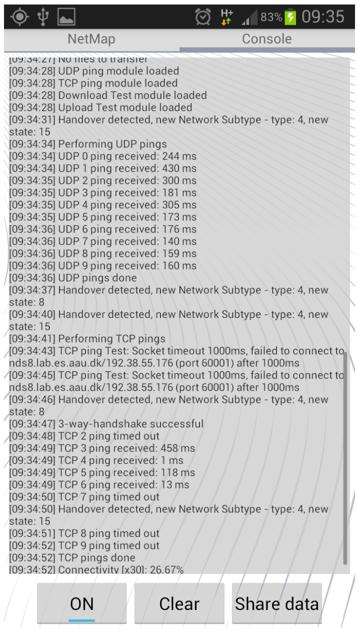


Figure 3.4: The console screen in the new layout of the application.

Added parameters to Meta Data: In the meta data a few new parameters have been added compared to the previous project [MTP12]. These are as follows:
MCC: The Mobile Country Code, which is a unique code that is different for each country around the world [wik13e].
MNC: The Mobile Network Code, which when combined with the MCC can be used to identify the carrier of the network [wik13e].
BER: A Bit Error Rate indicator as provided through the Android API [and13].
screenState: The state of the screen of the device, indicating whether the screen is on or off.
isCharging: The charging state of the device, indicating whether the device is charging or not.

3.2 Overview of Measured Parameters

In this section a preliminary overview of the parameters measured in the NetMap System will be given. In doing this a short technical description of the parameters will be given along with the argumentation for measuring them. Furthermore a short description of alternative measurement methods will be given.

3.2.1 Round Trip Time (RTT)

The RTT is, as defined by The Internet Engineering Task Force (IETF) in [For12], the time it takes from a request is transmitted from the device to the server, and until the reply is received from the server at the device. The RTT is measured using both TCP and UDP, and only by measuring on payload packets, that is for TCP the connection is initiated before the measurements are performed. For each RTT test, both TCP and UDP, a sequence of 10 requests/replies are performed.

The RTT is of interest because many applications operate using non-local data, why most requests and actions performed by the user will cause a connection being made and data being transfered. Therefore the delay induced by the network on the response time is of interest. The UDP and the TCP protocol were chosen based on them being the most widely used protocols in Internet traffic. The ICMP protocol could also be used to calculate the RTT, or the Connectivity (see Section 3.2.3), but this protocol is not used to transmit user data between devices, why it would be difficult to compare it to other use cases. In [LWG06] a series of experimental results show that there are a significant difference between measuring connectivity using TCP or ICMP, with TCP showing a rather large advantage.

When measuring the RTT it should be noted that it is not necessarily the same route that is being measured, i.e. the request to the server might travel by one route, while the reply might travel via another route. Alternative methods that could be considered includes one-way delay, and measurements using protocols such as ICMP packets. One-way latency measurements would make it possible to compare the uplink and downlink of the connection, but requires good time synchronization between client and host. Unfortunately, some Android devices can not directly be synced using NTP without getting root access. Furthermore, by avoiding the need to synchronize the time it will greatly decrease the power consumption, due to not having a scheme like NTP running in the background.

3.2.2 Throughput (Bulk Transfer Capacity)

The Bulk Transfer Capacity, or Throughput, as defined by the IETF in [For12], is the average amount of data that can be transmitted over a link per second, also known as bit rate or bits per second (bit/s). As the measurements are performed at the application layer, data represents the bits received excluding headers or overhead, and time is measured as the time between the first received bit and the last. The Throughput is measured for both Uplink and Downlink using TCP, which implements congestion control. Each measurement is performed for 15 seconds, during which as much data as possible is transfered. After the 15 seconds a final packet is transmitted that ends the measurement, after which the duration of the measurement and the amount of data transfered is stored at the receiving end.

It is interesting to know the throughput that a connection can deliver, especially in connection with content based applications such as streaming or cloud services. Furthermore, as the Throughput is the most common performance parameter used to describe a connection, both by users and by service providers, it is the parameter giving the best perception of how well a connection or network performs. Based on these arguments throughput is considered the most important parameter that is being measured by the NetMap system.

The implemented method for measuring the throughput is the most primitive procedure, where as much data as possible is pushed through the connection (roughly 9 MB for a Download measurement on a 5 Mbit/s connection), making it rather expensive in data usage. The throughput of a connection can also be estimated by finding the available bandwidth, which can be estimated by other methods that are cheaper in resources, for instance by exploiting knowledge about interarrival times of packets or using one-way delay variations.

3.2.3 Connectivity

The Connectivity is based on the IETF definition of the *two-way interval temporal connectivity* in [For12]. In the definition N request packets should be transmitted uniformly distributed between time T and time $T + \triangle T - W$, where W is the time to wait for a reply for each packet. For each sequence of N packets, if a single reply is received within the waiting time W the Connectivity sequence returns true, and false otherwise.

The Connectivity measurement in the NetMap system is only transmitting 1 packet, then it waits for the reply with a timeout of 1 second, after which it waits 1 second and starts the sequence over, which is done 30 times before the results are saved. In this way the Connectivity requests are not uniformly distributed, but they are not completely periodic either, because each request is spaced 1 second plus the varying RTT of the Connectivity request/reply. In this implementation each request/reply will be evaluated to be either true or false, which is stored as the result in a vector indicating whether the individual request/reply sequence was a success or not. Furthermore, it can be said that the Connectivity parameter, despite it running continuously, is a cheap parameter only using approximately 210 kB pr hour.

The reason for measuring this parameter is to have an indication of whether the device lost connectivity during the other measurements, while at the same time giving an indication of the continuous performance experienced. With the more strict timeout defined for the parameter the Connectivity can be interpreted as an indicator of when the network performance drops below some defined threshold, which in this case means that it takes more than a second to do a request/reply sequence. The threshold, or timeout, of 1 second, is chosen as a response slower than this would be noticeable for the average user. The biggest advantage of this parameter is that it is running constantly, why more or less every bigger fluctuation of the network performance will be caught. Because Connectivity is running constantly it will inevitably influence the other parameters, but as the conditions are the same for all the parameters, and the fact that Android devices more or less always will have other services doing communication in the background, the impact of Connectivity will be minimal.

The Connectivity measurements could also measure RTT or delay variation, as Connectivity operate in the same manner as the UDP RTT. This would give a continuous indication of the response time on the network but is for now left to the UDP ping task.

3.2.4 Handovers

The handovers are measured, or rather registered, event based, and no data is transmitted in order to measure them, which differs the handovers from the other parameters. All the handovers registered are done by using a *Phone State Listener* in the Android API, which when the connection state or the signal state is changed, reads the state of the *Cell ID (CID)*, *Location Area Code (LAC)*, *Network Type*, and *Network Sub Type*, and notifies the Handover module which then checks the individual states and evaluates whether it is a new state.

Furthermore, as the Phone State Listener only updates the state of the device while the screen is on, a watchdog was implemented that with a 2 second interval sends a screen-on signal immediately followed by a screen-off signal. This causes the Phone State Listener to perform a state update, effectively forcing any handovers to be registered, without the screen being turned on.

It was chosen to register the handovers because these were expected to impact the performance, e.g. in connection with a CID handover it is expected that the performance of RTT, Throughput and Connectivity will drop. By registering the CID and LAC handovers the individual cell towers can be identified and described in terms of performance as seen from the connected device. The performance of the tower is then, besides the RTT, Throughput and Connectivity, also expressed via the Network Type and Network Sub Type, which indicates what connection types the network supports.

As the handovers are registered through the Android API some limitations apply, exemplified with the neighboring cell list, which is a list of the neighboring cells to the one currently connected, including CID, LAC and signal strength. This information is only available on some devices depending on the implementation of the specific modem chipset. Furthermore, the handover types are chosen based on them being made available through the Android API. Another thing to note is that the states of the different handover types are registered on a much higher layer than where they actually happen, i.e. close to the hardware. This means that the update of states are delayed compared to real time notifications close to the hardware.

3.3 System Architecture

In this section the architecture of the system will be described, both regarding the application and the server. In doing this an overview of the modules and the communication in the system will be given.

The server and client application implementation can be located on *code/NetMapSystem* .

3.3.1 Application Architecture

In Figure 3.5 the architecture of the application can be seen.

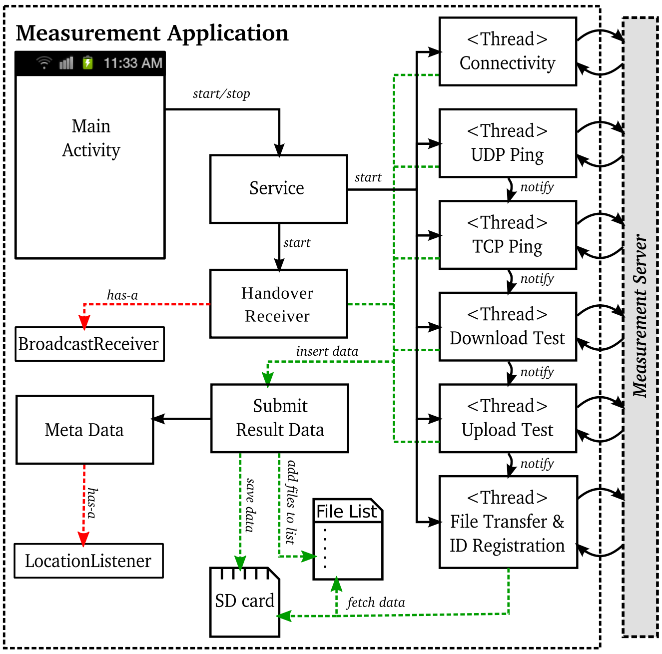


Figure 3.5: [MTP12, p.26✎]Architecture of the measurement application, which is running on the client devices.

From the figure the communication between the modules can be seen, which is used to ensure the scheduling of the processes, described in Section 3.7.

From the figure it can be seen that the service initiates all the modules, that are doing the measurements and the transferring of files. The connectivity is started, and kept running during the lifetime of the service, and the Handover Receiver is started which listens for handovers. Both are stopped when the Main Activity is stopped.

Then the schedule is initiated, that is, the UDP Ping is started, which when finished saves the results and notifies the TCP Ping module. The TCP Ping follows the same procedure, and when finished notifies the Download Test, then the Upload Test and finally the File Transfer module, which also handles the ID registration at the server.

Each of the measurement modules saves the gathered results through the Submit Result Data, which adds Meta Data to the results, and saves it all to files on the SD card. Each time new results and meta data is appended to a file, a check is performed of how many lines is added to the file. If the amount of lines in the file exceeds some limit, the file is added to the File List, and a new file for future results is created. The File List contains the files that are ready to be transmitted to the server. The File List is then checked each time the File Transfer module is executed, and the listed files are transmitted to the server.

3.3.2 Server Architecture

In Figure 3.6 the architecture of the server can be seen, including the modules that run on the server. From the figure it can be seen that the server has the modules equivalent to those on the application running.

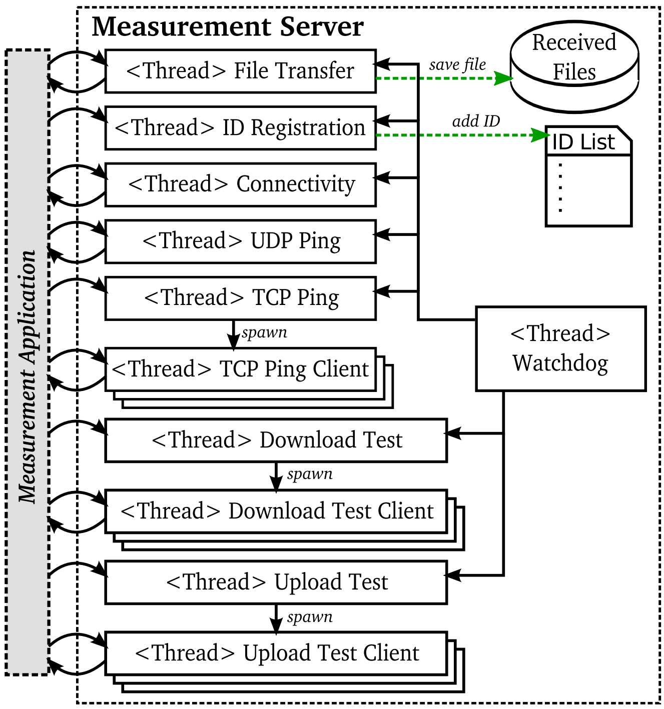


Figure 3.6: [MTP12, p.27✎]Architecture of the measurement Server.

Looking at the modules from the top and down, first is the File Transfer module that receives the files from the device, validates each transfer with a checksum, and saves the file at the server. Then the ID Registration module, that handles and updates the list of device IDs that are registered at the server. This is a simple security to be used in the File Transfer module. Then is the Connectivity module, that runs continuously and responds to the Connectivity module at the client application. The UDP Ping module is, like the Connectivity module, also continuously running and responding to the UDP Ping module at the client application. These two modules are not spawning threads to handle the connections because they both use UDP, and are implemented as simple FIFO queues. The following 3 modules, TCP Ping, Download Test, and Upload Test all use TCP, and all spawn client threads to handle the incoming connections, in order to be able to handle multiple connections at once. This is to ensure that the main threads are not blocked.

From the figure it can also be seen that a watchdog is implemented that ensures that if any of the modules are terminated unexpectedly, then they will be restarted again. This improves the durability of the server, and thereby the measurement system.

3.4 The Server Setup

In this section some hardware specification of the server will be listed. The server has been provided by the IT department at Aalborg University and is running as a virtual machine using VMware Virtual Platform[vmw13].

The hardware specification can be located on the cd: *hardware_info.html* . Below is a summary of the specification on the server:

Processor:	
Product:	AMD Opteron(TM) Processor 6274
Size:	2200MHz
Number of cores:	16
Memory:	
Size:	2 Gb DRAM
Type:	PC3-12800
Network:	
Product:	82545EM Gigabit Ethernet Controller
Speed:	1 Gbit/s

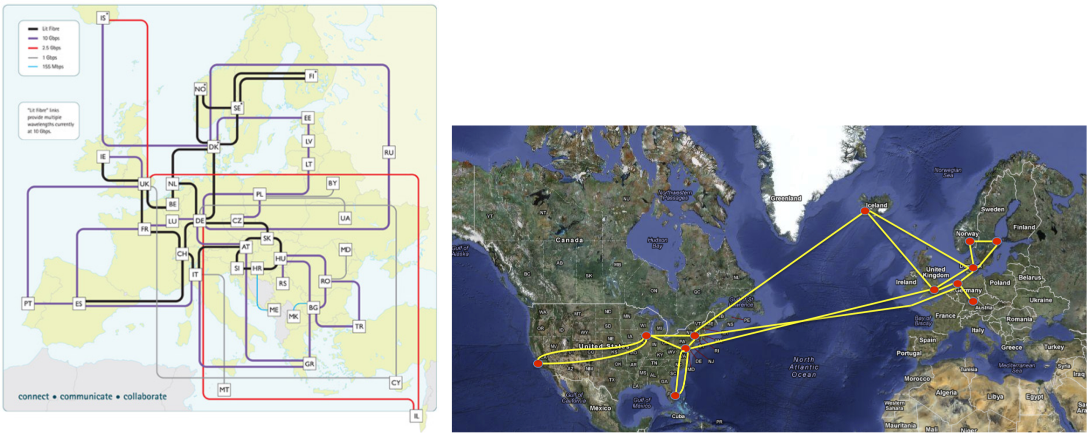
The server is a simple virtual machine used throughout this particular project for small scale tests. The server specification is deemed sufficient for these tests as no more than up to 4 devices will be running at the same time. Should the system be used by significantly more devices the server must also be upgraded to insure that all requests are handled equally to not create any bias in the measurements.

3.4.1 Server & Network Location

In the design of the server one of the important factors is the actual physical placement of the measurement server. Because the clients have the possibility to use different ISPs as carriers the location of the server should not favor any of the carriers. This could be achieved by placing a series of measurement servers on each of the carriers internal network to avoid any biased measurements.

Since the network distance to the measurement server depends on the carrier this can have an impact on the outcome of the measurements. In order to say how being on different network influence the measurement, the topology of the Internet will be described shortly with focus on Denmark as this is where the measurements are conducted.

[DMCP12, p.46] The Internet is a hierarchical structure consisting of smaller (regional, national or international) networks which are connected to the ”rest” of the Internet through Internet Exchange (IX) points. This means that for devices on different networks to be able to communicate, the communication traffic needs to go through the IX, even though the devices physically might be right next to each other. In Denmark there are 2 main IXs points, in Lyngby and in Ballerup [dix12], where a number of networks are connected to, which are connected to the rest of the Internet through other networks such as GÉANT and NORDUnet, see Figure 3.7.



(a) GÉANT [gea12]

(b) NORDUnet [nor12]

Figure 3.7: Topologies of 2 different international networks, connected to the danish IXs.

In the current implementation the server is located on the Aalborg University network, which is connected to Forskningsnettet that is connected to the danish IX in Lyngby. This can be seen on Figure 3.8

Figure 3.8: Topology of Forskningsnettet, which is connected to the danish IX in Lyngby. [fsk12]

[DMCP12, p.48✎] This means that whenever a service is connecting to the measurement server all the data must go through the danish IX to enter Forskningsnettet. So the distance from the devices to the server only varies depending upon what ISP the device is using. Forskningsnettet is a good location for the server due to no mobile cells being directly connected, why traffic from all carriers must go through the same point, making the carrier networks comparable. For instance, for the devices in Denmark using the carrier Hi3G, the traffic needs to go via Hi3G's network to the IX Netnod in Stockholm [net12] before reaching danish IX and Forskningsnettet, where if the carrier is TDC the traffic only needs to go via TDC's network to the danish IX in Lyngby before reaching Forskningsnettet.

3.5 Measurement Modules

In this section a thorough description of the different measurement modules will be given. The description will focus on how individual modules operates and communicate with the server. The purpose is to illustrate how the measurements are gathered and what choices have been made during the design process.

Each module is designed and implemented as a standalone module allowing them to be executed individually. This allows to add or remove any of the implemented modules since none of them are dependent upon each other. This is the case for all except the ***Data Storage*** (See Figure 3.5) module which is used by the measurement modules to save the data. Furthermore, when ever a module is storing its data a series of meta data is also logged and attached to the data. This is explained in greater detail in Section 3.8.

3.5.1 Connectivity Test

The purpose of the Connectivity module is to return a boolean value which indicates if a connection can be established with an external host (the server). The module is designed to transmit a request every one second, this is illustrated in Figure 3.9. A request is sent to the server, when the request is received by the server it then transmits a reply. If this reply is not received within a specific timeout period (listed just below) the result will return a false boolean value indicating that a connection to the server could not be established, and if a reply is received within the timeout period it will return true. Whenever one connectivity test has occurred (request sent and reply received) a 1 second wait time is introduced before the next connectivity is scheduled. This sequence will happen 30 times before the results are stored for later analysis, which is repeated while the system is running.

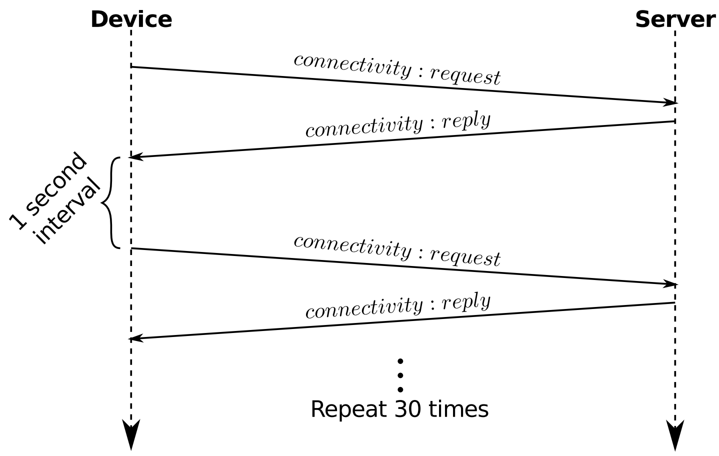


Figure 3.9: Connectivity Message Sequence Chart.

In recap the module has the following parameters specified:

- *Interval:* 1 second
- *Timeout:* 1 second
- *Packet size:* 60 bytes

With the given information the time it takes for a set of connectivity measurements to be performed can be estimated using the following equation

$$T = 30 + \sum_{i=1}^{30} \Delta t_i \tag{3.1}$$

where T is the total time used for a set of measurements and Δt is the round-trip time of each packet, both measured in seconds. With a timeout value of 1 second for each packet each set of measurements can take a maximum of 60 seconds. It should also be noted that as the round-trip time is influencing the time it takes between each consecutive sequence, some randomness is introduced to the starting time of each sequence, causing the measurements not to be completely periodic.

3.5.2 Round-Trip time Test: UDP

In the implementation of the UDP ping sequence, the measured RTT is logged and calculated as illustrated in Figure 3.11. A RTT measurement consist of one request packet sent to the server which then replies this request, when this reply is received by the device the time it took is captured. This is repeated a total of ten times, where each request will wait for its reply in order to continue. If a packet is lost during transmission, the socket will automatically timeout after a specified time.

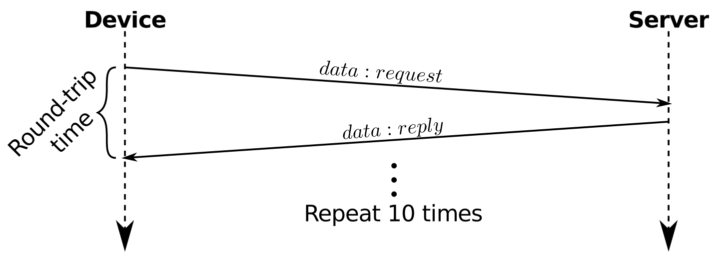


Figure 3.10: UDP Ping Message Sequence Chart.

By using this strategy where each reply is received before the next request is transmitted, one packet does not influence the others, i.e. not exceeding the link capacity. In recap the module has the following parameters specified:

- *Socket timeout:* 1 second
- *Number of pings:* 10
- *Packet size:* 23 bytes

3.5.3 Round-Trip time Test: TCP

For the TCP test sequence a few more steps have to be considered, due to the control packets induced by the reliability of the protocol. Transaction between two components using the TCP protocol consist of three steps, a *3-way handshake*, the exchange of data and a connection termination. Such sequence is illustrated in Figure 3.11 below.

These three steps will cause some small delay, and must therefore be considered when capturing the RTT measurements. TCP also incorporates a timeout for both the 3-way handshake and the data transmission. In the implementation, if a timeout occurs during the 3-way handshake a new connection is attempted, this can happen 3 times, after which the ping sequence is stopped. However, if the handshake is successful the module will start doing 10 ping attempts one by one where the RTT is measured. When the ping sequence is finished the connection is terminated. This sequence is ensured in the implementation by using a blocking connect function.

For the TCP ping test the ***Nagle's algorithm***[Nag84] has been disabled to avoid clustering of packet sequence. By disabling this algorithm the network is not allowed to reduce the number of small packets transmitted by the device, evidently avoiding unnecessary transmission delay hence providing more precise RTT measurements.

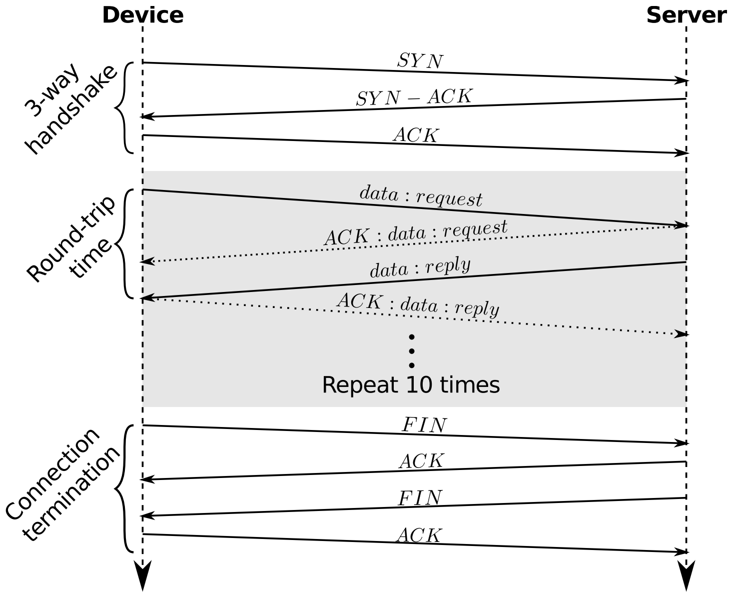


Figure 3.11: TCP Ping Message Sequence Chart.

Furthermore, the TCP protocol requires that for each data packet transferred, an acknowledgement (ACK) packet is sent back. Therefore the TCP ping sequence needs to first wait for the *ACK-packet*, and then just after the data reply is sent/received. This means that even though it only was the data transfer that needed to be measured, the application will have to wait for the *ACK-packet* as well. This is seen in Figure 3.11. As a result of the reliability of TCP, in the RTT measurements an extra delay is added by the *ACK-packets* that has to be considered.

However, upon further inspection of the packet sequence by using Wireshark on the server, it was noted that in the TCP ping sequence some of the requests and some of the replies were followed by individual ACK-packets, while for other request and reply packets the ACK was incorporated in the following reply or request packet. This was most often the case for the first request/reply sequence, where as for the remaining 9 sequences the ACKs were incorporated.

The time between the first request packet and the ACK packet for the reply packet was the same as the time between request and reply packets with incorporated ACKs. This, despite the behavior, indicates that whether the packets are acknowledged in connection with the next packet or separately, they are not further delayed on the server.

But in case where the individual ACK-packet is used and this gets lost in transmission, a retransmission will happen causing the RTT to increase. But due to the strict timeout set these samples will most often be discarded.

In recap the module has the following parameters specified:

- *Socket timeout:* 1 second
- *Number of pings:* 10
- *Packet size:* 23 bytes
- *Connection retries:* 3
- *Nagle's algorithm:* Disabled
- *3-way handshake and connection termination:* Not part of the RTT measurement

The TCP and UDP ping modules both makes 10 pings in a row when ever their task is scheduled, to get more values for calculation of results. By conducting the 10 pings the cases where an extreme RTT is measured, which could give the wrong indication of the average performance, can be weighed down by the last 9 pings and so forth. This is analyzed further in Section 7.1, page 65, Ping Method Analysis.

3.5.4 Throughput Test: Download

The Throughput tests are described from the devices point of view, meaning that the device is downloading from the server and uploading to the server. Both throughput tests use the TCP protocol and must therefore first initiate connection to the server using the *3-way handshake* as described in the TCP ping test. As in the TCP ping test the initial connection has a maximum of 3 attempts before it cancels the throughput test.

The purpose of the download throughput test is to download as much arbitrary data from the server in a specified time interval as possible. Figure 3.12 shows how a download throughput test communicates with the server. In the first step the initial connection is seen, and after a connection is established the device transmits a packet containing information of how long the throughput test should run. When this packet has been accepted by the server, the server then starts transmitting TCP packets as fast as possible, only limited by the link capacity enforced by the network. The device logs the starting time, and starts counting the amount of bytes received until the server disconnects (which is after the specified run time sent by the device), when the server disconnects the time is logged and these information is then stored for later analysis. The payload size of the TCP packets has been set to 1448 Bytes and is transmitted as fast as the TCP protocol allows. Because the TCP protocol is a streaming protocol it will split the payload into appropriate chunk sizes defined from the Window Size offered by the receiver which is set during the continuous transmission [Ste94, chap. 20].

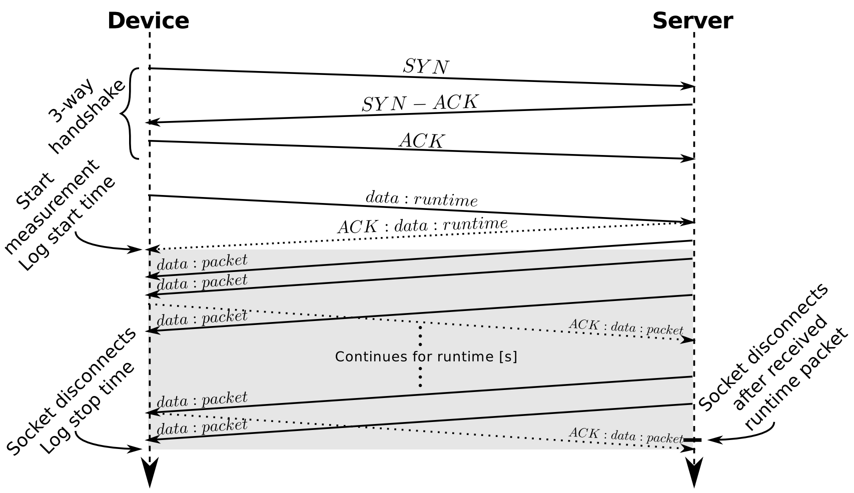


Figure 3.12: Download Throughput Test Message Sequence Chart.

In recap the module has the following parameters specified:

- *Socket timeout:* 2 seconds
- *Transmit time:* 15 seconds
- *Connection retries:* 3
- *Download TCP payload size:* 1448 bytes

- *Nagle's algorithm:* Enabled

3.5.5 Throughput Test: Upload

The upload throughput test is very similar to the download test but with the roles of both the server and client reversed, meaning that now the device transmits packets as fast as possible to the server. The whole setup procedure is the same as it first is doing the initial *3-way handshake*, when the connection is established the test run time is transmitted to the server, after which the device starts uploading data. When the server accepts the run time packet the start time is logged, and then the device transmits as much data as possible and the amount is logged. After 15 seconds the upload stops and when the last packet is received at the server the time is logged and the server calculates the average throughput used in the upload session. This result is then transmitted to the device for later analysis, and if the result exchange fails the device simply calculates a local average using the result gathered from the device itself.

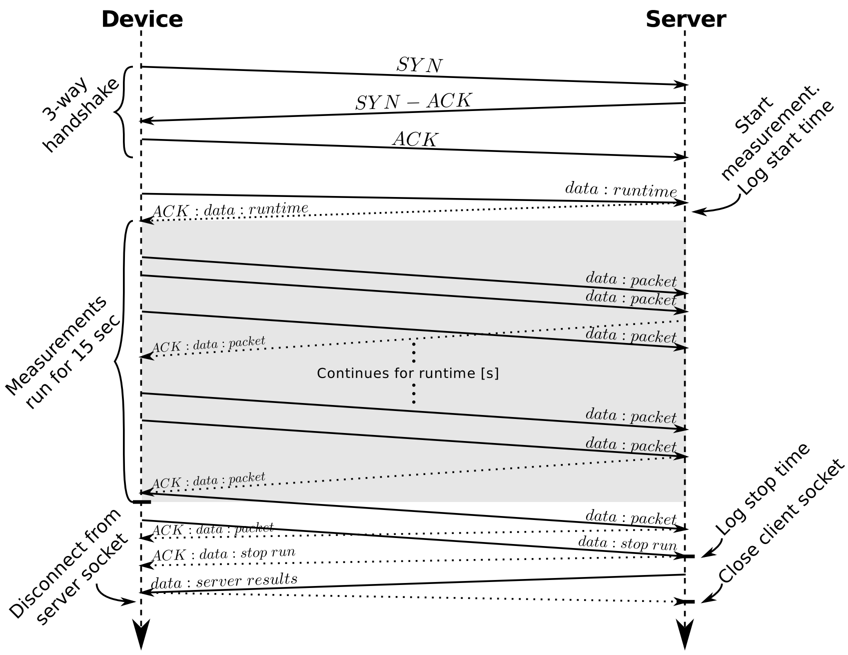


Figure 3.13: Upload Throughput Test Message Sequence Chart.

In recap the module has the following parameters specified:

- *Socket timeout:* 2 seconds
- *Transmit time:* 15 seconds
- *Connection retries:* 3
- *Upload TCP payload size:* 1448 bytes
- *Nagle's algorithm:* Enabled

3.5.6 Handover Events

The handovers happen as events and they are also captured event based. When the events happen, a listener registers this and notifies the handover class. In the handover class the states of the different types are checked for updated status, and marked as a handover if there is a new state.

Handovers of different types may occur at the same time, but are registered and stored individually. Following are descriptions of the different handover types, and their background.

CID: The Cell ID is a unique ID that is used to identify the different sectors or antennas at the base stations or cell towers[wik13b].

[WGS10]There are 3 types of CID handovers: Hard, soft, and softer. Hard handover is when the connection between device and antenna is dropped or lost before connection to another antenna is established. Soft handover is when connection to an antenna at a new base station is established before the connection to the current antenna is released. Softer handover is when the device establishes connection to a new antenna, or sector, but at the same base station as the currently connected antenna.

[FSW03]The handovers are controlled by the Radio Network Controller (RNC) which routes the data traffic towards the base stations that are active in the handover process. The soft and softer handover types makes it possible, and smooth, for the transport protocol on top of the IP protocol to keep a stream or connection active during a handover.

LAC: [wik13d]The Location Area Code is a code that describes a set of base stations or cell towers, or a grouping of these. The amount of cell towers with the same LAC may vary a lot depending on the setup of the network, and the area type that the LAC is in. That is, in rural areas where there is big spacing between the cell towers, less towers are on the same LAC compared to in urban areas.

The LAC is used for routing the packets, or calls, to the part of the network where the device is. However, when many users are on the same LAC, the paging traffic increases. A paging request is broadcast when a router is looking for a route to a device. Therefore, the amount of cell towers to assign to a LAC is a trade off between effectiveness in routing and keeping the amount of paging traffic low.

Net Type: The Network Type is what kind of connection the device is using, that is whether the device is connected via WiFi, cellular, or no connection. But as the measurements only are performed using cellular connections, such as 2G and 3G connections, the Net Type handovers only occur when the connection is totally lost, that is no connection, and when the connection is reestablished.

Net Sub Type: The Network Sub Type are the connection technology used with the connected Network Type, that is in cellular cases EDGE, UMTS, HSPA, etc.. The Net Sub Type handovers that most often occur are UMTS and HSPA, which both are 3G technologies, but they differ in the effectiveness of utilization of the connection available, in terms of throughput and power consumption.

Registration of Handovers - Using Watchdog What generally characterizes these 4 types of handovers is that they happen because of external factors of the device. The CID depends on the cell antenna or tower that the RNC finds best suited for the moment and the location. The LAC depends like the CID on the location. The Network Type depends (mostly) on whether there is a connection or not. And the Network Sub Type depends on the capabilities of the network and the connected cell antenna.

Because of this the handovers are registered as events by the use of a listener. But in Android some phone state listeners do not update values while the screen is off, which is done to reduce power usage. This is for instance the case for the listener used for updating the states of the handovers.

This is handled by implementing a watchdog, called *DroidKicker*, which at a fixed interval sends a *screen-on* signal, immediately followed by a *screen-off* signal. This causes the device to request updates from all registered listeners, but this does also causes the device to turn the screen on and off again. This is a temporary fix which is used to ensure the updates from the listeners in the test version of the NetMap system. In a final version, a better method for obtaining listener updates should be used.

Currently the *DroidKicker* runs at a 2 second interval.

3.5.7 UDP & TCP Requests Management

One of the tasks that the server implementation must handle is the possibility of multiple request occurring during a small period of time. On Figure 3.14 below an illustration is shown where the difference between how the server handles TCP and UDP requests.

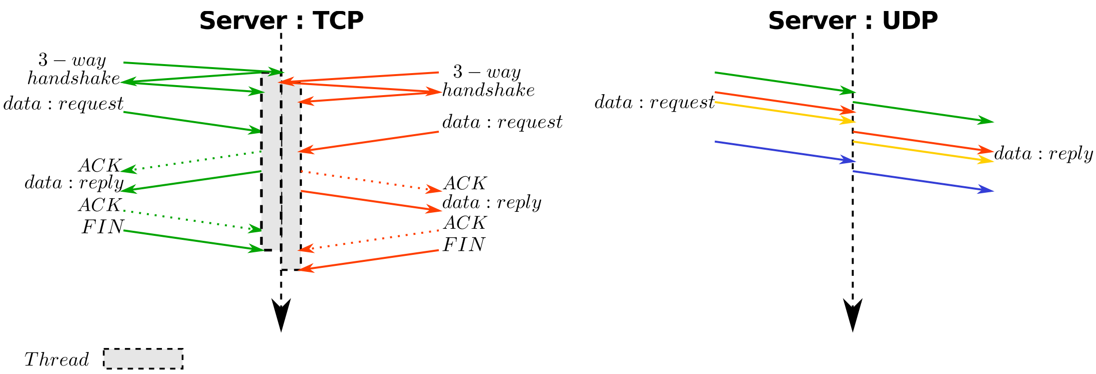


Figure 3.14: Illustration of how the server handles UDP and TCP requests.

The main difference between the two tasks is that the TCP is a multi-server environment, meaning that whenever a request is made, initiating with the 3-way handshake, the server spawns a new thread that is to handle the rest of communication between the server and a particular device, until the connection is terminated. On the other hand, the UDP task is first-in-first-serve (FIFS), which means that whenever a request packet is received by the server this packet will be replied to first, as illustrated on Figure 3.14.

3.6 File Transfer Module

In order to simplify the collecting of gathered measurement results from the devices, prior to processing of the data, a file transfer module has been developed. Each time a measurement module has added 500 lines to a result file, this file is added to the file transfer list. When the file transfer module is scheduled it will transfer all the files in the file transfer list.

First the module checks if the device has been successfully registered at the server, and if not it attempts to do so, before the actual file transfer takes place.

This is done, as indicated in Figure 3.15 using TCP, and by, pr file, first generating a MD5 checksum based on the content of the file, and transmitting this checksum to the server. Then the content of the file is transmitted to the server, that, when the file is done transferring, compares the checksum received from the client with at MD5 checksum generated by the server locally.

If the 2 checksums are equal, the file is saved and a status message is transfered back to the device, indicating the status of the file transfer.

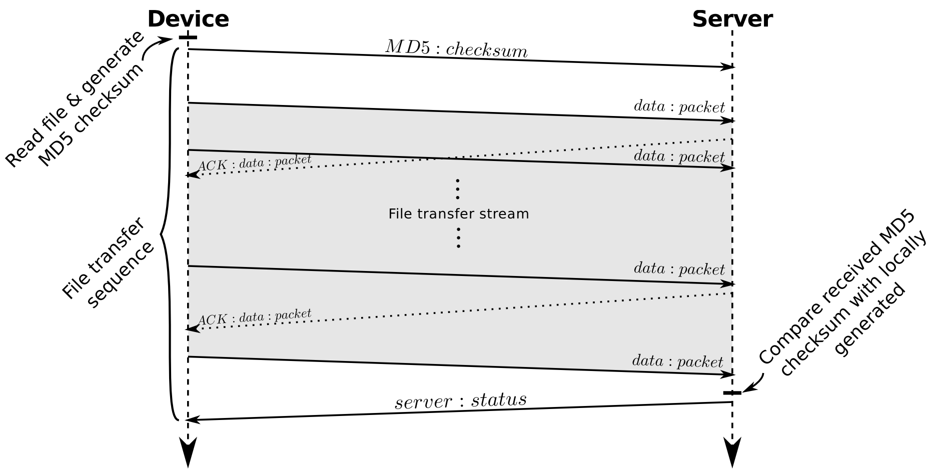


Figure 3.15: Sequence diagram showing the file transfer sequence including the MD5 checksum transfer, file transfer, and transfer status.

Back on the client, if the status message indicates a successful transfer, the file is deleted locally and the next file in the list is transfered. But if the status message indicates an erroneous transfer, the file is not deleted locally but added to the file transfer list again to be transfered the next time the file transfer module is scheduled.

In recap the module has the following parameters specified:

- *Max lines pr file:* 500 lines
- *Connection retries:* 3
- *Socket timeout:* 2 seconds

3.7 Scheduling of Application Processes

In this section the scheduling of the processes running in the application will be described and discussed.

The processes to schedule in the application are the processes measuring Handovers, Connectivity, UDP and TCP Ping, Throughput upload and download, and the File Transfer module, that transfers the measurement files to the server. The scheduling of the processes is seen in Figure 3.16.

From the figure it can be seen that the Handover process is event-based, as handovers happen based on the context of the device, why this process cannot be included in the scheduling. Furthermore it can be seen that the Connectivity is running continuously why this process is not part of the scheduling either. Then the 5 processes that are scheduled in a sequential manner can be seen. These processes, excluding the File Transfer, uses active measurements to measure their respective parameters. This means that they put load on the network, or connection, to test the capacity, delay, or other. Therefore it is necessary to ensure that no other active measurement is putting load on the connection while one of them is measuring. The File Transfer can also put load on the connection, depending on the amount and sizes of files it is about to transfer, why it must be included in the scheduling.

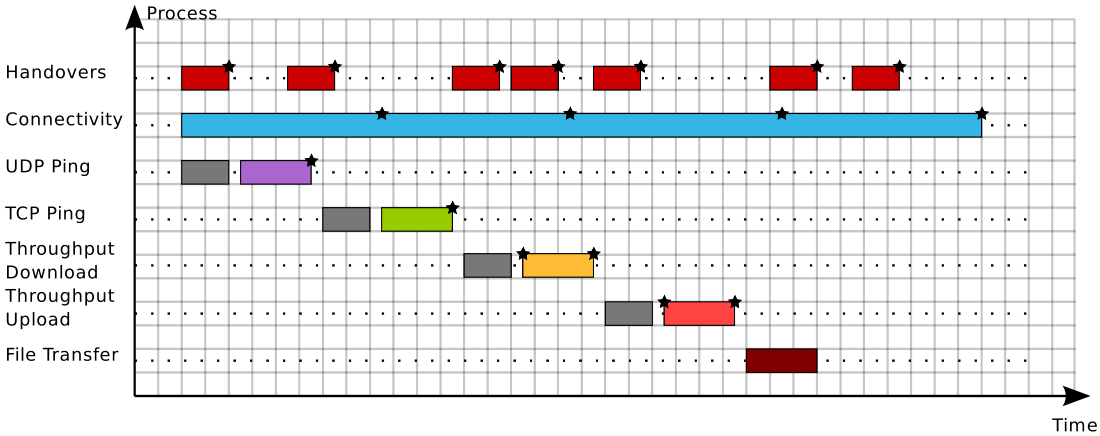


Figure 3.16: [MTP12, p.31✎]Scheduling of processes in the application. The grey items are pauses that are put in between measurement processes. The stars show where the meta data is collected and saved. The sizes of the processes are not to scale in terms of execution time.

The Handovers and Connectivity are not included in the schedule as the Handovers are measured, or counted, passively, and the Connectivity only transmits a single small packet every second, which is not expected to influence the other measurements. During a TCP throughput test the impact of the connectivity packet is small, but during the ping tests it will have a somewhat larger impact since one of the 10 pings could be skewed because of the connectivity packet. This will happen on a regular basis but as 10 pings are happening in succession this can be detected and dealt with in some statistical manner.

The scheduling is ensured by using inter process signaling in the implementation. Each process implements an object that is used to transmit notifications between the processes. Furthermore, the File Transfer process only checks for files to transfer once per schedule execution, why it for most of the time will do nothing.

Furthermore, to ensure that any packets or data left in socket buffers will not interfere with the other measurements, an inter test delay is used, which can be seen in the figure as the grey items. This inter test time is set to 5 seconds.

The current scheduling order shown in the figure can be interchanged or some modules can be left out if preferred. If some measurement metrics are more important than others the scheduling order can be created for such purpose. In the current implementation all metrics are deemed equally important which is why they are scheduled in a simple sequential order.

3.8 Data Protocol

[MTP12, p.28✎]This section describes how the measurements and meta data are stored in the result files, and is the exact same standard as defined in [MTP12], only with a few parameters added to the meta data (see Section 3.1).

The protocol will consist of three parts, one identifying the type of protocol, a part with the measurement data and lastly a part which contains the meta data. The protocol will be represented as a *String* using comma to separate values, and the different parts of the protocol will be split with a hash mark (#) to identify different sections of the protocol. This allows for a somewhat dynamic protocol that can be easily extended to add more values. The structure of the protocol is as follows:

```
Type # Network Measurements # Meta Data # (optional) Meta Data
```

The different fields of the protocol will now be described.

Type: There are four different types indicating which type of network measurements the data field contains. The four types are given as Integers and represent the following:

- **1** = *Ping*
- **2** = *Throughput Test*
- **3** = *Connectivity*
- **4** = *Handover*

Network Measurements: This field of the protocol depends on what value the type field has. This field can use four different patterns of values. These four possibilities are listed in tables in their chronological occurrence, along with a description and what data type they use.

Ping		
Value:	**Type:**	**Description:**
ping_{1-10}	Long[10]	RTT values in milliseconds.
protocol	Integer	Layer4 protocol used (1 = TCP, 2 = UDP).
size	Integer	Request packet size (bytes).
ipAddressGlobal	String	IPv4 address of the client used to connect to the server.
port	Integer	Port number used to establish connection to the server.
ipAddressLocal	String	The local IPv4 address at the client.

Table 3.1: Ping Data structure.

The IP addresses are displayed in human-readable notations, such as ”172.16.254.1”, and -1 if the client could not resolve the IP address.

Throughput Test		
Value:	**Type:**	**Description:**
bandwidthSubType	Integer	Display sub type of test (1 = Download, 2 = Upload).
testTime	Long	The run time of the test in milliseconds.
transferredData	Long	The amount of Bytes transferred in the test.

Table 3.2: Throughput Test structure.

When performing the throughput test the client will capture a series of meta data both before and after the test, hence the protocol will have the following structure:

```
Type=2 # Throughput Test Data # Meta Data # Meta Data
```

Connectivity		
Value:	**Type:**	**Description:**
conn_{1-30}	Integer[30]	Shows connectivity, 1 if the connection is available, -1 if not and 0 if test was not performed yet.

Table 3.3: Connectivity structure.

A Handover occurs whenever a previously logged state changes. There are four different types of handovers (*handOverType*) that can occur, and depending on what type of state that changes information is saved. The following four types can happen: The CID changes, the LAC changes, the device switches between WiFi and Mobile connection or the Mobile connection changes technology. A list of the possible technologies can be found at [Goo12b]. The field is structured as followed:

Handover		
Value:	**Type:**	**Description:**
handOverType	Integer	Identify the type of handover that has occurred.
newState	Integer	Display the new state data.

Table 3.4: Connectivity structure.

Where:

- ***handOverType = 1:*** New CID detected, updates *newState* to the new CID.
- ***handOverType = 2:*** New LAC detected, updates *newState* the new LAC.
- ***handOverType = 3:*** Device changes network type, updates *newState* to the new **Network type**.
- ***handOverType = 4:*** Network sub type changes, updates *newState* to the new **Network subtype**.

The different network types and subtypes can take on the following values:

Network types:	1 = WiFi, 0 = Mobile.
Network subtypes:	0 = unknown, 1 = GPRS, 2 = Edge, 3 = UMTS, 4 = CDMA, 8 = HSPDA, 9 = HSUPA, 10 = HSPA, 13 = LTE.

Meta Data: The meta data is static in its content, and is added at the end of the protocol as shown previously. The Meta Data field consists of the values defined in Table 3.5.

As it is not possible to predict how many neighboring cells a devices can see, this list of values must therefore be dynamic. To do this the following will be looped according to the amount of cells detected:

```
lengthOfList,LAC[1:length],CID[1:length],RSSI[1:length],NetworkType[1:length]
```

If the device does not detect any neighboring cells the first value will display -1, otherwise the above list will continue until the end of the list of neighboring cells.

Meta Data		
Value:	**Type:**	**Description:**
deviceId	String	Unique ID of the device given by the Android distribution.
time	Long	The current time in milliseconds when the meta data was gathered.
longitude	Float	Current location given in WGS84 coordinates.
latitude	Float	Current location given in WGS84 coordinates.
accuracy	Float	The accuracy of the fix in meters.
speed	Float	Speed of the device given in [m/s].
positionProvider	String	Tells which provider the GPS coordinates are received from (GPS, network).
LAC	Integer	Current Local Area Code.
CID	Integer	Current connection cell id.
PCS	Integer	Primary scrambling code, -1 if unknown.
RSSI	Integer	The currently measured Received Signal Strength Indication (RSSI) value.
MCC	Integer	Current Mobile Country Code.
MNC	Integer	Current Mobile Network Code.
operator	String	The name of the ISP.
networkType	Integer	Same type as listed under Hand Over.
networkSubType	Integer	Same type as listed under Hand Over.
phoneRadioType	Integer	Radio used to transmit voice calls (0 = none, 1 = GSM, 2 = CMDA, 3 = SIP).
dataConnectionState	Integer	Indicates the current data connection sate (0 = disconnected, 1 = connecting, 2 = connected, 3 = suspended).
cpuUsage	Float	Display the current CPU usage given in percentages.
totalRam	Integer	Display the total amount of RAM on the device, given in Megabytes.
availableRam	Integer	Shows the amount of free available RAM on the device.
batteryLevel	Float	The current battery level on the device, given from 0 to 1.
neighboringCellList	String	A list of the neighboring cells within proximity of the device. See below for description of this value.
BER	Integer	The currently measured GSM Bit Error Rate (BER) value, from 0 to 7 mapped to percentages from 0.14 to 18.10%.
screenState	Integer	The currently state of the screen (1 = on, 0 = off).
isCharging	Integer	Tells if the phone is currently charging (1 = yes, 0 = no).
model	String	Human-readable name of the device model.
manufacturer	String	Human-readable name of the device manufacturer.
sdkVersion	Integer	SDK version currently running on the device.

Table 3.5: Meta Data structure.

Chapter 4

Data Analysis with Focus on Movement Speed and Handovers

In this chapter an analysis is done based on measurements gathered in a car driving on a route between Brovst, Aalborg, and Aarhus. First an overview of the data used in the analysis will be presented along with a short description of what the analysis will cover, based on what was covered in the analysis performed in the previous project[MTP12].

4.1 Data Overview

The data is gathered in a car driving on a route between Brovst, Aalborg, and Aarhus, as illustrated on Figure 4.1. As it can be seen the route covers road from Brovst to Aalborg, from Aalborg via Hobro to Aarhus, and from Brovst via Løgstør and Hobro to Aarhus.

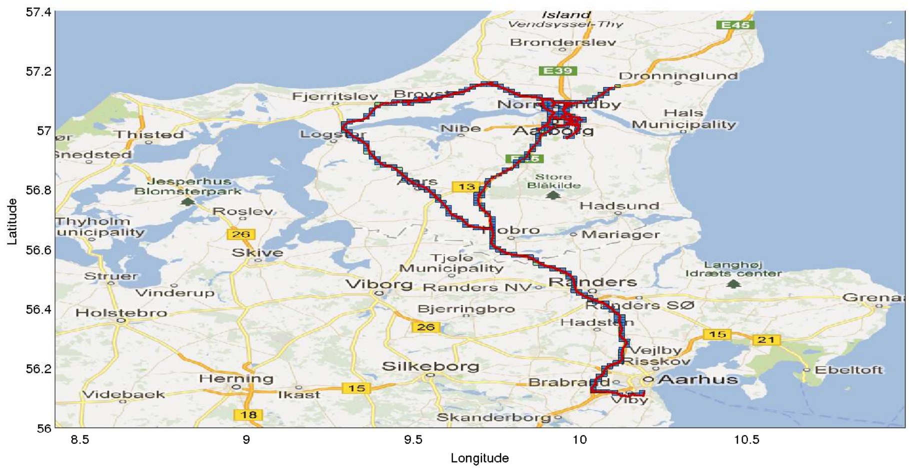


Figure 4.1: The measurement route between Brovst and Aarhus. The measurements are plotted on top of a Google Maps plot[goo12a].

This route has been chosen as it covers different speed limits and different area types, and that a car driving this route was available.

All the figures and calculations throughout this chapter are performed using MATLAB scripts, that can be found on the enclosed CD *part_1_matlab/* .

4.1.1 Recap of the Extent of Previous Analysis

In this chapter measurements from the route described in Figure 4.1 will be analyzed, where some of the analysis is based on procedures used in the previous project [MTP12], more specifically the concept of grouping the measurements in tiles according to location.

In that analysis the data was also evaluated over time, that is the hour of the day and day of the week, and especially for differences between the busy and non-busy periods. The busy period was chosen to be in the hours 7 to 17, and the non-busy the remaining hours of the day, based on the area type being semi-industrial. This showed some interesting results regarding the performance, namely that for some parameters the network almost consistently showed 20% better performance during non-busy period than during the busy period.

In Figure 4.2 the amount of measurements for this analysis according to what time of the day they were measured, can be seen.

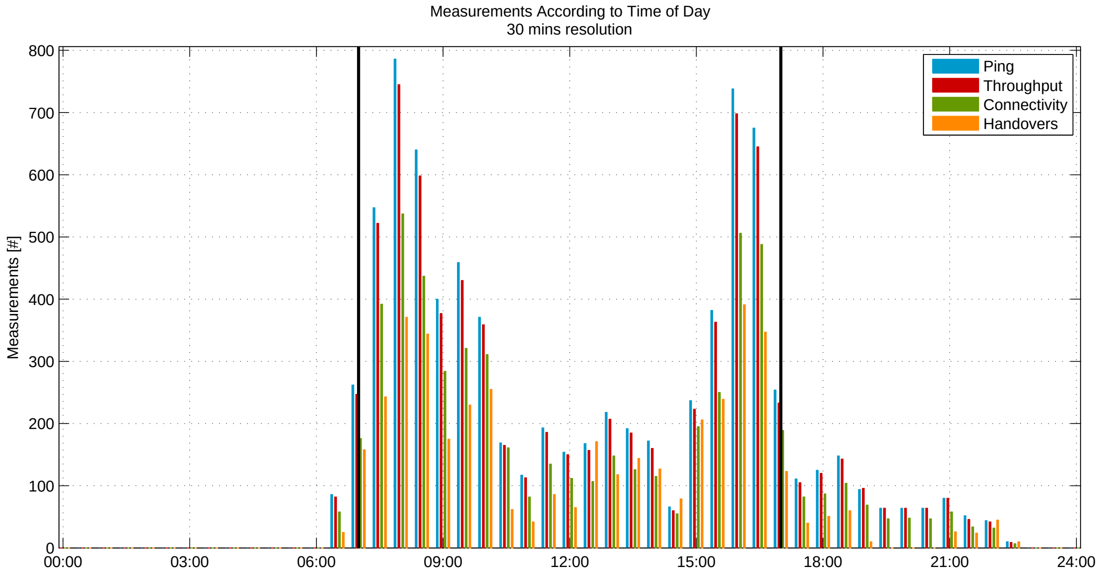


Figure 4.2: The amount of measurements pr parameter according to time of day.

From this it can be seen that the measurements are gathered in the hours between 6 and 22, with the majority of the measurements being gathered in the busy period between 7 and 17. Thereby the amount of measurements gathered in the non-busy period, from 17 to 7, is scarce, why the difference between the busy and non-busy periods will not be explored in this analysis.

In the analysis in [MTP12] the influence of location on the different factors on the performance of the network were analyzed by evaluating the measurements assigned to tiles on the measurement route. A tile is here defined as a square area bounded by GPS coordinates, where the measurements with GPS coordinates within the tile are assigned to it. In order to cover the entire route with tiles a grid covering the entire area where measurements are performed in is generated, after which the measurements are assigned to the tile covering them.

This showed some variations in performance of the various parameters. To complement the measurements performed while moving on the measurement route, 2 stationary tests were performed, where the devices were gathering measurements while being stationary. The data from these tests showed that the movement of a device have great impact on the experienced performance, as much better performance was experienced while stationary, even though the measurements were performed in the same area.

The analysis performed in [MTP12] showed that the type of area, the local location and the time of day all influence the experienced performance. However, the measurements were all performed while biking or walking on a small 2.5 km route, why the amount of handovers was low and the diversity in movement minimal. The influence of these 2 points is what the following analysis seeks to explore.

In Figure 4.3 the total amount of measurements, to be used in this analysis, gathered per tile along the route (seen in Figure 4.1) can be seen.

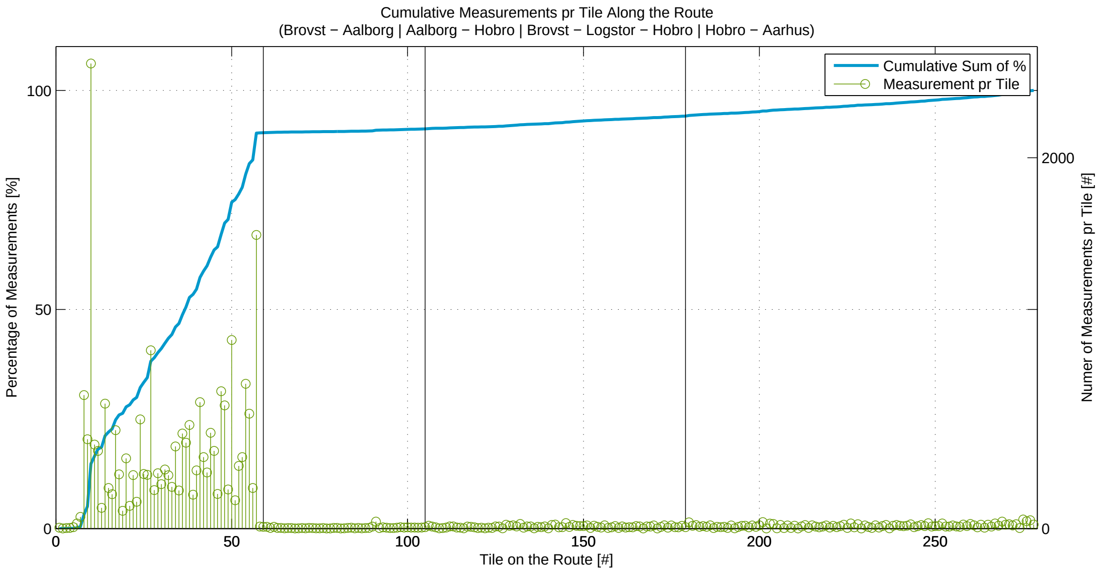


Figure 4.3: The amount of measurements gathered per tile on the route, with a tile size of 1000x1000 meters. The vertical black lines indicates the route division as stated in the figure title.

From this figure it is obvious that the amount of measurements is low in many of the tiles on the route. Therefore, in the following analysis, when evaluating the variation in performance caused by the location of the measurements on the route, only the measurements gathered on the part of the route running from Brovst to Aalborg will be used. But when evaluating the performance according to movement speed, all the measurements will be used.

4.1.2 Tile Size Analysis

In the following analysis the performance of the different parameters will be evaluated according to location. In doing this the measurements will be assigned to tiles, which is done to group the data points according to location. The choice of tile size will have some impact on the results and conclusions made based on the analysis; With bigger tiles, more measurements will be included per tile, but the fluctuations according to location will be reduced. Conversely with smaller tiles, less measurements will be included per tile, but the dynamics of the performance according to location will be more apparent.

The resulting amount of measurements per tile, according to tile size, is investigated by simulating different tile sizes and different amounts of measurements, uniformly distributed over the distance of the route. The result of this simulation is seen in Figure 4.4.

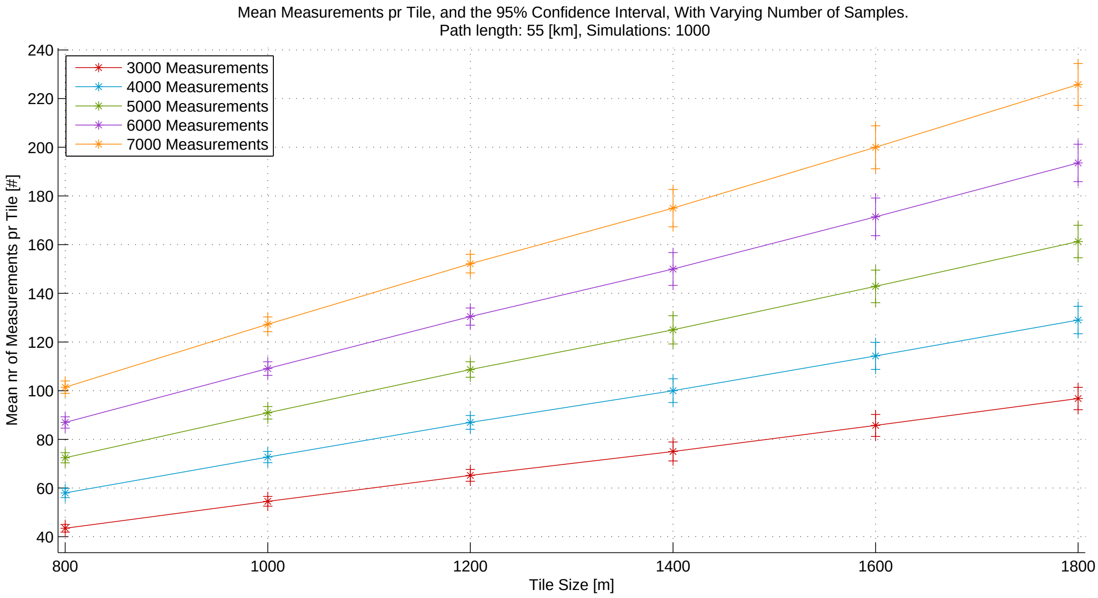


Figure 4.4: Simulations of the mean number of measurements gathered per tile, when uniformly distributed measurements along a route of 55 km (55 km is chosen because from Brovst to Aalborg University there is about 45 km so to account for deviations from the route a little longer distance is used). The markers (+) mark the 95% confidence interval, calculated using method described in Appendix B.1.

From the figure it can be seen that if around 4000 measurements of one parameter type are gathered along a route of 55 km, and the measurements are uniformly distributed, around 70 measurements of that parameter can be expected in each tile if a tile size of 1000x1000 meters is chosen.

Furthermore, as the throughput measurement is running for 15 seconds, and with most of the route being main road where the speed limit is 80 km/h a throughput test would stretch over 375 meters. So to limit the amount of measurements that stretches over 3 or more tiles, the tile size is chosen to be 1000x1000 meters.

4.1.3 Notes Regarding the Data used for Location Analysis

Following are some statistics about the gathered measurements on the route from Brovst to Aalborg, which the analysis will be based on. Note that the number of pings marked are groups of 10 pings, and the Connectivity points are groups of 30.

```
-------------------------------------------------------------------------
Current status of data points gathered on the route:
```

```
-----------------------------------------------------------------------------
Total TCP pings: 3689
Total UDP pings: 3730
Average TCP pings in tiles: 72.33, standard deviation: 71.89
Average UDP pings in tiles: 73.14, standard deviation: 70.00

Total Download measurements: 4433
Total Upload measurements: 4421
Average Download measurements in tiles: 86.92, standard deviation: 71.86
Average Upload measurements in tiles: 86.69, standard deviation: 69.79

Total Connectivity points: 5224
Average Connectivity points in tiles: 102.43, standard deviation: 105.10

Total Handover points: 3594
Average Handover points in tiles: 70.47, standard deviation: 70.38

Total amount of measurement points: 25091
-----------------------------------------------------------------------------
```

From this it can be seen that with the chosen tile size, the average number of measurements per parameter type per tile follows the values from the simulation shown in Figure 4.4.

4.1.4 Movement Speed of Devices During Measurements

In this section an overview of the movement speed of the devices while doing the measurements will be given. The speed is extracted from the location object in the Android API which also provides the GPS coordinates, given in m/s, and converted to km/h in this analysis for easier reference. Furthermore, the data is not limited to the part of the route between Brovst and Aalborg, but is from the entire route as illustrated on Figure 4.1.

In Figure 4.5 the amount of measurements of the different performance parameters according to different movement speeds can be seen.

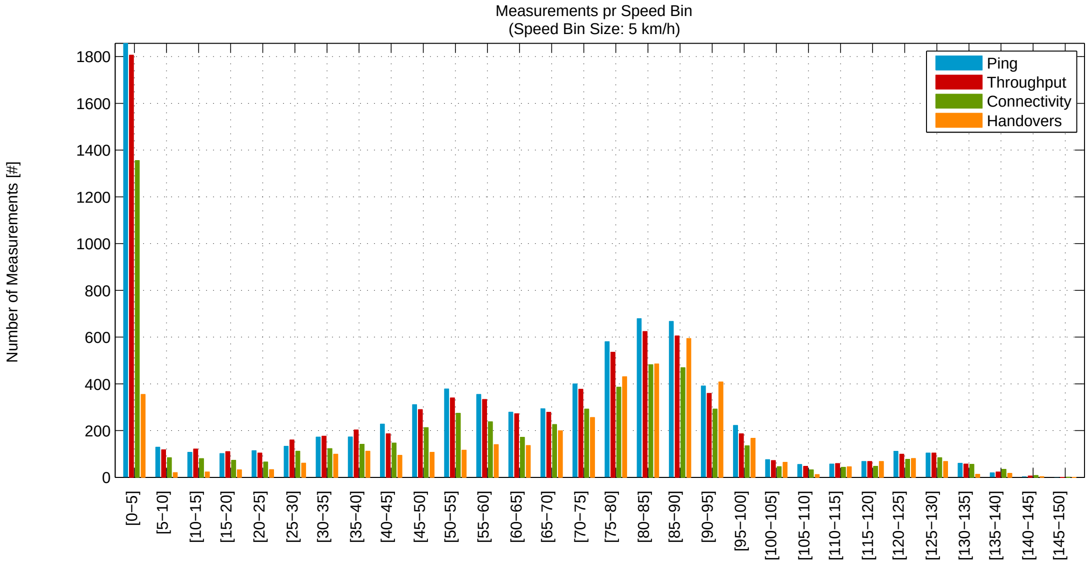


Figure 4.5: The amount of measurements per parameter according to movement speed.

From the figure it can be seen that there is a tendency that the measurements are gathered around 4 movement speeds. Firstly around 0 km/h, or stationary, which is measurements gathered when starting or stopping movement along the route, or when the measurement has not been stopped after ended movement. Secondly the movement speeds around 50 km/h, which typically are measurements performed in urban environment, where the speed is limited to this. The third movement speed is around 80 km/h, which typically are measurements performed on main road, where the speed limits are ranging from 60 to 90 km/h. The fourth and last movement speed is around 120 km/h which is on expressway.

In Figure 4.6 the movement speed while measuring averaged per tile is plotted. On the figure the vertical lines indicates the division of the route. The first part is the route is from Brovst to Aalborg, the second part is from Aalborg to Hobro, the third part is from Brovst via Logstor to Hobro, and the fourth part is from Hobro to Aarhus.

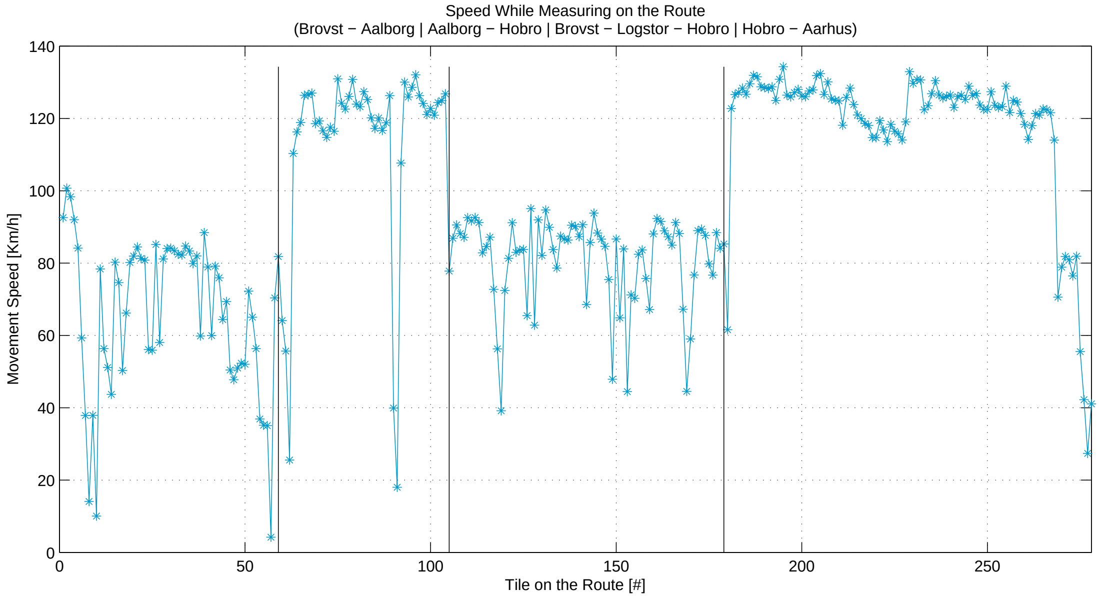


Figure 4.6: Movement speed measured along the measurement route shown in Figure 4.1. The 3 vertical lines indicate the division of the route as described earlier and indicated in the title of the figure.

From the figure it is apparent that the first part of the route mostly consists of main road (speed limit 80-90 km/h) and city (speed limit 50-60 km/h), while the second part mostly consists of expressway (speed limit 110-130 km/h). In the third part of the route there is only main road and city, and no expressway, while the fourth part, apart from the last 10 km, only is expressway.

In the following the parameters Round-Trip Time (RTT), Throughput, Connectivity and Handovers will be analyzed and the impact of the movement speed will be evaluated.

4.2 Analysis of Round Trip Times

In this section the RTTs measured on the route seen in Figure 4.1 will be analyzed and evaluated.

4.2.1 RTTs in Relation to Location

In Figure 4.7 the mean of the measured RTTs per tile along the route between Brovst and Aalborg can be seen.

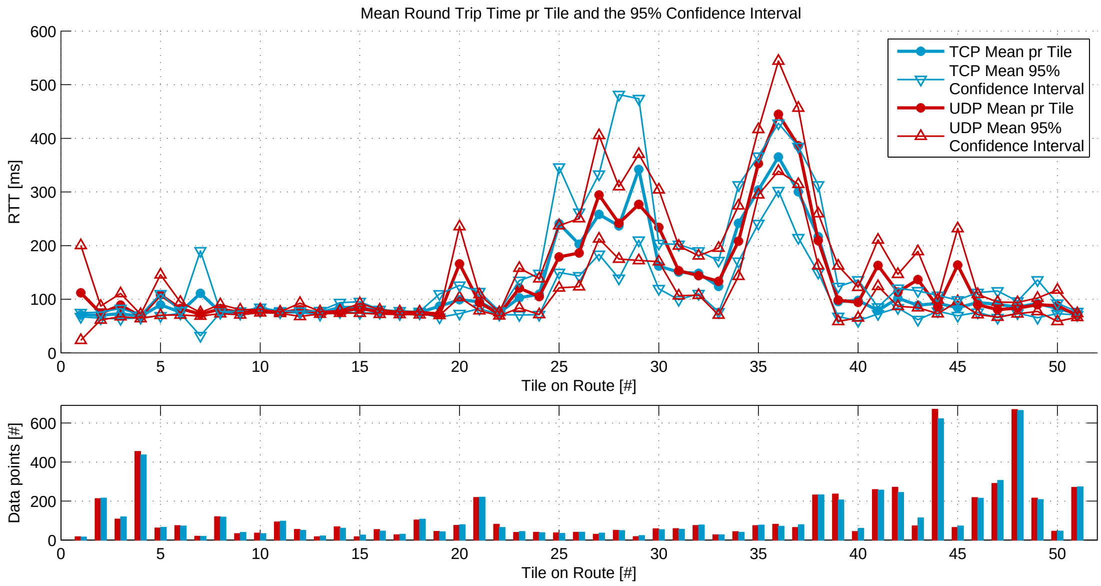


Figure 4.7: RTTs measured along the route between Brovst and Aalborg shown in Figure 4.1. The 95% confidence interval for the mean per tile is calculated based on the method described in Appendix B.1.

When looking at the initial part of the route, it can be seen that from Brovst to Aabybro (around tile 19), the RTTs are consistently low, while the following part from Aabybro to Nørresundby North (around tile 39) the RTTs are fluctuating with some very high peaks. The first of the big peaks is believed to be due to bad network coverage while the other big peak is believed to be caused by restrictions put on the wireless signals in the vicinity of airports, as the peak is right next to the runway of Aalborg airport.

From tile 39 to tile 51 the measurements are gathered in and around Aalborg and Nørresundby. From the RTTs in this part in can be seen that they are somewhat consistently low. This is believed to be due to the more dense coverage of the area by cell towers than in the more rural area outside the city. The more fluctuating RTTs in the city than in the more rural areas could be due to the higher density of users in the city, increasing the amount of cross traffic on the network.

The higher fluctuations in the urban areas compared to the rural areas are also apparent from the confidence intervals. For instance from tile 5 to tile 19 in the rural area, there are mostly fewer measurements than in the urban area from tile 39 to 51, but the confidence intervals are wider in the urban areas than in the rural. This indicates that the RTTs are more consistent in the rural area.

4.2.2 RTTs in Relation to Movement Speed

In Figure 4.8 the RTTs are evaluated according to movement speed of the device, that is the data is sorted according to speed in bins of 5 km/h. In the figure the 95% confidence interval for the RTTs in each speed bin is plotted in the top part and in the bottom part the amount of samples in each speed bin is illustrated. From this it can be seen that, when disregarding the stationary measurements at 0 km/h, the largest cluster of measurements are gathered while moving around 80 km/h, and the second largest while moving around 50 km/h. However, there are also some samples available between 5 and 35 km/h and from 100 km/h and up.

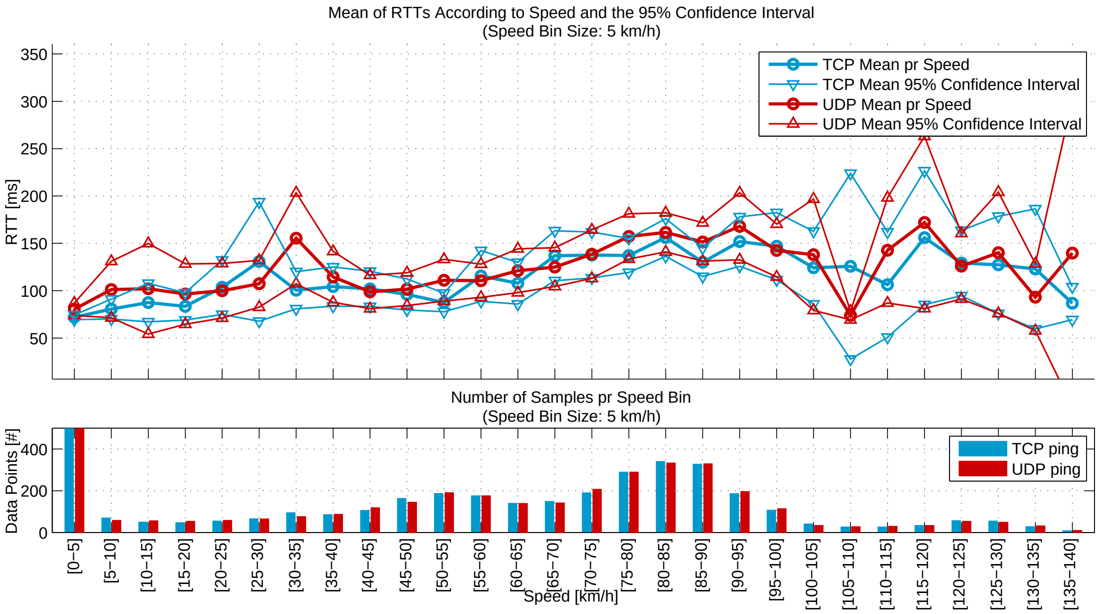


Figure 4.8: The mean of the RTTs sorted in 5km/h bins from the entire route. The 95% confidence interval for the mean per speed bin is calculated based on the method described in Appendix B.1.

When looking at the part of the figure where the main part of the data is represented, that is from 40 to 100 km/h, it can be seen that the mean increases slightly and almost linearly. In the same interval the 95% confidence interval is almost constantly the same size. This could indicate that when a device is moving faster it is more likely to experience high RTT values. This could be due to when moving at the higher speeds more handovers will occur, effectively delaying the pings. However, with the relatively low amount of measurements, and that the TCP RTT mean fluctuates a little, it is difficult to make rigorous conclusions.

To investigate this behavior further the RTTs are sorted according to some bigger speed intervals, which can be seen in Figure 4.9. The RTTs in these speed intervals are then sorted and plotted as histograms to be able to investigate the distribution of RTTs in the different intervals.

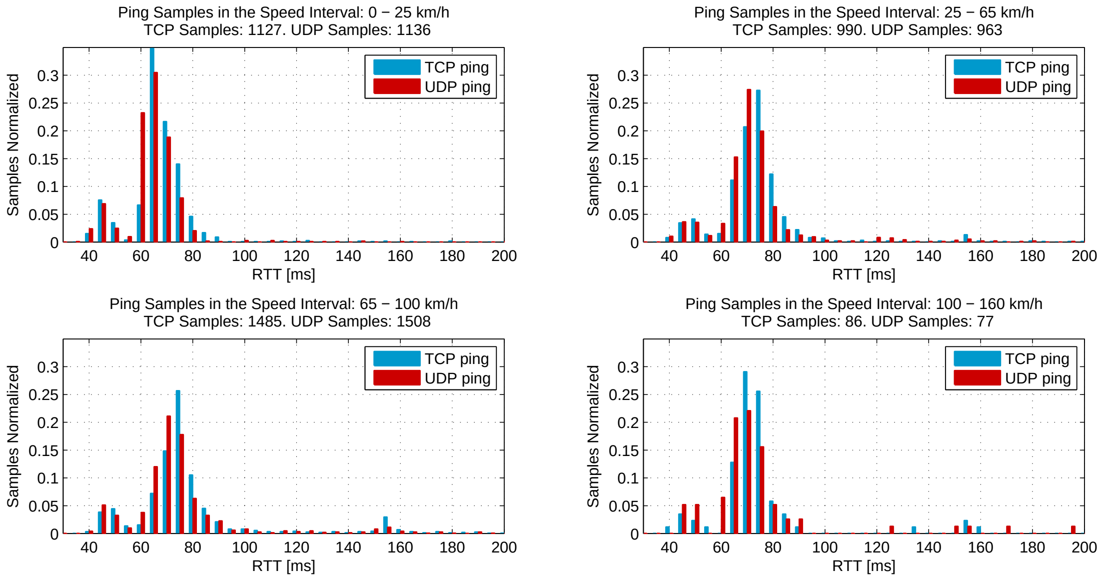


Figure 4.9: The distribution of RTTs within the 4 different speed intervals, plotted as histograms.

From this figure it can be seen that in the slowest speed interval the RTTs are mainly below 90 ms. The second speed interval also has the main peak below 90 ms, but a little lower than in the first speed interval, and there are some very small peaks around 120 and 150 ms. In the third speed interval the main peak is again below 90 ms, and again a little lower than both the first and the second speed interval. In the third interval there also is a small peak around 150 ms, which is a little higher than in the second interval.In the last speed interval the distribution looks much similar to the that in the third interval, but it should be noted that the amount of measurements in this interval is much lower than in the first three.

Generally the differences between the different speed intervals are small, why it seems that the impact of movement speed on the performance of RTTs is only very little, if any at all.

4.3 Analysis of Throughput

In this section the Throughput rates measured on the route seen in Figure 4.1 will be analyzed and evaluated.

4.3.1 Throughput in Relation to Location

In Figure 4.10 the average throughput rates measured in each tile along the route between Brovst and Aalborg are plotted, along with the 95% confidence intervals of the mean values. For the RTTs some tendencies were highlighted, and some of the same tendencies can be seen here with the throughput rates, however more obvious for Download than for Upload.

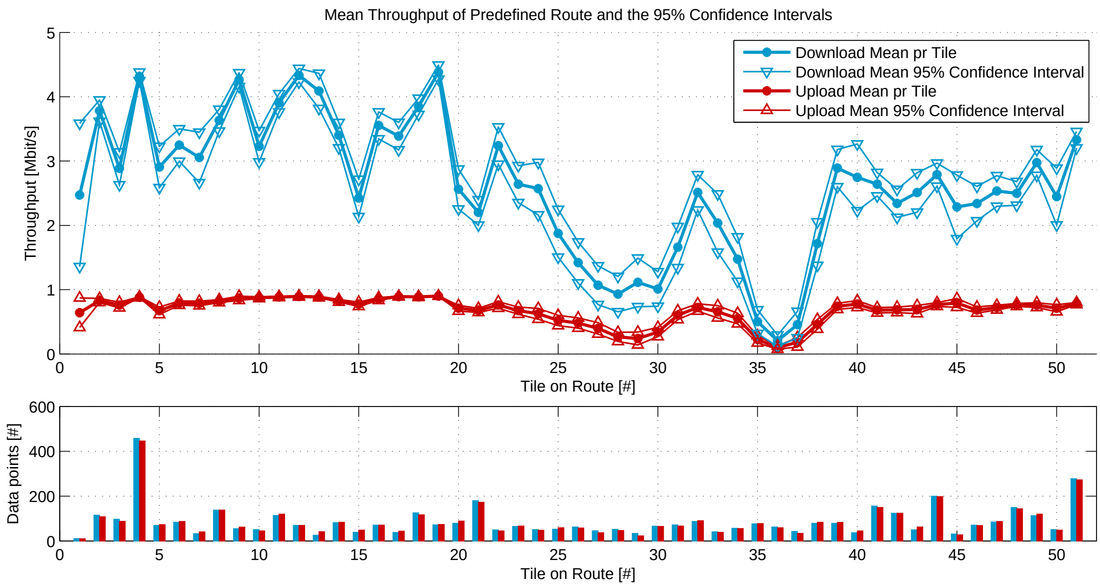


Figure 4.10: Average throughput rates in the tiles along the route between Brovst and Aalborg seen in Figure 4.1. The 95% confidence interval for the mean per tile is calculated based on the method described in Appendix B.1

In the initial part of the route the throughput shows high rates with only little fluctuations from tile 1 to tile 19, which is measured from Brovst to Nørresundby. After this some big drops in performance are present with the biggest drop around tile 36, which as mentioned earlier is believed to be due to network restrictions in the vicinity of the airport runway.

In the following stretch from tile 39 to tile 51, which is in and around Aalborg and Nørresundby, the throughput rates are somewhat steady around 2.5 and 0.7 Mbit/s for Download and Upload respectively. The reason for these rates being a little lower than the rates measured before Nørresundby is believed to be due to the much higher density and amount of users on the network, and thereby more cross traffic.

Throughout the route the fluctuations of the Download rates are much more apparent than the fluctuations of the Upload rates. This is believed to be caused by the limitations put on the throughput rates in the subscriptions by the service provider, where the Download is limited to 5 Mbit/s and the Upload to 1 Mbit/s. The reason for the skewness in the rates is that regular users mostly request and download content from providers around the world, and to a much lower degree upload content. Because of this usage tendency it is very likely, and logical, that the service providers assign resources accordingly to their networks, such that there are much more capacity from the Internet toward the users than in the other direction. But the utilization of the connection from the Internet to the users is most likely higher than the other direction.

For these reasons, the Download rate is more sensitive to changes in the load on the network, while other influences such as in signal strength and other also may have an impact. For the Upload rate, also the fact that the rate is limited at a much lower rate than what is physically possible with the used technology, the influences on the connection must be much more radical in order for the Upload rate to drop significantly.

4.3.2 Throughput in Relation to Movement Speed

In Figure 4.11 average of the throughput rates sorted according to speed, in 5 km/h bins, are seen. In the top figure the mean throughput rates and the 95% confidence intervals per 5 km/h speed bin are seen, while the bottom figure shows the amount of samples per speed bin. From this it can be seen that, besides the stationary measurements at 0 km/h, the samples are mainly gathered around 80 km/h and around 50 km/h, while also a smaller amount of measurements are present around 120 km/h and around 30 km/h.

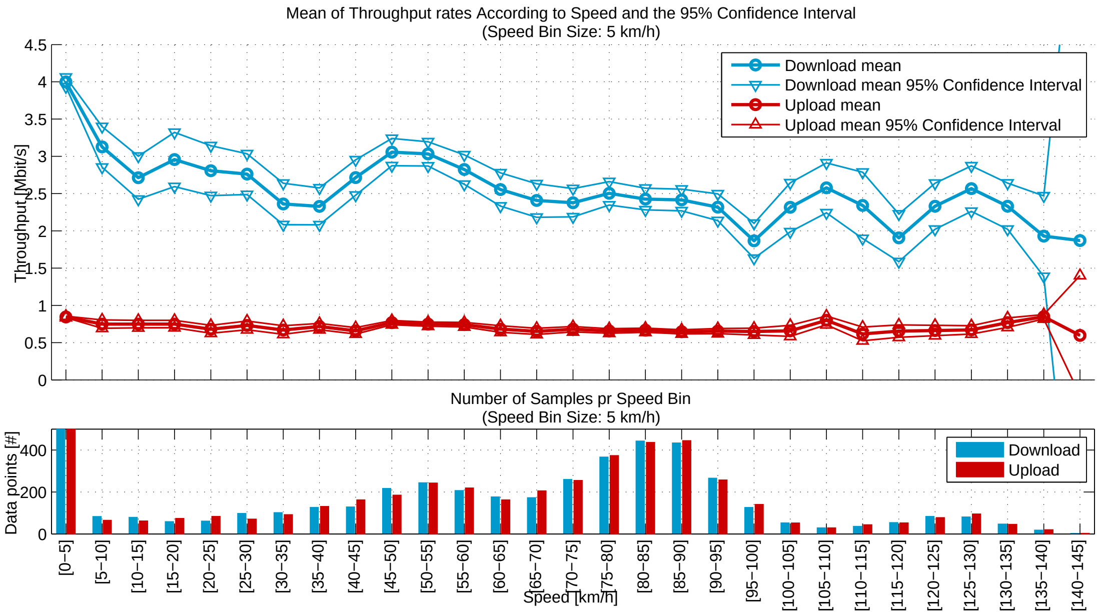


Figure 4.11: The mean of the throughput sorted according to different movement speeds of the device, and the amount of measurements per speed bin. The 95% confidence interval for the mean per speed bin is calculated based on the method described in Appendix B.1.

Again, as for the location related measurements, the Upload rates are much less fluctuating than the Download rates, which also is supported by the more narrow confidence intervals for Upload.

From the figure it can be seen that there is a slight tendency that the Download rates drop a little from 0 km/h, over 50 to 80 km/h, while the rates seem to be the same for 80, 110 and 130 km/h.

The drop from 0 to 80 km/h could indicate a dependency between the movement speed and the performance of the connection. This dependency could be caused by when a device moves faster the rate of handovers is increased, effectively lowering the overall performance due to more retransmissions and rerouting of packets.

The performance leveling out from 80 to 130 km/h could be described by the area types that the higher speeds are reached in. These speeds are reached on the expressway, which for most of the time being is located in rural areas, where the amount of users and load in the network typically is low. Conversely, the measurements gathered while the speed is from 0 km/h to 80 km/h is typically in more populated areas, where the amount of users on the network typically is higher, leading to more load on the network.

However, as the confidence intervals from 80 to 140 km/h are wider than from 50 to 100 km/h, it is difficult to make conclusions in about the higher speeds, while the results seem rather confident between 50 and 100 km/h.

To be able to analyze this further the measurements are grouped in 4 bigger speed intervals, and the distributions within these bigger intervals are seen in Figure 4.12.

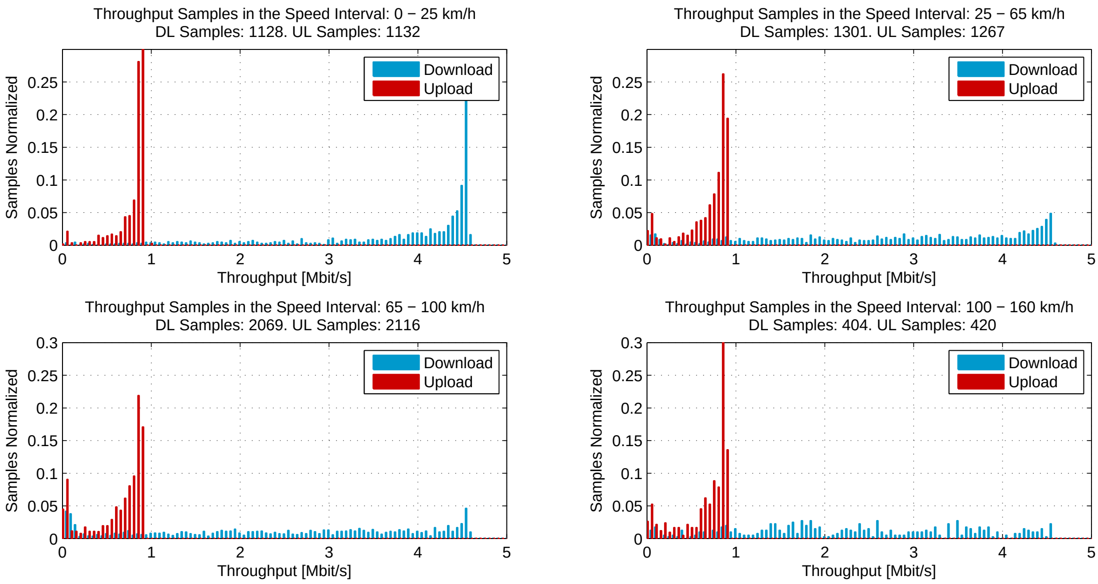


Figure 4.12: The distribution of throughput rates within the 4 different speed intervals, plotted as histograms.

From this figure it can be seen that the best performance is reached in the slowest speed interval (0 - 25 km/h), where for Upload the main peak is close to 1 Mbit/s, and for Download the main peak is close to 4.5 Mbit/s.

In the next speed interval (25 - 65 km/h) for the Upload the main peak is still close to 1 Mbit/s but the ascend from around 0.2 to 1 Mbit/s is a little less steep than for the first speed interval, and the peak close to 0 Mbit/s is a little lower here as well. For the Download rates, the main peak is still at 4.5 Mbit/s but now it is much smaller than in the first speed interval. At the same time, for the second speed interval, the measurements are more uniformly distributed from approximately 1 to 4 Mbit/s.

In the third speed interval (65 - 100 km/h) for the Upload rates the peak close to 0 Mbit/s is again higher than in the previous 2 speed intervals, and the peak at 1 Mbit/s is also a little lower. For the Download rates the peak at 4.5 Mbit/s is again a little lower than in the second interval, and while the uniform distribution between 1 and 4 Mbit/s is close to the same a peak at 0 Mbit/s is beginning to become significant.

In the last and fastest speed interval (100 - 160 km/h) for the Upload rates the peak just before 1 Mbit/s has increased again compared to the 2 previous speed intervals, and at the same time the peak close to 0 Mbit/s has lowered a little. For the Download rates the peak at 4.5 Mbit/s is close to non-existing, while the distribution between 1 and 4 Mbit/s is a little less uniform with small clusters of measurements at 0.8, 1.7, 2.4 and 3.5 Mbit/s.

Due to the much lower amount of measurements in the fastest speed interval it is difficult to include this in the conclusion, but when looking at the first 3 speed intervals, there is a tendency that the performance drops slightly when the movement speed is increased. But as mentioned earlier the different area types that are linked with the different movement speeds could also be some of the reason for the change in performance seen here.

4.4 Analysis of Connectivity

In this section the Connectivity measurements measured on the route shown in Figure 4.1 will be analyzed and evaluated.

4.4.1 Connectivity in Relation to Location

In Figure 4.13 the average Connectivity per tile on the route between Brovst and Aalborg is plotted, along with the 95% confidence interval per tile.

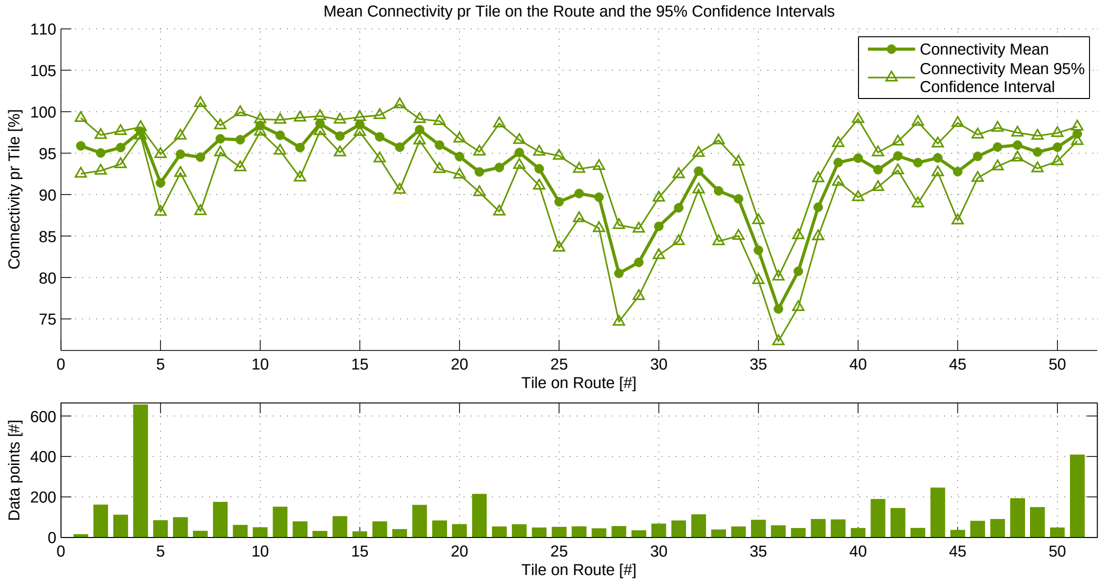


Figure 4.13: The Connectivity measurements averaged per tile they were measured in and plotted according to the route shown in Figure 4.1. The 95% confidence interval for the mean per tile is calculated based on the method described in Appendix B.1. Note that the y-axis range is from 70% and up.

In the initial part of the route, the Connectivity is mostly steady between 90 and 100%, but there are two big drops around tile 28 and tile 36. These are again as with the other parameters believed to be due to the vicinity of the airport and restrictions put on the cellular network. After these drops the Connectivity again stabilizes around 90% from tile 39 to tile 51, which is in and around Aalborg and Nørresundby.

From the confidence intervals along the route it can be seen that where it is widest it stretches over 12%, which indicates that generally there are very few dropped packets, or timeouts, along this part of the route.

4.4.2 Connectivity in Relation to Movement Speed

In Figure 4.14 Connectivity is evaluated in relation to movement speed, which is done by sorting the measurements according to 5 km/h speed bins, and taking the average of the Connectivity in each bin.

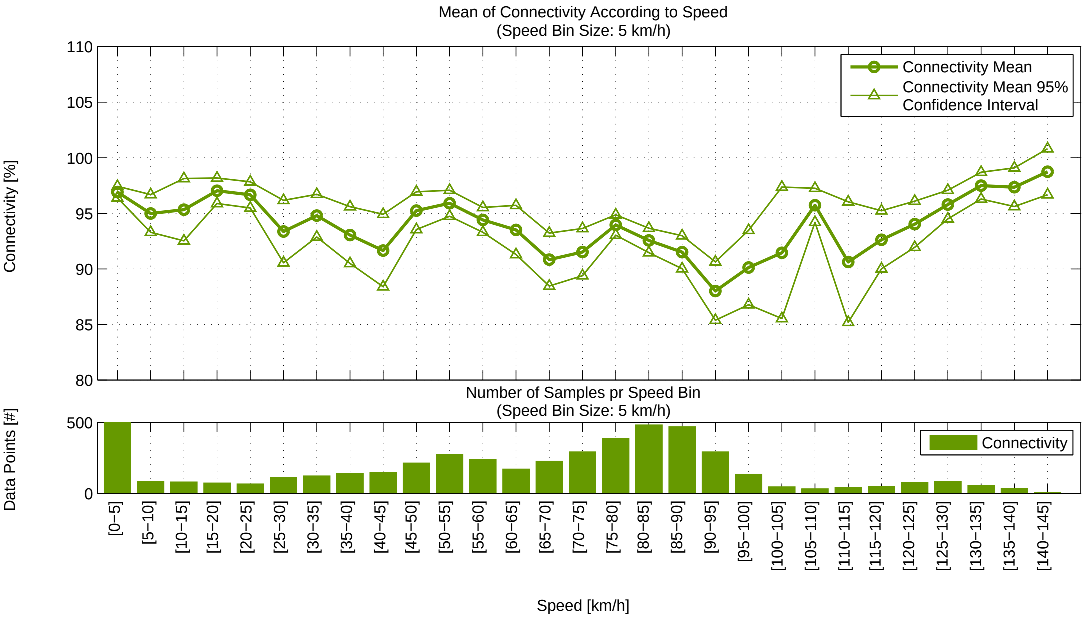


Figure 4.14: The Connectivity measurements averaged according to the speed the device moved with during the measurement. The 95% confidence interval for the mean per speed bin is calculated based on the method described in Appendix B.1. Note that the y-axis range is from 80% and up.

From the figure a little of the same tendencies as with throughput and RTTs can be seen, namely around 0, 50 and 80 km/h, where the Connectivity drops but only very little. Because the drop in Connectivity is so little it is difficult to make the same conclusions as with the other parameters.

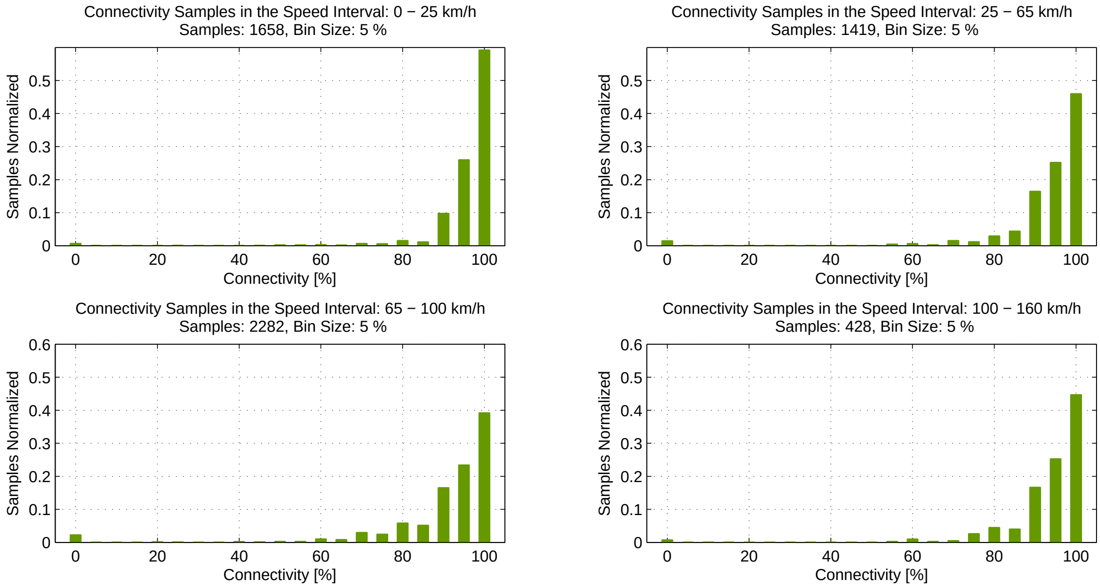


Figure 4.15: Histograms showing the distribution of the average Connectivity in each respective speed interval.

But to investigate the behavior of the Connectivity in the detail, the measurements are divided according to 4 speed intervals. This is seen in Figure 4.15 where in each speed interval the Connectivity measurements are sorted in histograms.

From the figure it can be seen that in the slowest speed interval (0 - 25 km/h) from the right the 2 first bars are 0.25 or higher, the third bar is 0.1 and the rest 0.01 or less. At 0% Connectivity there only is a very small bar.

For the second speed interval (25 - 65 km/h), again from the right, the first 2 bars again are 0.25 or higher with the first bar lowered a little, the third bar has increased to above 0.15, and the fourth and fifth have also increased. The bar at 0% Connectivity is also a little higher than in the first speed interval. This indicates that in this speed interval, the Connectivity has a little greater chance of failing a single or even all the samples in 30 tests.

This tendency is even more outspoken in the third speed interval (65 - 100 km/h) where, from the right, the first bar now is below 0.4, the second and third bar is close to the same, but the following bars have increased a little. The bar at 0% Connectivity is also a little higher here.

The last speed interval (100 - 160 km/h) the distribution looks like the distribution of the second speed interval, that is, from the right, the first and second bars have increased a little again compared to the third speed interval. Furthermore, the bar at 0% Connectivity has dropped a little in height.

The distribution and the evolution of the Connectivity through the first 3 speed intervals could indicate that the Connectivity and the performance drops slightly when the device is moving faster. In the last speed interval the distribution does not follow the evolution in the other intervals, which is believed to be due to different area types where these movements speeds typically is reached. That is, in the first 3 speed intervals, the movement speeds can be, and are typically, reached in and around urban areas, while the movement speeds in the last speed interval is too high and therefore only reached on the expressway, and therefore typically in more rural areas.

4.5 Analysis of Handovers

In this section the handovers registered in the on the route shown in Figure 4.1 will be evaluated and analyzed.

4.5.1 Handovers in Relation to Location

In Figure 4.16 the amount of handovers of the different types registered on the route between Brovst and Aalborg can be seen. In the top part of the figure the CID and LAC handovers are seen, and in the bottom part the Net Type and Sub Net Type.

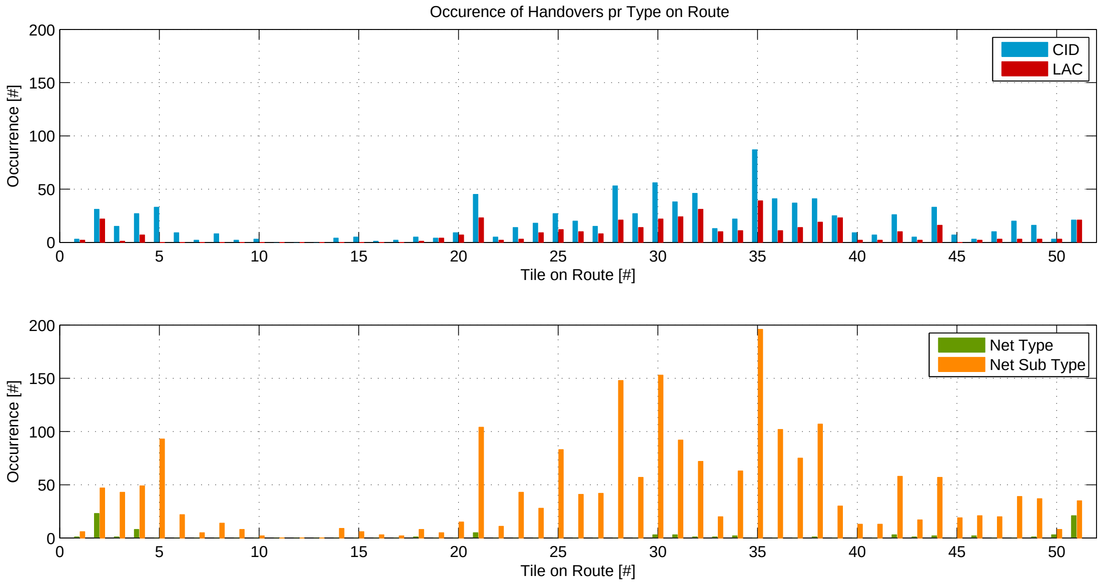


Figure 4.16: The occurrence of handovers per tile on the route shown in Figure 4.1.

The occurrence of the handovers on this part of the route can be divided into 2 parts; from tile 1 to tile 6, which is in Brovst, and from tile 20 to tile 51 which is in and around Nørresundby and Aalborg. This makes sense as cell towers tend to be placed with closer spacing in urban areas than in the rural[att13], why handovers between towers are more frequent in these areas.

From the top part of the figure it can be seen that the amount of LAC handovers is lower than the amount of CID handovers and that the concentrations of LAC handovers are in the same areas as the CID handovers.

From the bottom part it can be seen that the Net Sub Type handovers very much follows the distribution of the CID and LAC handovers. Conversely, the Net Type handovers are very few, only with small spikes at tile 2 and at tile 51.

Linking CID Handovers with Cell Tower Locations It is believed that the CID handovers is the handover type that is causing the biggest change in the performance, which is when the device from one cell antenna to another cell antenna, where there might be more than one antenna (and thereby CID) per cell tower (most often 3, covering 120 degrees each). Therefore this type of handover will now be analyzed further, that is the measurements will be attempted linked with cell antenna locations in the area.

[oci12]The cell tower locations are extracted from the cell ID database Open Cell ID. This is an database containing GPS locations of where cell antennas with a CID and a LAC have been registered. This is an open source database and the information in the database is added by a number of different client applications, that add the registered CID with GPS coordinates of where the device was located at registration. This means that the position of the cell antennas extracted might be very imprecise compared to the true position, depending on what location service is used by the device when registering the CID and location. But for this case the approximate location of the cell antennas is sufficient.

Another cell tower location database is the `http://www.mastedatabasen.dk/`, which is an official danish site containing the exact locations of all radio tower positions in Denmark. But where this is precise in the locations of the towers, the provider using the tower, and the technologies supported by each tower, it contains no information about the CID or LAC of the antennas on the towers. This makes the linking of measurements with CIDs and cell towers difficult, but for visual interpretation the tower locations used by TDC[1], and supporting UMTS, in the vicinity are marked on the figure with '**X**'.

For simplicity the analysis is done in a small bounded area where the amount of measurements is relatively high, and where the measurements are performed on the same stretch of road. In this small area the CIDs registered in the measurements of RTT, Throughput and Connectivity are noted and the location of the cell antennas with these CIDs are found from Open Cell ID. The cell antenna locations are then plotted and the measurements related to the CID of a antenna is plotted with a corresponding color. This is seen in Figure 4.17, which shows the antenna locations and the related measurements on the route between Aabybro and the airport, outside of Nørresundby.

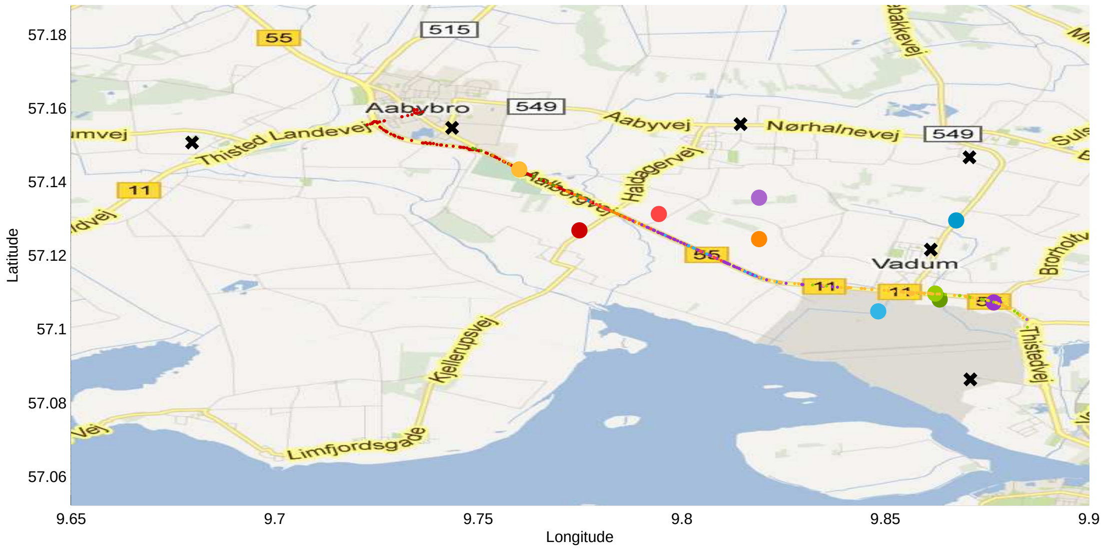


Figure 4.17: The cell antennas with unique CIDs (big dots) and the measurements (small dots) with corresponding CIDs in small bounded area. The measurements marked are positions from all other types of measurements than actual Handovers. True cell tower locations in the vicinity are also marked (with **X**). The data points are plotted on top of a Google Maps plot[goo12a].

From the figure it can be seen that there are 10 cell antennas on this stretch of road, or rather the measurements on this stretch of road contain 10 different CIDs.

It is however difficult to see the density of measurements on the stretch, so to better see the amount of measurements on the stretch they are sorted in bins according to the longitude GPS coordinate. This is seen in the top part of Figure 4.18 where the measurements are sorted according to longitude in bins of 0.005 (in this case approximately 260 to 320 meters), and plotted with the color corresponding to the connected CID.

In the bottom part of Figure 4.18 the CID handovers on this stretch of road is plotted, where the colors relate to the CIDs, and also here sorted according to longitude bins of 0.005.

[1]TDC is the service provider of the subscriptions used for performing the measurements.

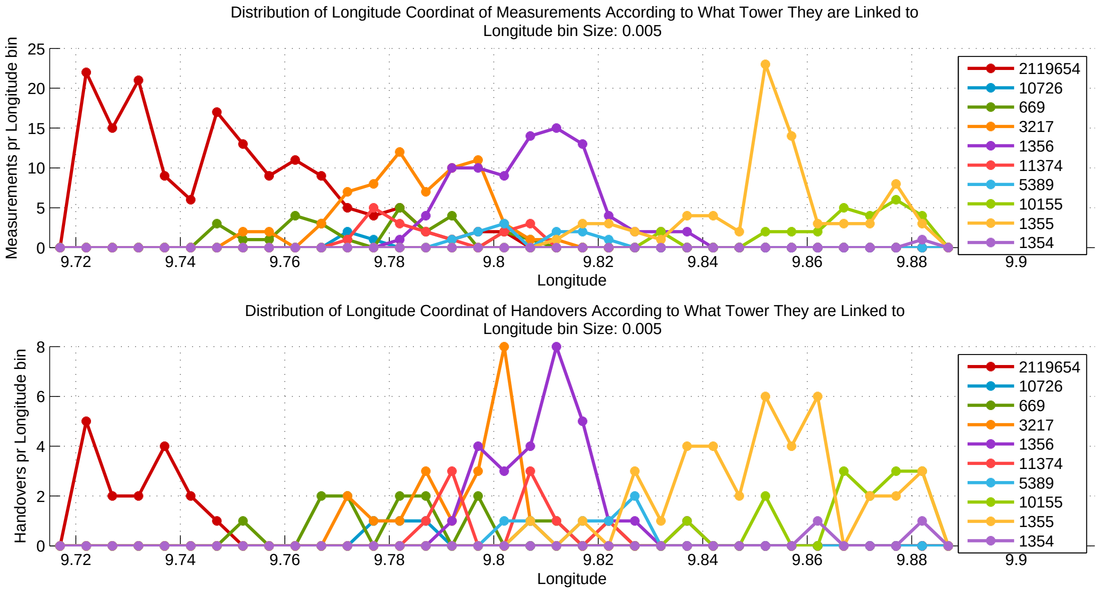


Figure 4.18: Top: The number of measurements marked according to longitude GPS coordinate and colored according to CID. Bottom: Number of CID handovers marked according to longitude GPS coordinate and colored according to CID.

From the top part of the figure it can be seen that there are 4 CIDs that the main part of the measurements are registered with, namely CID 2119654, 3217, 1356 and 1355. From the bottom part of the figure the amount of handovers per CID confirms that these CIDs are the most frequent.

Handovers from one cell antenna to another happens when the signal strength from the one cell antenna drops below some threshold, and the cell antenna with the highest signal strength is chosen[WGS10]. Therefore, when moving along the same stretch of road in the same direction, intuitively the same handover sequence from one specific antenna to another specific antenna should happen often.

From the data with CIDs 3217 (dark orange) and 1356 (dark purple), there is a tendency that the handovers happen on the rightmost edge of the cluster of measurements with the same CID. This could indicate that handovers to these CIDs typically are performed while moving west and away from Nørresundby.

Likewise, for the CID 2119654 (dark red), it seems that most of the handovers are registered in the leftmost edge of the measurement cluster, why handovers to this CID typically happens when moving east and leaving Aabybro.

For the fourth big cluster of measurements connected to the same CID, namely 1355 (light yellow), shows a little different tendency, in that the handovers are spread more out and the measurements show a peak in the middle of the handover cluster. This could indicate that the CID is used while moving in both directions, both east and west.

In order to understand the impact of these CID handovers on the performance, the average Throughput measured in this small bounded area will be compared with the number of CID Handovers, which is seen in Figure 4.19.

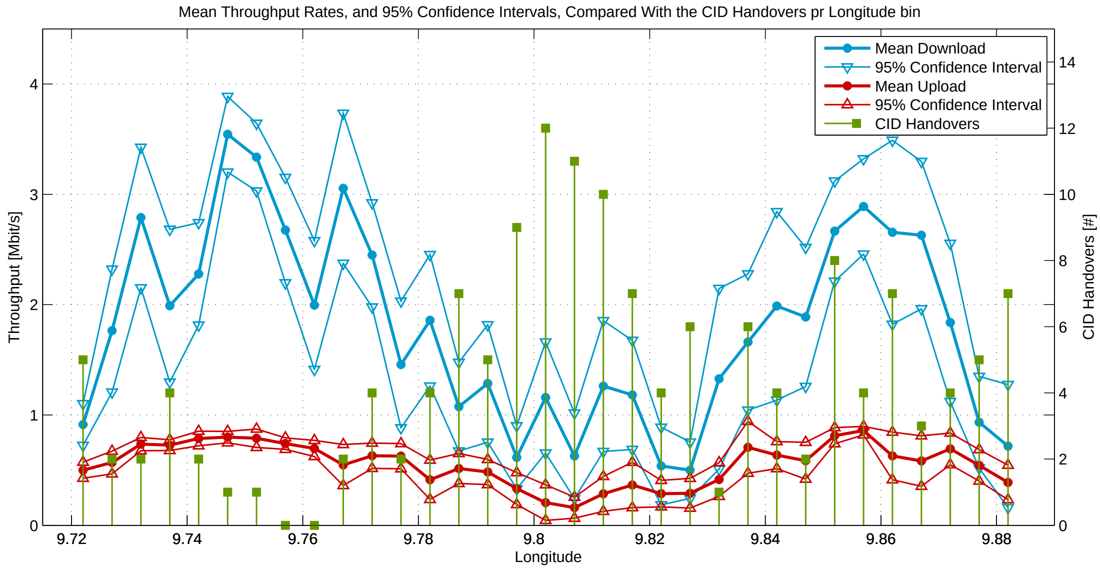


Figure 4.19: The mean Throughput measured in the small bounded area sorted in 0.005 bins according to Longitude, and the 95% confidence intervals for the mean per longitude bin, which is calculated based on the method described in Appendix B.1. Also the total number of CID handovers per longitude bin is plotted.

From the leftmost part, from 9.72 to 9.78, the throughput rates are high while the amounts of handovers mostly are low. In the center part, between 9.78 and 9.82, the throughput rates are low and the amount of handovers are high. And in the rightmost part, from 9.82 to 9.88, the throughput rates are high and the amounts of handovers are fluctuating but not as high as in the center part. For all the throughput rates, and both for Download and Upload, the confidence interval is more or less the same width, only fluctuating a little.

This indicates that the amount of handovers has some impact on the throughput rates experienced in that area. This can be explained by that if a handover happens during a data transfer, some of the packets might be lost and will need to be retransmitted, which takes longer time than with no packet loss. Another possibility is that the connection was handed over to another access network, or part of it, why packets must be routed via a new route to the destination, which might take longer time than the previous network. So even though the device might be continuously connected during the handover, the quality of the new link or connection might be different than the previous, e.g. due to load differences.

In any case, the throughput will not be at the maximum capability of the connection right away, as the throughput uses TCP which utilizes congestion control, why i takes a little time for the data rate to settle[SRCQ09].

4.5.2 Handovers in Relation to Movement Speed

From Figure 4.20 the occurrence of the different handover types, normalized per speed bin can be seen, along with the amount of the different handover types according to movement speed.

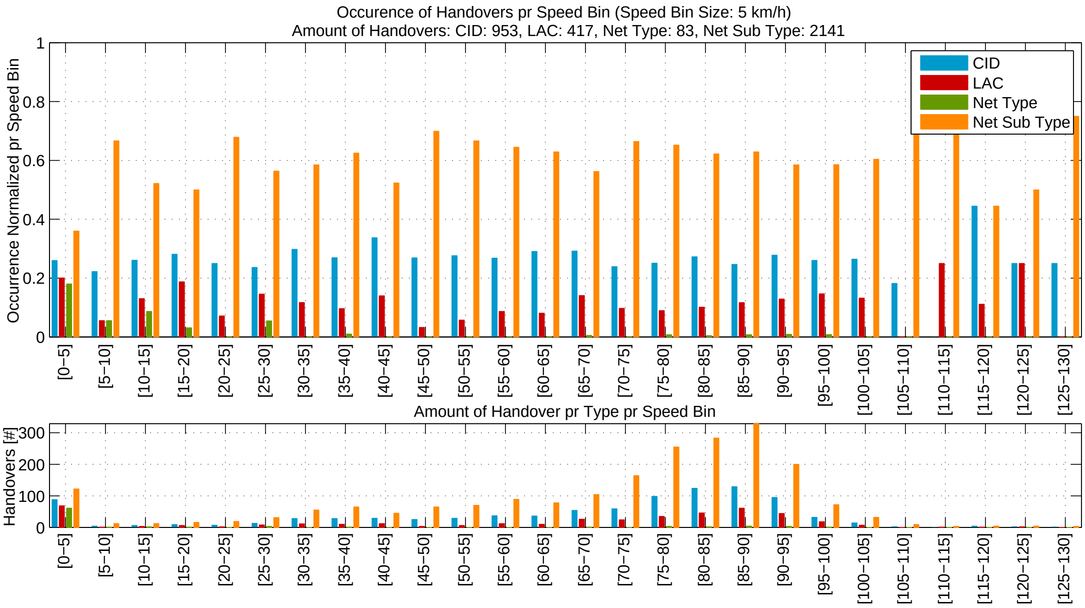


Figure 4.20: Handovers per 5km/h speed bin, normalized per speed bin.

From the figure it is apparent that Net Sub Type handovers are the most frequent at all speeds, followed by CID, then LAC, and lastly Net Type handovers. Furthermore, it can be seen that the ratio between the different types of handovers is close to constant for all the different movement speeds. Only little variation is present, which is difficult to say whether it is caused by the change in movement speed, or the lack of measurements.

4.6 Comparison of Correlation Between Parameters

In this section the relationship between the different parameters will be analyzed, which was also done in the analysis in the previous project [MTP12]. This is done by calculating the correlation coefficient between the different parameters. In the following it is said that 2 parameters are highly correlated if the correlation coefficient is 0.8 or higher. They are medium correlated if the coefficient is between 0.5 and 0.8, and they are not correlated if the coefficient is below 0.5.

The correlation coefficient is calculated as described in Appendix B.2.

4.6.1 Correlation Between Variation of Different Parameters Over the Route

First the correlation coefficients between the parameters over the route between Brovst and Aalborg will be calculated, which is seen in Figure 4.21. Here the correlation coefficients are calculated using the mean values for the parameters in the tiles on the route, which are shown in the Figures 4.7, 4.10, 4.13, and 4.16. For the handovers the number of handovers per type per tile is used. Note that the Net Type handovers are omitted as there only are very few of these.

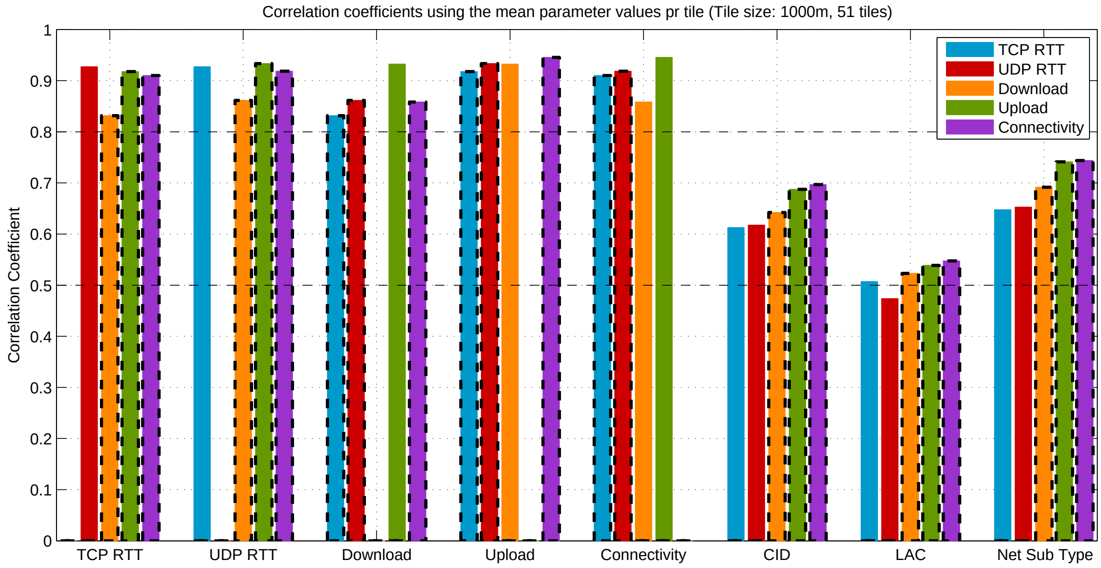


Figure 4.21: The correlation coefficients between the different parameters for all tiles between Brovst and Aalborg. Note that values for the leftmost 5 parameters are there twice, e.g. for correlation between UDP RTT and TCP RTT is the same value as for TCP RTT and UDP RTT. Values for the handovers only occur once, as there were no, or only very little, difference in the ratio between the individual handover types (see Figure 4.20). If the parameters show negative correlation the bar is marked with a dashed line on the edge.

From the figure it can be seen that TCP and UDP RTT, Download and Upload, and Connectivity all are highly correlated, and that these 5 parameters have medium correlation with the 3 handover types shown, however a little higher for CID and Net Sub Type than for LAC.

The correlation between the CID handovers and the performance parameters is believed to be present because, as described earlier, when the device makes a handover from one cell antenna to another mid stream, some packets are likely to go lost. For the parameters using TCP, that is TCP RTT and Up- and Download, some retransmission will happen, while for the parameters using UDP, that is UDP RTT and Connectivity, this will result in lost measurements.

The correlation between the Net Sub Type and the performance parameters is more difficult to explain. It could be that one connection technology, such as UMTS, is used during idle period in between measurements, and when the connection is used during measurements the connection technology changed to another type, e.g. HSPA. However, it could be that the connection technology is chosen according to what the currently connected cell antenna supports. If this is the case, then if the change from a slow technology to a faster typically happens in a specific area, the performance parameters typically will show good performance in this area.

The high correlation between the 5 performance parameters does support what has been described in the previous sections, when these parameters were evaluated in relation to location. However, this could be caused by the big fluctuations that happen near the airport, which is a special case. To see if this is the case the correlation coefficients for the same parameters are calculated again, but now the values measured around the airport, from tile 23 to 38, are discarded. This can be seen in Figure 4.22.

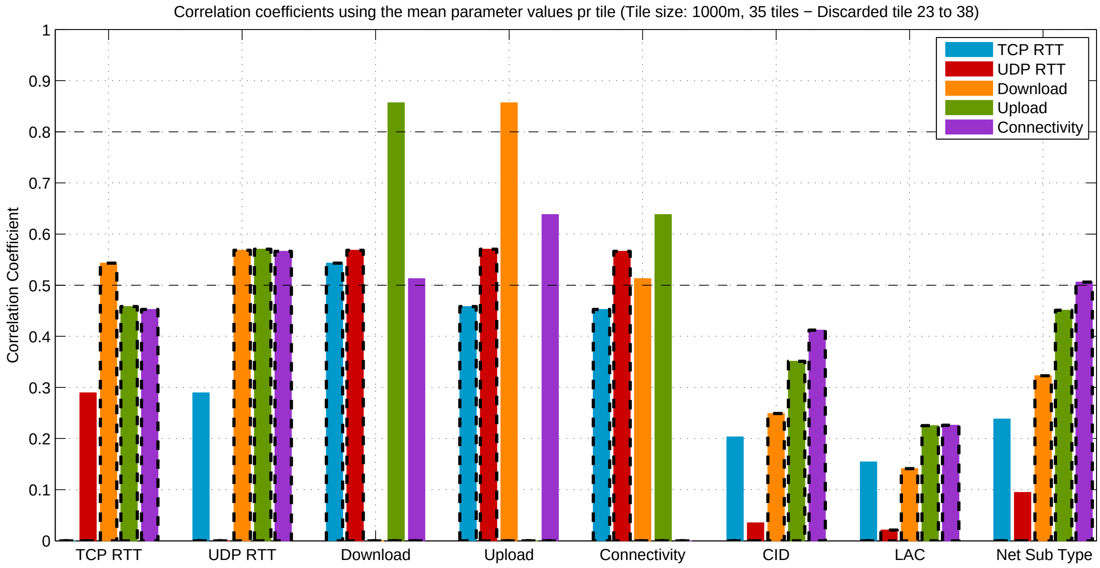


Figure 4.22: The correlation coefficients between the different parameters, for reduced set of tiles. Note that values for the leftmost 5 parameters are there twice, e.g. for correlation between UDP RTT and TCP RTT is the same value as for TCP RTT and UDP RTT. Values for the handovers only occur once, as there were no, or only very little, difference in the ratio between the individual handover types (see Figure 4.20). If the parameters show negative correlation the bar is marked with a dashed line on the edge.

From this figure it is very clear that the big drop in performance experienced near the airport has a big impact on the correlation coefficients. With the data points near the airport removed only the correlation between Download and Upload is still high.

One of the biggest changes is the correlation between TCP and UDP RTT, which now is below 0.3, and thereby no correlation. This is remarkable as the TCP and UDP RTT mean curves seem to follow each other in Figure 4.7 in the area that is not near the airport. This might then be because some of the fluctuations that are there are bigger for UDP than for TCP RTT, why the correlation drops. Generally, the correlation between TCP RTT and the other parameters seem to be the values that have changed the most from the previous calculation.

For the correlation between the 5 performance parameters and the handovers, all the values have dropped, but now it is the correlation between UDP RTT and the handovers that has changed the most.

The low correlation between the different parameters when evaluated along the path indicates that there is no immediate connection between how the parameters are affected along the path. This could be caused by the tile size of 1000x1000 meters, which is a relatively big area when looking at network performance. This result can be related to the result obtained in the previous project[MTP12], where generally higher correlation coefficients was seen when evaluating performance over the path. But in that analysis the tiles were 100x100 meters, why the much more local fluctuations in the performance were analyzed.

4.6.2 Correlation Between Variation of Different Parameters at Different Movement Speeds

Now the correlation coefficients are calculated between the different parameters according to their values at different movement speeds, which is shown in Figure 4.23. The correlation coefficients are calculated using the average values for RTT, Throughput and Connectivity per 5 km/h speed bin, which are shown in the Figures 4.8, 4.11, and 4.14.

The values for the first speed bin, that is at 0-5 km/h, is disregarded as these are stationary measurements and measurements performed at startup of the application. These are removed as it is the measurements performed while moving that are of interest. Furthermore, from the plots of performance according to different movement speeds it can be seen that the amount of measurements gathered at speeds higher than 100 km/h are very low, why the performance values for parameters above 100 km/h are disregarded as well.

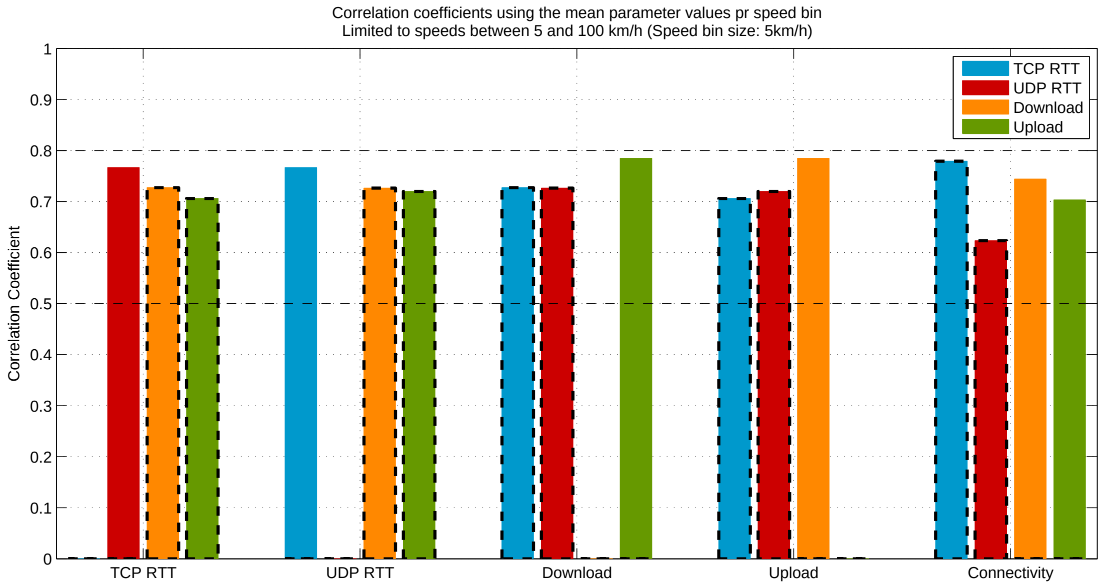


Figure 4.23: The correlation coefficients between the different parameters and the performance of these at movement speed between 5 and 100 km/h. Note that all values are there twice (except for Connectivity), e.g. for correlation between UDP RTT and TCP RTT is the same value as for TCP RTT and UDP RTT. If the parameters show negative correlation the bar is marked with a dashed line on the edge.

From the figure it can be seen that all the parameters show mutual medium correlation, and most of the correlation coefficients are 0.7 or higher, as it only is the correlation between UDP RTT and Connectivity that is below 0.7.

The medium, but high medium, correlation between the various parameters supports what was said earlier about the parameter values in relation to movement speed, and especially at speeds between 5 and 100 km/h. This indicates that the performance of the parameters generally are affected in the same way when the device is moving faster, or at least that is the tendency.

4.7 Summary of Analysis with Focus on Movement Speed and Handovers

Throughout the analysis it has become apparent that the area type, and especially the density of users in the area, has the greatest impact on the measurements and the performance experienced. This was seen when looking at the different parameters in the tiles on the route, where different area types such as urban, rural and a mixture of these were experienced, with a following change in performance.

In this analysis it was, due to when the majority of the measurements were performed, not possible to properly evaluate the impact of time of day on the measurements. The measurements were performed in the hours from 6 to 22 with the majority being between 7 and 16. Therefore the analysis of the impact of the busy and non-busy periods was not applied here.

But through the analysis of the different parameters the impact of movement speed and the connection between movement speed and performance was analyzed. This analysis showed that generally when moving faster the performance would drop slightly. This seemed to be the case both for TCP and UDP RTT, Upload and Download, and Connectivity, when evaluating the performance at the speeds around 0, 50, 80 and 120 km/h. This slight drop was believed to be caused by the rerouting of packets that happens when making a handover from one cell antenna to another midstream, and that the rates of handovers are increased when moving faster.

A rerouting of packets would cause a short delay in the communication, and if the frequency of handovers is increased, it could be assumed that the performance would drop accordingly. However, when the very high speeds were reached the device would typically be in a more rural area, where there are less users and longer distance between cell towers. So the question is, when the distance between cell towers and movement speed is increased, is the ratio between these two the same as when moving slower in urban areas where cell towers are placed with smaller spacing. This does not seem to be the case and therefore the slight drop in performance.

Also the handovers was subject to thorough analysis, where the handovers and performance in a small bounded area was explored. From this it was concluded that the CID handovers do have a great influence on the performance of the network. This might then conversely be what keeps the performance up when a device is moving fast in more rural areas, that is that there are less handovers due to the bigger spacing between cell towers.

These conclusions are based on a somewhat limited amount of measurements, so in order to make the conclusions stronger more measurements would be needed. Furthermore, the conclusions and tendencies described through this analysis are done based on available information through the measurements, why in order to get a more complete conclusion of what influences the performance of the network, deeper analysis and more detailed analysis is needed.

Chapter 5

Data Analysis of Area Type

In this chapter the data used for the analysis in Chapter 4 will be analyzed for differences based on area type. This is done based on the conclusions made in the analysis in Chapter 4 but also based on the analysis performed in [MTP12], which both conclude that the area type have some influence on the performance of the connection and network. All the figures and calculations in this chapter are performed using MATLAB scripts, that can be located on the enclosed CD *part_1_matlab/* .

First the area types will be defined followed by analysis of the measurements in the different area types.

5.1 Definition of Area Types

In this section the different area types that the data will be divided according to, will be defined. These area types are chosen based on the measurements available why other types might suit other cases better.

The area types are chosen based mainly on knowledge of the size of the towns and cities, why a somewhat large area around the big cities are marked as urban. The data is divided according to 3 area types, and these are as follows:

Urban Area This area type is covering bigger cities, and the immediate neighboring areas, resulting in areas covering the cities and small a zone around them. This is done to get the measurements gathered while being connected to the same cell towers as the users in the corresponding cities. In this area type there typically are many cell towers that are located with small spacing apart, and there are many users.

In the available data, areas of the urban type are covering Aalborg and Nørresundby, Randers, and Aarhus, which is the biggest cities covered by the measurements.

Small Town Area This area type is, as the name indicates, covering small towns along the route where measurements are performed. In this area type there typically are a few cell towers pr town servicing the users in the town.

In the available data, areas of the small town type are covering Aabybro, Brovst, Aars, Haverslev Støvring and Ødum. These are the small towns that the route passes through.

Rural Area This area type is, with the previous area types applied the remaining area. In this type of area the cell towers typically are placed with large spacing, and the amount of users is very low, compared to the size of area.

In the available data, this area type is covering the remaining of the measurements that have not been included in the previous area types.

Figure 5.1: The locations of the measurements colored according to the area type they were measured in. The area types on the route are plotted on top of a Google Maps plot[goo12a].

5.2 Analysis of Measurements in Different Area Types

In this section the RTTs and Throughput rates will be evaluated according to the area type. First off the RTTs are evaluated, which is seen in Figure 5.2.

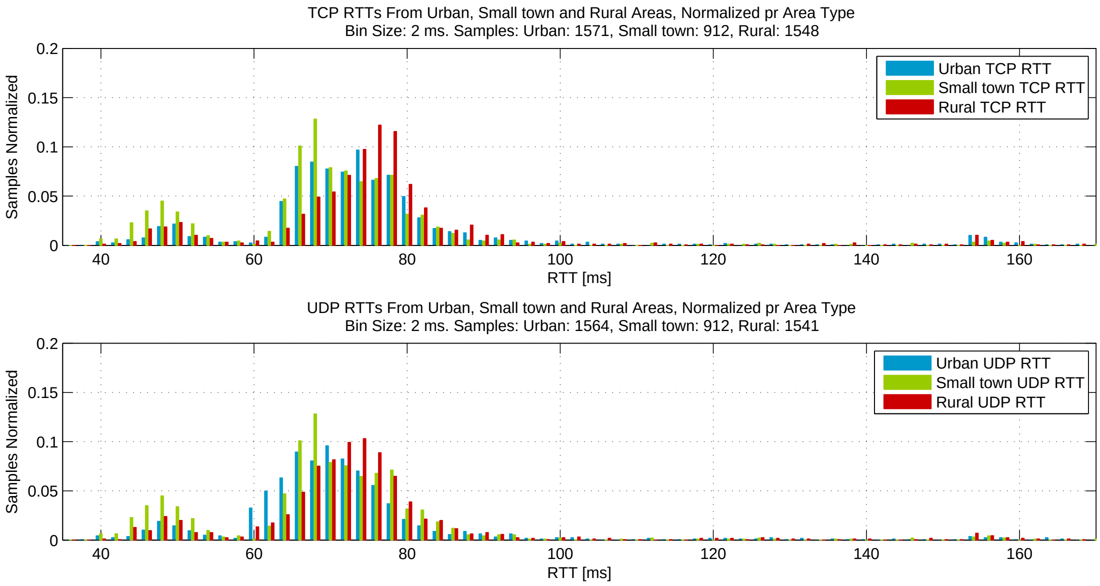


Figure 5.2: Histogram of the TCP and UDP RTTs measured in the different area types. The number of measurements are normalized according to area type, for easier comparison.

From the figure it can be seen that the RTT measurements from urban areas mainly are between 65 and 85 ms, while there also is a very small population between 45 and 55 ms. For RTTs from small town areas it can be seen that they tend to be a little faster with the main population in the same area but with a peak at 68 ms, while the population between 45 and 55 ms also is a little bigger than for urban areas. The RTTs from the rural areas also have the main population in the same area, but with a peak around 76 ms, and the population between 45 and 55 ms being the same size as for urban.

Based on this it can be concluded that the best performance is obtained in the small town areas, followed by the urban areas, and lastly the rural areas. But it should be noted that the difference in the performance of RTTs between the different areas is small, which also was mentioned in the analysis in Section 4.2. However, this does again indicate that areas with relatively few users, compared to the amount of cell towers, that is the small town areas, offer the best performance.

In Figure 5.3 the Throughput rates measured in the different area types are seen.

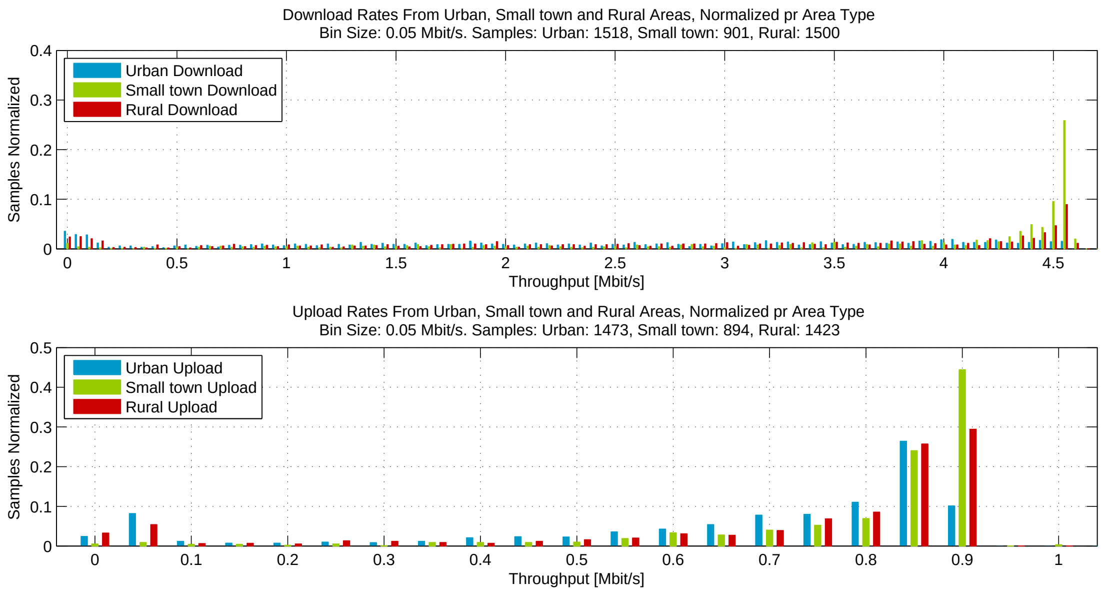


Figure 5.3: Histogram of the Throughput rates measured in the different area types. The number of measurements are normalized according to area type, for easier comparison.

From the top part of the figure the Download rates can be seen. From this it can be seen that the area type with the highest amount of high rates is the small town, followed by rural and then urban. Both the small town and the rural area type have a small ramp up in amount of rates from 4.2 to 4.5 Mbit/s, while urban stays more uniform. From around the rate 0.5 to 4.2 Mbit/s all the area types are more or less uniform, while the urban and the rural area types have some small populations around 0 Mbit/s

From the bottom part of the figure the Upload rates can be seen. From this some of the same tendencies as with the Download rates can be seen. In the Small town area type there is a ramp up towards the throughput limit at 1 Mbit/s where it peaks at 0.9 Mbit/s, while the rural area type peaks just before at 0.8 Mbit/s. The urban area type also peaks at 0.8 Mbit/s, but the bar at 0.9 Mbit/s is very low, compared to the other area types. Again only the urban and rural area types have some small populations around 0 Mbit/s.

From this it can again be concluded that the best performance is achieved in the small town areas, but now the next best is achieved in the rural areas, and the lastly the urban area type. This can however be explained by a combination of the amount of users in the areas and the distance between the cell towers.

In the small town areas there are few cell towers placed with low spacing but also not that many users. In the rural areas the cell towers are placed with large spacing but there are very few users. And in the urban areas the cell towers are placed with low spacing, but the amount of users is very high.

5.2.1 Conclusion of Area Type Analysis

Based on this analysis it can be argued that because the utilization of the network is lowest in the small town area, followed by the rural area and finally the network utilization is highest in the urban areas, the performance typically is better in the small town areas.

The utilization of the network can furthermore be seen as a combination of the number of users in an area and the resources available on the network, in terms of bandwidth, cell towers, and distance to the backbone. This explains why small town area type is the one achieving the best performance, as it is a good combination between few users, and number of cell towers, or capacity on the network.

Chapter 6

Device Difference Analysis

The goal of this chapter is to explore the impact put on the measurements by different device models. To do this 4 different devices will be used to gather measurements in the same context, and based on these measurements the difference in performance caused by the devices will be analyzed and evaluated. All the figures and the calculations in this chapter are performed using MATLAB scripts, that can be found on the enclosed CD *part_1_matlab/* .

6.1 Measurement Setup

Following is the setup used in performing the measurements, including the different devices and the scenario.

6.1.1 Measurement Scenario

In order be able to evaluate the impact on the performance of the devices, the measurements on the different devices must be done with context being as similar as possible. This means that the measurements will be performed on the same location, during the same time period, and using the same network.

The 4 devices will gather measurements for 24 hours from 15 to 15 o'clock from Monday to Tuesday, and will be placed on a window ledge with the screen facing up. The devices will gather the measurement using the same network as used in gathering the measurements analyzed in Chapter 4, where the Download and Upload throughput rates from the service provider are limited to 5 and 1 Mbit/s respectively.

During the 24 hours the 4 devices will be the only devices connecting to the server and performing measurements, why measurements from no other device will influence the scenario.

The NetMap system will furthermore be running with the same setup as described in Chapter 3, but with the inter test time set to 10 seconds, in order to not consume all the available data on the subscriptions.

6.1.2 The Devices used in the Analysis

In Tabel 6.1 a overview of the key features in this context can be seen. The 4 devices used are the low-end smartphone Samsung Galaxy Y, the mid-range smartphone Sony Xperia U, the mid-range tablet Samsung Galaxy Tab 10.1 and the high-end tablet Asus Nexus 7.

	Samsung Galaxy Y [wik13g]	**Sony Xperia U**[wik13h]	**Samsung Galaxy Tab 10.1**[wik13f]	**Asus Nexus 7**[wik13a]
CPU	Broadcom BCM21553 ARM11 832 MHz processor, ARMv6 (Single-core)	1 GHz Dual-core ARM Cortex-A9	1 GHz Dual-core Nvidia Tegra 2 processor	ARM Cortex-A9 Nvidia Tegra 3 T30L 1.2 GHz Quad-core (1.3 GHz Single-core mode) 1MB L2 cache
Memory	384 MB	512 MB	1 GB	1 GB
Android version	2.3.5 (Upgraded to 2.3.6)	2.3.7 (Upgraded to 4.0.4)	3.1 (Upgraded to 4.0.4)	4.1 (Upgraded to 4.2.2)
Connectivity	Wi-Fi 802.11 b/g/n, Cellular GSM, UMTS, HSDPA	Wi-Fi 802.11 b/g/n, Cellular GSM, UMTS, HSDPA	Wi-Fi 802.11 a/b/g/n, Cellular GSM, UMTS, HSPA, HSPA+	Wi-Fi 802.11 b/g/n, Cellular GSM, UMTS, HSPA, HSPA+
Compatible Networks	GSM 850/900/1800/1900, HSDPA 900/2100	GSM 850/900/1800/1900, HSDPA 850/900/1900/2100	GSM 850/900/1800/1900, HSPA 850/900/1900/2100	GSM 850/900/1800/1900, HSPA 850/900/1700/1900/2100
Launched	Aug 2011	Feb 2012	Jun 2011	Jul 2012

Table 6.1: The devices used to generate the data that is used in the comparison. Note that the Xperia U and Galaxy Y only have HSDPA which is HSPA downlink, and thereby not HSPA capabilities in the uplink.

6.1.2.1 The Measurements used in the Analysis

Following is some information about the measurements used in the analysis.

```
----------------------------------------------------------------------------
Notes about data:
----------------------------------------------------------------------------
Device: Xperia U
    RTT Measurements x10: 2204 (TCP: 1102, UDP: 1102)
    Throughput Tests: 2204, (Download: 1102, Upload: 1102)
    Connectivity Tests x30: 2404, (Timeouts: 917)

Device: Nexus 7
    RTT Measurements x10: 2213 (TCP: 1106, UDP: 1107)
    Throughput Tests: 2214, (Download: 1107, Upload: 1107)
    Connectivity Tests x30: 2378, (Timeouts: 2485)

Device: Galaxy Y
    RTT Measurements x10: 2056 (TCP: 1028, UDP: 1028)
    Throughput Tests: 2056, (Download: 1028, Upload: 1028)
    Connectivity Tests x30: 2147, (Timeouts: 11166)

Device: Galaxy Tab
    RTT Measurements x10: 2218 (TCP: 1109, UDP: 1109)
    Throughput Tests: 2218, (Download: 1109, Upload: 1109)
    Connectivity Tests x30: 2303, (Timeouts: 67)

----------------------------------------------------------------------------
```

From the measurements it was further seen that the Xperia U, the Nexus 7, and the Galaxy Tab only used UMTS during all the measurements, while the Galaxy Y only used HSDPA.

Furthermore, it would also be interesting to compare the RSSI and BER from the different devices, but among the 4 devices, only the Xperia U and the Galaxy Y return these values. This is because it is optional whether the hardware producers should return these values if requested via standard interfaces, such as used by Android[rss13].

6.2 Analysis of Measurements

In this section the measurements gathered in the previously described scenario will be analyzed.

6.2.1 Analysis of Round Trip Times

In Figure 6.1 the RTTs measured on the different devices can be seen during the 24 hours where the measurements were performed. From this figure it can be seen that the RTTs are more or less constant during 24 hours when the devices are stationary.

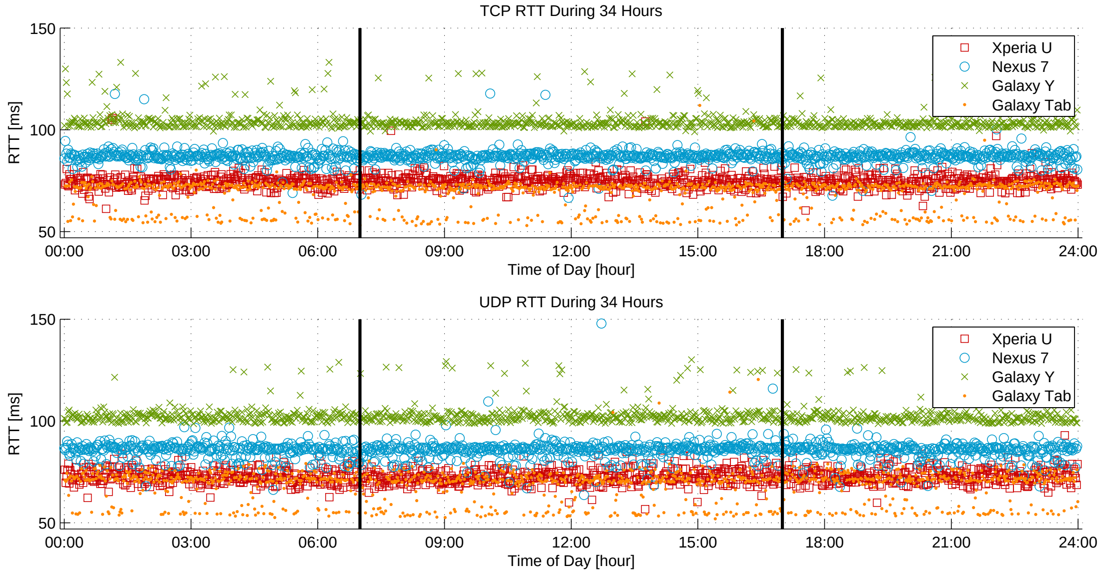


Figure 6.1: The measured RTT with 4 different devices during 24 hours.

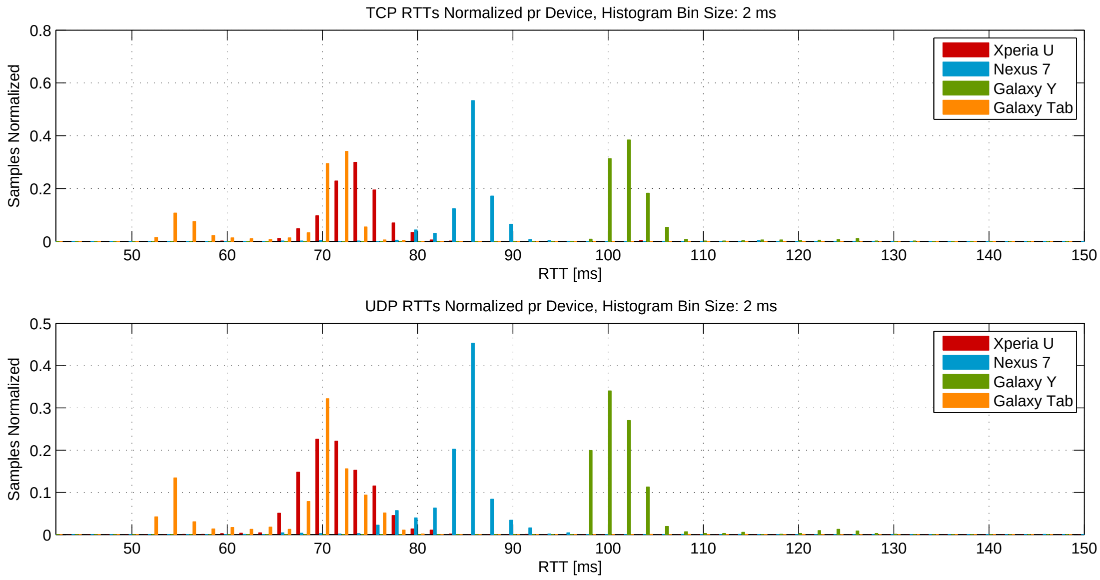


Figure 6.2: Histogram of the RTTs measured with 4 different devices during 24 hours, normalized pr device.

In Figure 6.2 a histogram of the RTTs measured for the different devices can be seen.

From Figure 6.2 it can be seen that the Galaxy Tab has the fastest RTTs between 50 and 60 ms, and a big cluster between 65 and 80 ms. The second fastest is the Xperia U with all the RTTs from this device between 65 and 80 ms. Third fastest is the Nexus 7 with all RTTs between 80 and 90 ms, and the device with the poorest RTTs is the Galaxy Y with RTTs between 100 and 110 ms.

It is interesting that the Galaxy Tab is the only one of the 3 mid/high end devices that obtains RTTs below 65 ms. It is also interesting that the Nexus 7 has slightly slower RTTs than the other two. Conversely it makes better sense that the Galaxy Y has the poorest RTTs, as it is the only low end device of the 4.

6.2.2 Analysis of Throughput Rates

In Figure 6.3 the Throughput rates measured during 24 hours on the different devices can be seen. From the figure it can be seen that throughput is much more affected by the time of day, which is especially noticeable for the busy period between the hours 7 and 17. Furthermore, it can be seen that at 1 o'clock the Download suffers from a sudden drop in performance, which is assumed to be due to backup of the server being performed.

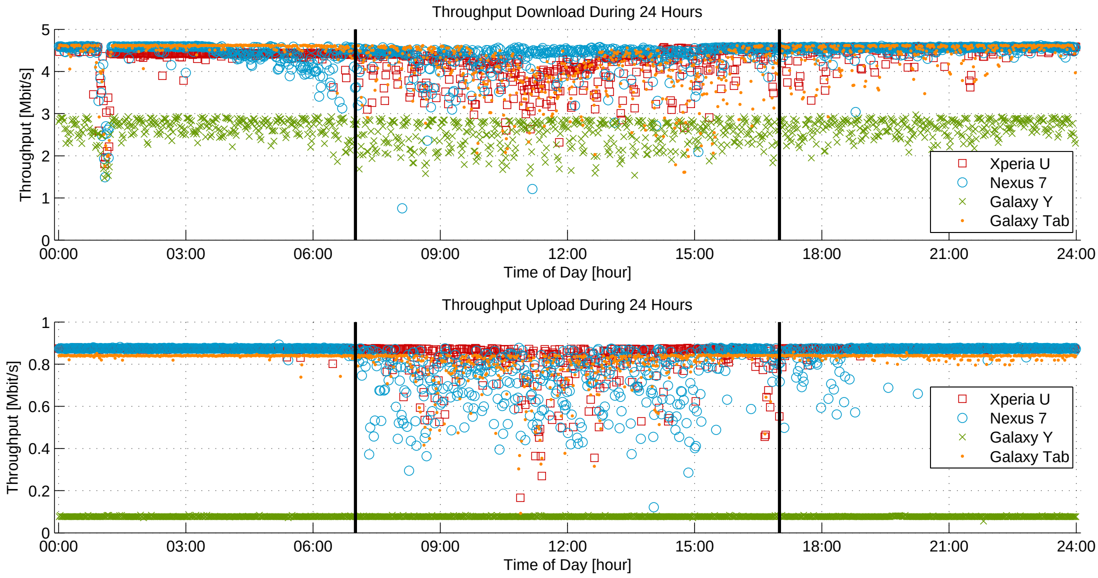


Figure 6.3: The throughput rates measured with 4 different devices during 24 hours.

In Figure 6.4 histograms of the Throughput rates measured can be seen. From the top part of this figure it can be seen that the Galaxy Tab again is the one achieving the highest rates, but now the Xperia U and the Nexus 7 achieve close to the same rates. However, for the upload the Galaxy Tab shows a little slower rates than for the Xperia U and the Nexus 7.

Also here the Galaxy Y is the poorest, as it was the case with the RTTs. The Download rates do not exceed 3 Mbit/s while the Upload rates are very low, below 0.1 Mbit/s. The lower Download rates could be explained by the Download rate being the parameter with the biggest dynamics, why a poorer performing device often would experience lower rates. But the extremely low Upload rates are quite surprising, as it is expected that most devices at least should be able to achieve 1 Mbit/s in both Up- and Download.

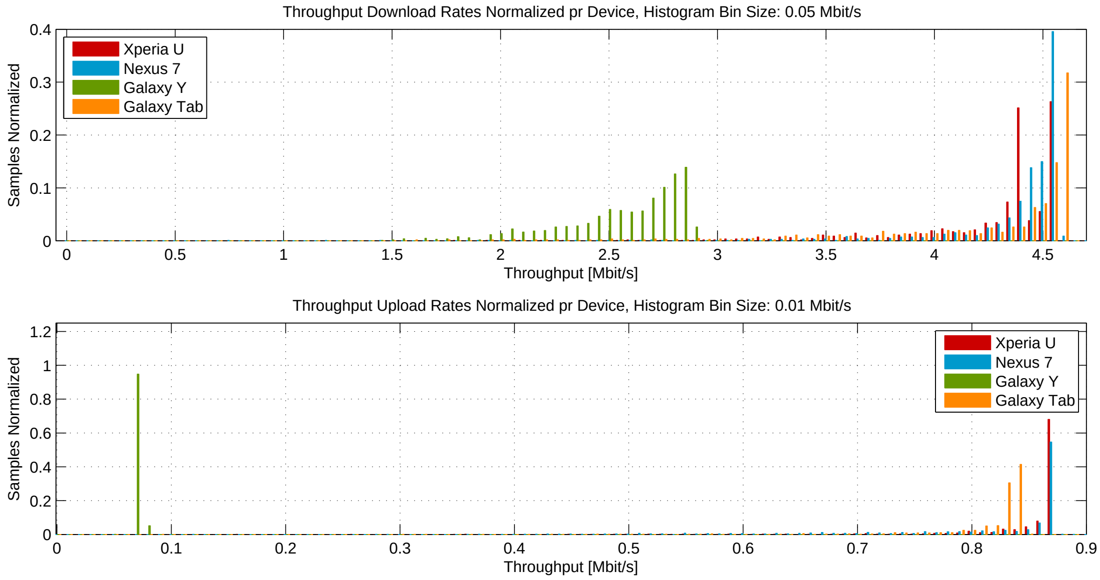


Figure 6.4: Histogram of the throughput rates measured with 4 different devices during 24 hours, normalized pr device.

6.2.3 Analysis of Connectivity

In the top part of Figure 6.5 the mean Connectivity pr 30 minutes, along with a smoothed curve of the points, can be seen. In the bottom part of the figure a histogram of the Connectivity for the different devices can be seen.

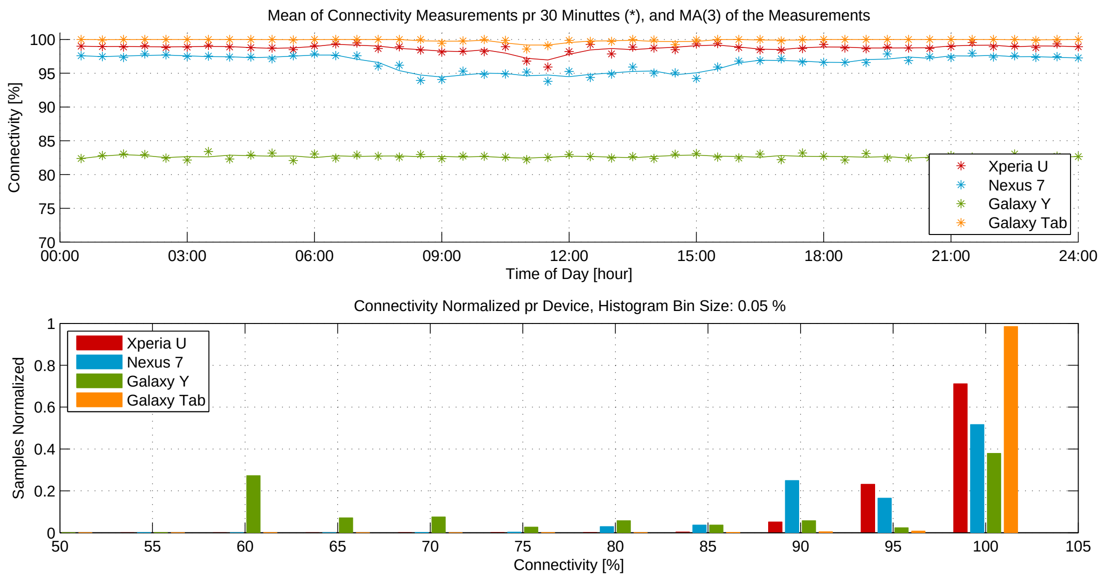


Figure 6.5: Top: Connectivity measured with 4 different devices during 24 hours. Bottom: Histogram of the measured Connectivity during 24 hours, normalized pr device.

From the top part of the figure it can be seen that the Connectivity for the different devices again follows the tendencies with the RTTs, where the Galaxy Tab has the highest Connectivity, followed by the Xperia U and the Nexus 7, and the Galaxy Y being the poorest again.

From the bottom part of the figure, from the histogram of the Connectivity for the different devices, it can be seen that the Connectivity for all the devices follows the same tendency from 80% to 100%, which is a ramp up. Furthermore, it can be seen that approximately half of the Connectivity measurements from the Galaxy Y is between 60% and 70%.

The low Connectivity measurements for the Galaxy Y can be explained by the extremely low Upload rates seen earlier. When the Upload test is running the buffers get filled quickly, and for the duration of the Upload test the Connectivity will timeout because of packets being queued up. Upon closer inspection it has been confirmed that this actually is the case, but only during the Upload test.

6.2.4 Conclusion of Device Difference Analysis

Based on the plots of the parameters RTT, Throughput and Connectivity previously presented, it can be concluded that the poorest performing device is the Galaxy Y, while the Galaxy Tab, the Xperia U and the Nexus 7 are much more equal in performance, perhaps with a small lead to Galaxy Tab.

So it can be concluded that the individual device can have a great impact on the experienced performance. This is naturally depending on the specifications of the individual device, as seen with the Galaxy Y. In the meta data the device make and model is also gathered, why, in a final measurement system, when sufficient measurements is gathered in an area the differences between different devices could be analyzed.

From the plots of the different parameters in this chapter, it can further be concluded that some devices are more alike performance wise than other. This could also be used in a final system, when looking at performance of individual devices in an area, some types could be grouped, and from this the performance of a specific device type could be predicted based on measurements from other devices, similar in performance.

Chapter 7

Analysis of Measurement Methods

In this chapter a thorough analysis of the currently implemented methods for measuring RTTs and throughput in the NetMap system will be made. The analysis will focus on describing the methods in detail, and based on this alternative methods of extracting results from the measurements will be suggested. This is especially with focus on what information is needed in a Network Performance Map (NPM), as described in 2.1. All the figures and calculations performed in this chapter are performed using MATLAB scripts, that can be found on the enclosed CD *part_1_matlab/* .

7.1 Ping Method Analysis

Currently NetMap measures RTT using both TCP and UDP by transmitting a train of 10 packets of 23 Bytes between the client and server. The packets within each train are sent in succession, meaning that as one response has been received, the next packet is transmitted as fast as the system allows it. In the analysis the average RTT of the 10 packets is used as the RTT value of the corresponding test.

The reason behind the averaging is that some of the extreme values that exist are smoothed in the results. An alternative could be to evaluate the individual measurements of each sequence, as this would increase the amount of available measurements and show the smaller fluctuations, which could then be filtered while processing the information if required.

In the following the ping method will be analyzed, by looking at the RTT of each individual packet in the 10 pings, to determine the distribution of the 10 RTTs and how they behave on different connection types: 3G and WiFi.

[utr13]When using the 3G connection the data is transmitted through the Radio Access Network, which includes mobile devices that transmits data using UMTS technologies to base stations and Radio Network Controllers, from where the data is transferred to the core network.

[acc13]When using the WiFi connection the device transmits data to the wireless access point, or router, from where the data is routed through the Digital Subscriber Line (DSL) gateway to some local exchange before reaching the access network and then the core network.

7.1.1 Distribution of RTT for Individual Packets Measured on 3G Connection

In this section the RTTs of the individual packets in the ping sequence, and the variation of these, will be evaluated. This is done by for each ping sequence of 10 packets, the smallest, or fastest, RTT is subtracted from all 10 RTTs of the sequence. In doing this for each ping sequence the fastest RTT is identified and the difference for the other RTTs to the fastest is found. This is done for all the available ping sequences and sorted in histograms.

The data used in this analysis is the same as described and used in Chapter 4. In Figure 7.1 the amount of sequences where packet 1, 2, 3, etc have the fastest RTT is counted.

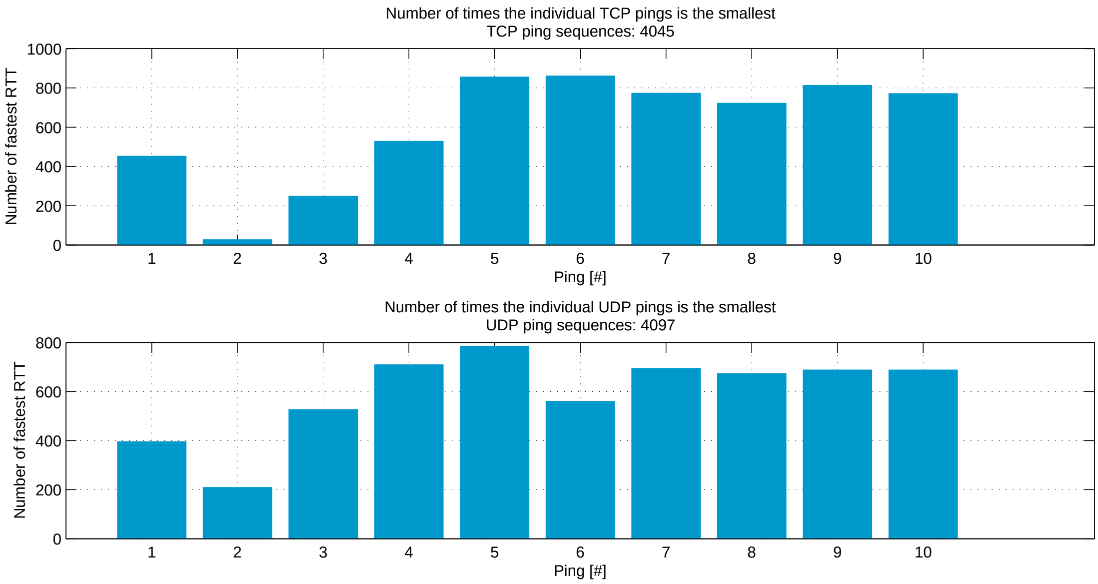


Figure 7.1: The number of times the individual ping has the fastest RTT, for TCP and UDP respectively. As the RTTs are represented in ms, there are ping sequences where more than one ping is the fastest, why the bars sum to more than the number of sequences.

From this figure it can be seen that for TCP it is less often that ping 1 to 4 in the sequence has the fastest RTT, and especially rare for ping 2 and 3. Ping 1 and 4 is actually closest to the expected amount, given that the fastest RTT is uniformly distributed among the individual pings, and given that only 1 ping can be the fastest. For the UDP sequences ping 1 and 2 are less often the fastest RTT, and especially ping 2.

In Figure 7.2 the difference between the fastest TCP RTT and the RTTs of the other TCP pings are sorted in histograms. The fastest RTTs are not included in the histogram, as this RTT is set as reference point, 0, to the other entries in the sequence, meaning that the other RTTs seen are relative to the fastest RTT.

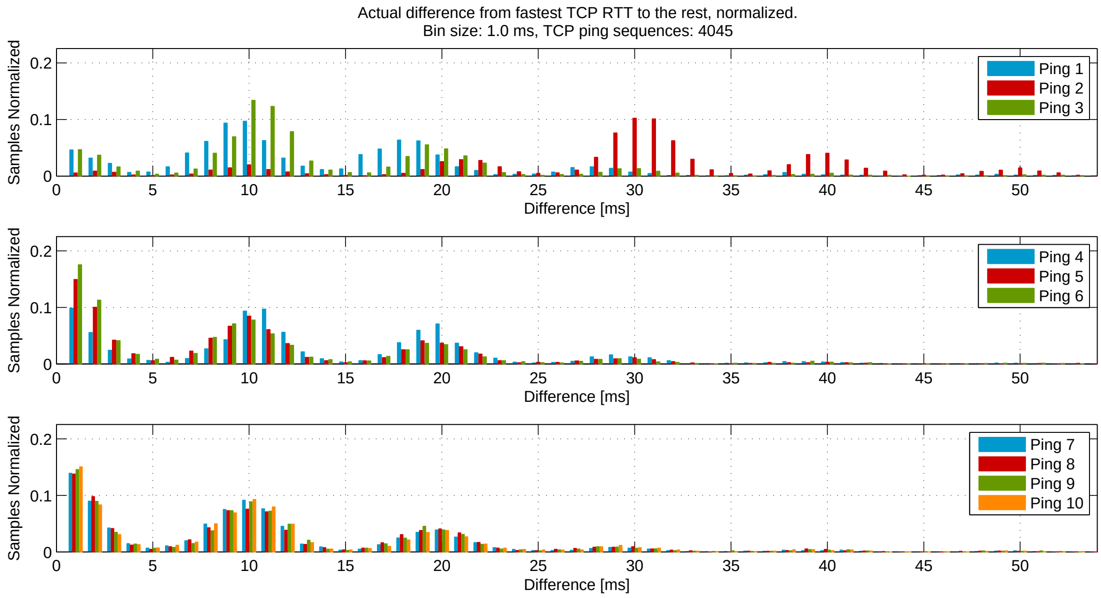


Figure 7.2: Histogram showing the TCP RTTs of the individual ping subtracted the fastest RTT of the sequence.

From this figure it can be seen that ping 2 most often is 30 or 40 ms slower than the fastest RTT, and only very rare below 25 ms. Furthermore it can be seen that ping 1 and 3 actually follows the tendency of ping 4 to 10, with the exception of having a little less below 5 ms than the others. It is very interesting that ping 2 is so distinguishable from the general tendency of the other pings, which is that most of them are located below 5ms, around 10, and around 20 ms slower than the fastest RTT.

In Figure 7.3 the difference between the fastest UDP RTT and the RTTs of the other UDP pings are sorted in histograms. The fastest RTTs are again not included in the histogram, as it is used as a reference point.

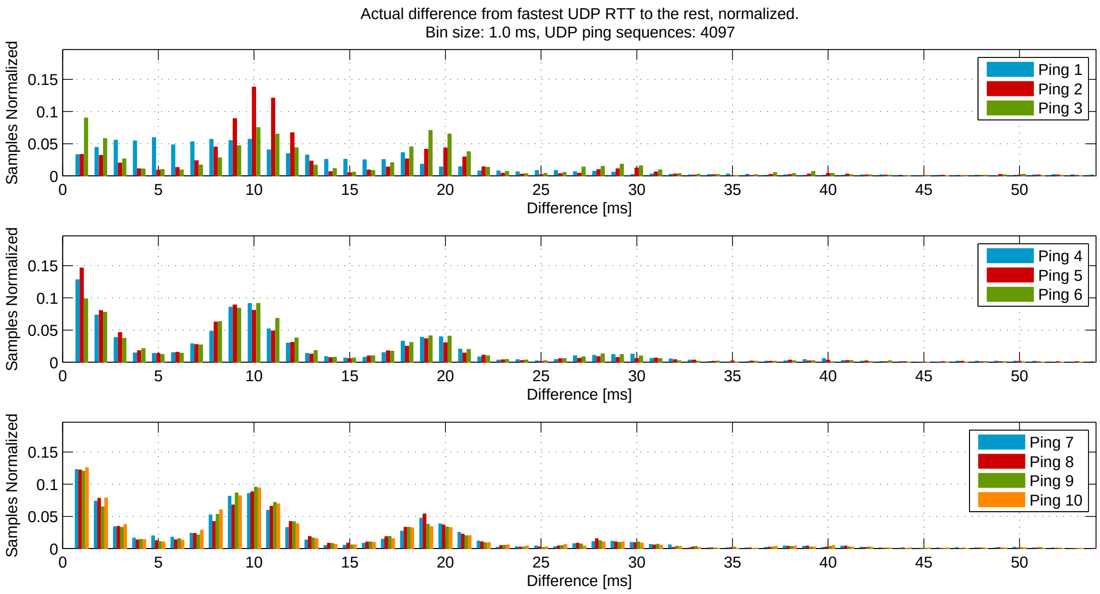


Figure 7.3: Histogram showing the UDP RTTs of the individual ping subtracted the fastest RTT of the sequence.

From the figure it can be seen that ping 1 and 2 are the two diverting the most from the rest. Ping 1 is almost uniform between 2 and 12 ms after the fastest RTT, after which it decays slowly. Ping 2 is very rarely below 5 ms and has a high amount around 10 ms, and otherwise follows the tendencies of the other pings.

Overall, from Figures 7.2 and 7.3, it can be concluded that the RTTs of TCP and UDP do not quite behave in the same manner, but for both protocols the RTT for the first 2-3 pings are behaving differently compared to the RTTs of the following pings.

This behavior of TCP and UDP will be investigated further in the next section.

7.1.2 Device Specific Analysis of Individual RTTs Measured on 3G Connection

In this section the behavior of TCP and UDP will be investigated further according to the device model that the RTTs were measured on. This is done by using the data gathered with the 4 different devices as described and used in Chapter 6.

In the top part of the following four figures the individual TCP and UDP pings for each device model are sorted into histograms and normalized, while the bottom part shows the cumulative sum of the normalized RTTs, again for the individual TCP and UDP pings for each device.

In Figure 7.4 the RTTs measured with the Xperia U can be seen, which is the same device as used to gather the measurements used in the previous section. From this figure it is very clear that TCP ping 2 for most of the sequences is 10-20 ms slower compared to the other pings in the sequence. For the UDP RTTs the almost uniform tendency for ping 1 can be seen, and also that ping 2 and 3 typically are a little slower than the others. These observations correspond with what was noted in the previous section.

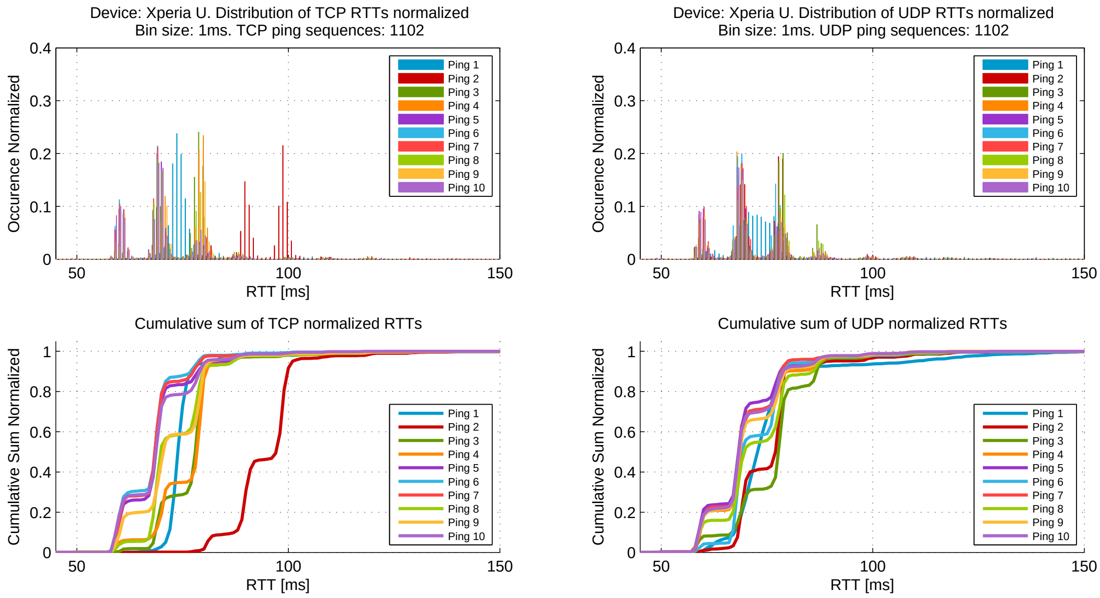


Figure 7.4: Histogram of the TCP and UDP RTTs measured with the Xperia U, placed stationary for 24 hours.

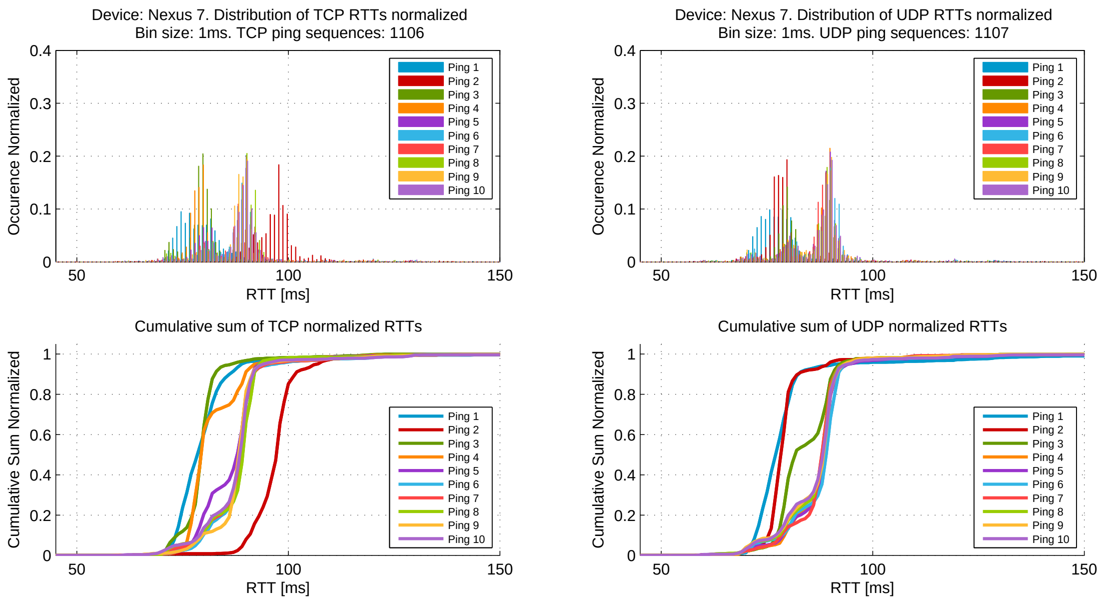


Figure 7.5: Histogram of the TCP and UDP RTTs measured with the Nexus 7, placed stationary for 24 hours.

In Figure 7.5 the RTTs measured using the Nexus 7 can be seen. For the TCP RTTs ping 2 is again typically the slowest in the sequence, while the other 9 pings tend to cluster around 90 and 80 ms. For the UDP RTTs the pings more or less are divided into 2 clusters around 80 and 90 ms. Compared to the Xperia U, the Nexus 7 delivers much more consistent RTTs both for TCP and UDP, that is the spread is much smaller than for the Xperia U.

In Figure 7.6 the RTTs measured with the Galaxy Y can be seen. From this figure it can be seen that, as concluded in Chapter 6, that this is the device delivering the poorest results of the 4. For the TCP RTTs ping 2 again typically is the slowest, while the other 9 pings, with a little exception, are very consistent around 100 ms. For the UDP RTTs again ping 1 has an almost uniform tendency, while the other 9 pings being consistently placed around 100 ms. So despite the Galaxy Y delivering relatively poor RTTs for both TCP and UDP, it seems to be a very constant device in performance.

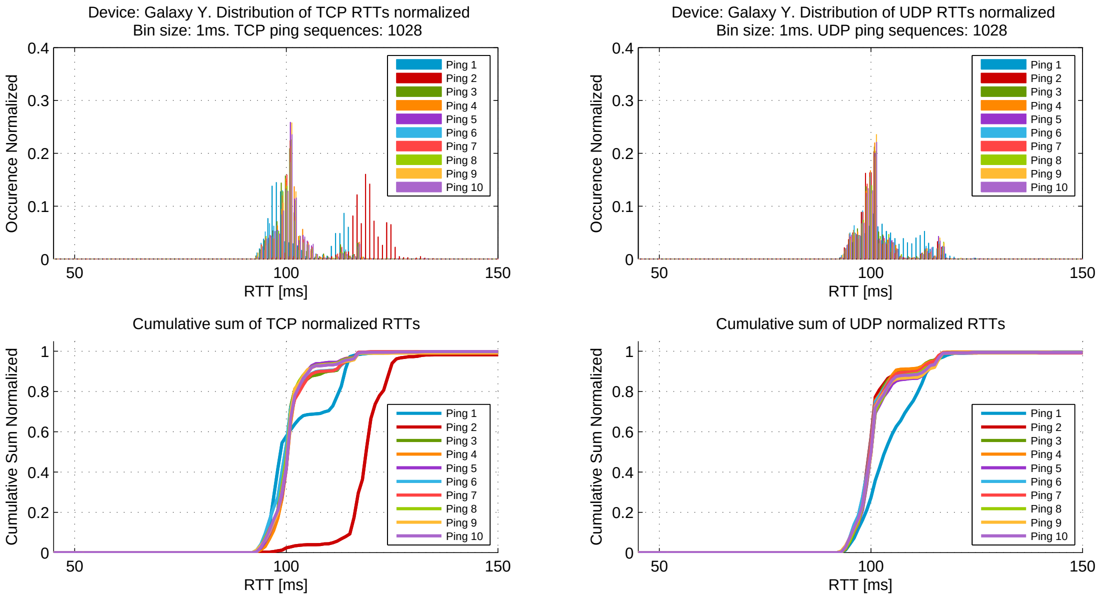


Figure 7.6: Histogram of the TCP and UDP RTTs measured with the Galaxy Y, placed stationary for 24 hours.

In Figure 7.7 the RTTs measured using the Galaxy Tab can be seen. For the TCP RTTs it can be seen that also here ping 2 typically is a little slower than the other pings, with the exception of ping 1. Upon further inspection of the RTTs of ping 1 it was found that a large population (70%) of the RTTs are located between 210 and 240 ms, both for TCP and UDP RTTs. But besides ping 1 for both TCP and UDP, the Galaxy Tab is actually the device delivering the most consistent and fastest RTTs of all 4 devices.

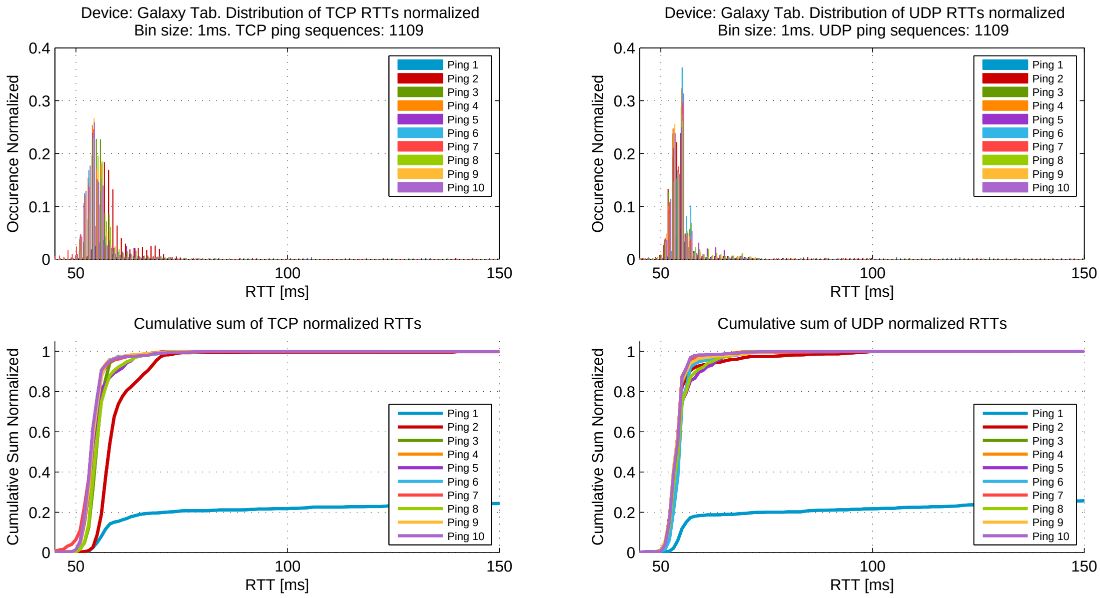


Figure 7.7: Histogram of the TCP and UDP RTTs measured with the Galaxy Tab, placed stationary for 24 hours. Note that ping 1 for both TCP and UDP RTTs has 70% of the population between 220 and 240 ms.

These plots indicate that the implementation of the protocol stack on the individual device has an impact on how well it performs, noticeable when looking at the first 1-3 pings for TCP and the first 1 ping for UDP. This assumption is based on the fact that the same socket API is used on all devices, and that the devices measure the RTTs on the same 3G network using the same ISP.

For the TCP the slower RTTs of ping 2 for all the devices could however indicate that the 3G network, and the way it handles TCP streams has some impact on the measurements, but the great variance between the different pings on the different devices indicate that it is the devices that have the greatest impact. After the first few pings the RTTs seem to stabilize around the fastest RTT, which could indicate that the implementation of the protocol needs some time, or a few packets, in order to reach peak performance.

For the UDP the device specific tendency is even more outspoken than for TCP. The first UDP ping is for 2 of the devices slower, and for 1 of the 2 extremely slow, while for the other 2 no noticeable difference in speed is seen. This indicates that the implementation of the protocol on the devices has an impact on the performance of the RTTs. Furthermore, the fact that for 2 of the devices all RTTs, except for ping 1, is clustered within 10 ms, while for the other 2 devices the RTTs are distributed over 2-4 spikes and 20-40 ms, could indicate an impact of the device specific implementation.

In order to be able to say more about what impact the network, or rather the connection and access network type, has on the measurements, RTT measurements performed on a WiFi/DSL connection will be evaluated in the next section.

7.1.3 Analysis of Individual RTTs Measured on a WiFi Connection

In this section the behavior of individual RTTs measured on a WiFi/DSL (from here on simply referred to as WiFi) connection will be analyzed.

The data is generated by measuring with a Sony Xperia U (see Table 6.1) on a wireless LAN, which is connected to a DSL connection limited to 10 Mbit/s download and 1 Mbit/s upload by the service provider Telenor. By using this connection it is ensured that the measurements traverse another network before entering the danish research network where the server is placed. The measurements were performed during 14 hours, from 18 in the evening to 8 in the morning. During this period it is expected that the cross traffic will be relatively low, and thereby the effect on the measurements will be minimal.

In Figure 7.8 the amount of sequences where packet 1, 2, 3, etc have the fastest RTT is counted, for measurements performed on a WiFi connection.

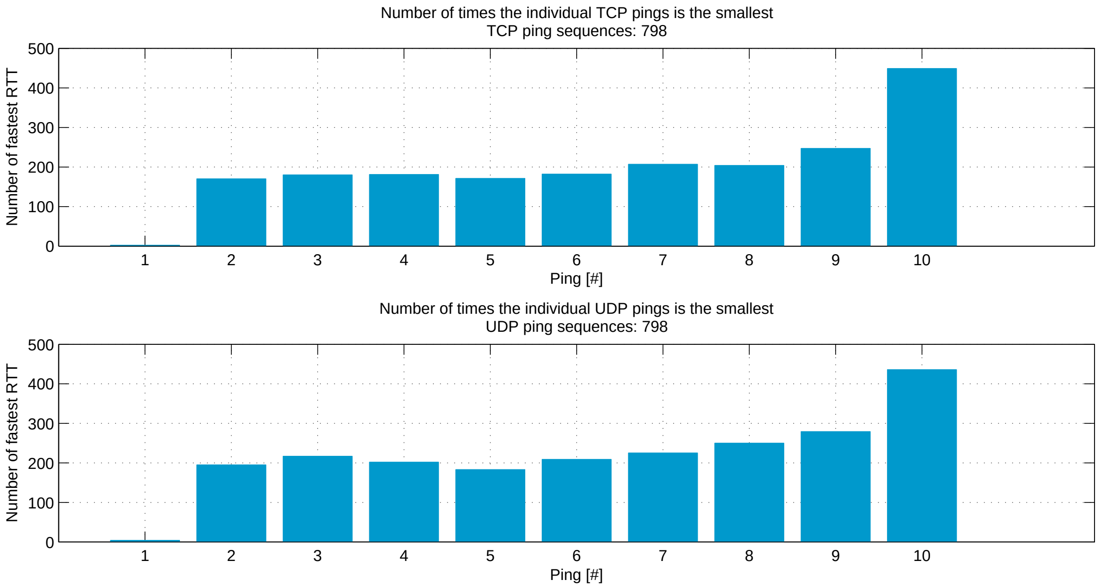


Figure 7.8: The number of times the individual ping has the fastest RTT, for TCP and UDP respectively, measured on WiFi. As the RTTs are represented in ms, there are ping sequences where more than one ping is the fastest, why the bars sum to more than the number of sequences.

From this figure, if compared with the equivalent for the measurements performed on a 3G connection in Figure 7.1, 2 significant differences can be noted. Firstly, where for 3G ping 2 most rarely was the fastest RTT, both for TCP and UDP, here for WiFi it is now ping 1, which intuitively makes more sense due to route discovery. Secondly, for 3G there were no single ping that most often achieved the fastest RTT. This is for now WiFi the case, as ping 10 is the one most often achieving the fastest RTT, again both for TCP and UDP, however it most likely shares the fastest RTT with another ping as the sum of the bars are much more than the amount of ping sequences.

In Figure 7.9 the RTTs measured with the Xperia U on the WiFi connection can be seen.

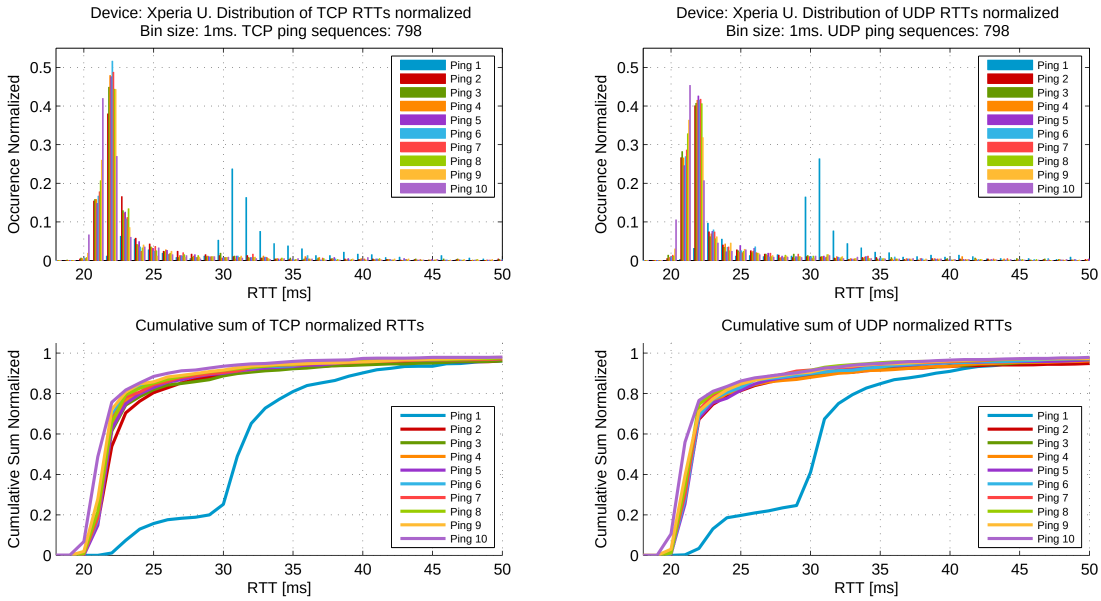


Figure 7.9: Histogram of the TCP and UDP RTTs measured with the Xperia U, placed stationary on a WiFi connection for 14 hours.

From this figure, when compared to the equivalent for measurements performed on a 3G connection in Figure 7.4, the significant differences can again be seen. First off, the reason for ping 1 to only very rarely having the fastest RTT is seen, as it consequently, and both for TCP and UDP is 5 to 10 ms slower than the RTTs of the other 9 pings. Furthermore, it can be seen that the 10 ms periodicity that was seen in the measurements performed on the 3G connection now have disappeared, and instead the RTTs of all the pings (except the first) are located in one big cluster between 20 and 25 ms.

7.1.4 Variation of RTTs

In this section the spread of the RTTs per ping sequence will be analyzed. This will be done by using the RTTs measured on both 3G and WiFi and only from the Xperia U device (see Table 6.1). The 3G measurements are from two different data sets, one is the stationary measurements gathered and used in Chapter 6, and the other is the measurements gathered while driving between Brovst, Aalborg and Aarhus, as described in Chapter 4. The WiFi data set is the measurements gathered and described in the previous subsection.

The spread of the RTTs are analyzed by for each ping sequence of 10 RTTs calculating the mean and the 95% confidence interval of the mean. Then the size of each confidence interval for each ping sequence is found, see Figure 7.10

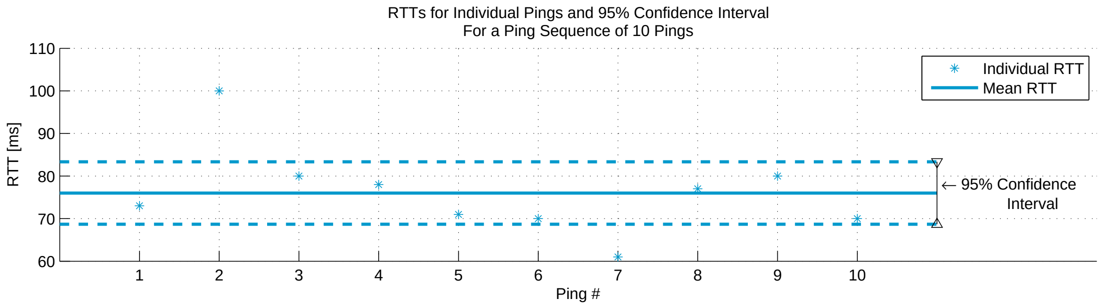


Figure 7.10: A single ping sequence of 10 RTTs, where the mean and the 95% confidence interval of the mean is marked. The confidence interval is found as described in Appendix B.1.

For each ping sequence in the 3 data sets, the size of the 95% confidence interval is calculated, and when done for all ping sequences, they are sorted in a histogram, which is seen in Figure 7.11. This will describe the variation in the spread of RTTs per ping sequence.

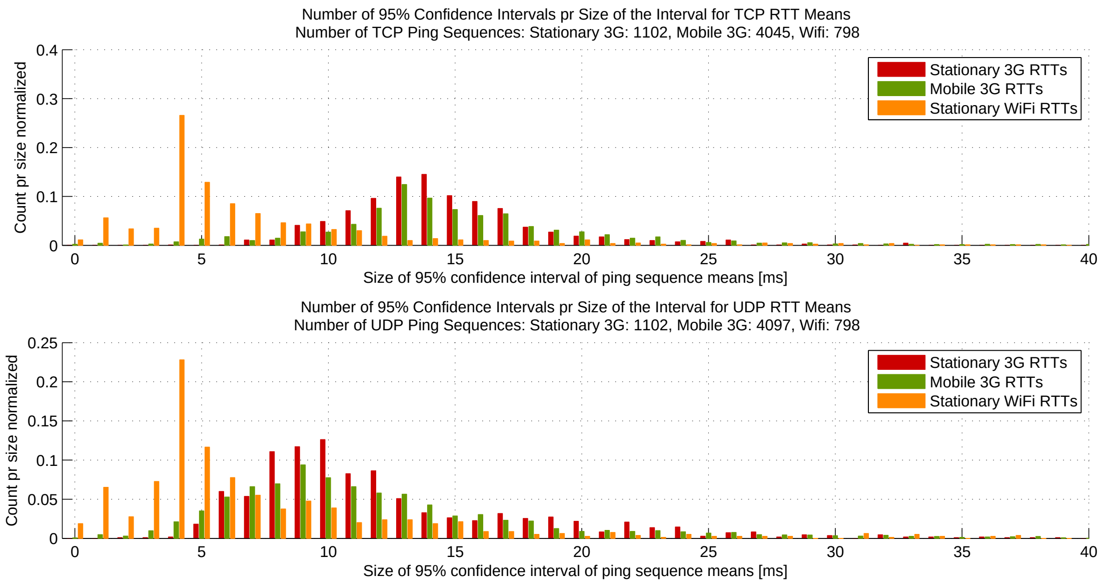


Figure 7.11: Amount of confidence intervals of the mean RTT of different sizes, compared for 3G, stationary and mobile, and WiFi measurements for same device model.

From this figure the difference between the spread of RTTs measured on a WiFi connection and RTTs measured on a 3G connection while moving and while being stationary. From this it can be seen that, as described earlier, the variation of the spread of the RTTs measured on a WiFi connection is smaller than the RTT measurements performed on a 3G connection. Furthermore, it can be seen that the variation of the spread of the mobile 3G measurements is a little bigger than for the stationary 3G measurements. This makes sense, as the when moving around while measuring, there is a possibility that handovers will occur, or that the received signal properties will change, during the measurement of the ping sequence. This could then cause some of the RTTs to fluctuate significantly, compared to the mean, which will cause the confidence interval to increase in size.

Furthermore, a difference between the spread of TCP and UDP RTTs can be seen. The confidence intervals for RTTs measured on WiFi TCP and UDP are similar, but the confidence intervals for RTTs on peak about 4-5 ms later for TCP than for UDP, meaning that the confidence intervals typically are wider for TCP than for UDP. This is most likely caused when packets are dropped while a TCP retransmission is taking place, increasing the RTT compared to the other RTTs in the sequence. But for UDP when a packet is dropped, it is not included in the calculation, and thereby not compared to the other RTTs in the sequence, as it does not exist.

7.1.5 Summary and Discussion of Individual RTT Analysis

Based on this analysis of the individual RTTs, by using 2 different datasets for 3G measurements and one set for WiFi measurements, it can be concluded that the individual pings in the sequence are affected differently by different devices, that is the implementation of the protocol stack or the handling of the protocols varies for different devices.

The reason for the different implementations across devices could be explained by the chip-sets being used in the various models, where each chip-set, beside the physical differences, might have different priorities regarding power usage and priorities, or an alternative scheduling. Especially changes to priorities across devices could help explain the differences for the 4 devices and how the RTTs appear in 1 or 4 spikes, but these are only speculations.

Through the analysis of measurements performed on 3G, it was noted that for TCP the second ping most often delivered RTTs slower than the mean, which was consistent for all devices tested. Furthermore, the distribution of the RTTs for the different pings also showed variation on the different devices.

Likewise, for the UDP measurements on 3G the first 2 pings also behaved differently than the other 8 pings, and the distribution of RTTs on the different devices also varied.

Optimized Result Extraction

From the analysis performed on the RTTs of the individual pings in this section, it was apparent that the some level of detail is lost when only taking the mean of the 10 RTTs and using that as the result. The removed information from taking the mean includes the distribution and thereby the variation in the RTTs of each ping sequence.

Optimally all the individual RTTs are kept and a distribution is fitted to the data when the performance in an area is requested. This will however require a lot of storage space and computational resources each time the data is requested. A more economical alternative to this could be to only store the mean value and some information about the spread of the RTTs per ping sequence.

The spread of the ping sequence is interesting, because it says something about the variability of the network, and because the 10 RTTs are measured as fast as possible, spread is an approximation of a momentarily variance of the network. But in order to gain a higher confidence level in the variance measured, more pings could be used, i.e. 15 or 20. This will make the results stronger statistically, while increasing the data usage minimally as only 23 bytes are transmitted per ping packet. The amount of information saved as results per RTT measurement could then be reduced to 2 or 3 values (mean, spread information, and number of successful pings) instead of the current 10.

By increasing the number of pings used per measurement, the impact of the device could then potentially be reduced, as 1 packet of 10 has much higher influence than 1 of 20 packets. This is referring to the variability seen when comparing the different devices, where sometimes 1 particular ping of the 10 could skew the mean value significantly.

Furthermore, from the analysis it seemed that ping 1 and/or ping 2 often showed RTTs deviating from the norm of the other 8 or 9 pings, and therefore it could be chosen to disregard these. This is however a little problematic, as it is not known with certainty whether it is the devices causing this behavior, or if it is the network. If it is the devices that cause this, and they are removed, then the RTT measurements and results will get more focused on the network performance. If it is not caused by the devices, then by disregarding them, important characteristics about the network will go lost. Therefore, this would need to be analyzed in greater detail.

Possible Further Analysis of RTT Measurements

Further analysis and evaluation of the RTT measurements and the method for measuring them could be done. For instance the impact of the size of the request and reply packet could be analyzed, that is, how would the different pings behave if the packet size was increased. This is for now left undone, as one of the aims of the system is to use as little data as possible in estimating the network performance.

Another area that could be analyzed further, is the difference in impact of the connection technology used by the device. That is, do the individual pings operate differently while using UMTS compared to when using HSPA. This does however require a device that can control whether only to use the one or the other. Such a device is not available to the project group why this also is not looked into now.

Lastly, one could also set up a controlled test environment, where the different measurement methods could be analyzed. Such environment could be done either by using a real hardware setup or simulations. Cellular network hardware is very expensive and requires special permissions to set up, why such hardware is not easily accessible. One could also use some of the known network simulation tools such as OMNeT++[omn13] or NS-3[ns313] to test the measurement method. By this approach it is difficult to get a realistic view of the network, but it gives the possibility to control some of the unknown parameters.

7.2 Throughput Method Analysis

The Throughput measuring method currently implemented in the NetMap system is a brute force method using TCP that, within 15 seconds, transmits as much data as possible.

The 15 seconds should be seen as the time allowed for either the device or the server to transmit data. In the Upload test the device transmits for 15 seconds and terminates with an end-packet, and on the server the transmission of data is timed from the first packet to the end-packet is received. Likewise, in the Download test the server transmits for 15 seconds and terminates with an end-packet, and on the device the transmission of data is timed from the first data packet to the end-packet is received. Both types the test might therefore run a little longer than the 15 seconds, why these times, and not the 15 seconds are used in the calculation of throughput.

The amount of data put through the connection between the device and the server is controlled by the congestion avoidance mechanism implemented in the TCP. The congestion avoidance increases the congestion window size until a timeout, after which it lowers the congestion window size and slowly increases the congestion window size again. This behavior means that the utilization of the connection fluctuates around the highest throughput available, and not constantly kept high.

The result of a throughput test is, as mentioned, calculated from the total data transfered and the total test time. Because of the relatively long test time of 15 seconds, the result from this procedure should not be affected too much by the variation caused by the congestion avoidance. But another factor that might influence the result is the slow start that the TCP uses in the start of the stream, where the congestion window size is doubled until a threshold, from where the congestion avoidance takes over. This slow start might influence the result as it is not taken into consideration, that is the data transfered during this period and the time duration of the period are not removed before calculation of the throughput result.

The impact of the slow start and the fluctuations caused by the congestion avoidance on the results will be analyzed in the following section.

7.2.1 Analysis of TCP Streams

In this section the behavior of the TCP stream during the Download and Upload test will be analyzed. This is done by capturing the traffic on the server and evaluating and comparing it to the results obtained and saved in the NetMap system. The traffic is only captured on the server side, as the found methods for capturing packets on the Android device turned out to have too great impact on the performance of the throughput measurements, why the traces could not be used.

7.2.2 Measurement Scenario

10 sets of Download and Upload measurements were made, while the packets were captured on the server. Only packets going to and from TCP ports 60003 and 60004 were captured, and between each test the capturing was stopped, saved, and a new started.

On the device the application was started and the stop button was pressed just after, which meant that both the handover listener and the connectivity module were stopped during the RTT measurements, before the Throughput measurement were scheduled. By doing this, no other measurement were running while the throughput tests were performed. Furthermore, for each set of Upload and Download tests the result obtained by the system was stored as well for later comparison.

7.2.3 Measurements and Results

For both Download and Upload the initial connection setup packets, that is the SYN, SYN/ACK and ACK packets, are discarded. Then the throughput is calculated by using a Moving Average process, calculating the throughout for the 20 previous packets, which is used to account for the extreme spikes that otherwise would be present in throughput. For the Download test, only packets with source port 60004 are used, and for Upload only packets with destination port 60003.

This gives the throughput experienced during the test, along with the fluctuations caused by the network and the congestion avoidance in TCP that tries to regulate according to this. Furthermore, the slow start that follows when using TCP can also be seen, which happens in the initial phase of the data transfer. After the slow start, the congestion avoidance takes over, and tries to optimize the throughput according to the link. The slow start and the initial rise in throughput controlled by the congestion avoidance, is called the rise time.

The result from the NetMap system will then be lower than the mean calculated after the rise time, as the low throughput achieved during the rise time is included in the result. The rise time can, and most likely will, variate from test to test. Therefore it is found for each test as the time from the first packet, until the first time where the throughput drops after being higher than the result obtained from the NetMap system.

Other methods for obtaining the rise time could be used, such as when the throughput rates settles, i.e. changes are below some threshold for a time duration, but for now the previous method is used, because it (upon visual inspection of the tests) gives good results.

In the following figures the mean of all the throughput rates (including rates before rise time) will not be compared to the NetMap results since these two essentially are the same.

In Figure 7.12 the throughput calculated from the packets of one of the 10 Download and Upload sets can be seen. Furthermore, the throughput registered by the NetMap system in this sample is marked with a black line, and the mean throughput calculated for the data after the initial rise time is also marked.

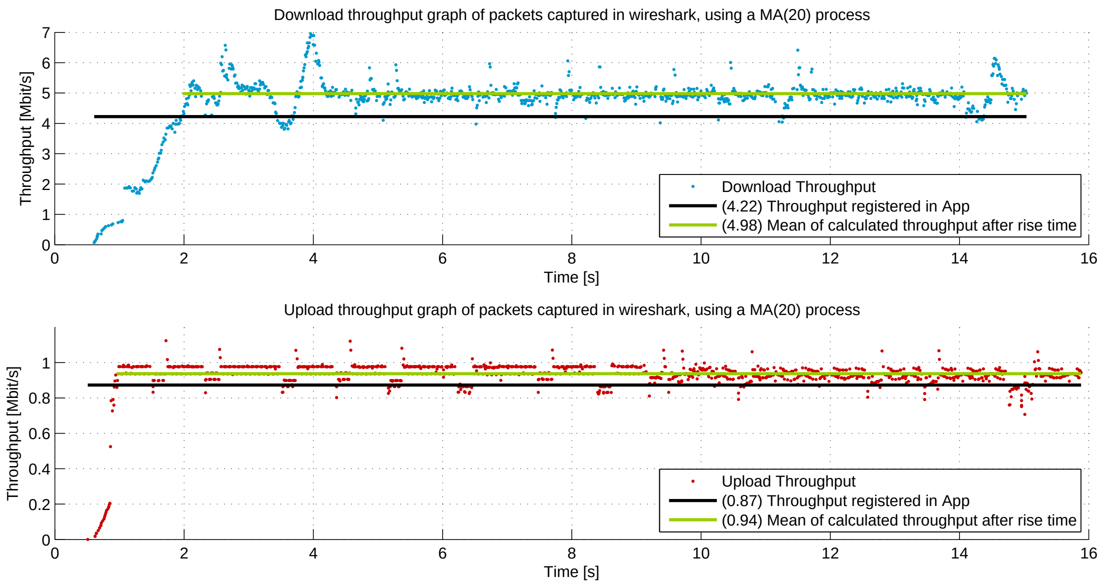


Figure 7.12: The calculated Upload and Download throughput from one sample. The rates are found by using a Moving Average process, finding the average throughput over the last 20 packets. The difference in length of the measurements is caused by where the data is transmitted from, see Section 3.5 for detailed description. The calculations is made without protocol header, so the throughput is calculated on application layer.

From the figure it can be seen that for the Download the slow start raises the throughput until around 2 Mbit/s from where the congestion avoidance takes over and slowly increases the throughput to about 5 Mbit/s, which is the limitation set by the service provider. From the remaining data transfer the throughput rate fluctuates around 5 Mbit/s, where congestion avoidance causes the rate to rise above and fall below the 5 Mbit/s.

For the Upload the maximum throughput rate of 1 Mbit/s is achieved during the slow start. For the remaining of the data the throughput rate fluctuates around 1 Mbit/s, again controlled by the congestion avoidance.

For the Download the peaks higher than 5 Mbit/s are caused by the fact that the server has a much faster connection than the device, why when buffers are empty on the connection path, more data can be transfered briefly, until the device, or the devices ”part” of the network, lets the server know that it should lower the transfer rate.

For the Upload it is the device that is transmitting and the fluctuations are much smaller. That is because it is not the servers part of the network that tells the device to slow down, but the devices part of the network, which is closer to the device, effectively making the fluctuations smaller and correcting the rate faster.

In Figure 7.13 the occurrence of throughput rates according to the time in the test are seen in a histogram. That is for each interval of 2 seconds the occurrence of the different throughput rates are counted, and for each interval the occurrences are normalized.

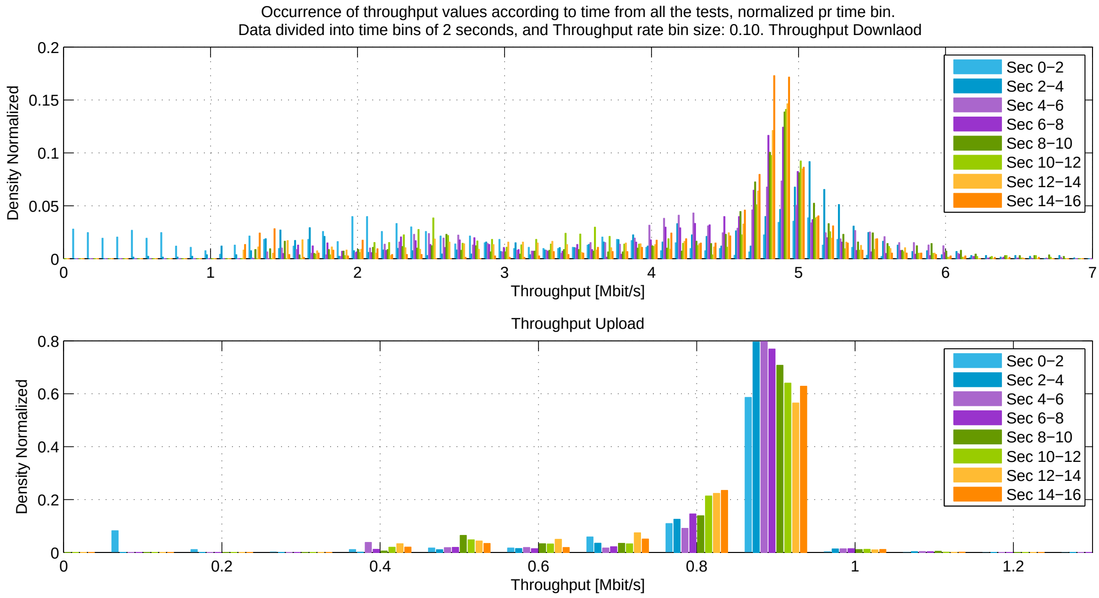


Figure 7.13: Histogram of the throughput rates calculated at 2 second time intervals from all the tests.

This plot shows that generally the 10 different sets of Download and Upload tests follows the same tendency as the single sample shown in Figure 7.12. From the top part of the figure it can be seen that in the download tests it is only in the first 2 seconds where rates below 1 Mbit/s are obtained. Furthermore, from 4 to 16 seconds in the tests the majority of the throughput rates achieved are around 5 Mbit/s.

From the bottom part of the figure it can be seen that for Upload the maximum speed limit set by the ISP is reached fast, that is for the 0-2 second interval approximately 70% of the occurrences are 0.8 Mbit/s or higher. The maximum speed limit is reached within the exponential rise time in the TCP slow start, which also is seen in the Download histogram. Because of this the throughput calculated for Upload after the rise time will be closer to the result from the NetMap system, when compared to the Download.

7.2.4 Accuracy of Throughput Measurements

In this section the accuracy of the throughput result from the application will be compared to the mean throughput calculated from the packet traces after the rise time. That is, the mean value of the throughput based on the values measured after the rise time divided with the value obtained through the NetMap system. This is done based on the assumption that the throughput calculated from the data after the rise time is assumed to be the estimate closest to the real value of the network at the time of measurement.

In Figure 7.14 the calculated accuracy for the Download and Upload rates can be seen.

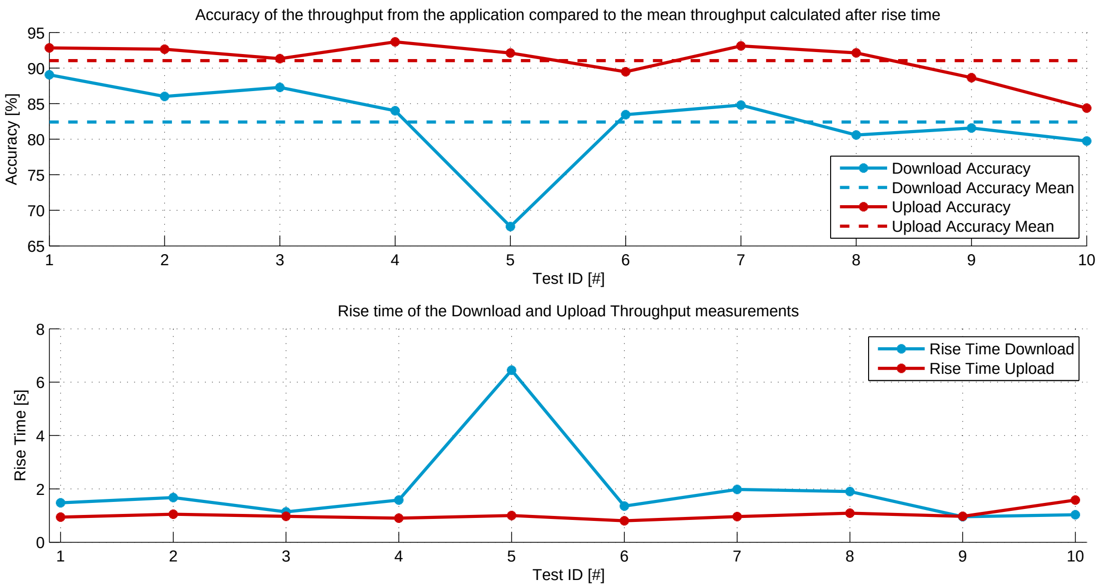


Figure 7.14: Top: The accuracy of the throughput rates obtained by the system, compared to the throughput calculated after the rise time.

From the top part of the figure it can be seen that the accuracy for the Download is slightly lower than for the Upload. This is caused by the longer rise time for the Download tests than for the Upload tests, which, as mentioned before, means that the results gathered from the NetMap system is much closer to that calculated after the rise time in the Upload because the maximum throughput limit is reached much earlier in the transmission.

In the bottom part of the figure the rise time for the different test sets can be seen, both for Download and Upload. From this it can be seen what has been stated earlier, that the rise time for the Download generally is a little (around 1 second) longer than the rise time for Upload.

7.2.5 Summary and Discussion of TCP Stream Analysis

From this small analysis of the behavior of the TCP streams that are used to measure the throughput in the NetMap system, the impact of the rise time on the obtained results was seen. More specifically that the rise time of the Download has a greater impact on the Download results than the impact of rise time for Upload on the Upload results. This could be utilized in optimizing the Throughput measurement method in order to get more precise results in relation to the maximum capacity of the connection, but also in relation to the amount of data needed to be transfered to obtain a more correct result.

To get a more precise result, the data transfered during the rise time could be skipped before starting counting the bytes transfered. That is, for Download the first 2-4 seconds could be used to warm up the connection and giving TCP and the congestion avoidance time to settle around the maximum throughput rate. Then after the initial time the bytes should be counted for the remaining time of the test to estimate the throughput and the maximum capacity of the connection.

Furthermore, by doing this the duration of the test could be lowered such that only 5 seconds are used to measure the throughput, after the initial 2-4 seconds. This procedure, with a total throughput test time of 9 seconds, could potentially lower the data usage of the test with 30%.

Alternatively, instead of just skipping the initial 4 seconds worth of data and rates, this could also be used to estimate the performance of the connection. That is, the data transmitted during this time could be used to estimate the rise time characteristics, where the lower the rise time, and the more data transfered during the rise time, the better the connection. The evaluation of the rise time characteristics should be done by using the result from the 5 seconds of throughput estimation, for instance as a ratio between them or a percentage of the maximum throughput.

Optimized Result Extraction

As mentioned earlier the current implementation of the throughput estimation simply uses the average of the whole transfer captured on the application layer. This method is not incorrect, but as seen throughout this section it does lack some level of detail. In the following, two different methods for obtaining more detailed throughput estimations are suggested.

The first suggestion is time duration oriented, where the idea is to define a duration that covers the rise time, i.e. 2 seconds for Download and 1 second for Upload, after which 5 seconds are used to estimate the maximum throughput rate. In doing this the characteristics of the rise time, and the rates during this, can be used as a metric for the connection in addition to the actual maximum throughput rate.

The second suggestion is rate oriented, where the duration of the throughput estimation is set to 10 or 15 seconds, during which the throughput rate is sampled. The rate of sampling could, if set too high, influence the throughput rate, why this must be tested, but could be 10 samples pr second. After the data transfer is done, the slowest 20-30% of the rates (the rise time) are discarded, and maybe also the highest 10-15% (the peaks), before the mean throughput is calculated from the remaining rates. This procedure is inspired by Speedtest[spe13].

Both methods offer the potential for getting a higher level of detail of the throughput estimation, which is the goal. Furthermore, by increasing the level of detail obtained pr measurement, the amount of data needed to transfer in order to estimate the maximum throughput rate can be lowered, making the system more efficient.

Part II

Novel Methods of Estimating Throughput

Overview of Part II

This part of the report will focus on analyzing and researching alternative and novel methods for estimating throughput on any given network, with focus on adaption into the NetMap system.

Chapter 8 introduces the terminology that will be used throughout this part, followed by a brief description of the cellular technologies considered. Also a description of existing methods for estimating throughput or available bandwidth is given.

In Chapter 9 the first of the considered techniques is described and analyzed, the Packet Pair Technique (PPT). The technique is tested with various parameters, followed by a conclusion and discussion that sums up the results obtained.

Chapter 10 describes the second of the considered techniques, that is the Combined Probe Gap Model (PGM) and Probe Rate Model (PRM) Technique. This technique is also tested with various parameters followed by a couple of tests using WiFi and finalized with a conclusion and discussion.

An overall conclusion and discussion of this part is given in Chapter 11, which sums up the results obtained using each of the techniques, and discusses if these are viable as replacements of the existing throughput estimation method.

Chapter 8

Introduction

In this chapter a description of the network metrics capacity and available bandwidth will be given, followed by an outline to the network technologies that are used while measuring these.

8.1 Network Terminology

The metrics, ”Available Bandwidth” and ”Capacity”, can help specify what is being measured and under which circumstances the measurements are performed.

A simple illustration has been made to show what is meant by each metric which can be seen in Figure 8.1. The term channel describes the communication path between a device and the backbone of the network and should not be confused with wireless communication channels which contain multiple communication paths.

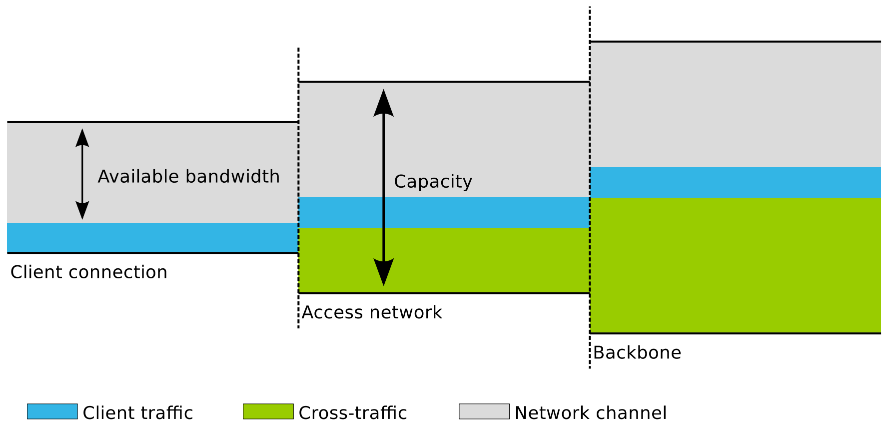


Figure 8.1: Illustration of available bandwidth and capacity of a network channel seen from a single device.

To get a better understanding of both metrics, a more thorough description will be given. The definitions are based on work by Prasad et. al [PDMC03], who describes the parameters capacity and available bandwidth, as well as methods for calculating these.

Capacity: The capacity metric describes the amount of data that can be transmitted on a hop at a specific layer. As shown in Figure 8.1 a path often consist of multiple hops, and the path capacity is found from the hop with the smallest capacity, which is considered to be the rate listed by the operator. The measurements performed are based on what can be transmitted on the application layer, and because of this the overhead added on the layers below must be taken into account when calculating the capacity. The formula for doing this calculation can be seen in Equation 8.1 below.

$$C_{L4} = \frac{C_{L3} \cdot L_{L4}}{H_{L3} + L_{L4}} \tag{8.1}$$

Where C_{L4} is the capacity on the application layer of the TCP/IP model[1], L is packet size and H is the total overhead at a given layer.

The rates provided by the operators includes the transmission of overhead traffic, meaning that when using Equation 8.1, the highest capacity is obtained by maximizing the packet size within the limitations of the cellular network channel.

Available Bandwidth: The available bandwidth describes how much of the capacity a user has access to on a hop or a path. It is found as a function of the capacity C as well as the average utilization $\overline{u}$ for each hop i on a path. The averaged utilization $\overline{u}$ can be found using Equation 8.2, where τ is an averaging timescale and $u(x)$ is the instantaneous available bandwidth.

$$\overline{u}(t-\tau, t) = \frac{1}{\tau}\int_{t-\tau}^{t} u(x)dx \tag{8.2}$$

The function uses the basic behavior of packet transfer, where each packet is transferred using all of the available bandwidth, meaning that transferring with 50% of the available bandwidth is equal to the channel being fully utilized for half of the averaging timescale, and being idle for the remaining half. This also means that the averaging from the time period $(t - \tau, t)$ persists when calculating the available bandwidth, making this a value averaged over time as well. Normally this method would have to be repeated over the entire path, but as the measurements considered in this report are all end to end measurements, the entire path will be considered as a single hop. The function for finding the available bandwidth for the path can be seen in Equation 8.3.

$$avbw = (1 - \overline{u})C_{L4} \tag{8.3}$$

The capacity has been considered previously, meaning that the only variable that should be considered here is the averaged utilization. From figure 8.1 it can be seen that parts of the path will be occupied by cross-traffic, and therefore the remaining capacity of the channel averaged over a period of time is the rate which is described as the available bandwidth.

Without knowledge of the network utilization on a path or a hop, Equation 8.3 will not be directly applicable, as the parameter $\overline{u}$ is required for the equation to be useful. This issue can be solved by instead looking at the remaining capacity, which can be found by maximizing the utilization, and using this for estimating the available bandwidth. By using this approach the equation for available bandwidth can be rewritten, and the result can be seen in Equation 8.4..

$$avbw = \overline{u}_{meas} \cdot C_{L4} \tag{8.4}$$

[1] TCP/IP model layers: Link (1), Internet (2), Transport (3) and Application (4)

The measured available bandwidth using end to end measurements is not necessarily related to the hop with the smallest capacity, but the hop with the smallest capacity acts as a bottleneck for the maximum available bandwidth that can be measured.

8.2 Mobile Communication Technologies

Throughout the work with the NetMap system, measurements have been collected on the cellular network using 2nd and 3rd Generation Technologies. To get a better understanding of the technologies, a short description of EDGE (2.75G), W-CDMA (3G) and HSPA (3.5G) will be given in this section. The information here is based on the technical specifications available through the 3rd Generation Partnership Project (3GGP) [3gg13].

The main parameter to consider here is the Transmission Time Interval (TTI), which is a parameter that refers to the interval or frame in which data is transferred between the MAC and PHY layers. For both UMTS and HSPA the TTI also acts as an interval between updates of the transport block sizes, which allows for dynamic adaptation of the rate during transmission. An example of how this works with a 10 ms TTI can be seen in Figure 8.2.

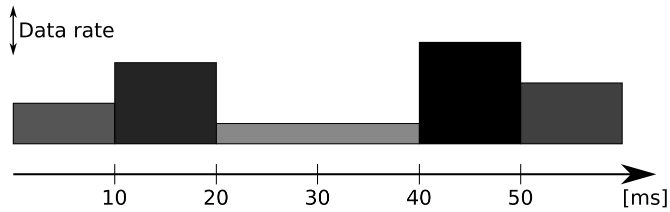


Figure 8.2: Example of a constant transfer where the rate is adjusted due to a Transmission Time Interval (TTI) of 10 ms.

EDGE (2.75G) Enhanced Data rates for GSM Evolution (EDGE) is the slowest of the three technologies considered, and while it is considered to be part of 2G it still complies with the standards defined for 3G by 3GPP, and therefore is often referred to as a 2.75G technology.

- **Modulation:** A combination of Gaussian minimum-shift keying (GMSK) and 8-ary phase-shift keying (8PSK).
- **Multiple Access:** Time and frequency division (TDMA and FDMA), where 8 slots are allocated per channel due to TDMA.
- **Frame Interval:** 4.615 ms.

One thing to note about EDGE is that the data packets, which often are TCP/IP packets, are too long to transmit in a single time slot. This means that more time slots and thereby frame intervals are used for packet transfer, resulting in a pseudo TTI of approximately 20 ms.

W-CDMA (3G) Sometimes referred to as Universal Mobile Telecommunications System (UMTS), Wideband Code Division Multiple Access (W-CDMA) is the basic 3G technology that is used during measurements.

- **Modulation**: Quadrature phase-shift keying (QPSK).

- **Multiple Access:** Code division multiple access (CDMA).
- **TTI:** 10 ms.

HSPA (3.5G) High Speed Packet Access (HSPA) is an improvement to basic 3G technology described above, where higher rates and lower TTI can be achieved.

- **Modulation:** QPSK, 16- and 64 Quadrature Amplitude Modulation (QAM) depending on the transmission direction and release version.
- **TTI:** 2 ms

It should be noted that the difference between each of the cellular technologies is much larger than what is described here, but for the novel bandwidth estimation techniques considered in this part of the report the TTI is expected to be the parameter with the biggest impact.

8.3 Existing Throughput Measurement Methods

Available bandwidth is one of the key network performance metrics for describing network quality. Accurate and timely measurement of bandwidth is challenging due to the dynamic characteristic of network traffic especially when moving into the mobile domain. Therefore, many researchers have developed different techniques to provide better estimation of bandwidth. In this section some different estimation techniques will be presented and discussed with the focus on adaptation into the NetMap system.

Active probing is the most feasible technique for estimating available bandwidth, and many methods using this technique have been developed. Using active measurements rather than passive means inducing measurement traffic into the network when estimating the available bandwidth. One of the main goals for choosing alternative estimating techniques is to lower the amount of measurement traffic created when carrying out measurements.

In [HZZ$^+$12] and [SKK03] the authors present a series of different end-to-end available bandwidth estimation techniques which they divide into two categories: (1) PGM (Probe Gap Model) methods and (2) PRM (Probe Rate Model) methods.

1. **Probe Gap Model (PGM)-based methods:** exploits the information in the time gap between the arrivals of two successive probe packets at the receiver. Which means that a probe pair is sent with some specified time gap $\triangle_{in}$ (time interval between two probe packets) and reaches the receiver with a time gap $\triangle_{out}$. Assuming that a bottleneck is present in the network, such that the queue does not become empty between the departure of the first probe in the pair and the arrival of the second probe, then $\triangle_{out}$ is the time it takes for the bottleneck to transmit the second probe in the pair which can be used to estimate the available bandwidth. An example tool is the Nettimer [KL01] which utilize this method.
2. **Probe Rate Model (PRM)-based methods:** uses the concept of self-induced congestion. If one sends probe traffic at a rate lower than the available bandwidth along the path, then the arrival rate of probe traffic at the receiver will match the rate at the sender. In contrast, if the probe traffic is sent at a rate higher than the available bandwidth, then queues will build up inside the network and the probe traffic will be delayed. As a result, the probes' rate at the receiver will be less than their sending rate. Thus, one can measure the available bandwidth by searching for the turning point at which the probe sending and receiving rates start matching.

Two methods are described in [PDMC03], Self-Loading Periodic Streams (SLoPS) and Trains of Packet Pairs (TOPP), both using the Probe Rate Model. They are both used to estimate the available bandwidth, and both involve monitoring variations in the delay between packets in a flow.

An implementation using a variation of TOPP is the method ASSOLO [GRT09], which utilizes a stream with varying rate called a chirp to estimate the available bandwidth. The method tests a span of rates using two mirrored exponential functions (similar structure to a sigmoid function), which means that the resolution decreases as the function goes towards the limit values. The method can however update the limit values while measuring if needed. Using the default setup for the method ASSOLO often takes less than a second to complete with a traffic intensity of about 300 Kbit/s. It should however be noted by default the method runs multiple times and filters the results to get better estimates.

A method such as ASSOLO can provide good results granted that it is run on a Real-Time Operating System (RTOS). Most of the methods using delay variations have only been tested in simulated networks, where network parameters are clearly defined. This is in heavy contrast to the networks measured by the NetMap application, where all information is collected at the endpoints for the entire path, and not at the individual hops which would be beneficial.

Finally a method called Self-Loading Decreasing Rate Train (SLDRT [HZZ+12]) is considered which also utilize the Probe Rate Model. This method sends a single train of packets with decreasing rate, and uses the measured one-way latency to estimate the available bandwidth. The reason behind this is that initially the packets sent with a high rate will induce a queuing delay that can be observed from the latency, but as the rate decreases this additional delay will disappear and the bandwidth can be found from the rate at which this happens. The single packet train means that the method generally has a low data usage, with the exact amount of data depending on the parameters used for the measurement.

In the following chapters two alternative techniques for estimating available bandwidth are considered. First an adaptive alternative to the Nettimer using the Probe Gap Model will be analyzed in Chapter 9, and then a technique using the SLDRT method which utilizes the Probe Rate Model in Chapter 10, to determine if these methods can be used as potential alternatives to the already implemented method, which measures the bulk transfer capacity using TCP.

Chapter 9

Packet Pair Technique

In this chapter an alternative technique for estimating the bandwidth throughput will be presented. The goal is to minimize the amount of data used when trying to estimating the bandwidth on any given network link. Mobile cellular subscriptions usually have a monthly limit on how much data a user is allowed to generate, the goal is then to implement a lightweight bandwidth estimation method which could be used as an alternative to the already implemented TCP throughput test in the NetMap system.

9.1 The Technique

[KL13] In this section the general principle of the packet pair property of FIFO-queueing networks (for convenience referred to as PPT) is described, and it is shown how it can be used to measure bottleneck link bandwidth. The idea of this principle is that if two packets are sent close enough together in time to cause the packets to queue up at the bottleneck link, then the packets should arrive at their destination with the same time spacing as when they exited the bottleneck link. To better understand this Figure 9.1 illustrates this principle.

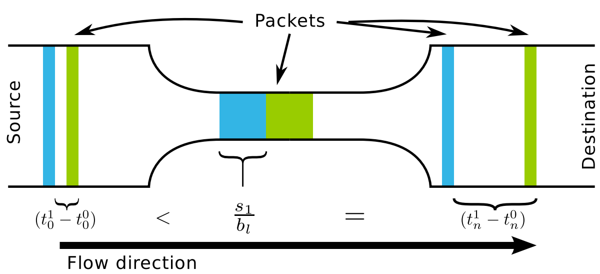


Figure 9.1: Illustration of the packet pair queuing principle, where two packets of the same size are traveling from the source to the destination. The wide part of the pipe represent high capacity links while the narrow illustrates the bottleneck along the link. The increase in space between the packets is caused by the queuing delay introduced at the bottleneck.

The technique predicts the difference in arrival times of two packets of the same size traveling from the same source to the same destination:

$$t_n^1 - t_n^0 = max\left(\frac{s_1}{b_l}, t_0^1 - t_0^0\right) \tag{9.1}$$

Where t_n^0 and t_n^1 are the arrival times of the first and second packet at link n or the destination, while t_0^0 and t_0^1 are transmission times of the first and second packet at the source. s_1 is the size of the second packet, and b_l is the bandwidth of the bottleneck link. The intuitiveness of Equation 9.1 is that if two packets are sent close enough together in time to cause the packets to be queued up at the bottleneck link ($\frac{s_1}{b_l} > t_0^1 - t_0^0$), then the packets will arrive at the destination with the same spacing ($t_n^1 - t_n^0$) as when they left left the bottleneck link ($\frac{s_1}{b_l}$).

For this technique to provide true results, a few assumptions are made. First, it is assumed that the two packets are queued only at the bottleneck link and at no later link on the path. Secondly, it is assumed that no packets will be queued between the two measurement packets as a result of cross-traffic. In the event that any of these assumptions are violated Equation 9.1 will not hold, but by introducing some statistical filtering techniques over multiple packets it should still be possible to use the obtained results.

Using the packet pair property, Equation 9.1 can be rewritten for b_l to obtain the bandwidth of the bottleneck link:

$$b_l = \frac{s_1}{t_n^1 - t_n^0} \tag{9.2}$$

9.2 Implementation

To test whether or not this technique is applicable in the particular scenario a simple implementation has been made, with the purpose of conducting a series of test measurements. The purpose of the system is to transmit a series of UDP packets and time the arrivals which are very time critical. As the technique should be usable in the NetMap system running on an Android device, the client has been written in Java.

The PPT should measure bidirectionally, meaning that both the down- and uplink of the network path should be measured. These two measurements should not be made simultaneously, but scheduled after each other. To comply with the rest of the NetMap system, the client must initiate the connection to the server in order to start the tests.

As mentioned previously in Section 9.1, the PPT is very sensitive and susceptible to changes in the measurement path. To accommodate for this more packets should be transmitted in a sequence, and these sequences should be executed with an arbitrary amount of time delay between each sequence. Each sequence will be noted as a burst sequence $\boldsymbol{S}$, and each burst will consist of $\boldsymbol{L}$ amount of packets. Between each burst a delay of $\boldsymbol{D}$ milliseconds should be present, and each packet is set to have size of $\boldsymbol{P}$ bytes.

Both the server and client implementation is located in *code/PPT* . To better illustrate how the measurements are being conducted Figure 9.2 shows a message sequence diagram of the system.

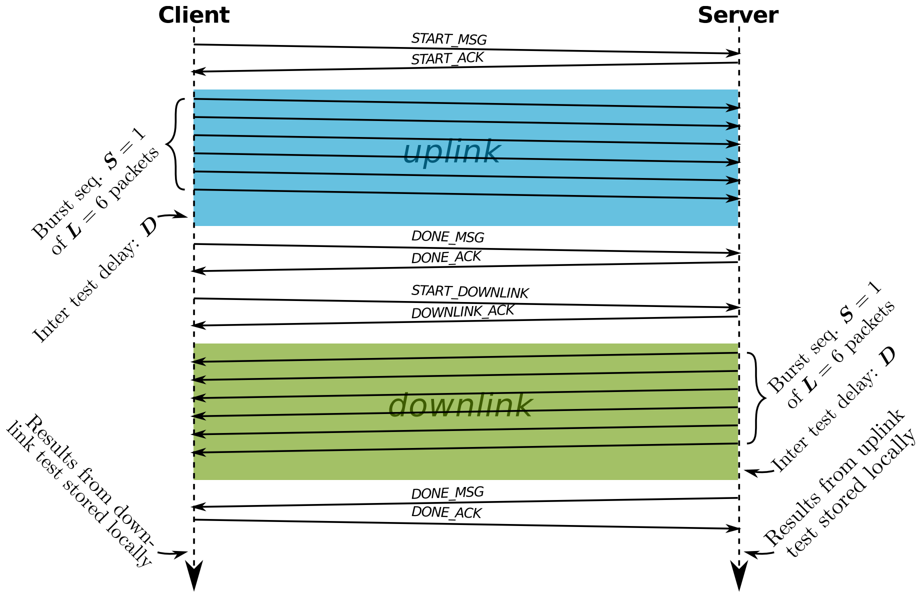


Figure 9.2: Message sequence chart for the Packet Pair Technique. First a uplink throughput test is made, which is marked by the blue area, and right after that a downlink throughput test is made, which is marked by the green area. Inside the colored areas each measurement consists of a number of burst sequences, specified by the variable S. The sequences will run in succession, with each burst transmitting L number of packets. Between each burst a inter-test delay D is present, and after each test has finished the generated restults are stored locally. Note: The figure only shows 1 sequence each for the up- and downlink test. More sequences can occur, but for the sake of simplicity the figure only shows one each.

The current implementation is created with simplicity in mind, which is why the measurement results are stored locally on both sides, but should later be moved to the client in order to conform with the rest of the NetMap system, and so that they can be displayed to the end users of the system. Furthermore, the system should handle multiple requests by assigning a unique thread to each request, to avoid queuing and delay in a single thread at the server. As with all the existing methods and techniques, these measurements are all conducted and captured on the application layer.

9.3 Local Network Test Setup

To see how the PPT works and how different parameter values affect the results, a controlled test scenario is used. The PPT is tested locally meaning that the client application is executed from a computer connected locally to the server's network, such that the throughput between the server and the client is limited by the local network. Figure 9.3 illustrates the controlled network setup.

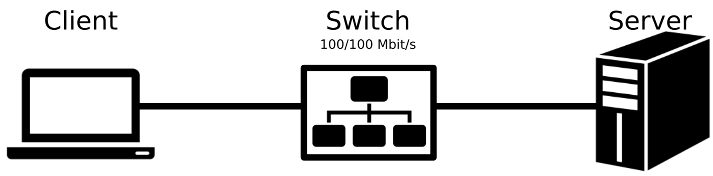


Figure 9.3: Illustration of the local test setup, where the client is connected directly to the servers local network instead of having to connect over the Internet.

This controlled network setup is, as seen in the figure, limited by the capability of the switch, which has a throughput capacity of 100/100 Mbit/s duplex. The purpose of the PPT is then to measure the available bandwidth at the bottleneck (the switch), which should give an indication of the throughput speeds a client should be able to achieve.

The following tuning parameters allows for configuration of the PPT, which is expected to have an effect on the obtained results:

- $\boldsymbol{P}$ = Packet size (in bytes).
- $\boldsymbol{S}$ = Number of burst sequences.
- $\boldsymbol{L}$ = Number of packets per burst sequence.
- $\boldsymbol{D}$ = Delay (in milliseconds) between each burst sequence.

To simplify the tests some of the parameters will be set to static values. The number of bursts and delay between each burst will be the same throughout the rest of this section. The number of bursts has been set to $\boldsymbol{S} = 5$ and the delay set to $\boldsymbol{D} = 5000ms$, which leaves the packet size $\boldsymbol{P}$ and number of packet per burst $\boldsymbol{L}$. Four different packet sizes will be tested: $\boldsymbol{P} = \{500,\ 1470,\ 5 \cdot 1470,\ 10 \cdot 1470\}Bytes$ and each of these will be tested with burst lengths of $\boldsymbol{L} = \{10,\ 20,\ 50\}$ which adds up to a total of 12 different parameter setups. A data packet size of 1470 Bytes is chosen because when also adding the UDP header to the size it is still possible to fit it into a single Ethernet frame, and hence by multiplying it with 5 and 10 the network stack is forced to fragment the packets in 5 or 10 and therefore transmit more Ethernet frames over the network's backbone.

9.4 Local Packet Pair Test Results

With the 12 different parameter setups specified, the goal is to see how well the PPT estimates the available bandwidth while trying to minimize the load it induces on the network. The currently implemented method for estimating throughput (see Section 3.5, page 18) will on a 100 Mbit/s link produces approximately **180** megabytes of traffic in its 15 second runtime (not considering TCP slow start). The amount of traffic transmitted depends on the available bandwidth at the bottleneck of the network path.

Packet size $\boldsymbol{P} = 500$: First the setup with the smallest packets, and how well they perform when estimating the bandwidth is evaluated.

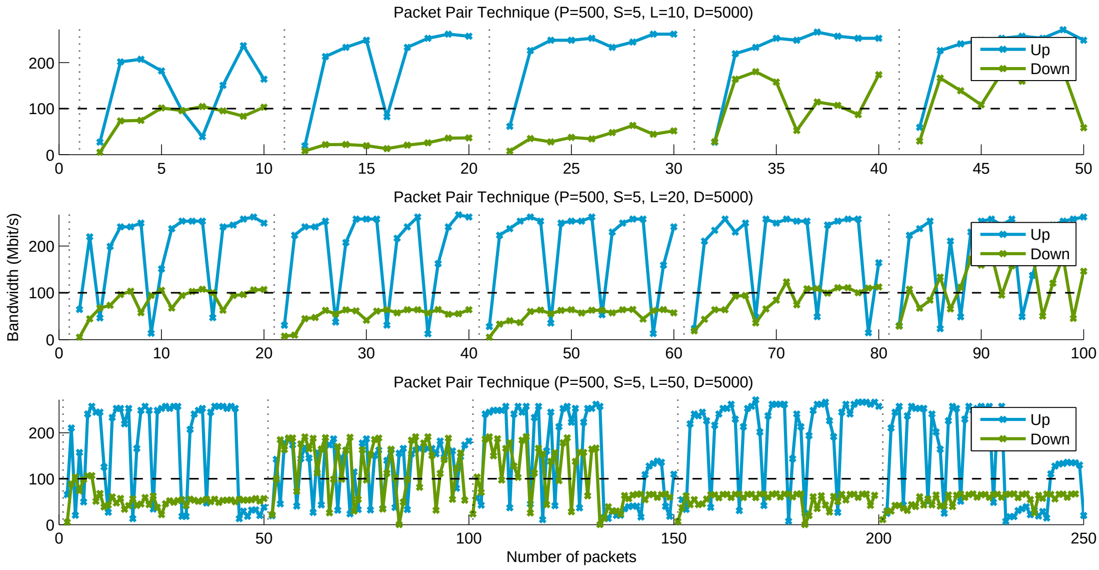


Figure 9.4: The Packet Pair Technique shows the estimated bandwidth obtained through burst sequences of different length. The black dotted line marks the target bandwidth estimation. On the x-axis the cumulated amount of packets transmitted in the 5 burst sequence is seen, where each vertical dotted line marks the start of a new burst sequence.

The first thing to notice on Figure 9.4 is the evident difference between the uplink and downlink estimations, as the uplink estimation is generally higher throughout all the tests. In the uplink test the client transmits UDP packets as fast as possible to the server, which marks the time of arrival for use in the estimation of the available bandwidth. It can be seen that the system has difficulties in trying to estimate the targeted bandwidth, which might be a result of the small packets being transmitted too fast, the inability of the systems to mark the arrival in a high enough time resolution, or because the packets never get queued at the bottleneck because they are processed too fast.

The tests give an indication of the impact that different burst lengths have on the estimation of the bandwidth. It can be seen that the bandwidth estimation quickly seems to settle to something static, but with some random extreme case values occurring randomly. There is therefore no need for long burst trains, and bursts consisting of 30 packets or less seems reasonable. The test generates between 0.02 and 0.11 megabytes of traffic for the 5 burst sequences if no packets are dropped.

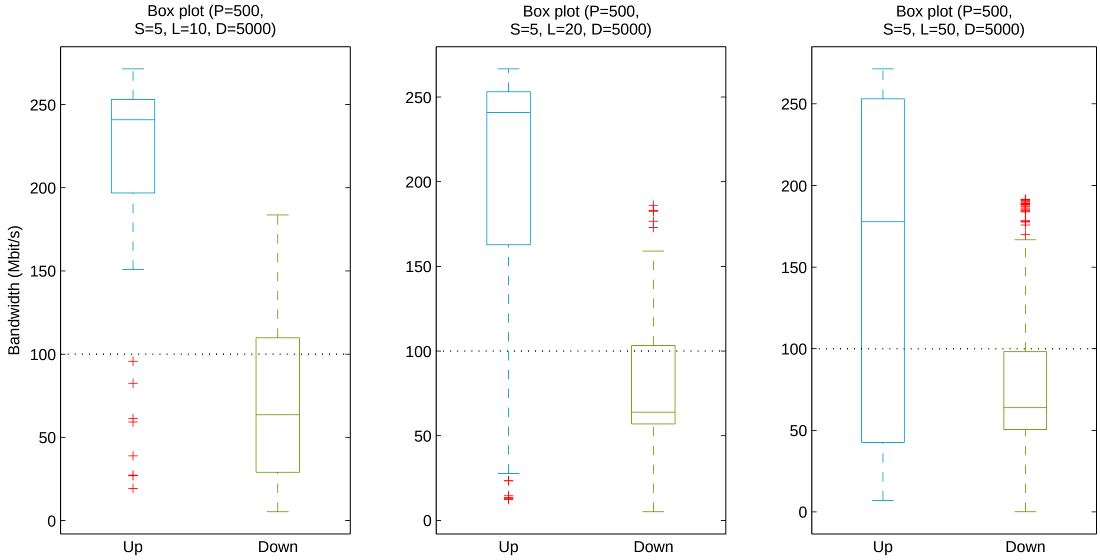


Figure 9.5: Boxplot of the results from Figure 9.4. The edges of the boxes are the 25th and 75th percentiles, the vertical line in the box shows the median and the black dotted line shows the true bandwidth. The whiskers[1] extend to the most extreme data points that are not considered outliers, and the outliers are plotted individually.

Figure 9.5 shows boxplots of the 5 bursts with the different packet lengths L accumulated into three plots. In the three plots it can be seen, as mentioned before, that the uplink estimations all seem to overshoot and that by adding more packets, and hence more data points, the extreme case values simply fall into the 25th-75th percentile area, as the box is expanded.

Packet size $P = 1470$: On Figure 9.6 and 9.7 the same tests are performed but this time the packet size has been increased to 1470 Bytes, utilizing almost the maximum payload of a Ethernet frame. By keeping the size of the UDP packets under one Ethernet frame no reassembling is needed by the network stack.

On Figure 9.6 it can be seen that by increasing the packet size better estimations can be obtained, especially in the downlink test. The uplink test still shows some large fluctuations in its estimations, and both the up- and downlink tests are still very sensitive. On average the packets were close to the estimation, especially the downlink tests which can be seen on Figure 9.7. It is worth mentioning that the median seems to provide an accurate estimation of the bandwidth when compared to the mean or average value, as it can be seen in the following table:

[1] The default is a w of 1.5. Points are drawn as outliers if they are larger than $q_3 + w(q_3 - q_1)$ or smaller than $q_1 - w(q_3 - q_1)$, where q_1 and q_3 are the 25th and 75th percentiles, respectively.

Burst length $L = 10$:	Uplink average:	238.53 Mbit/s	Downlink average:	98.95 Mbit/s
Burst length $L = 10$:	Uplink median:	97.64 Mbit/s	Downlink median:	87.87 Mbit/s
Burst length $L = 20$:	Uplink average:	210.97 Mbit/s	Downlink average:	114.99 Mbit/s
Burst length $L = 20$:	Uplink median:	93.27 Mbit/s	Downlink median:	74.86 Mbit/s
Burst length $L = 50$:	Uplink average:	146.00 Mbit/s	Downlink average:	102.99 Mbit/s
Burst length $L = 50$:	Uplink median:	94.48 Mbit/s	Downlink median:	100.99 Mbit/s

As it can be seen in the table, the median is much closer to the true bandwidth and could therefore be used as a simple filter. The tests used between 0.07 and 0.35 megabytes for the 5 burst sequences if no packets were dropped.

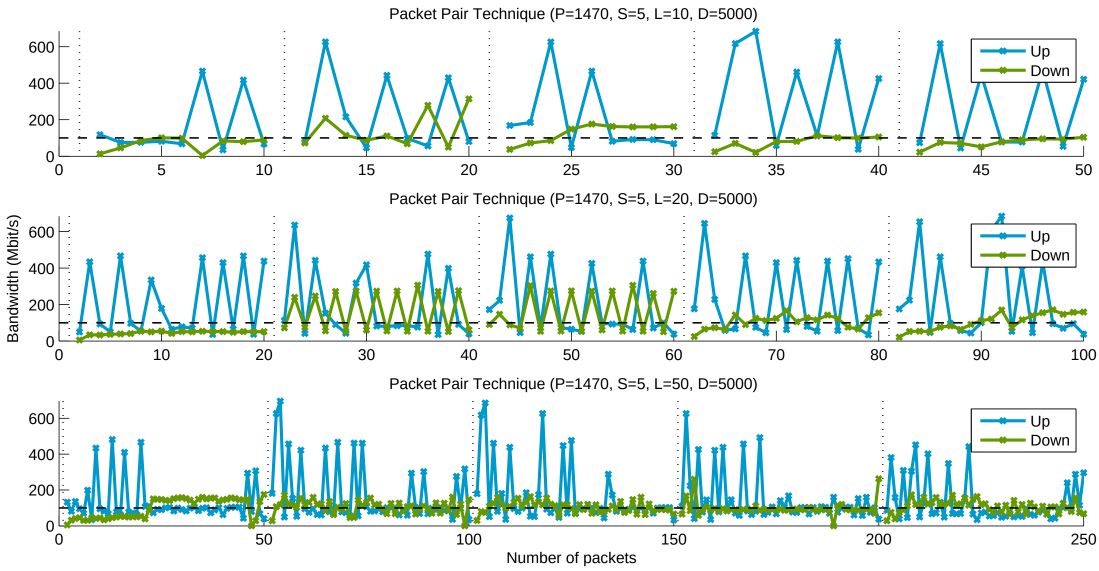


Figure 9.6: Packet Pair Technique showing the estimated bandwidth obtained through burst sequences of different length. The black dotted line marks the true bandwidth. On the x-axis the cumulated amount of packets transmitted in the 5 burst sequences is seen, where each vertical dotted line marks the start of a new burst sequence.

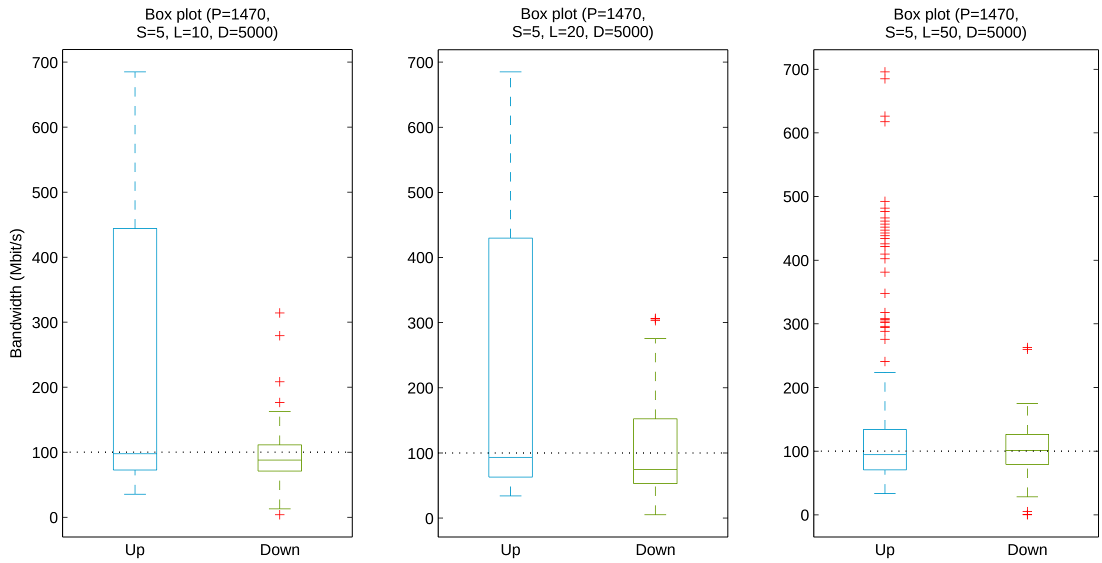


Figure 9.7: Boxplot of the results from Figure 9.6. The edges of the boxes are the 25th and 75th percentiles, the vertical line in the box shows the median and the black dotted line shows the true bandwidth. The whiskers extend to the most extreme data points that are not considered outliers, and the outliers are plotted individually.

Packet size $P = 5 \cdot 1470$: In this test the packet size has been increased by a factor of 5, which essentially means that the network will have to fragment the UDP packets into at least 5 Ethernet frames. As the timing of the packet arrival happens on the application layer, all the fragmented UDP packets must arrive at the target before being pushed to the application layer. By doing this the time it takes for transmitting one packet is increased, removing some of the critical timing issues which might cause problems in time stamping the arrival of packets. Using fragmented packets does however induce a higher probability of packet loss as all the UDP fragments must arrive at the target to complete a packet.

For these tests it can be seen in Figure 9.8 that both the up- and downlink estimations are more accurate compared to the previous two tests. There is some packet loss present in the test, especially when looking at the bottom plot in the figure, where the downlink test shows that the last two burst sequences are almost incomplete. This is a direct result of the slow processing of received packets on the client, where the buffer is filled and eventually drops packets. The server is much faster at reading the packets and generally drops less packets than the client. Below is a list of packet loss in the different tests:

Burst length $L = 10$:	Uplink packet loss:	0%	Downlink packet loss:	6%
Burst length $L = 20$:	Uplink packet loss:	0%	Downlink packet loss:	9%
Burst length $L = 50$:	Uplink packet loss:	0%	Downlink packet loss:	40%

These tests use between 0.35 and 1.75 megabytes for the 5 burst sequences if no packets are dropped. On Figure 9.9 it once more shows that the median provides the most accurate estimation of the bandwidth.

Note: Because of the cumulative display of packets along the x-axis and the packet loss, the 5 sequence marks will not always align properly. Each sequence always contains some of the L packets transmitted, where after an arbitrary amount the packets begins to be dropped. An example of this can be seen in the top plot in Figure 9.8 where the 1 packet of the first burst sequence is lost such that the blue and green lines to follow the same x-ticks.

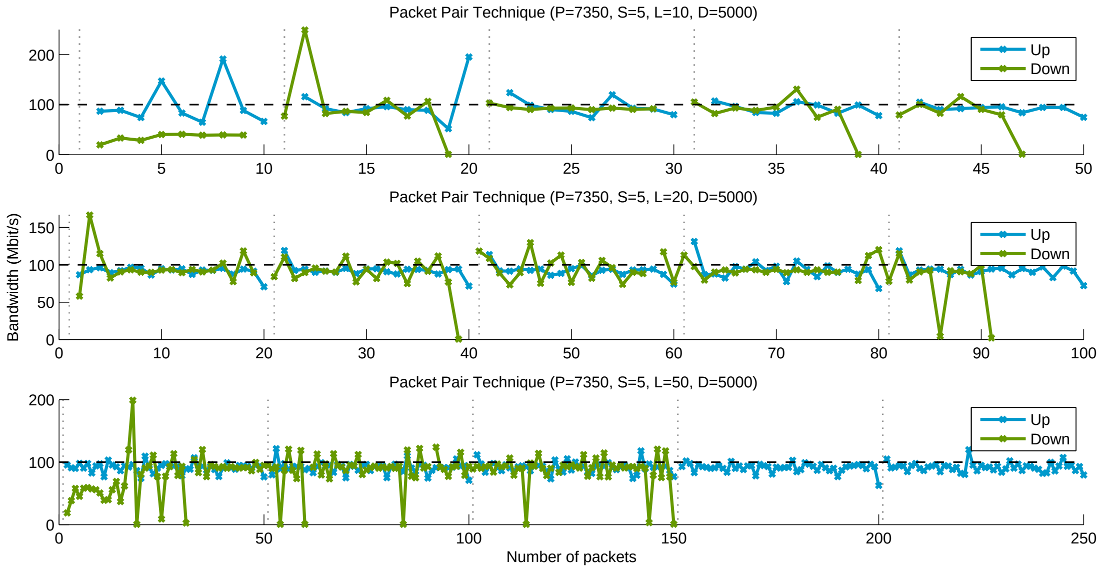


Figure 9.8: Packet Pair Technique showing the estimated bandwidth obtained through burst sequences of different length. The black dotted line marks the true bandwidth. On the x-axis the cumulated amount of packets transmitted in the 5 burst sequences is seen, where each vertical dotted line marks the start of a new burst sequence.

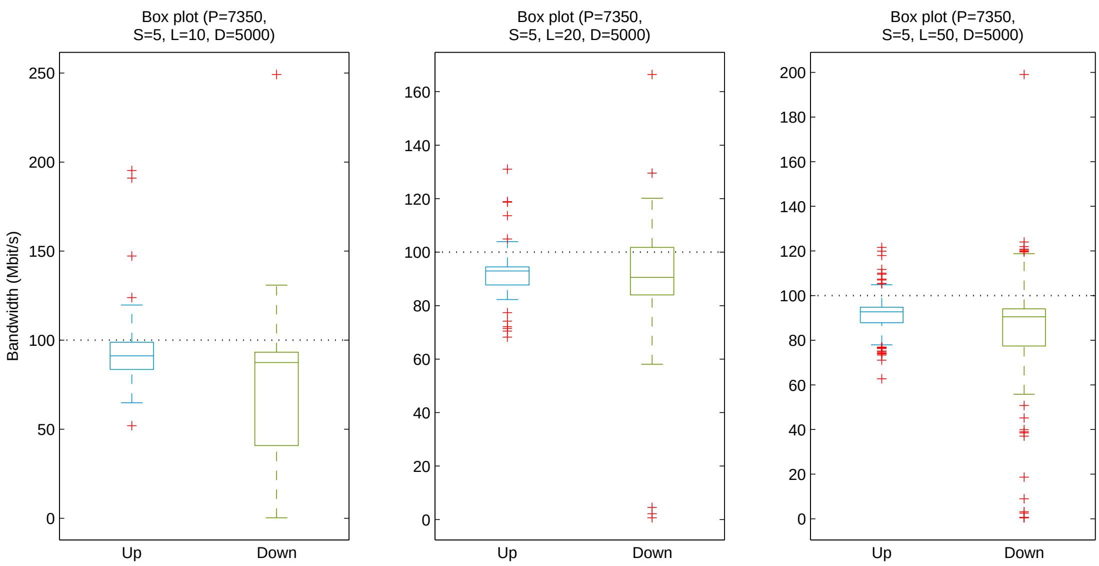


Figure 9.9: Boxplot of the results from Figure 9.8. The edges of the boxes are the 25th and 75th percentiles, the vertical line in the box shows the median and the black dotted line shows the true bandwidth. The whiskers extend to the most extreme data points that are not considered outliers, and the outliers are plotted individually.

Packet size $P = 10 \cdot 1470$: In this last test the packet size has been increased with another factor of 5, adding up to a total requirement of at least 10 fragmented UDP packets to complete one transmission. As mentioned in the previous section, increasing the packet size will increase the probability of packet loss. This is expressed on Figure 9.10 where the downlink test has high packet loss, especially in the two bottom plots where packet loss is already detected in the first burst sequence.

Burst length $L = 10$:	Uplink packet loss:	0%	Downlink packet loss:	16%
Burst length $L = 20$:	Uplink packet loss:	0%	Downlink packet loss:	47%
Burst length $L = 50$:	Uplink packet loss:	0%	Downlink packet loss:	72.8%

In the above table it can be seen that increasing the length of the bursts only introduce higher packet loss. In Figure 9.11 it is seen that the distribution of bandwidth estimations is closer to the true bandwidth, and that using the median gives the result closest to the true bandwidth. The tests uses between 0.7 and 3.5 megabytes for the 5 burst sequences if no packets are dropped.

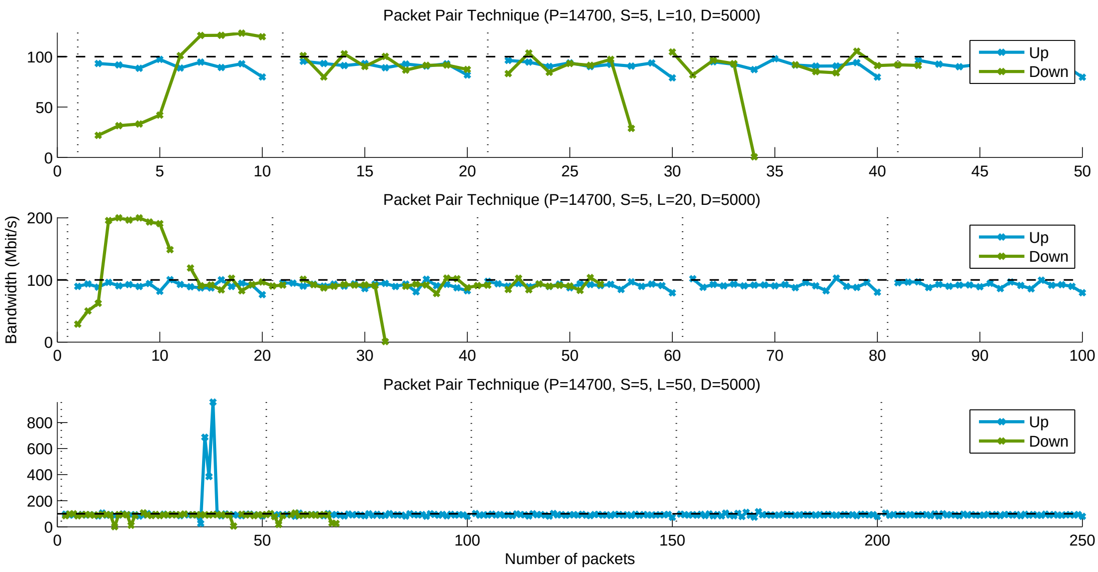


Figure 9.10: Packet Pair Technique showing the estimated bandwidth obtained through burst sequences of different length. The black dotted line marks the true bandwidth. On the x-axis the cumulated amount of packets transmitted in the 5 burst sequences is seen, where each vertical dotted line marks the start of a new burst sequence.

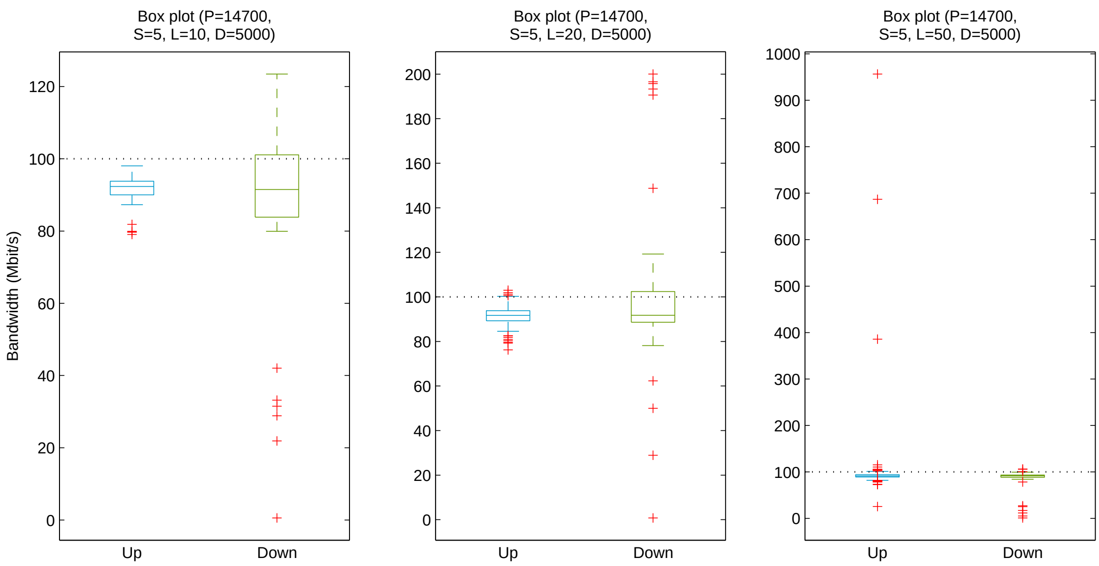


Figure 9.11: Boxplot of the results from Figure 9.10. The edges of the boxes are the 25th and 75th percentiles, the vertical line in the box shows the median and the black dotted line shows the true bandwidth. The whiskers extend to the most extreme data points that are not considered outliers, and the outliers are plotted individually.

9.4.1 Summary

From the last four paragraphs it is seen that the Packet Pair Technique (PPT) can be used to estimate the available bandwidth on a network path. Because of its many tuning parameters: *Packet size, number of burst sequences, length of burst sequence and delay between each burst*, the PPT is very flexible when optimizing its estimation of the bandwidth. From the previous tests the following parameter values provide the best estimations of the true bandwidth:

- Packet size $\boldsymbol{P}$ should be at least 1470 bytes and not more than 5 times bigger.
- Number of burst sequences $\boldsymbol{S}$ depends on how many sequences that should be used when trying to estimate the bandwidth. More burst sequences gives more data to estimate from.
- Packets per burst sequence $\boldsymbol{L}$ should be between 10 and 30, as some of the tests showed that no more is needed for the burst sequence to settle on an estimate, and if the estimate is off target it seems better to do a new burst after the inter-test delay.
- Inter-test delay $\boldsymbol{D}$ has been set to 5 seconds between each burst during testing, which seemed fine. Lowering the time between each test could mean that when a new burst sequence is initiated packets from the previous one will still be present in the network and its buffers. Increasing the inter-test delay would further insure that no packets are present.
- As a filter technique the median showed good properties as it separates the higher half of the samples from the lower half, reducing the importance of outliers, e.g. because they may be measurement errors, and provides the result which occur more often in the data set.

Lastly, the PPT allows for a very significant reduction in the load put on the network, and the amount of bytes used to do the bandwidth estimations. The already implemented bandwidth estimation can generate 180 megabytes during a 15 second (downlink) run, while a test using the PPT with the following parameters $\boldsymbol{P} = 5 \cdot 1470$, $\boldsymbol{S} = 3$, $\boldsymbol{L} = 30$ and $\boldsymbol{D} = 5000$ and run for approximately 15 seconds generates around 0.63 megabytes of data, which is a 99.6% reduction of data usage (in this particular scenario).

It should be noted that there is a radical difference between a high-speed LAN setup and cellular networks on which the measurements should be performed, and as a result the tuning parameters listed previously might not be the optimal values for estimating the available bandwidth. The values will however still be used as a starting point in the next section.

9.5 Cellular Network Test Setup

In this section the PPT is deployed onto the mobile domain, where the goal is to see how it performs in a more dynamic and realistic network environment. In this particular setup the targeted bandwidth is never truly known, as seen in the many tests and measurements performed previously in this project. In this setup the *Sony Xperia U* (See Table 6.1, page 60) device was used with a subscription from TDC. The subscription is limited to a maximum throughput of **5 Mbit/s downlink** and **1 Mbit/s uplink**, which should then be close to the bandwidth that the PPT is supposed to estimate.

Figure 9.12 illustrates the test setup and highlights where the bottleneck may be present. It is assumed that somewhere on the path between the Client and through the Access Network is where the bottleneck is present and lowers the link capacity for this particular measurement device. The bottleneck is essentially set by the technology used to transmit data between the device and cell tower, these technologies were presented in Section 8.2, page 87 and depending on what digital modulation used on the technologies different data rates can be achieved. However, these data rates can only be achieved if no bandwidth constraints are set by the ISPs, such as the one in our particular tests.

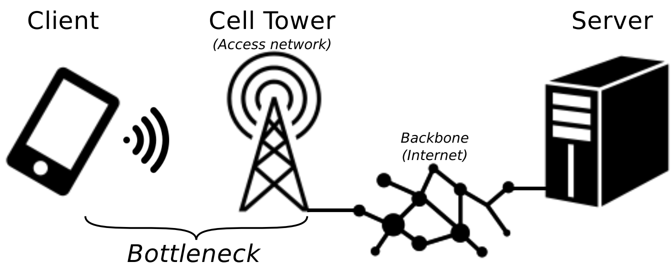


Figure 9.12: Illustration of the mobile test setup where the client application is deployed on a mobile Android device, which then connects to the measurement server via the cellular network. The measurements are performed on the application layer and the test setup does not take retransmission and modulation techniques performed on the lower layers into consideration.

In the previous section a set of parameter values were found from measurements performed on a local network, and the following parameters are based on these observations:

- $\boldsymbol{P = 3 \cdot 1470}$ $[B]$
- $\boldsymbol{S = 5}$
- $\boldsymbol{L = 30}$
- $\boldsymbol{D = 2000}$ $[ms]$

The inter-test delay ($\boldsymbol{D}$) has been changed to 2 seconds to match the 15 second TCP throughput test runtime, such that a 2 second delay is present between each burst sequence. For 5 bursts this adds up to 10 seconds plus additional network delay, which only rarely should exceed the 15 seconds that the TCP throughput test uses.

The PPT is deployed onto the Android OS which means that the device will gather the measurements, where in the previous setup, Section 9.3, the PPT client were running from a computer. One of the goals, other than to estimate the currently available bandwidth, is to verify a final deployment of the software for possible use in a final NetMap implementation. This means that the software should be able to perform and estimate the bandwidth precisely. To make this verification the phone will execute a series of PPT measurement, and following each PPT run, a TCP upload and download test will be scheduled respectively, resulting in a schedule that looks as follows. Between each tests a 5 second delay is present, like in the NetMap implementation:

1. PPT uplink estimation test
2. PPT downlink estimation test
3. TCP uplink throughput test
4. TCP downlink throughput test

Because of the poor insight into the network that the measurements are conducted on, the actual available bandwidth is never truly known, but for the sake of simplicity it will be assumed that the TCP throughput method provides the most precise estimate of the bandwidth, why it will be used as reference.

9.5.1 Sources of Error:

Because of the real life deployment and test scenario, a series of errors might be present which could impact the final conclusion, which is based on the measurements performed. Some of these sources of error are:

Unforeseen changes in the cellular network:
Because of the current implementation of the software, the scheduling order of the measurements can not be changed, which means that when comparing the PPT and TCP uplink or PPT and TCP downlink measurements, it can not be assured that the cellular network is performing equally and providing the same quality during the different tests.

Impact of the Android OS:
As mentioned previously in the project, the amount of different hardware platforms that support the Android OS will have some impact on the performance of the measurements. This phenomenon was elaborated on in greater details in Chapter 6, page 59, but for now all the measurements will be conducted on the same type of hardware device.

9.6 Cellular Network Packet Pair Technique (PPT) Test Results

With the parameters for the PPT specified, a series of measurements will be made to see how well the PPT estimates the available bandwidth, when compared to the TCP throughput method. From the previous internal tests it was seen that the median was a good method of displaying the estimated bandwidth.

In the previous tests the estimations were calculated using Equation 9.1, where all the burst sequences were combined and then the median was calculated. Because of the high sensitivity in the cellular network another estimation method could be used, where each burst sequence gives an average which then can be used to estimate the available bandwidth. Figure 9.13 illustrates the difference between the methods just described.

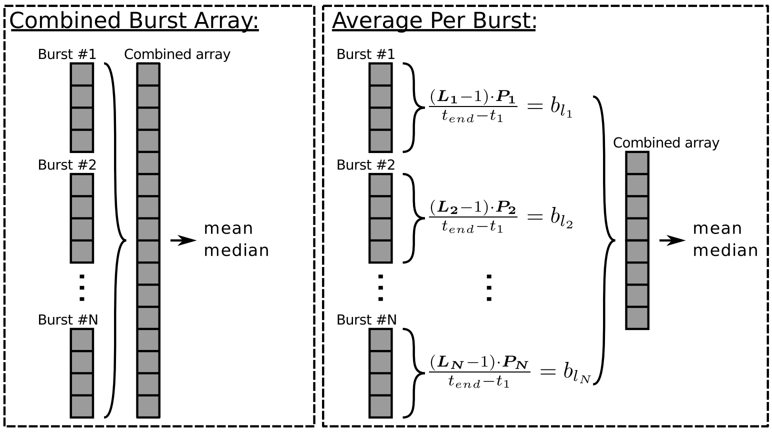


Figure 9.13: Illustration of the different methods showing how the measured data can be used for estimating the available bandwidth.

The first estimation method *Combined Burst Array (CBA)* is very sensitive to extreme values in the measurements, where the other method *Average Per Burst (APB)* averages over a burst sequence to reduce the sensitivity. Since the APB method averages over each burst this does however mean that it gives a set of only 5 measurements under the current PPT parameters, where the CBA method provides a more populated set to estimate from.

Bandwidth Estimation using the *CBA* or *APB* method: To get a better understanding of how the two different methods estimate the available bandwidth, a small illustration based on a simulation will be made. The simulation will try to perform an estimation from one single burst where the burst length $\boldsymbol{L}$ is increased, in order to determine the difference between the methods. Below are the equations used to calculate the estimations:

$$CBA = \frac{1}{N}\sum_{i=1}^{N}\frac{P}{\triangle t_i} \qquad\qquad APB = \frac{P}{\frac{1}{N}\sum_{i=1}^{N}\triangle t_i} \tag{9.3}$$

The equations are the same as presented in Figure 9.13. The main difference between the equations is how packet size influences the result. In the CBA the packet size is divided with each inter-arrival time, which essentially makes the bandwidth estimation more sensitive. The APB is less sensitive since it sums up all the inter-arrival times before the division with the packet size.

To determine which distribution to use for the inter-arrival simulations, some empirical data will be used to see which method fits best. On the left plot in Figure 9.14 the histogram from data gathered in Section 9.7 is seen. For simplicity an exponential distribution is fitted to the data to generate a mean which will be used to simulate inter-arrival times from, which can be see on the right side of the figure.

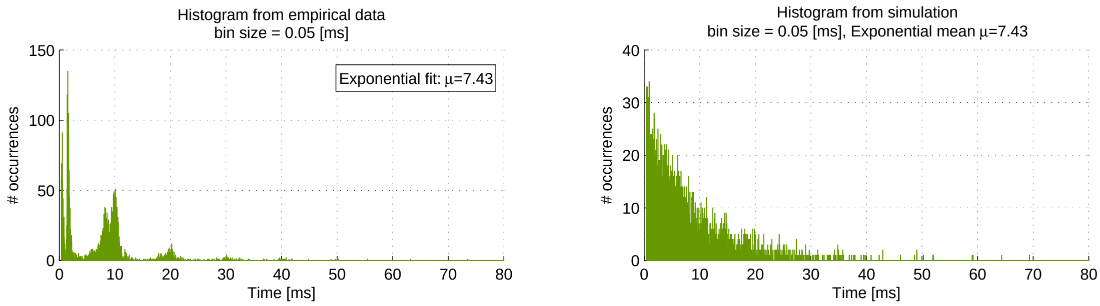


Figure 9.14: On the left side a histogram of empirical inter-arrival times is seen. The data is gathered using the PPT system and is used later in Section 9.7, page 110. On the right a histogram of a exponential distribution using a mean estimated from the empirical data.

As it can be seen in Figure 9.15 the exponential distribution is a poor fit to the empirical data, but for simplicity it is kept as the purpose is to see how the equations in 9.3 can be used to estimate the bandwidth. This can be seen in Figure 9.15 where a series of exponential random inter-arrival times are drawn based on the sample size (x-axis) to estimate the bandwidth. On the left side of the figure, showing the bandwidth estimation, it can be seen that the CBA method is more sensitive and that extreme values have a rather large impact on the estimation of the bandwidth. Because of the exponential distribution many of the inter-arrival times are low, which gives a high bandwidth estimation.

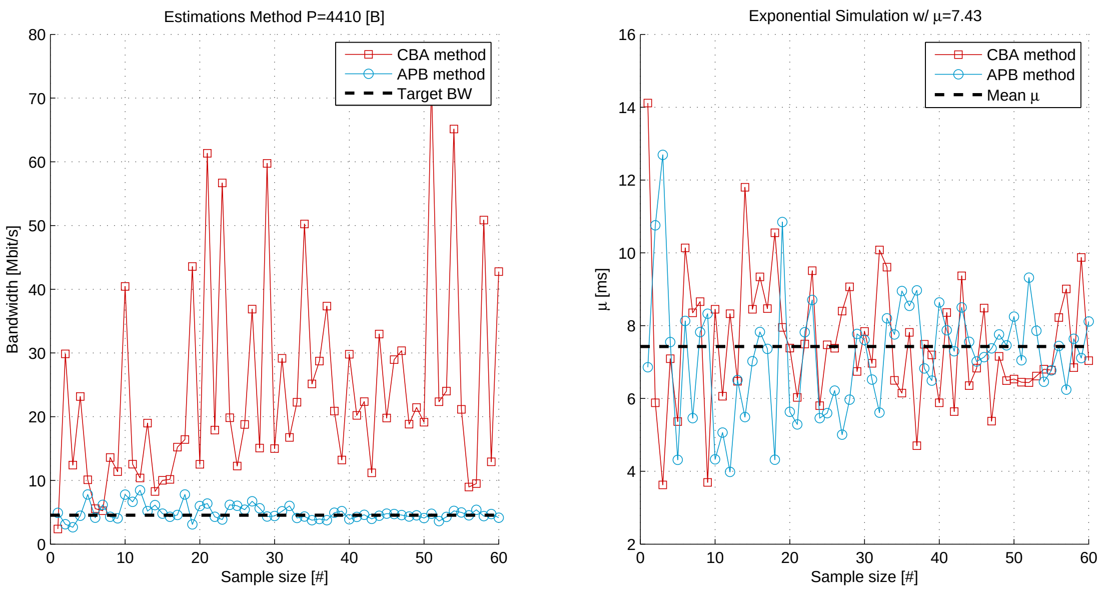


Figure 9.15: Bandwidth estimation using Equations 9.3 and an exponential distribution to draw simulation samples from. On the left is the bandwidth estimation, where on the right the simulation samples can be seen estimating the mean μ using the Equations 9.3, but without using the packet size P.

The APB method sums all the inter-arrival times before the dividing with packet size, which removes the extreme estimations that are seen using the CBA method. The APB estimations are much closer to the actual bandwidth $\frac{P}{\mu} = 593.68[B/ms] \approx 4.53[Mbit/s]$ which is marked by the black line. From the figure the APB method provides the closest bandwidth estimations. On the right side of the figure the same equations are used to calculate the mean μ from the exponentially distributed inter-arrivals times but omitting the packet size P. By omitting the packet size from the Equations 9.3 they both are a simple sample mean which illustrates the importance of how to incorporate the bandwidth estimations, since both methods are valid estimation techniques as illustrated in the right plot of Figure 9.15

In the following subsections the 2 described estimation methods will be applied to measurements performed on a cellular connection with the parameters described in the start of Section 9.5, and evaluated by comparing the results with the TCP throughput test results.

9.6.1 Combined Burst Array (CBA) Method:

On Figure 9.16 the estimation from 15 measurement runs is seen. The distribution of measurements can be seen for the CBA with each sample plotted as a boxplot. By using this method it can be seen that each sample has some extreme values which will skew the distribution (marked by red crosses). From the boxplot it can be seen that most of the TCP throughput measurements are in, or at least in close proximity to, the 25-75% of the PPT measurement distribution, and that the median often gives a close approximation to the TCP measurements.

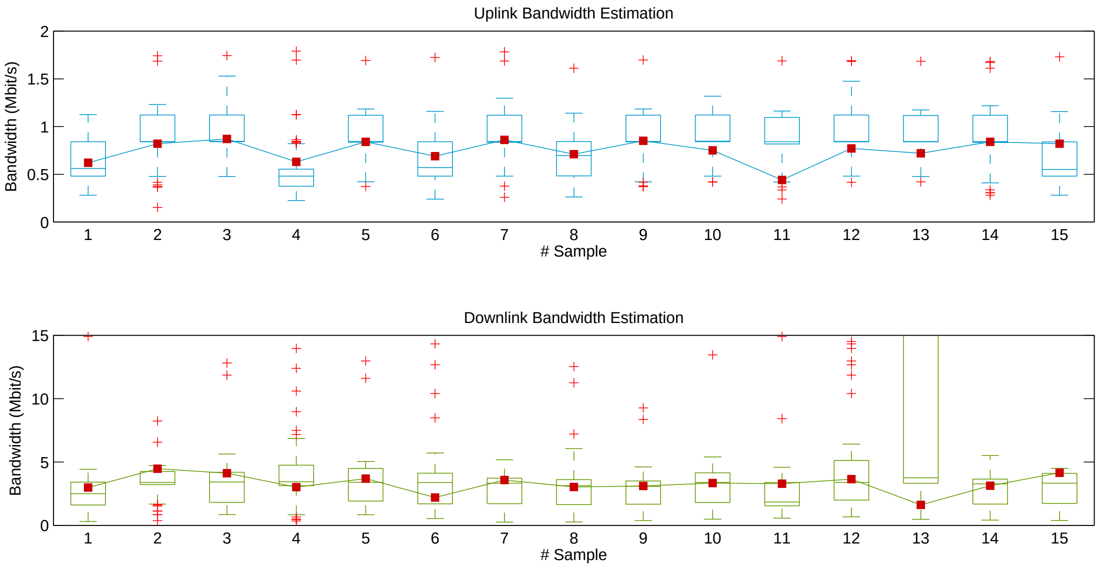


Figure 9.16: Boxplot of the CBA method of 15 measurement runs where PPT, TCP upload and TCP download is scheduled in this order. The blue (downlink) and green (uplink) line with red square marks the TCP throughput test results.

Figure 9.17 compares the final estimation results obtained from the gathered measurements. The uplink and downlink measurements from both the TCP throughput and PPT mean and median estimations are shown, and from the figure it can be seen that both the TCP tests and PPT median estimation give somewhat the same bandwidth results, where both measurements lie close to each other in both the uplink and downlink tests.

From the figure it can also be seen that sometimes the PPT estimations do not follow the fluctuations in the TCP measurements, which can be explained by the fact that when ever either the uplink or downlink tests from the TCP or the PPT is scheduled there is at least a 15 second difference between them, which might lead to sudden changes in the network.

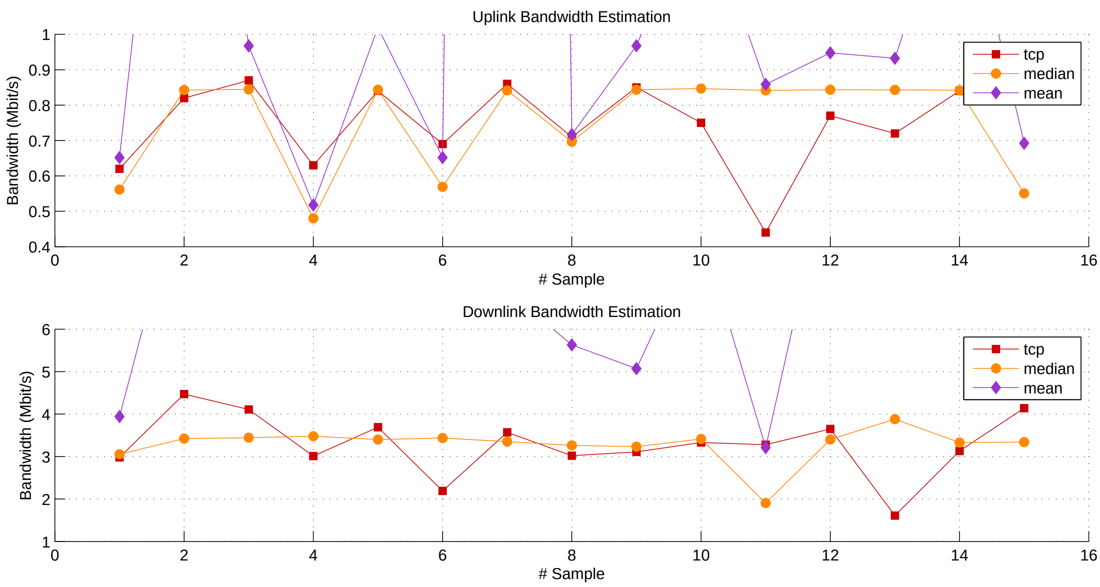


Figure 9.17: CBA method where the TCP throughput results are plotted along with the mean and median of the PPT measurement results. **Note:** Most of the mean marks are left outside because of the extreme values which cause the high estimations.

9.6.2 Average Per Burst (APB) Method:

In this subsection the same data is evaluated, but now instead of combining all the gathered data into a big array, each burst sequence is now evaluated and the outcome used to estimate the available bandwidth. Figure 9.19 shows the boxplot and distribution of the data, and the first thing to notice is the more dense estimations and that next to no extreme values are present.

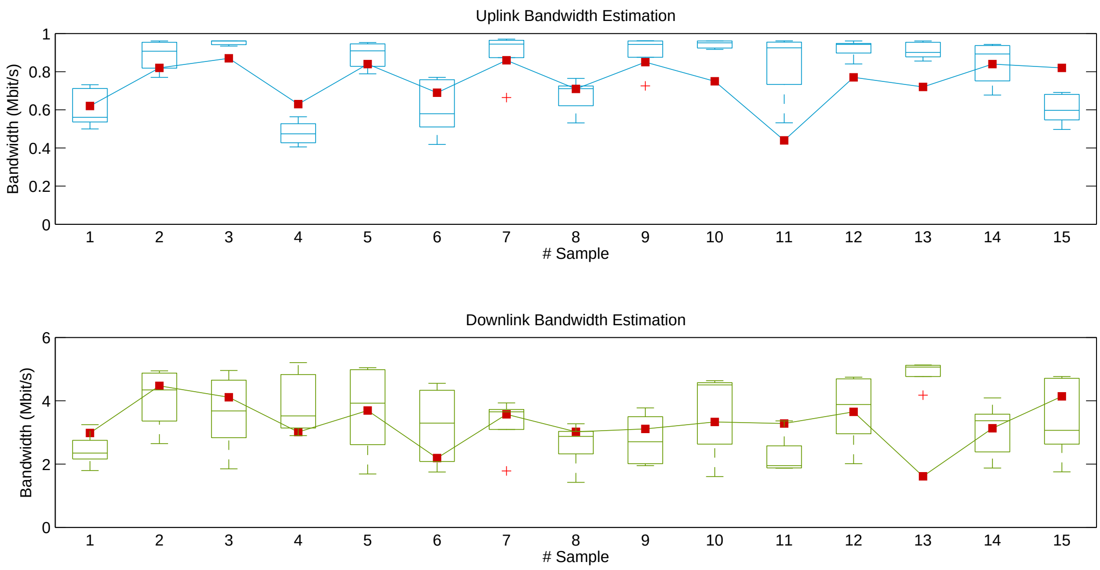


Figure 9.18: Boxplot of 15 measurement runs displayed using the APB method. The PPT, TCP upload and TCP download are scheduled in the order just written, and the blue (downlink) and green (uplink) lines with red squares mark the TCP throughput test results.

When comparing the TCP measurements and the PPT estimations, it seems as if the APB method provides more precise estimations of the actual performance of the network, since most of the results are closer to the results of the TCP measurements and the PPT estimations follow the TCP fluctuations better.

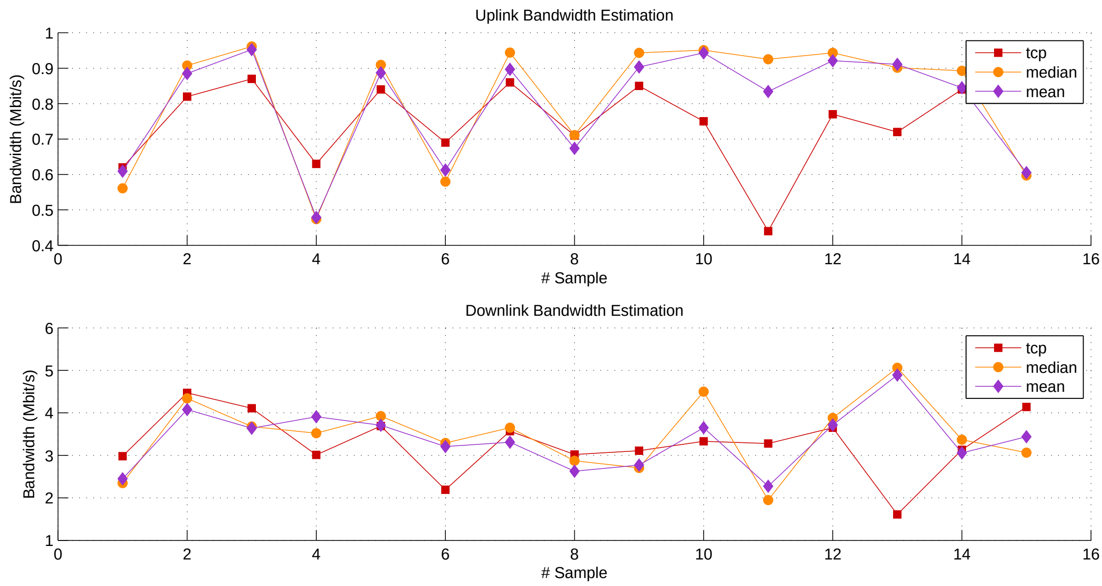


Figure 9.19: APB method where the TCP throughput results are plotted along with the mean and median of the PPT measurement results.

9.6.3 Differences between methods:

To emphasize the differences between the two methods Figures 9.20 and 9.21 show a series of extra runs. A total of 60 measurements have been made and the results are plotted with the different methods. Figure 9.20 shows the results when applying the CBA method and Figure 9.21 when applying the APB method.

In Figure 9.20 the first thing to notice is the high sensitivity to extreme values, as the means of the estimations are located high above the actual limit of the subscriptions. This is the case both for uplink and downlink as most of the mean values lie higher than what is technically possible, as the mobile subscription is limited to 5 Mbit/s downlink and 1 Mbit/s uplink.

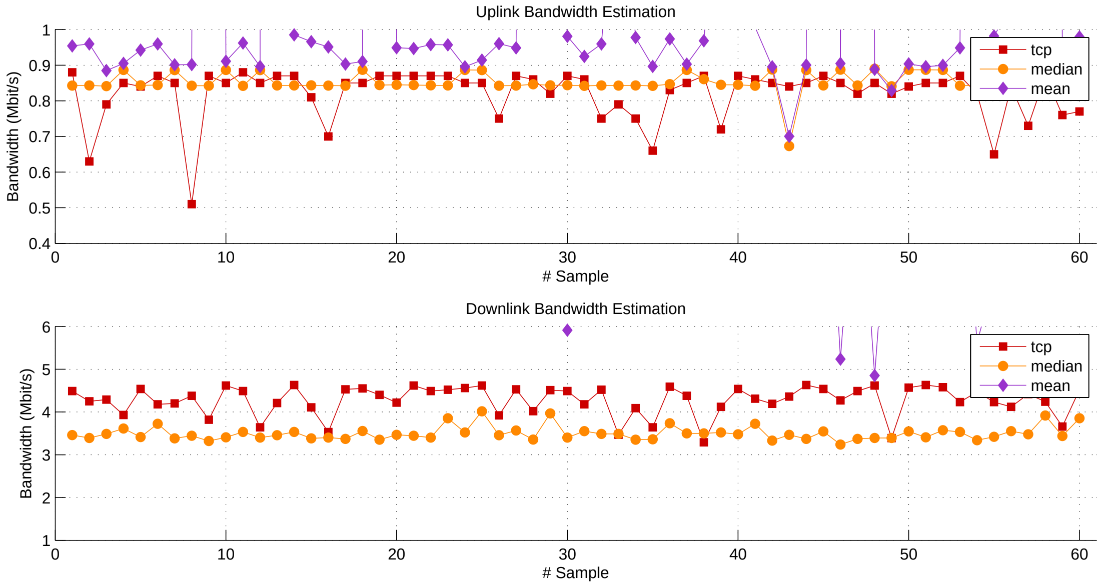


Figure 9.20: The CBA method applied to 60 measurement samples.

In the uplink on Figure 9.20 the median and TCP show very similar bandwidth estimations, but in downlink the median always shows a lower estimation of the available bandwidth when compared to the TCP results.

In Figure 9.21, for uplink, the mean shows more steady results while the median has increased a little such that both show a bandwidth estimation that is a little higher than the TCP results. For the downlink, both the mean and the median show estimations very close to the TCP results, but also here both are located a little higher than the TCP results.

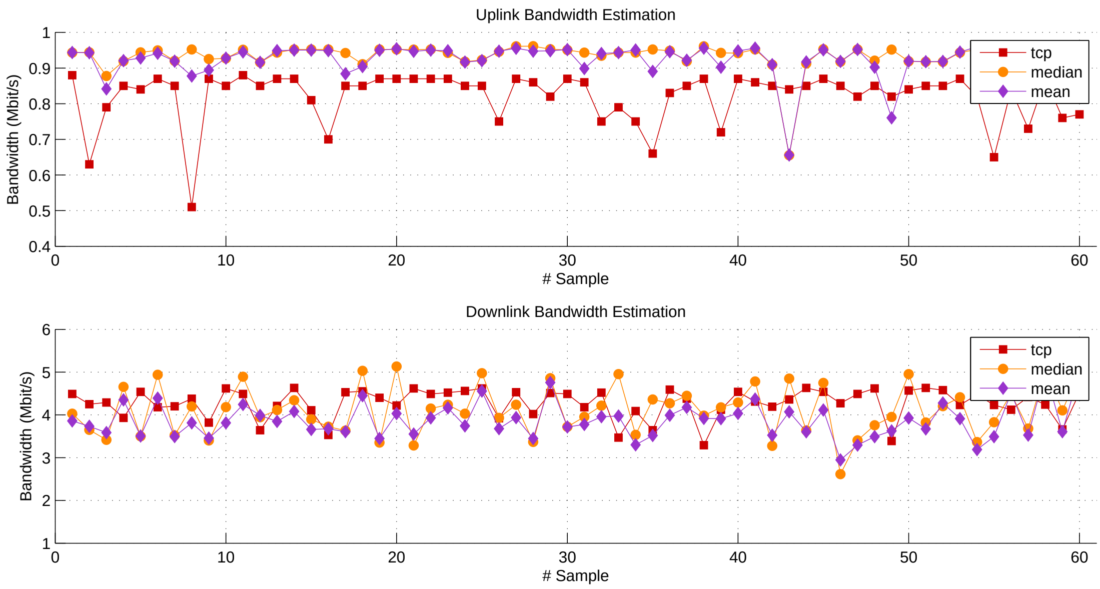


Figure 9.21: The APB method applied to 60 measurement samples.

Based on the figures it seems as if the **Average Per Burst (APB)** method provides the best estimation when comparing the 2 methods.

Accuracy in the methods: When looking at the accuracy between the TCP throughput estimations and those gathered using the PPT method, it is possible to determine the efficiency of the PPT method. This is done by looking at the absolute values between the estimations, to determine how close the PPT estimations are to the TCP throughput measurements. The accuracy will be calculated using the same data used in Figures 9.20 and 9.21, but only using the median calculated from the PPT method.

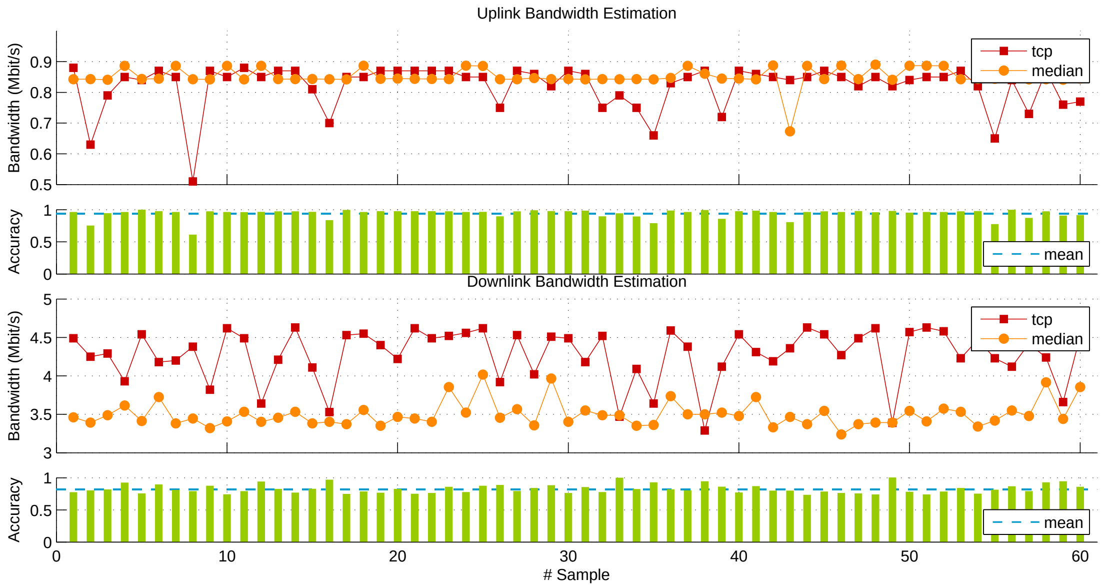


Figure 9.22: CBA showing the accuracy between the TCP throughput measurement and the PPT method from 60 measurement samples. In the bottom plot the accuracy between each sample is presented as a bar plot that also contains the average between all accuracy samples.

Figure 9.22 shows the same results as displayed in Figure 9.20, but with the accuracy between the TCP throughput measurements and the PPT median calculations displayed. From the figure it can be seen that the uplink results show very high accuracy in the measurements, but when looking at the downlink it can be seen that a (close to) constant 1 Mbit/s difference is present which gives an accuracy close to 80%. This can also be seen when looking at the average accuracy for all the samples, which can be seen in the table below:

Average accuracy: (Combined Burst Array using median)	
Uplink:	93.64 % average accuracy from 60 samples.
Downlink:	82.12 % average accuracy from 60 samples.

Table 9.1

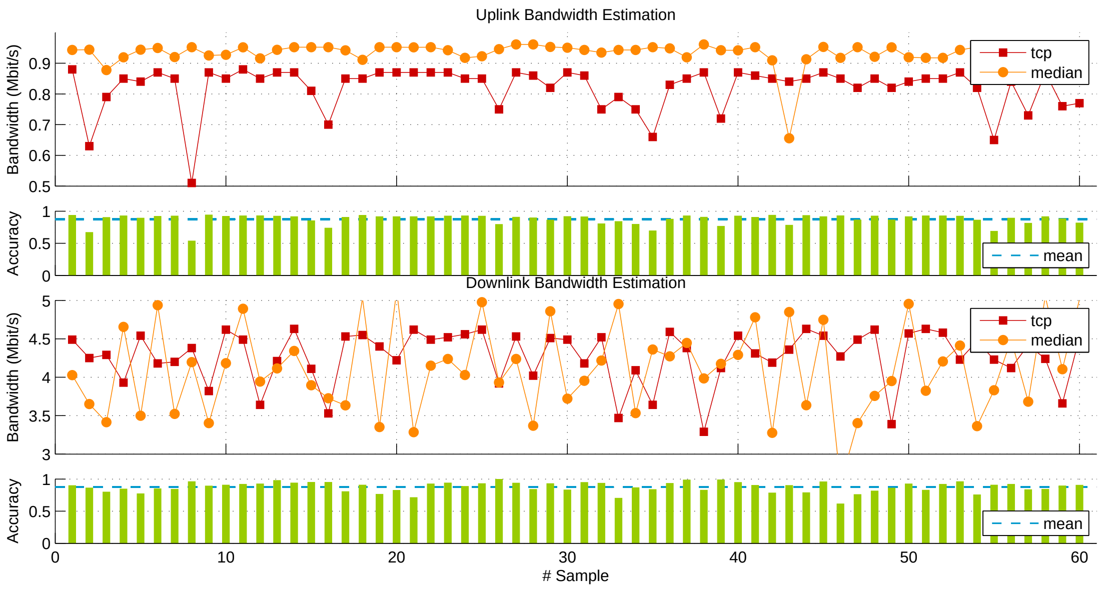


Figure 9.23: APB showing the accuracy between the TCP throughput measurement and the PPT method from 60 measurement samples. In the bottom plot the accuracy between each sample is presented as a bar plot that also contains the average between all accuracy samples.

Figure 9.23 shows the same results as displayed in Figure 9.21 but again only the PPT median calculations are looked at. When looking at the uplink results a worse accuracy than before is seen, but the downlink now provides a better accuracy. The uplink median estimations are all located close to the subscription limit of 1 Mbit/s which could indicate that they estimate the actual available bandwidth. The downlink results all fluctuate between one another indicating that they provide somewhat the same results. The average accuracy from all the samples can be seen in the table below:

Average accuracy:	**(Average Per Burst (APB) using medians)**
Uplink:	87.44 % average accuracy from 60 samples.
Downlink:	87.41 % average accuracy from 60 samples.

Table 9.2

It should be noted that the accuracy results are based on the assumption that the TCP results are accurate approximations of the true values of the bandwidth. Furthermore, another uncertainty of the accuracy is that the measurements by the different methods are not performed simultaneously, why the network might change in the meantime, causing the results to differ.

9.7 New Mobile Test Results

An area that needs to be investigated is the accuracy of the PPT method, in the case where the subscription has a higher or no speed limit. This will be tested in this section, where a subscription from Telenor[tel13] is used, which has no speed limitation. This means that the limitation will be caused by the capabilities of the device and the network in the area where the measurements are performed. In the previous tests the speed was limited by the ISP to 5 Mbit/s download and 1 Mbit/s upload. The device used is still the Sony Xperia U (see Table 6.1), which can reach a maximum peak rate of up to 14 Mbit/s in the downlink and 5.76 Mbit/s in the uplink [wik13c].

A series of measurements will be conducted using the same procedure as described in Section 9.5. The focus point of the samples is to determine the performance of the PPT when the potential throughput is increased.

9.7.1 Combined Burst Array (CBA) Results:

First the results when combining the estimation into one big array are analyzed. In Figure 9.24 the PPT uplink and downlink median estimations, compared to the TCP throughput measurements, are seen. In the bottom of each plots the accuracy between the two tests is seen, and as it was seen in Section 9.6, the difference in how accurate the samples are is somewhat random as some have very high accuracy and others have very low. But it can be seen in both the uplink and downlink results that the PPT method performs just as well when the throughput rates are increased on the cellular network.

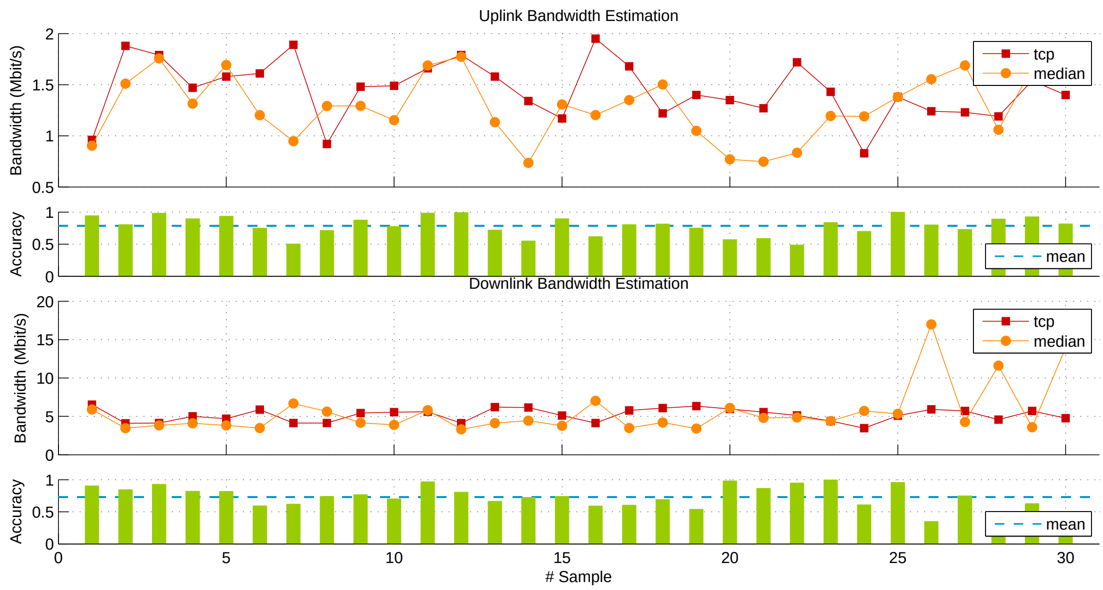


Figure 9.24: The results obtained using the CBA method and the accuracy between the TCP throughput measurements and the PPT method, from 30 measurement samples. The samples are all captured using the Telenor subscription, which has no throughput speed limitation.

In Table 9.3 the average accuracy is presented. If the result is compared to that in Table 9.1 a drop in accuracy is seen, but when looking at some of the individual samples the results are almost equal. The reason for the drop in average accuracy is the extreme values in some of the samples which skew the results.

Average accuracy: (Combined Burst Array (CBA) using medians)	
Uplink:	78.71 % average accuracy from 30 samples.
Downlink:	72.86 % average accuracy from 30 samples.

Table 9.3

9.7.2 Average Per Burst (APB) Results:

Again looking at the same measurements but now displayed using APB, which can be seen in Figure 9.25. From this a slightly better accuracy between each sample and on average, when compared to the CBA method, is seen. As with the CBA method some of the unique samples still show poor accuracy between the PPT and TCP tests, which can be explained by sudden changes in the network. But as seen in the previous section, the *Average Per Burst* method still provides better estimates than the *Combined Burst Array* method.

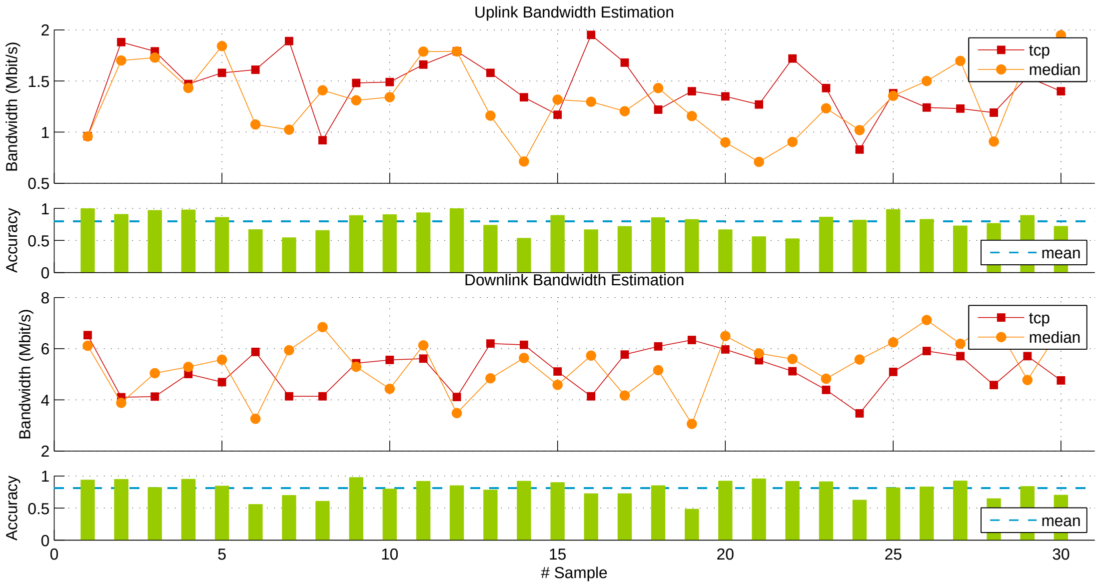


Figure 9.25: Results obtained using the APB method and the accuracy between the TCP throughput measurement and the PPT method, from 30 measurement samples. The samples are all captured using the Telenor subscription, which has no throughput speed limitation.

When looking at the average accuracy in Table 9.4 and comparing it to that in Table 9.3, a better accuracy for APB is seen. However if compared to the results in Table 9.2 (for the limited throughput rates), a poorer accuracy can once again be seen here.

This drop in accuracy might be caused by some of the compared samples being very far from each other in the particular measurements gathered. As stated in the Sources of Error (Section 9.5.1) the state of the cellular network could change a lot while conducting the measurement, and because the PPT and the TCP throughput are not executed at the same time the comparison of the particular samples should not be seen as a rigorous result.

When looking at the results in Figure 9.25 the PPT and TCP results are all located close to each other, and all samples show overall higher throughputs (due to the subscription), indicating that both the TCP throughput and the PPT methods can be deemed correct.

Average accuracy:	**(Average Per Burst using medians)**
Uplink:	79.41 % average accuracy from 30 samples.
Downlink:	81.07 % average accuracy from 30 samples.

Table 9.4

9.8 Summary and Conclusion

It is worth mentioning that there is a big difference between using the UDP (PPT) and the TCP (TCP throughput) protocol, why the comparisons between the measurement methods should not be seen as rigorous results. As mentioned briefly under Sources of Error (Section 9.5.1), a series of different factors play a crucial role and can not be ruled out when comparing the different results, along with the fact that the two protocols behave differently.

From the different measurements conducted throughout this section, the Packet Pair Technique showed good properties when trying to estimate available bandwidth. The technique was tested both on an internal network setup, and later moved onto a realistic network scenario seen from a final project goal. The system was deployed onto a mobile device where the system did show some big fluctuations from the fact that it ran on Android OS and because of the TTI caused by the cellular network technologies, but with use of different estimation or filtering methods some good estimations were achieved.

The PPT reduces the amount of load and traffic put into the network significantly. An example of this can be seen list below. The list is based on the 60 samples shown in Figures 9.20 and 9.21.

Packet Pair Technique:		
	Uplink:	Average amount of traffic created per test from 60 samples: 0.63 Megabytes.
		Average packet loss: 0.04%.
	Downlink:	Average amount of traffic created per test from 60 samples: 0.59 Megabytes.
		Average packet loss: 7.22%
TCP Throughput Test:		
	Uplink:	Average amount of traffic created per test from 60 samples: 1.62 Megabytes.
	Downlink:	Average amount of traffic created per test from 60 samples: 8.29 Megabytes.
The Difference:		
	Uplink:	PPT uses 38.90% of the traffic data compared to TCP throughput test.
	Downlink:	PPT uses 7.06% of the traffic data compared to TCP throughput test.

Table 9.5

Because of the many different parameters that can be changed in the PPT, it is very flexible and has the possibility to be optimized depending on the network it is currently trying to estimate. For example, the parameters could be changed according to the current cellular network type (2G/3G/LTE), for instance creating more samples for a better estimations.

Furthermore, the PPT did not show any issues when it was tested on a subscription which had no data rate limit as previously. The technique displayed the same behavior as the tests conducted on the limited data rate subscription indicating that it can be used on higher data rates.

Chapter 10

Combined PGM and PRM Technique

In addition to the Packet Pair Technique described, in Chapter 9, another technique which uses parts of both the Probe Gap Model (PGM) and Probe Rate Model (PRM) has been created. This technique is based on methods mentioned in Section 8.3, more specifically it is based on a Self-Loading Decreasing Rate Train (SLDRT) model based on PRM, but instead of using one-way latencies to estimate the available bandwidth this technique uses the interarrival time between received packets, which is what PGM utilizes.

The chapter starts with a description of the technique, which explains the principle as well as the considerations behind the implementation. Then the actual implementation is described, explaining how to estimate the available bandwidth based on the measurements. Following this some tests are performed on the cellular network using different parameters, and the results are analyzed to see if it could be a possible replacement to the existing TCP measurement method (Bulk transfer capacity). The tests on the cellular network are compared to tests performed using WiFi as it is expected that different behavior can be observed, and as this network provides higher speeds it should be possible to find the limitations of the technique. Finally a conclusion is made based on the measurements and the analysis, followed by a short discussion regarding the technique.

10.1 Description

This technique uses equally sized UDP packets sent with varying rate from the mobile device to the server, which can then be used to estimate the available bandwidth in the uplink direction. It has been chosen only to perform upload measurements as the uplink capacity often is set lower than the downlink capacity, which has an impact on the timing of transmission rate as well as the interarrival time measurements. In the test implementation a default packet size of 1000 bytes is used for the estimation, which results in an initial transmission rate (Δt) of 7.63 ms for a 1 Mbit/s connection or 1.53 ms for a 5 Mbit/s connection. If these values are compared to the Transmission Time Interval (TTI) values described in 8.2 it is seen that only HSPA is expected to provide a sufficiently low resolution (2 ms), and only for the 1 Mbit/s throughput limit of the upload.

Additional tests have been performed using this technique, but with different parameter values to see how these might influence the estimations. The tests will not be used in the conclusion, as they mainly serve to see if the default values described in Section 10.3 are sufficient. The tests can be found in Appendix A, page 148.

To get a better understanding of how the technique works, the message sequence chart can be seen in Figure 10.1.

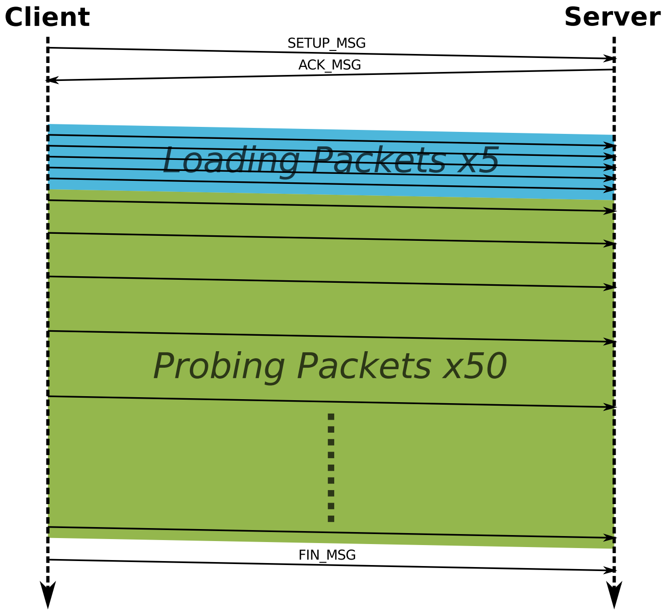


Figure 10.1: Communication flow of the Combined Technique for measuring available bandwidth. Note that only a few of the probing packets are shown in the figure for easier viewing.

As shown on the figure the technique uses two types of packets, loading and probing, which are both important for estimating the available bandwidth. To get a better understanding of the technique, a short description of both packet types follows.

Loading Packets: These packets are used to generate congestion on the network which results in packets being queued somewhere between client and server. A set amount of packets are transmitted at a static rate which will be referred to as **Initial Rate** (R_{init}) for the remainder of this chapter. The rate should be larger than the bottleneck capacity of the connection to ensure packets are queued, but not too much larger as this is expected to decrease the accuracy of the estimations.

The queue will remain after the loading packets have been transmitted, which should give an increase in interarrival time for the first of the probing packets as it is expected that the queuing delay will vary. This assumption is based on the work done by the author of pathChirp, who describes how the assumption of cross-traffic having a constant bit-rate is an oversimplification, and that the queuing delay is more likely to vary as shown in Figure 10.2 [Cot03].

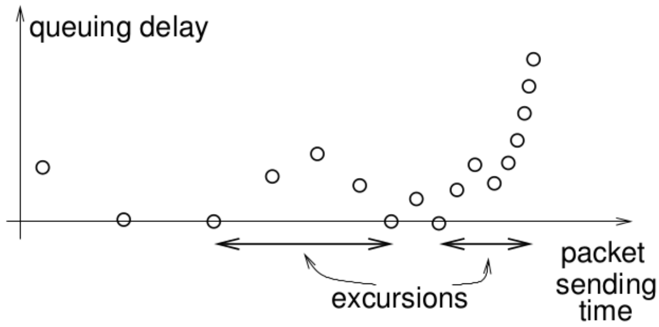


Figure 10.2: Typical queuing delay signature, with marked excursions showing additional delay caused by bursts of cross-traffic [Cot03].

Probing Packets: This type of packets are used to find the available bandwidth, and does so by changing the rate of transmission throughout a measurement. A packet train using this method is often referred to as a ”Chirp”. For the measurements performed using the proposed technique, the chirps will start at a rate at or just above the known capacity of the bottleneck, as it was described for the loading packets, and then gradually lowering the rate. The probing packets combined with the queue created by the loading packets should make it possible to estimate the available bandwidth. It is expected that the queues on the path will use the principle of FIFO, which was explained in Figure 9.1 in Chapter 9, and it is this principle that this measuring technique tries to utilize.

While the probing packets are transmitted the queue will eventually disperse as the rate of transmission is continually lowered. It is expected that this can be seen on the interarrival times at the receiver, as they should initially be high as a result of the queuing delay variation shown in Figure 10.2, but when the queue is gone it should drop a little only to start increasing again due to the changing transmission rate at the client (sender). An illustration of the expected behavior of the interarrival times can be seen in Figure 10.3. From the actual measurements more variations in the interarrival times are likely to occur, but this will be handled by post processing of the data.

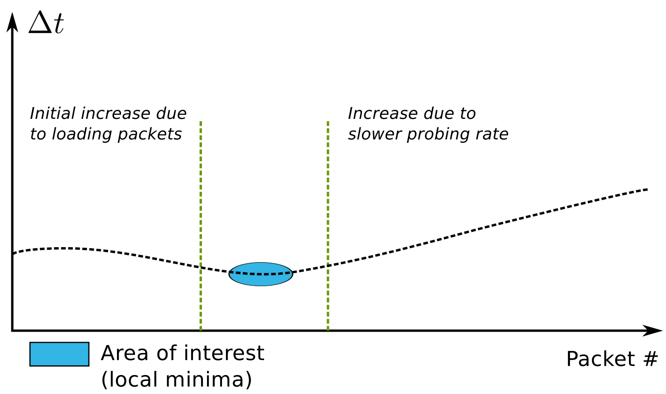


Figure 10.3: Example showing the expected interarrival times (Δt) of a measurement sequence.

As shown on the figure the location marked is a local minimum, which should indicate the change in interarrival time from packets being queued to packets following the expected transfer rate. As some variance is expected in the interarrival time there might be several minima, which has to be considered in the test implementation.

10.2 Test Implementation

As described previously the current test implementation only measures upload, as the requirements to timing is lower as a result of a lower capacity in the uplink direction. The communication between client and server is kept as low as possible, to avoid complicating the technique and to reduce the network intrusiveness as much as possible. The client and sever implementations are both done in Java, and the source code can be found in *code/Combined/Mixed_NetMap* (Client) and *code/Combined/Mixed_Server* (Server).

As it can be seen on Figure 10.1 a measurement is initiated by the client sending a setup message (*SETUP_MSG*), this message works as both an initialization message to the server, but also contains parameters that will be used by the client for measuring, which allows the server to dynamically adapt to the measurements. The server acknowledges the transmission (*ACK_MSG*), letting the client know that it can start the measurement sequence.

After initiating a measurement a series of loading packets are transmitted, which should induce a queue at the bottleneck on the network path. This is the most intrusive part of the measurement as this is where packets are transmitted at the highest rate, which should be no lower than the capacity listed by the network provider. The packet size should not have any direct influence on the results, but larger packets will increase the interarrival time needed between packets to reach a specified rate, as it can be seen from equation 10.2.

$$r_i = \frac{r_{i-1}}{\alpha} \quad [kb/s] \tag{10.1}$$

The above equation describes how the rate, r, changes according to value of α. The initial rate that is used for all the loading packets is denoted R_{init}, and should be no lower than the capacity specified by the provider as mentioned previously. The value of α must be larger than 1 to ensure a decreasing rate. Additionally the value of α should be chosen according to the amount of packets used for probing, the measurement span and the accuracy of the measurements. By increasing α the rate will drop faster which decreases the precision on the first packets.

The probing packets are transmitted immediately following the loading packets. The interarrival time Δt is found from the rate described in equation 10.1 as well as the packet size, p, that is used for both the loading and probing packets.

$$\Delta t_i = \frac{125 \cdot p}{16 \cdot r_i} \quad [ms] \tag{10.2}$$

After the probing packets have been sent the client transmits an end-of-transmission message (*FIN_MSG*), which tells the server to save the result and return to its waiting state again. Alternatively the server will return to the waiting state if a timeout occurs during the measuring routine, and no result will be saved from the measurement.

10.3 Test and Results

For testing the technique the implementation has been running on a Sony Xperia U, which has been communicating with the server that has been used for previous implementations of the NetMap system. The application framework is the same as what is currently implemented, but with all other measurements disabled. This is both to ensure that no additional bias is added to the measurements, and because this is the simplest method of implementing the technique, as the framework is module based.

The implementation of the technique is based on a set of parameters that controls the behavior of the measurement. The parameters are as follows

- **Alpha** (α): The value that decides how fast the rate changes during probing measurements.
- **Initial Rate** (R_{init}): The initial rate chosen for the measurements given in kbit/s.
- **Packet Size** (p): The packet size used for both loading and probing packets given in bytes.
- **Loading Packets** (d): The number of packets that are transferred using the initial rate to create congestion on the path.

A standard set of values for these parameters has been chosen as default values, and these values are: $\alpha = 1.05$, $R_{init} = 1024$ kb/s, $p = 1000$ bytes and $d = 5$. Each set of measurements will consist of 10 measurements with a 5 second pause between each, which will allow for some statistical methods to be used later in the data processing and ensure that buffers are empty when each measurement is performed.

During a measurement 58 packets are transmitted, 56 of them are 1000 bytes while the last two are 100 bytes, which results in approximately 55 kB of data used for each measurement. As each set of measurements consist of 10 individual measurements it leads to approximately 550 kB of data for a set, without counting the overhead as this is not seen as part of the available bandwidth.

To get an idea of what output is produced using this technique, the top of Figure 10.4 shows the basic output that is generated for a measurement set, which consists of 10 measurements performed using the method described in 10.1.

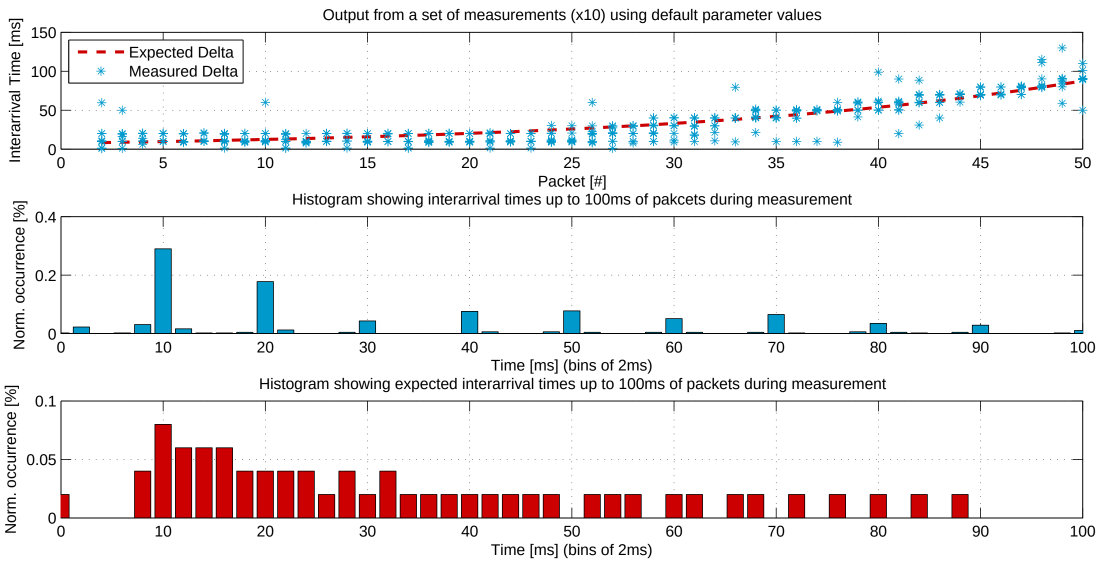


Figure 10.4: Basic output for set of measurements. The measured values are found by the server, while the expected values are transmitted within each packet.

The basic output from a set of measurements is the interarrival times, or delta values, which are marked by the blue dots in the top and plotted as a histogram in the middle of Figure 10.4. The histogram shows a tendency where the measurements are centered around bins with 10 ms accuracy. It is likely that the distribution in Figure 10.4 is due to the Transmission Time Interval (TTI) that is present for each of the technologies used for the measurements, which was described in 8.2. The bottom part of Figure 10.4 shows the expected interarrival times, as received from the client, and here it is seen that an accuracy down to 1-2 ms is required to experience the same interarrival times on bloth the client and server.

Looking further into the results, the top of Figure 10.5 shows the median values found from the set of measurements, and from the figure it can be seen that there is a slight resemblance with the expected output shown in Figure 10.3. It has been chosen to use the median value as Figure 10.4 shows that there are some outliers present in the measurements, and to ensure these do not create unnecessary skew to the measurements the median is used instead of the mean to create averages.

To get a more suitable result for finding the wanted minima, the median values from the top of Figure 10.5 are smoothed using a method known as *robust locally weighted scatter plot smoothing*, which uses local regression [loc13] on subsets of 20% of the median values for each point in the smoothing, as well as a weighting value depending on the deviation. The median and smoothed results can be seen in Figure 10.5.

From the smoothed plot the minima values are found and marked with a red dot indicating the interarrival time as well as the packet numbers for which the minima are found. The packet numbers found from the minima in the smoothed plot are then transferred to the median plot seen in the top of Figure 10.5 and plotted with a blue dot, showing the corresponding interarrival times from this plot whose values are considered more realistic as they are not smoothed.

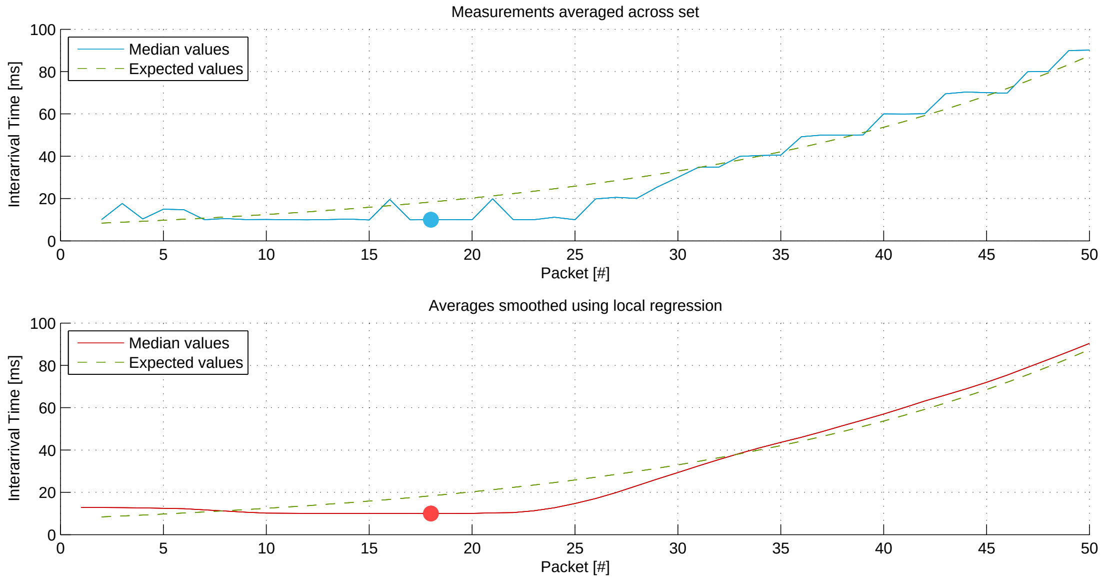


Figure 10.5: Averaged (median) and smoothed representation of a set of measurements. The top part of the figure shows the median of a measurement set consisting of 10 measurements. The green dashed line shows expected values found from equations 10.1 and 10.2. The dots indicate the interarrival times at packets that are found as minima in the smooth plot.

The smoothed output shows the expected characteristics, and the amount of local minimas in each measurement set is reduced to mostly be within 1-4 points. When multiple minima are found, the median of the interarrival times at all the minima points is used.

When the interarrival times at the minima have been found from Figure 10.5, the conversion to available bandwidth is done using the approach from Equation 10.2, which leads to the following equation:

$$r_i = \frac{125 \cdot p}{16 \cdot \Delta t_i} \quad [kb/s] \tag{10.3}$$

After the interarrival times have been found using the minima for both the median and smooth curve in Figure 10.5, and the interarrival times obtained have been converted to available bandwidth or rate using Equation 10.3, the resulting values are plotted in Figure 10.6.

The results found from the minima in Figure 10.5 are plotted as Measurement #1 in Figure 10.6, meaning that the entire process of sending 10 packet trains, finding the minima and converting to available bandwidth has been repeated 10 times to produce the results shown in 10.6.

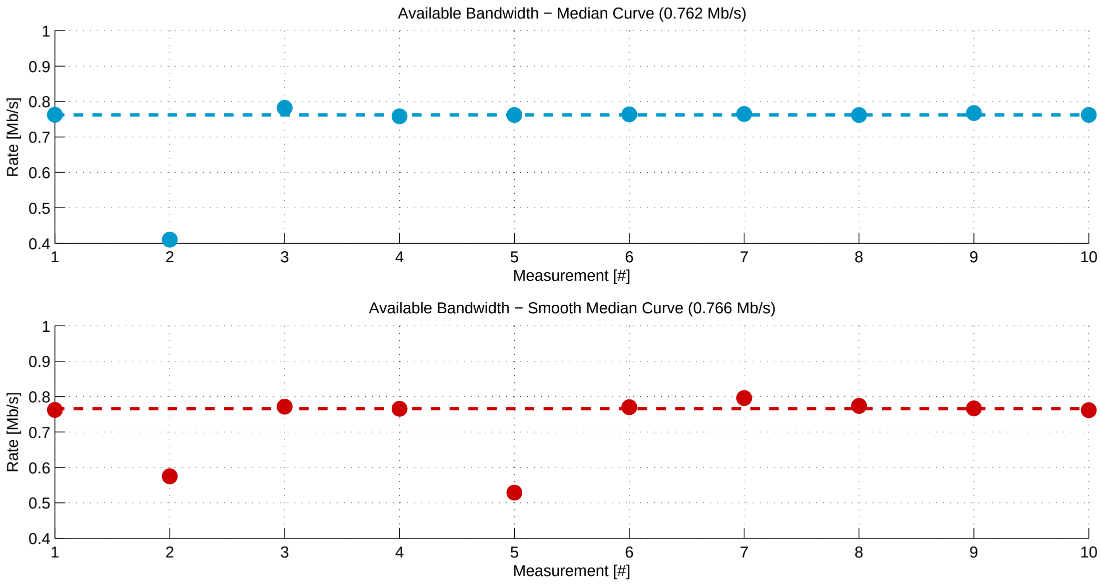


Figure 10.6: Results obtained from 10 measurements, with Measurement #1 shown in greater detail in Figure 10.5. The colors used in this figure refers the method used for finding the interarrival times (either median or smooth in Figure 10.5), and the dashed lines are the overall median for all 10 measurements, which is the value shown in the title of each of the plots.

Looking at the results obtained it can be seen that the estimation of the available bandwidth is fairly stable in the majority of measurements, but there are still measurements that vary a lot from the overall mean. The overall estimation is approximately 0.76 Mbit/s for both the median and the smooth approach, on a connection that is limited to 1 Mbit/s.

The results shown here are made using the default set of parameters, which were described in the beginning of this section. Additional measurements have been performed using different parameter sets, which can be seen in Appendix A, page 148. The results obtained from the appendix did however not show noticeable improvement on the estimation, and will as a result not be considered for further testing.

10.4 WiFi Test

The behavior of the interarrival times for a single set of measurements shown in Figure 10.4 was considered previously in this chapter, and it was believed that the clustering of interarrival times was caused by the TTI, which was described for the cellular technologies used in section 8.2. Additional measurements have been performed using WiFi, to see if the cellular network has any influence on the measurements. The measurements are performed with the same type of device and software as the primary tests performed on the cellular network.

10.4.1 Test #1 - Default Parameters

The first test is performed using measurements collected with the default set of parameter values, allowing for a comparison with the already described measurements performed on the cellular network. The histogram showing the distribution of measurements using WiFi with parameters $\alpha = 1.05$, $R_{init} = 1024$ kb/s, $p = 1000$ bytes and $d = 5$ can be seen in Figure 10.7.

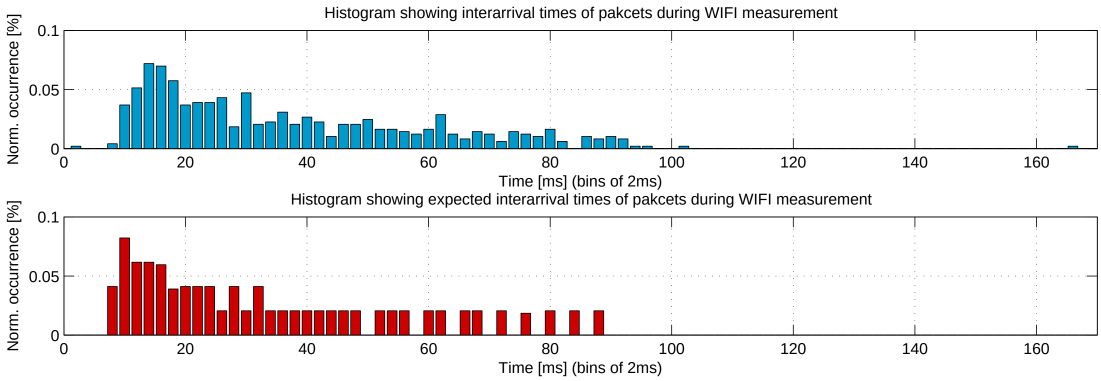


Figure 10.7: Histograms of both the measured and expected interarrival time using WiFi.

The histogram shows that the 10 ms clustering of interarrival times seen in the measurements on the cellular network has disappeared, and the measured values now show a behavior closer to the expected interarrival times. It is believed that the clustering is due to the technologies, as EDGE has a TTI of 20ms, W-CDMA has 10ms while HSPA has 2ms as it was described in Section 8.2. The histogram shown in Figure 10.7 is made with $R_{init} = 1024$ kb/s which is below the upload speed that can be expected using WiFi, and the results obtained indicate that the initial rate is too low as well, as no minima are found for any of the 10 measurement sets performed, which can be seen for a single set in Figure 10.8.

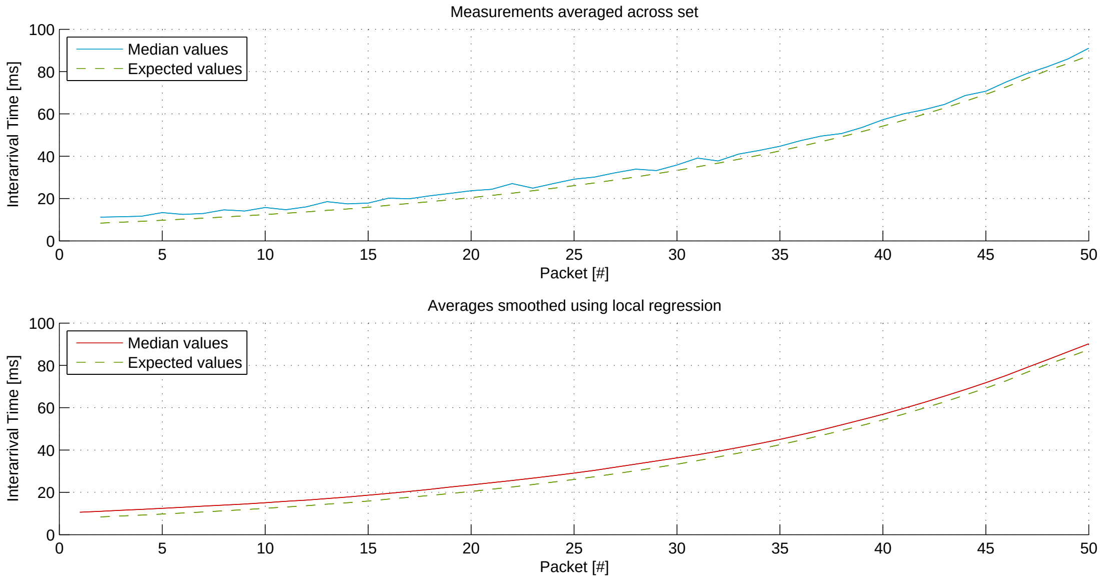


Figure 10.8: Median and smoothed representations of a set of measurements performed using WiFi. A more detailed description can be found below Figure 10.5.

As it can be seen in the figure the difference between the expected and measured interarrival time is much less than what was seen in the mobile measurements in Figure 10.5, and as the curve is gradually rising there are no minima to estimate the available bandwidth from. As the initial rate is 1024 kb/s the most likely reason is that no queue is invoked on the network by the loading packets.

The above test using WiFi was primarily made to see the behavior of interarrival times, as seen previously in Figure 10.7, and even though the available bandwidth could not be estimated as the initial rate (R_{init}) was too low compared to what is possible using WiFi, the behavior of the packets still clarify that the cellular network, or more specifically the technologies used on the network, are causing the clustering of interarrival times.

To see if the technique functions under higher speeds, another test will be performed using WiFi, but with a higher initial speed, $R_{init} = 20Mbit/s$, meaning that the initial interarrival time will be 0.38 ms for 1000 byte packets.

10.4.2 Test #2 - High Initial Rate

The following test is made to test the limits of the estimation technique. The parameters are as follows: $\alpha = 1.05$, $R_{init} = 20480$ kb/s, $p = 1000$ bytes and $d = 5$. Except for the initial rate R_{init} which has been increases, the parameters remain at their default values. A single measurement set can be seen in Figure 10.9, showing the individual measurements as well as the histogram of both the measured and expected interarrival times.

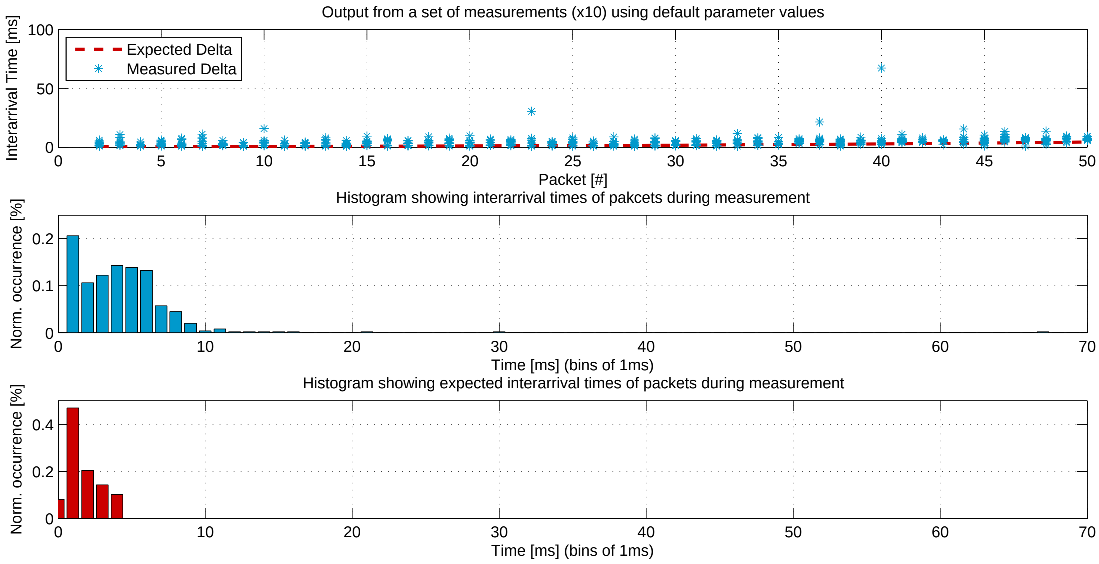


Figure 10.9: A measurement set performed using WiFi with $R_{init} = 20Mbit/s$. Note: The bin size has been reduced to 1ms.

As it can be seen on the middle and bottom plots of Figure 10.9, the measured interarrival times are generally higher than the expected values. To further emphasize these results, graphs showing the averaged and expected results for the same measurement can be seen in Figure 10.10.

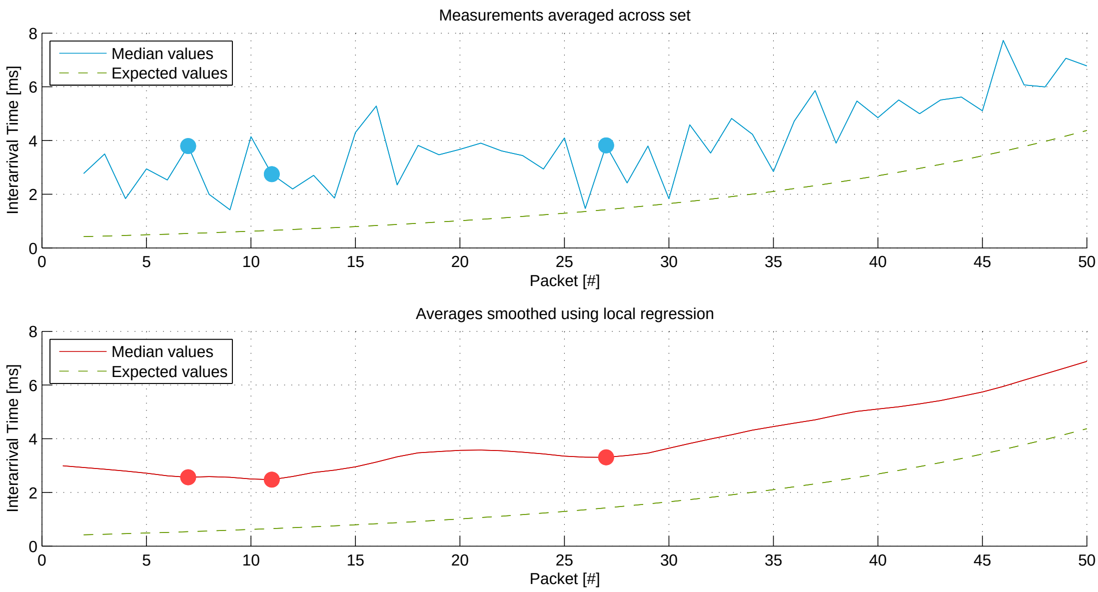


Figure 10.10: Averaged and smoothed representations of a set of measurements performed using WiFi with $R_{init} = 20Mbit/s$. A more detailed description can be found below Figure 10.5.

As expected the higher interarrival times are also visible here, but if compared to the 10.8 it should be noted that both the expected and measured values in this test are much smaller. As it can be seen in the smoothed curve in the bottom of Figure 10.10 the behavior is slightly different from the measurement shown for cellular networks in Figure 10.5. From this measurement set the curve has more of a wave shape, which results in multiple minina. It is expected that this might be caused by small variations on the network, which when combined when the low expected interarrival times of less than 5 ms is enough to create additional minima in the measurements.

Generally the minima found for all the measurement sets when using WiFi are more spread than they were for the cellular networks, which combined with the results showing that the interarrival times are higher than expected could indicate that the limit for this technique using the current implementation is between 2.5 and 3 Mbit/s. The results obtained for the measurements using WiFi can be seen in Figure 10.11, which backs up the theory of a limit to the speeds that can be measured.

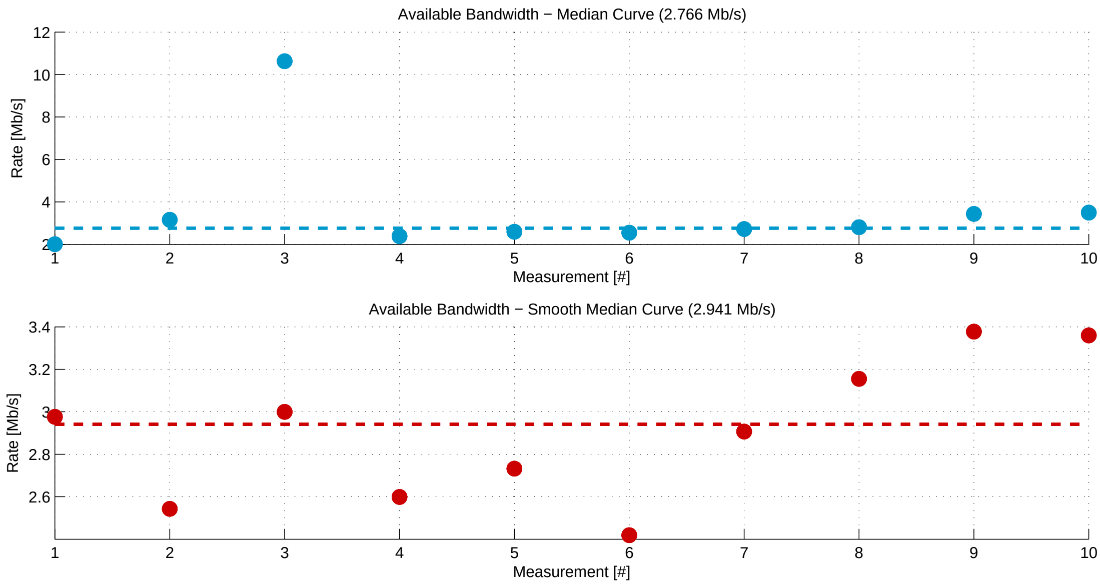


Figure 10.11: Results obtained from WiFi measurements using $R_{init} = 20Mbit/s$. A more detailed description of the plots can be found below Figure 10.6.

Looking at the results it is seen that none of them show a stable performance across multiple measurements. Common for both is the overall average which, as mentioned previously, is around 2.5 and 3 Mbit/s. Additional tests performed using Speedtest [spe13] (External server) and Iperf [ipe13] (Internal measurement server) have shown upload speeds around 15 Mbit/s, which strengthens the suspicion that the results obtained in 10.11 show the highest rate that can be measured using the current implementation.

10.5 Conclusion

From the results obtained throughout this chapter and the Appendix A, it is seen that the Combined PGM and PRM Technique in its current state has limited potential to be used as an alternative to the already implemented technique which measures the available bandwidth as bulk transfer capacity using TCP. The biggest gain from using this method compared to the existing is the amount of traffic generated, which is substantially lower than the method currently implemented.

Finding the difference between this method and the already implemented method is difficult, as measurements can not be done in parallel without creating additional bias to both measurements. In the previous chapter tests were performed sequentially with the measured upload speeds being around 0.8 - 1 Mbit/s using both methods, which is still slightly higher than what this technique was able to estimate. The biggest limitation to the technique seems to be the TTI from the different mobile technologies, which severely limits the accuracy of measured interarrival times.

The results obtained using WiFi showed what is believed to be a limitation to the speeds that can be measured, and this alone means that in its current state the technique, or the software implementing it, will not be a good substitute for the existing NetMap implementation which measures bulk transfer capacity. It might be possible to obtain better results by optimizing the software, but with the way the cellular network handles data packets, additional testing must be performed to be able to draw final conclusions.

10.6 Discussion

Currently each set of measurements consists of 10 measurements of the type shown in Figure 10.1, with a fairly large delay of 5 seconds between each measurement. The measurement itself is very fast, as the packets are transmitted with very small interarrival times as shown in the bottom of Figure 10.4, but with the delay between each measurement of 5 seconds, each set of measurements currently take more than a minute to complete. It is expected that both the amount of measurements required for a set as well as the delay between each measurement can be optimized further, to allow for faster measurements.

The amount of variation seen in the measurements from Figure 10.6 suggests doing multiple measurements and taking the median of these as it has been done in all the tests, but the amount of measurements required could be reduced. This reduction could be made as a set value, e.g. 5 measurements, but an alternative way could be to terminate a measurement when the overall median stabilizes, as the results show a tendency for the overall median to be very close to the majority of the estimations obtained from individual measurement sets. This would be a good way to do dynamic termination of the measurements, but as all the results are found using post processing, the current test implementation does not support this method.

The initial rate parameter (R_{init}) required by the Combined Technique is difficult to determine beforehand, which will lead to issues when performing on different networks and connections. One way of obtaining a value could be to perform a special test when the application is initialized, and redo this each time a network handover occurs, e.g. changing from 2G to HSPA, as this should be sufficient to have the initial rate within range for the measurements.

As the transmission rates increase, the requirements to both accuracy and precision increase as well. As the accuracy requirements for interarrival times exceed what can be expected of the network in terms of TTI, even when measuring as low as 1 Mbit/s (which requires Δt = 7.63ms using 1000 byte packets), measuring interarrival times might not be a suitable solution for cellular networks. The test using WiFi showed no issues with clustering of the interarrival times, but seemed unable to estimate values higher than approximately 3 Mbit/s.

For the technique to be generally useful, it should be capable of measuring speeds higher than what is currently possible, which is expected to require optimization of the software. The clustering of interarrival times using the cellular network will most likely also be a bigger issue for higher speeds, but as this can not be tested using the current implementation it is unknown to what degree this will influence the measurements.

Chapter 11

Conclusion and Discussion of Alternative Bandwidth Estimation Methods

In this part of the report two novel techniques of estimating the available bandwidth have been considered. Both techniques are based on existing work, but take a different approach as the techniques are modified to work, and tested, on mobile devices and networks. The currently implemented technique in the NetMap system estimates the bulk transfer capacity by fully utilizing the connection for a period of time, but as a result it has a high data usage which is an unwanted feature in a system made for crowd-sourcing. The two novel techniques both try to limit the data usage by using different approaches to estimate the available bandwidth, which is a metric describing how much capacity a user can utilize at a given time, as described in Section 8. The novel techniques both use UDP for transferring data, while the current implementation uses TCP which automatically handles transmission speed and retransmission.

The first technique is the Packet Pair Technique (PPT), which uses small packet bursts and measures the interarrival time between the packets. Two filtering methods have been used with this technique: The first method is Combined Burst Array (CBA), which uses the individual interarrival times of all the packets in the burst, and gives an estimation based on each of these. The second filtering method is Average Per Burst (APB), which uses the interarrival time of each whole burst, imitating the calculation of bulk transfer capacity, but using less data.

The second technique considered is the Combined PGM and PRM Technique, which uses a train of packets with predefined interarrival times to estimate the available bandwidth. The estimation works by creating a queue on the network in the beginning of the measurement, and then gradually transmitting packets at a decreasing rate until the queue disperses, at which point the available bandwidth can be estimated.

Conclusion of Alternative Bandwidth Estimation Methods

The reason for considering alternatives to the already implemented bulk transfer capacity method was to see if the data usage could be reduced. Both of the techniques considered throughout this chapter transmit significantly less data than the current implementation, but the issue then is how accurate the techniques are in comparison.

The results obtained using PPT on the cellular network followed the estimations from the current TCP implementation with approximately 80% accuracy, but only using between 7% and 39% of the data. Of the two filtering methods considered within this technique, APB was found to be the one giving the most accurate results.

The Combined PGM and PRM Technique was only tested in the uplink direction because of high requirements to precision on higher throughput rates. The rates obtained for the available bandwidth were generally lower than what was seen from the test using the PPT, with most of the measurements estimating around 0.78 Mbit/s with a known limit of 1 Mbit/s set by the provider. After analyzing the results it was found that the Transmission Time Interval (TTI) of the different cellular network technologies used would limit the accuracy of measurements, such that the majority of measured interarrival times would be clustered in bins with 10 ms spacing, as it is clearly seen in Figure 10.4.

If one of the techniques were to replace the currently implemented technique the PPT would be the preferred choice, as it is capable of estimating the available bandwidth in both directions with a reasonable accuracy, but using significantly less data than the currently implemented method. The Combined Technique has potential as an estimation tool, but with the way the cellular network handles data it will not suffice as an alternative to the current implementation.

Discussion of Alternative Bandwidth Estimation Methods

While working with the novel techniques in this part of the report, one of the biggest concerns has been the correctness of the measurements and estimations performed. The measurements are mainly performed on the cellular network where the internal setup is unknown, and only basic information about the technologies used on the network is available. If the estimation techniques were to be tested more thoroughly it would require a testing framework, where the behavior of the network is known and the performance is identical to the actual network. Such a framework could have been created using *The Network Simulator* (ns-2 or 3), but was omitted due to the time constraints of the project. Instead the estimations performed using the novel techniques are compared to measurements performed with the existing bulk transfer capacity method, which is considered the most realistic estimation of the throughput or available bandwidth.

Both of the techniques considered use the interarrival time of packets to estimate the available bandwidth, but as it was found, the accuracy of measured interarrival times was severely limited due to the TTI present in all of the considered network technologies. The impact of the reduced accuracy impacted both techniques, but was mainly noticeable during the estimations performed with the Combined Technique as it worked with predefined interarrival times for each packet in a test, whereas the PPT method did not specify the rate of packets beforehand. With the newer network technologies the limitation to the accuracy is lower, but as the available bandwidth increases the requirement for accuracy increases as well, meaning that the impact will be present for all technologies as it is related to both the available bandwidth and the TTI.

Part III

Final Words

Overview of Part III

In this part a final overview and wrap up of the report will be given. First, in Chapter 12 a summary of the conclusions and results reached during the various chapters of the report will be given in, followed by a final conclusion answering the problem description given in Chapter 2.

In Chapter 13 a discussion of the conclusions and the results achieved will be made, where the focus will be on how the results and discoveries made throughout the project can be incorporated into the NetMap project.

Lastly in Chapter 14, a look into the future of the NetMap system will be made. This will consist of an overall perspective of the NetMap system and the issues that should be attended in order to reach a release and deploy ready version of the NetMap system.

Chapter 12

Conclusion

In this chapter first the conclusions of the different chapters of the report will be summarized, followed by a final conclusion that will attempt to answer the Problem Formulation given in Chapter 2.

12.1 Project Overview and Summary

In this section an overview in the form of a summary will be given of the overall content and conclusions drawn throughout the report.

12.1.1 The NetMap System and Field Tests

To be able to analyze the impact of speed in this project, a new data set was gathered while driving in car on a route between Brovst, Aalborg and Aarhus, as seen in Chapter 4. The four parameters, RTT, Throughput, Connectivity and Handovers were analyzed with focus on the movement speeds that were registered while measuring.

From the analysis of the data and the impact of the movement speed, it was seen that the RTT, Throughput and Connectivity all showed a slight linear degradation in performance according to the movement speed, as it is seen in Figures 4.8, 4.11 and 4.14. This indicated that with an increase in speed, a deterioration in the network performance could be expected. However, this conclusion should not be considered rigorous because of the small sample size that the analysis is based on. In order to be able to make a stronger conclusion more samples are needed. Secondly, a more controlled environment could rule out some of the other influencing factors, such as varying number of users on the network while measurements are being collected.

By analyzing the CID handovers in a small bounded area it was concluded that in an area where handovers tend to occur the performance is affected, especially when looking at the throughput measurements. The drop in performance was assumed to be caused by the rerouting of traffic that happens when performing a handover, and by measures in the TCP protocol causing the rate to slowly adapt to a new connection.

In order to analyze if the individual parameters were affected similarly by the location of the measurements, the correlation between the parameters was calculated. This showed that the parameters, when mapped into square 1000x1000 meter tiles, did not show high correlation between each other. This is believed to be due to the much bigger tiles used here, when compared to the previous project [MTP12] where the tile size was 100x100 meters and the correlation between the different parameters were higher. The smaller tile size was used as the area covered by the route was much smaller and the amount of measurements higher. With the 1000 meter tiles, it is possible for the performance of a parameter to change a lot within a tile, why the local changes in performance are not that outspoken, as it was the case in the previous project.

The correlation of the parameters according to different movement speeds was also found. From this it was seen that the parameters RTT, Throughput, and Connectivity all showed close to high correlation. This also agreed with the results seen while analyzing the parameters according to movement speed.

The measurements were also analyzed according to area type, where 3 types were considered: *Urban*, *Small Town* and *Rural*. It was seen that the small town areas typically showed better performance than the urban and rural areas. From this it was assumed that the reason for the small town areas typically achieving the best performance was caused by a combination of relatively few users and many network resources.

As a measurement system based on crowdsourcing will be subject to many different device models, the performance of 4 different devices, ranging from low-end to high-end, was analyzed. The measurements showed that there were noticeable differences between the devices, which supports the need for information about the device that the measurements are performed on, to be used when evaluating the performance of a network in a given location.

12.1.2 The Current Measurement Methods

In Chapter 7, a more detailed look into the RTT and Throughput measurement methods was made. The goal of this was to get a more detailed understanding of the methods and see what the communication looks like during the measurements, and if possible propose an alternative result extraction strategy in order to get more detailed results from the measurements.

In the design of the NetMap application it was chosen that the RTT method should measure 10 RTTs per ping sequence. The idea behind conducting 10 samples for each measurement was to then take the mean of the samples, which would reduce the impact of any extreme outliers. This method was previously used in the results presented in [DMCP12], [MTP12] and in Chapters 4, 5 and 6. However, by looking further into the distribution of the 10 samples some unique patterns were seen, indicating that the 10 RTT samples were not normally distributed as assumed to be if they were treated equally by network and device. By analyzing the individual RTT, the mean of the 10 samples, and the spread of the samples, an alternative method for extracting results was proposed. This method includes using the mean, information about the spread, and the number of RTT samples that the values are based on. The information about the spread could be in form of the sample variance or a confidence interval of the mean. Furthermore it was suggested to do more pings per sequence as this would make the statistical values stronger.

In the current implementation of the throughput estimation the maximum rate is estimated by using a bulk transfer capacity method, and the result is the total amount of data transfered during the duration of the test, which is a simple average. This method does however not take into account the rise time of the throughput caused by the TCP slow start and congestion avoidance mechanisms, why it actually is not the maximum throughput that is achieved. The method was analyzed by looking at the fluctuations of the throughput by using a moving average, and based on the rates achieved the rise time was estimated for 10 upload and 10 download measurements. Based on this analysis two alternative methods for extracting results from the measurement method was proposed. In the first method the first few seconds of the measurement should be skipped before estimating the maximum throughput. The characteristics of the throughput rates during the first few seconds could then be used as a metric as well. In the second method the throughput rate is sampled many times during the measurement. Then a percentage of the lowest and the highest rates are discarded before calculating the maximum throughput from the remaining rates. Again, here the discarded rates could be used as a metric for estimating further characteristics of the connection, as with the previous method.

12.1.3 Novel Bandwidth Estimation Techniques

The biggest resource consumer in the system is the TCP throughput measurement, which on average can use up to 8 Megabytes per download test, with a 5 Mbit/s connection limit. This is a lot of resources to ask users to provide, and since the throughput parameter is the most descriptive performance parameter, novel and alternative techniques are needed.

This was the main motivation behind Part II, where two alternatives to the TCP throughput measurement method are proposed and tested, for possible implementation in the NetMap system. The *Packet Pair Technique* presented in Chapter 9 is based on the *Probe Gap Model-based* method, which exploits the information of the interarrival time between two successive probe packets at the receiver.

Another technique was also tested in Chapter 10, the *Combined PGM and PRM Method*, which uses a combination of both the *Probe Gap Model-based* and *Probe Rate Model-based* methods, where the latter method uses the concept of self-induced congestion.

Both techniques were implemented on mobile devices and tested on a real cellular network, meaning that they both have a server and a client, where the client was designed to be deployed onto mobile devices running Android OS.

Packet Pair Technique (PPT) The PPT showed good properties and results, as the technique reduced the amount of data created on the network by up to 90% of what the TCP download throughput test used, and 60% less of what the upload throughput test did. Because of the design of the technique the amount of data induced was always the same, where for the TCP throughput method the data usage depends on the rate limit of the subscription, where higher rates would produce more data during the 15 second run time. The PPT method uses UDP where it sends a series of burst sequences, that each contain a predefined number of packets. It is a bit problematic comparing the two protocols as they behave differently and are used for different purposes, with UDP being a simple and fast and TCP being a reliable protocol. From the fact that the methods where implemented on Android devices, and tested on real cellular networks, a series of factors influenced the tests which were noted as sources of error.

To determine the accuracy of the PPT, its results were compared to corresponding results from the TCP throughput test. From the all the samples taken it was seen that the PPT in many cases provided the same estimate as its counterpart, the TCP throughput method. Two statistical filtering methods were presented in Section 9.6 to estimate the available bandwidth, *Combined Burst Array (CBA)* and *Average Per Burst (APB)*. The APB method, where an estimate from each burst sequence was calculated and then the median from each burst estimated, showed the best properties and accuracy when compared to the results of the TCP throughput method. However, because the technique does not incorporate any retransmission or feedback during the test runs, and because it uses UDP, it has a high probability of losing packets during transmission. From this it was found that there was a fine line between transmitting too many packets, and transmitting just enough to create a bottleneck on the network path, as transmitting too many packets simply lead to the packets being dropped.

Based on the accuracy of the method compared to the TCP throughput method, and due to the very low data usage while estimating the bandwidth, the PPT is evaluated as a good alternative to the TCP throughput method.

Combined Probe Gap Model (PGM) and Probe Rate Model (PRM) Technique The Combined PGM and PRM Technique was tested in a similar manner as the PPT, where a series of small scale tests were conducted using an Android device and a cellular data connection. The results showed that the technique in its current state has a limited potential, as to replace the already existing TCP throughput estimation method. Like for the PPT, the combined technique did reduce the amount of data traffic induced on the network significantly, however as the Combined Technique heavily relies on packets being transmitted and received with exact timing, it proved problematic when used in a cellular network environment. The technique was tested both on a WiFi and a cellular network connection. The results obtained using the WiFi connection indicated that speeds no higher than 3 Mbit/s could be estimated, which was explained by the fact that when the system needs to transmit packets at a high frequency. E.g. if trying to estimate a 20 Mbit/s link using Equation (10.2), it would require packets to be sent and received with a 0.38 ms interarrival time in the start of the measurement.

However, the technique did produce close estimates for the available uplink bandwidth when compared to the results obtained from the TCP throughput test. One of the issues with this particular technique was the timing of packets, as mentioned before. It was also seen that when transmitting packets using a cellular connection the packets showed a tendency to cluster in intervals of approximately 10 milliseconds, but this was however not the case for the tests performed on a WiFi connection. This clustering of the interarrival times of packets are expected to be caused by the Transmission Time Interval (TTI), which is different for the various mobile communication technologies. It causes large uncertainty, as it skews the accuracy that packets can be sent and received with. The TTI for the cellular technologies considered are between 2 and 20 ms, with the most common being 10 ms for UMTS or 3G transmission. For the technique to be generally useful, it should be capable of both measuring speeds higher than what is currently possible, and circumventing the constraints created by the TTI.

Because of the limitations of the technique, it is currently not considered a good alternative to the TCP throughput method.

12.2 Final Conclusion

Following are the conclusions that will attempt to directly answer the problem description, which was stated in Section 2.3, page 5.

12.2.1 Impact of Movement Speed, Area Type and Device Model on Performance

Based on the analysis performed in Part I, it can be concluded that movement speed only has a little impact on the performance of a connection, causing the performance to decrease slightly when moving faster in the interval from 0 to 100 km/h. The little impact that is present is assumed to mainly be caused by the change in how often handovers between cell towers or antennas are performed, when moving faster.

From the analysis on the impact of area type, it can be concluded that the small town area type is where the best performance can be achieved, which is assumed to be due to a good combination of a low number of users and sufficient network resources available.

Lastly, from the analysis on impact on the performance caused by devices, it was seen that devices in different price ranges caused different performance measurements during a 24 hour measurement.

12.2.2 Alternative Bandwidth Estimation Methods

From Part II it can be concluded that alternative methods to the TCP throughput measurement method exists, and that they are deployable to the NetMap system, exemplified with the Packet Pair Technique (PPT) and the Combined Probe Gap Model (PGM) and Probe Rate Model (PRM) Technique. The accuracy of the methods were evaluated by comparing their results with results from the TCP throughput estimation method, and it was seen that the PPT method was capable of providing a sufficient estimate with low data usage, while the Combined Technique showed limitations to speed as well as accuracy.

From this it is concluded that the PPT can be regarded as a suitable replacement to the TCP throughput estimation method in the NetMap system.

Chapter 13

Discussion

In this chapter the conclusions and results obtained throughout the project will be discussed.

Discussion of the Analysis of Measurements

In Chapter 4 the measurements were analyzed for impact of movement speed and handovers, and in Chapter 5 the same measurements were analyzed for the impact of the area type they were gathered in. To some extend it could be argued that the movement speed and area type have some overlap, as in small town and urban areas there typically will be more occurrences of slow movement speed than in rural areas, due to the speed limits in these areas. This can be supported by the throughput rates assigned according to the 4 big movement speed intervals, which was done in Figure 4.12, page 42, and compared to the throughput rates assigned according to area type, seen in Figure 5.3, page 57. If it is assumed that in the small town and urban areas the movement speed typically is between 0 and 65 km/h, and that rural areas typically are from 65 km/h and up (also including express ways), then the connection between performance in different area types and at different movement speeds is noticeable.

In Chapter 4 the CIDs and the handovers between these were analyzed, and it was seen that the CID handovers did have some negative impact on the performance of the connection. But to get a more precise conclusion of the impact on the performance, further analysis of this should be done. This could for instance be done in an area where there only are 2 cell towers, such that it is ensured that the only handovers occurring are between these 2 towers. Then by repeatedly moving through the handover area and forcing the handovers, the direct impact on the performance could be obtained.

In Chapter 6 the difference in performance of various devices was analyzed, based on measurements performed while the devices were stationary. These measurements should, to fully understand the performance of the devices, be accompanied by measurements performed by the same devices while moving. In doing this it would be possible to see if the devices behave differently when also performing handovers, and if handovers happen at the same locations for different devices. This could be an interesting factor, as it might say something about how well the various antennas receive and transmit the signals, and thereby, when the devices are forced to perform handovers.

Discussion of Novel Bandwidth Estimating Methods

From the work with the novel bandwidth estimation methods it is apparent that the PPT method is the only one actually fulfilling the goal of both estimating cellular network connections satisfactory, compared to the TCP throughput estimation method, and reducing the data usage significantly. However, the method of evaluating the performance by comparing it to the current TCP throughput estimation method has some strengths and weaknesses.

Of the strengths is that both are real implementations, and that measurements are performed on real networks, that are being compared, as opposed to simulating parts of the setup. This ensures that the method actually will work when applied to a real scenario. But that the comparison has been performed in this manner also entails a number of uncertainties, in that the network is not controlled, why it is possible that the performance of it will fluctuate, causing the compared results to be unequal, as it is not the same that is measured.

That the accuracy of the method is based on the degree of similarity with the other method, is based on an assumption that the TCP throughput method is very close to the true max value. This means that it is assumed that TCP, by the use of congestion avoidance, keeps the utilization of the connection as close to full as possible, which, based on Figure 7.12, page 78 showing a TCP throughput sequence, seems like a fair assumption. But as it is also seen in this figure, the rise time in the beginning of the measurement causes the final result to deviate from the actual maximum throughput, which should be taken into account when evaluating further measurements.

However, the fact that the TCP throughput and the PPT uses different techniques in estimating the bandwidth, and that they also use different protocols (TCP and UDP), could make the comparison problematic. First of all the, UDP and TCP protocols are designed and behave differently. Secondly, the two protocols could be handled differently by the network, or the service provider, why estimation results might be different.

From Section 8 it can be argued that estimating the available bandwidth and measuring the maximum throughput gives comparable results, as both try to maximize utilization of a channel using different approaches.

Chapter 14

Future Work

In this chapter a number of areas of future work will be suggested, which are evaluated to be necessary in the further development of the NetMap project. Some of these have also been mentioned in [MTP12].

NetMap System and Next Generation Networks

Currently the NetMap system performs measurements on 2nd and 3rd generation cellular networks, but with most providers and mobile devices now offering Long-Term Evolution (LTE) access, this should be considered as well. LTE offers much higher throughput rates than the previous generation technologies, with speeds up to 70 Mbit/s, but still using payment plans that only allow for a limited data usage per month. For most of the measurements performed by the NetMap system this will not be an issue, but as the bandwidth measurement uses TCP bulk transfer the data usage with the current implementation can easily surpass 100 megabytes of data per measurement. One way of coping with the higher throughput rates would be to limit the TCP measurements to run for less time, or lower the frequency at which the measurements are performed.

Correctness of the Measurement Methods

As mentioned earlier throughout the report, the topics concerning the correctness of the data and the method for gathering the data are of great importance. If the measurements are biased to any degree it would deem the data invalid, e.g. if the measurements are biased towards some carriers they can not be compared to each other.

Currently all the conclusions have been made from data gathered using real cellular networks. This means that there are some influencing factors in the tests conducted throughout the project which can not be controlled. One way of telling whether or not the measurement methods are correct is to tests them in a controlled test environment. A physical test setup can be used to control the known influencing factors. Details about the physical environment and cross traffic can be ignored, and a link capacity can be provided which the application then must try to estimate. Currently such a system will require a real scale implementation of a cellular network, including signal modulation and handovers between cell towers. Such a test setup is both complex and expensive, which is why an implementation using simulation tools would be more preferable. Tools such as NS-3 [ns313] and OMNeT++ [omn13] could be used to create such simulations, but will require the dynamics of the considered cellular networks to be incorporated in the simulation framework. Furthermore, the already collected measurements and results could also be used in the setup and development of such a simulation tool.

It was argued during the project that the implementation of the TCP bulk transfer capacity measurement was lacking some details in its measurements, because it only produces an average data rate from the 15 second throughput test, which simplifies the result to some degree. Factors such as TCP rise time and congestion avoidance are not considered in the measurement, but were found to contain valuable information from the analysis performed in the project. This means that in a future release of the application, the intermediate test results should be logged or sampled when doing the TCP throughput, such that results similar to those displayed in Figure 7.12, page 78 can be recreated.

In Part II two alternative bandwidth estimation techniques were presented. The Packet Pair Technique (PPT) in Chapter 9 displayed some good properties as a replacement to the already existing TCP throughput test. However, the technique was never tested using different measurement scenarios as its TCP counterpart was. The technique should therefore first be exposed to some of the same scenarios as those used in [MTP12] (a predefined measurement route around Aalborg University campus), before comparing the results to determine how accurate the PPT is. Alternatively the technique could be verified using simulations as mentioned previously.

The Future Performance Map and Server Structure

The final goal with the NetMap system is to use the measurements to create a Network Performance Map (NPM). The NPM should consist of a grid structure of *tiles*, with each tile consisting of a series of GPS point associated with the measurement points that fall within the tile. The NPM should be capable of handling large amounts of data, which makes it important that extraction and presentation of data is done efficiently. With the current handling of tiles these issues have not been of concern, as large scale coverage maps (the need for many tiles) and high data density (more data processing time required) are not considered. These potential issues must therefore be considered, and efficient methods of extracting and representing the data must be applied, such that the measurement data can be delivered or presented to end users efficiently. This issue was already listed in the previous project [MTP12], but is still left for future work.

Already in [DMCP12], which described the first release of the NetMap system, it was seen that server scalability was a major concern that should be dealt with in future releases of the application. In the current server implementation no load balancing is considered, which can result in a bottleneck and weak point in the system. If the server, or link to the server, is overloaded with measurement traffic it could induce bias to the measurements conducted by the many clients. A better measurement server infrastructure, which also considers multiple data collection points, must be created before a new public release of the application is possible.

Bibliography

[24N12] 24Nordjyske. Tv interview: Netmap, June 2012.
http://www.youtube.com/watch?v=3BfpIv1rkmc.

[3gg13] 3ggp specifications, 2013.
http://www.3gpp.org/specification-numbering.

[acc13] Access network at wikipedia, 2013.
http://en.wikipedia.org/wiki/Access_network.

[and13] Ber at android developer, 2013.
http://developer.android.com/reference/android/telephony/SignalStrength.html#getGsmBitErrorRate().

[att13] Cell id at at&t, 2013.
http://developer.att.com/developer/tier2page.jsp?passedItemId=3100144.

[cor13] Correlation and dependence at wikipedia, 2013.
http://en.wikipedia.org/wiki/Correlation_and_dependence.

[Cot03] Les Cottrell. pathchirp: Efficient available bandwidth estimation for network paths, 2003.
http://www.osti.gov/energycitations/product.biblio.jsp?osti_id=813038.

[dix12] Danish internet exchange point, may. 2012.
http://www.dix.dk/.

[DMCP12] Martin Sloth Dam, Lars Møller Mikkelsen, Steffen Riber Christensen, and Michael Sølvkær Pedersen. Mismatch probability modelling of central decision process with multiple information providers and empirical network models. Technical report, May 2012.
http://dam.lab.es.aau.dk/.

[For12] The Internet Engineering Task Force. Ip performance metrics, May 2012.
http://datatracker.ietf.org/wg/ippm/charter/.

[fsk12] Forskningsnettet website, may. 2012.
http://www.forskningsnettet.dk/.

[FSW03] I. Forkel, M. Schinnenburg, and B. Wouters. Performance evaluation of soft handover in a realistic umts network. In *Vehicular Technology Conference, 2003. VTC 2003-Spring. The 57th IEEE Semiannual*, volume 3, pages 1979–1983 vol.3, 2003.

[gea12] Géant project website, may. 2012.
`http://www.geant.net/pages/home.aspx`.

[goo12a] Google maps, 2012.
`https://maps.google.com/`.

[Goo12b] Android Google. Android developer, list of telephony services on the device, October 2012.
`http://developer.android.com/reference/android/telephony/TelephonyManager.html`.

[GRT09] E. Goldoni, G. Rossi, and A. Torelli. Assolo, a new method for available bandwidth estimation. In *Internet Monitoring and Protection, 2009. ICIMP '09. Fourth International Conference on*, pages 130 –136, may 2009.

[HZZ+12] Zhiguo Hu, Dalu Zhang, Anqi Zhu, Zhiwei Chen, and Hualei Zhou. Sldrt: A measurement technique for available bandwidth on multi-hop path with bursty cross traffic. *Computer Networks*, 56(14):3247 – 3260, 2012.

[ing12a] ing.dk. Aalborg-app afslører dækningen på det danske mobilnet, June 2012.
`http://ing.dk/artikel/129895-aalborg-app-afsloerer-daekningen-paa-det-danske-mobilnet`.

[ing12b] ing.dk. Ny app afslører forskellige reaktionstider i danske mobilnet, June 2012.
`http://ing.dk/artikel/129891-ny-app-afsloerer-forskellige-reaktionstider-i-danske-mobilnet`.

[ing12c] ing.dk. Popularitet får studerende til at lynopdatere app om mobildækning, June 2012.
`http://ing.dk/artikel/129976-popularitet-faar-studerende-til-at-lynopdatere-app-om-mobildaekning`.

[ing12d] ing.dk. Studerendes app skal lukke sorte huller i næstveds mobildækning, June 2012.
`http://ing.dk/artikel/130016-studerendes-app-skal-lukke-sorte-huller-i-naestveds-mobildaekning`.

[ing13] ing.dk. Ny app skal afsløre de sorte huller i mobildækningen, May 2013.
`http://ing.dk/artikel/ny-app-skal-afsloere-de-sorte-huller-i-mobildaekningen-157123`.

[ios13] ios developer library (apple), May 2013.
`http://developer.apple.com/library/ios/navigation/`.

[ipe13] Iperf, 2013.
`http://code.google.com/p/iperf/`.

[KL01] Mary Baker Kevin Lai. Nettimer: A tool for measuring bottleneck link bandwidth. pages 1–12, January 2001.

[KL13] Stanford University Kevin Lai, Mary Baker. Packet pair technique, April 2013.
`http://static.usenix.org/event/usits01/full_papers/lai/lai_html/node2.html`.

[loc13] Local regression (wiki), 2013.
https://en.wikipedia.org/wiki/Local_regression.

[LWG06] Yang Jinmin Li Wenwei, Zhang Dafang and Xie Gaogang. On evaluating the differences of tcp and icmp in network measurement, October 2006.
http://sourcedb.ict.cas.cn/cn/ictthesis/200907/P020090722621963110682.pdf.

[MTP12] Lars Møller Mikkelsen, Steffen Riber Thomsen, and Michael Sølvkær Pedersen. Measuring network performance & quality using android based devices. Technical report, Dec 2012.

[Nag84] J. Nagle. Congestion control in ip/tcp internetworks, 1984.
http://tools.ietf.org/html/rfc896.

[net12] Netnod website, may. 2012.
http://www.netnod.se/.

[nor12] Nordic infrastructure for research and education website, may. 2012.
http://www.nordu.net/ndnweb/home.html.

[ns313] Ns-3, discrete-event network simulator, May 2013.
http://www.nsnam.org/.

[oci12] Opencellid, 2012.
http://www.opencellid.org/.

[omn13] Omnet++, May 2013.
http://www.omnetpp.org/.

[PDMC03] R. Prasad, C. Dovrolis, M. Murray, and K. Claffy. Bandwidth estimation: metrics, measurement techniques, and tools. *Network, IEEE*, 17(6):27 – 35, nov.-dec. 2003.

[Ros09] Sheldon M. Ross. *Probability and Statistics for Engineers and Scientists*. Acedemic Press, 4th edition edition, 2009.

[rss13] Rssi and ber google code, 2013.
https://code.google.com/p/android/issues/detail?id=9271.

[SKK03] Jacob Strauss, Dina Katabi, and Frans Kaashoek. A measurement study of available bandwidth estimation tools. In *Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement*, IMC '03, pages 39–44, New York, NY, USA, 2003. ACM.

[spe13] Speedtest.net, 2013.
http://www.speedtest.net/.

[SRCQ09] P.A. Shah, M. Rehan, H.M.O. Chughtai, and A. Qayyum. On reducing throughput degradation of tcp connection after vertical handover. In *Multitopic Conference, 2009. INMIC 2009. IEEE 13th International*, pages 1–4, 2009.

[Ste94] W. Richard Stevens. *TCP/IP Illustrated, Volume 1*. Addison Wesley, 1st edition edition, 1994.

[Tel12] Teletekniskselskab. Internet quality more than bandwidth, June 2012.
http://teletekniskselskab.wordpress.com/.

[tel13] Telenor, May 2013.
http://telenor.dk.

[utr13] Utran at wikipedia, 2013.
http://en.wikipedia.org/wiki/UTRAN.

[vmw13] Vmware, May 2013.
http://vmware.com.

[WGS10] M.P. Wylie-Green and T. Svensson. Throughput, capacity, handover and latency performance in a 3gpp lte fdd field trial. In *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*, pages 1–6, 2010.

[wik13a] Asus nexus 7 specs at wikipedia, 2013.
http://en.wikipedia.org/wiki/Nexus_7.

[wik13b] Cell id at wikipedia, 2013.
http://en.wikipedia.org/wiki/Cell_ID.

[wik13c] High speed packet access (hspa), May 2013.
http://en.wikipedia.org/wiki/High_Speed_Packet_Access.

[wik13d] Lac at wikipedia, 2013.
http://en.wikipedia.org/wiki/Mobility_management.

[wik13e] Mcc and mnc at wikipedia, 2013.
http://en.wikipedia.org/wiki/Mobile_country_code.

[wik13f] Samsung galaxy tab 10.1 specs at wikipedia, 2013.
http://en.wikipedia.org/wiki/Samsung_Galaxy_Tab_10.1.

[wik13g] Samsung galaxy y specs at wikipedia, 2013.
http://en.wikipedia.org/wiki/Samsung_Galaxy_Y.

[wik13h] Sony xperia u specs at wikipedia, 2013.
http://en.wikipedia.org/wiki/Sony_Xperia_U.

[win13] Msdn library (microsoft developer network), May 2013.
http://msdn.microsoft.com/library/.

Acronyms

DSL Digital Subscriber Line

PPT Packet Pair Technique

PGM Probe Gap Model

PRM Probe Rate Model

ISP Internet Service Provider

NetMap Network Mapping

NPM Network Performance Map

IETF The Internet Engineering Task Force

OSI Open Systems Interconnection

RTT Round-Trip Time

IX Internet Exchange

ICMP Internet Control Message Protocol

RSSI Received Signal Strength Indication

BER Bit Error Rate

MCC Mobile Country Code

MNC Mobile Network Code

CID Cell ID

APB Average Per Burst

CBA Combined Burst Array

RNC Radio Network Controller

LAC Location Area Code

CSV Comma-Separated Values

MTU Maximum Transmission Unit

FIFO First In, First Out

RFC Request for Comments

RTOS Real-Time Operating System

RRC Radio Resource Control

HSPA High Speed Packet Acces

GERAN GSM EDGE Radio Access Network

TTI Transmission Time Interval

EDGE Enhanced Data rates for GSM Evolution

W-CDMA Wideband Code Division Multiple Access

UMTS Universal Mobile Telecommunications System

LTE Long-Term Evolution

Part IV

Appendices

Appendix **A**

Additional Results - Combined PGM & PRM Technique

In Section 10 a novel technique utilizing a combination of PGM and PRM for estimating the available bandwidth is described. This appendix will show the results that were not shown in the section, and a description of the results will be given. The parameters that were also described in Section 10 are as follows:

- **Alpha** (α): The value that decides how fast the rate changes during probing measurements.
- **Initial Rate** (R_{init}): The initial rate chosen for the measurements given in kbit/s.
- **Packet Size** (p): The packet size used for both loading and probing packets given in bytes.
- **Loading Packets** (d): The number of packets that are transferred using the initial rate to create congestion on the path.

The tests have been performed using a Sony Xperia U, and all tests are performed at the same location. As the measurements are performed at different time instances the tests can not be directly compared, but to get a frame of reference between the results it will still be done loosely. Four tests have been performed in addition to the one described in Section 10 of the report, and the choice of parameters for each of these can be seen in Table A.1.

Test #	**Alpha** (α)	**Initial Rate** (R_{init})	**Packet Size** (p)	**Loading Packets** (d)
1	1.025	1024	1000	5
2	1.05	2048	1000	5
3	1.05	1024	1400	5
4	1.05	1024	1000	10

Table A.1: The different set of parameters that have been used for testing.

The results obtained in the following tests are based on the methods used in Section 10. It is recommended that readers unfamiliar with the methods familiarize themselves with these prior to continuing with the appendix.

Test #1: $[\alpha = 1.025 \quad R_{init} = 1024 \quad p = 1000 \quad d = 5]$:

For the first test the value of Alpha (α) is lowered to 1.025, which decreases the speed at which the probing rate changes over a measurement, as described in Equation 10.1. This evens out the measurements performed over the span of bandwidth considered, and should give better accuracy for estimations closer to R_{init}. The results can be seen in Figure A.1.

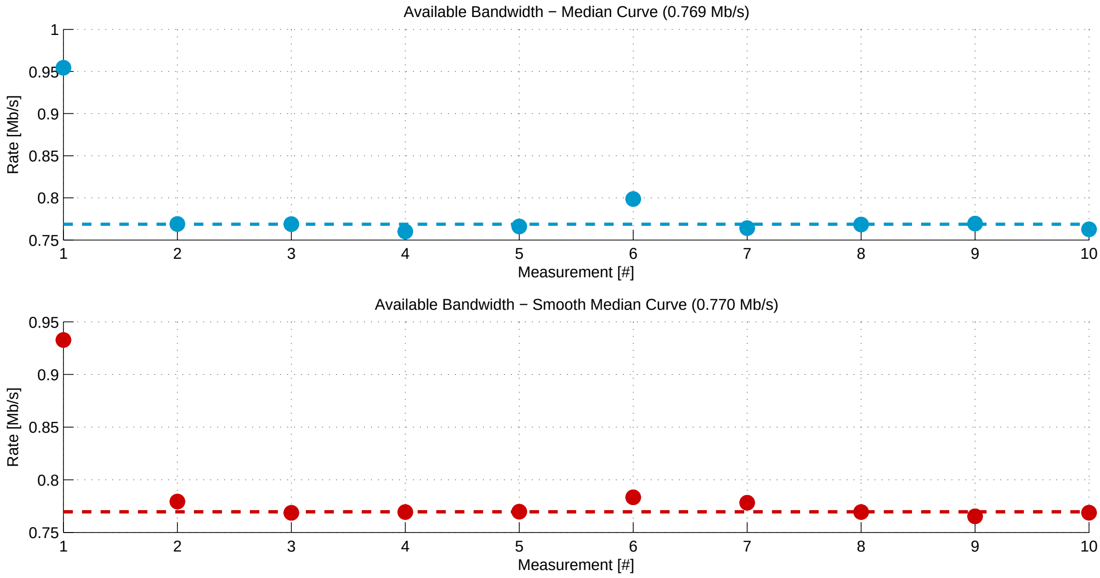


Figure A.1: Results obtained using parameter values defined in Test 1.

The results obtained using a different value of Alpha (α) are not much different from those obtained using the default parameter values. The majority of measurements are almost identical, but measurement 1 is different as the estimation is much higher than for the rest of the measurements. The overall median values are very close to those found using default parameter values, and there is no noticeable difference between the median (blue) and smooth (red) estimations.

Test #2: $[\alpha = 1.05 \quad R_{init} = 2048 \quad p = 1000 \quad d = 5]$:

In this test the initial rate (R_{init}) is doubled such that the loading packets are transmitted with 2 Mbit/s and the probing starts at this rate as well, even though the upload on the cellular network is limited to 1 Mbit/s. By increasing the initial rate the interarrival times between packets will me smaller, which makes the requirement of the system for both accuracy and precision higher. The results can be seen in Figure A.2.

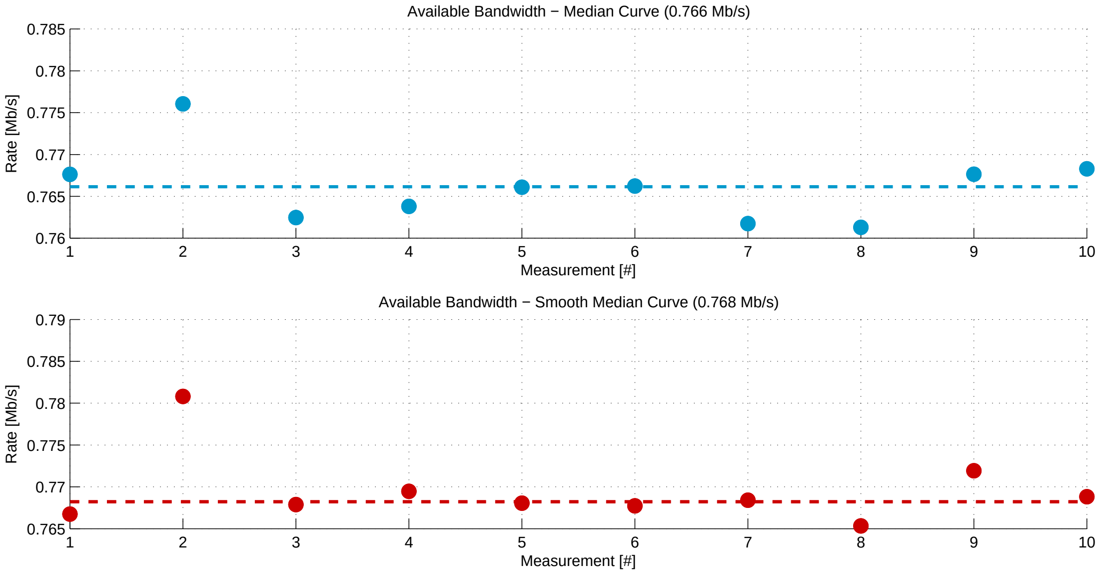


Figure A.2: Results obtained using parameter values defined in Test 2.

As it can be seen in the figure, changing the initial rate (R_{init}) only has little effect on the results when compared to the previous test and the results from Chapter 10. The variation between individual measurements is smaller, as the biggest difference between any measurement and the median is a little over 0.01 Mbit/s for measurement 2 when looking at the results using the smooth curve. The overall median of both methods follow the previous estimations as well, as it is still around 0.76 - 0.77 Mbit/s.

Test #3: $[\alpha = 1.025 \quad R_{init} = 1024 \quad p = 1400 \quad d = 5]$:

With an increased packet size (p) of 1400 bytes, the number of packets that needs to be transmitted each second to reach the required rate is lower. The result of this is a lower requirement for the precision and accuracy, which should have a positive impact on the results which can be seen in Figure A.3.

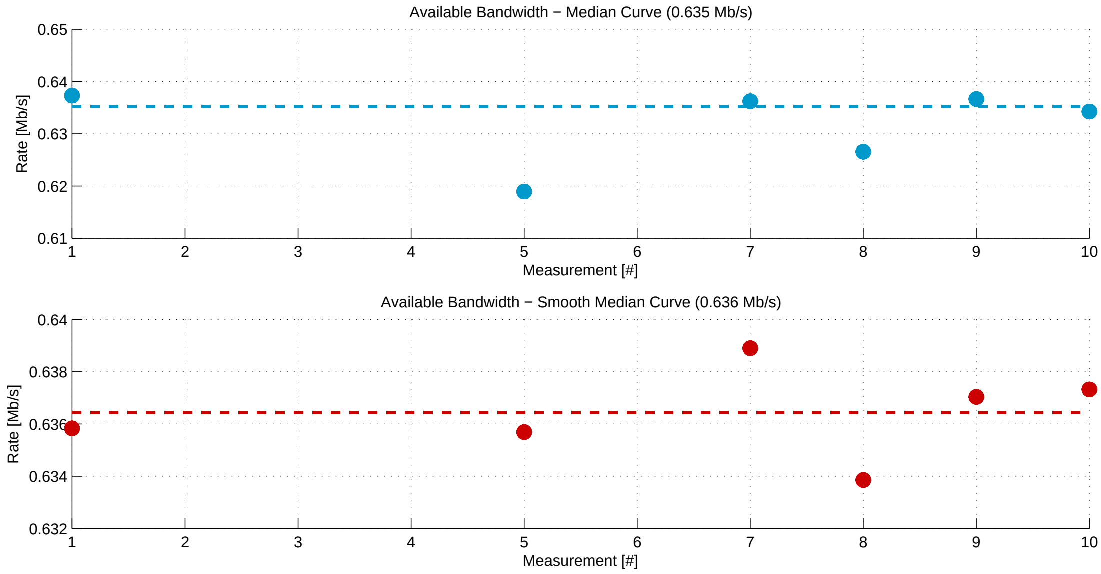


Figure A.3: Results obtained using parameter values defined in Test 3.

As it can be seen measurements 2, 3 and 6 are missing from the results, which is a result of not being able to locate any minima for the measurements. It is expected that either the loading packets do not create a noticeable queue on the network, which could be a result of the loading packets being sent in a burst, or the granularity caused by the TTI hides the minima from the receiver by creating a unfortunate bias to the measured interarrival times. Looking at the measurements that did provide an estimation, a median value of approximately 0.64 Mbit/s is found for both the median and smooth curve, which is a little over 0.1 Mbit/s less than what was seen with default parameter values, as well as in the tests described previously.

Test #4: $[\alpha = 1.025 \quad R_{init} = 1024 \quad p = 1000 \quad d = 10]$:

The last test uses 10 loading packets (d) instead of 5, which is expected to emphasize the packet queuing that should happen before the probing packets are transmitted.

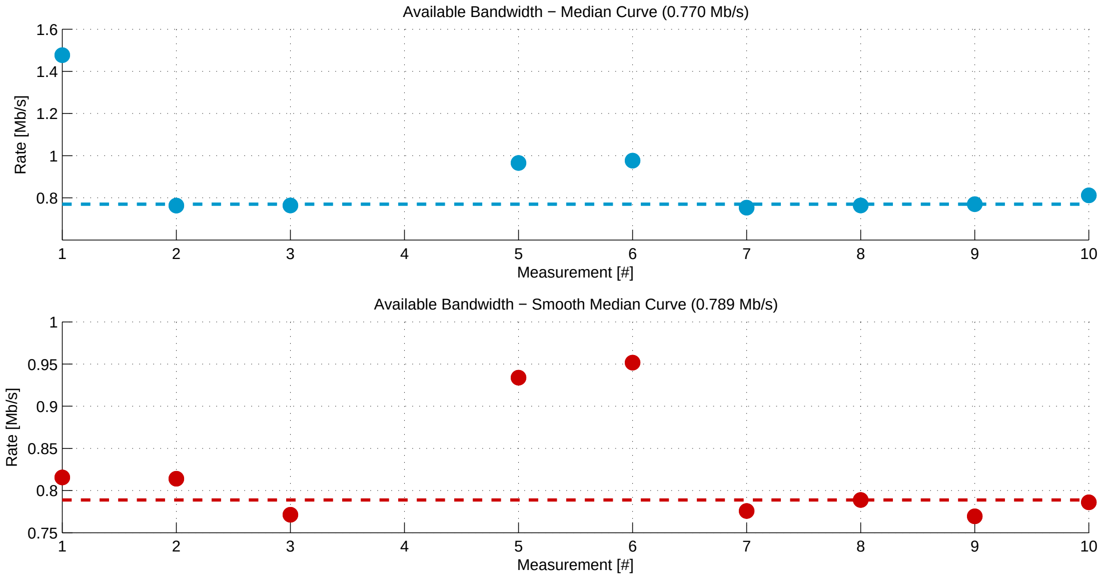


Figure A.4: Results obtained using parameter values defined in Test 4.

It is seen that changing the amount of loading packets (d) has little influence on the results. The overall medians are still around 0.78 Mbit/s, with a few of the measurements varying noticeably from the median. It is worth noting that for measurement 4 no minima could be found, which is once again expected to be a result of loading packets not creating any queue. Following the missing measurement it can also be seen that there are two measurements with higher than average estimates, which could indicate some kind of change to the network during the measurement, which has affected the result.

Appendix B

Applied Statistical and Probabilistic Methods

B.1 Method for Calculating 95% Confidence Interval

[Ros09]The confidence interval is in the analysis found by assuming the data to be normally distributed around the true mean, in a tile, at a movement speed, etc. The 95% confidence interval of the true mean of the data is then as stated in Equation B.1.

$$\mu \in \left(\bar{x} - t_{\alpha/2,n-1}\frac{s}{\sqrt{n}}, \bar{x} + t_{\alpha/2,n-1}\frac{s}{\sqrt{n}}\right) \tag{B.1}$$

Where:

$$\begin{aligned}
\mu &= \text{The true mean of the data} \\
n &= \text{The number of samples in the data} \\
\alpha = 0.05 &= \text{The } 100(1-\alpha)\% \text{ center confidence interval} \\
\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i &= \text{The sample mean of the data} \\
s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2} &= \text{The sample standard deviation of the data} \\
t_{\alpha/2,n-1} &= \text{Student's t inverse value of } \alpha/2 \text{ center percentile with} \\
& \; n-1 \text{ degrees of freedom}
\end{aligned}$$

B.2 Correlation Coefficient

[cor13]The correlation coefficient is obtained by first using the covariance function C, as seen in Equation B.2.

$$C\left(x_1, x_2\right) = E\left[\left(x_1 - \mu_1\right)\left(x_2 - \mu_2\right)\right] \tag{B.2}$$

By using the covariance function the correlation coefficient R can be calculated as seen in Equation B.3.

$$R\left(x,y\right)=\frac{C\left(x,y\right)}{\sqrt{C\left(x_1,x_2\right)C\left(y_1,y_2\right)}} \tag{B.3}$$

In practice the correlation coefficient are obtained by using the Matlab build-in function *corrcoef()*, which when *R* = *corrcoef([A])* where

$$A=\begin{bmatrix}x & y & z\end{bmatrix} \qquad \text{then} \qquad R=\begin{bmatrix}1 & R\left(x,y\right) & R\left(x,z\right)\\ R\left(y,x\right) & 1 & R\left(y,z\right)\\ R\left(z,x\right) & R\left(z,y\right) & 1\end{bmatrix} \tag{B.4}$$