Measurements and Analysis of Application-Perceived Throughput via Mobile Links

Markus Fiedler, Lennart Isaksson, Stefan Chevul
Blekinge Institute of Technology
Department of Telecommunication Systems
371 79 Karlskrona, Sweden
{lennart.isaksson,stefan.chevul,markus.fiedler}@bth.se

Johan Karlsson, Peter Lindberg
AerotechTelub AB
Department of Communications
351 80 Växjö, Sweden
{johan.karlsson,peter.lindberg}@aerotechtelub.se

Abstract

Application-perceived throughput plays a major role for the performance of networked applications and thus for user experience. Especially in mobile environments, throughput is limited, which is a challenge for applications and users. On this background, this tutorial paper investigates the process of user-perceived throughput in GPRS and UMTS systems seen over rather small averaging intervals. It discusses the corresponding active measurements mimicking the needs of streaming applications and analyzes results with aid of summary statistics, histograms and autocorrelation coefficients. These results reveal a clear influence of the network, seen from variations and autocorrelation of application-perceived throughput mostly on the one-second time scale. Such an influence has to be considered when choosing the right kind of network for a specific task. To that aim, the investigated metrics will serve as input to a multi-criteria decision process to determine the proper network for a specific service. The results indicate that, applications have to cope with considerably large jitter when trying to use the nominal throughputs. In GPRS, the promised average throughputs are not even reached in downlink direction; instead, a quite large amount of packet loss occurs. With aid of causality arguments for an equivalent bottleneck, bounds for the extra delay of the first packet sent via mobile links can in fact be derived from throughput measurements.

Keywords
Throughput, user-perceived Quality of Service, UMTS, GPRS, higher-order statistics, equivalent bottleneck.

1. Introduction

Networked applications, no matter whether connected in a wired or wireless way, rely upon the ability of timely data delivery. The achievable throughput is a quality measure for the very task of a communication system, which is to transport data in time. This is of particular importance for nowadays trendy *streaming applications* such as TV, telephony (as part of Triple Play) and gaming. Especially the higher throughput offered by (beyond-)3G mobile systems as compared

to earlier generations seems to be paving the way for these types of applications into mobile environments.

Throughput is thus one of the most essential enablers for networked applications, if not the most important one. While in general, throughput is defined on network or transport level (*e.g.* for TCP), the *application-perceived throughput* that is investigated in this paper reflects the perspective of the application, *i.e.* captures the behavior of all communication stacks in-between the endpoints. Streaming multimedia applications require some amount of throughput on a regular basis. For an elastic application such as file transfer, the achieved throughput determines the download time. For situation-dependent services, *e.g.* for Intelligent Transport Systems and services (ITS), short response times are of outmost importance. The background of the investigations summarized in this tutorial is the task of automatically choosing the right type of network for ITS applications, comprising streaming, messaging and interactive services. More information on the related PIITSA (Personal Information for Intelligent Transport Systems through Seamless communications and Autonomous decisions) project and the particular tasks to be solved is found in [10, 24].

Looking at the OSI model, the conditions on layer n affect layer $n+1$, $n+2$, *etc.*. In the end, the application and thus the user is affected by any kind of problem in the lower layers. For instance, traffic disturbances on network level are packet reordering, delays and loss. The latter can be compensated *e.g.* by retransmissions, which on the other hand increases the delay. Non-interactive applications usually cope well with (approximately) constant delays, even if they are comparably large. This is however not the case for interactive applications, where long delays disturb the interaction of both communication partners. End-to-end throughput variations reveal loss and delay variations, but not one-way delays [13, 10]; the latter can be measured on the network level as described in [4]. On the other hand, comparative throughput measurements on small time scales are capable of revealing existence, nature and severeness of a bottleneck [13]. Reference [10] discusses the concept of application-perceived throughput in detail and provides formulae to calculate application-perceived throughput requirements for streaming, messaging and interactive applications based on user patience, processing times and data volumes.

The current tutorial provides a survey on how to measure throughput as perceived by the user and its applications, combining the appealing possibilities of scaling studies with those of comparative end-to-end measurements. Our active measurements of application-perceived throughput values averaged over some short time interval (typically around one second) during an observation window of duration (typically around one minute) followed by a comparison between key higher-order statistics of sent and received data streams clearly reflect impairments that happen on OSI layers 1 to 7. Initially, we avoid interfering traffic in order to get a clear picture of the best-case throughput properties of a mobile connection. Having several connections sharing the same resources, the user-perceived throughput is likely to be reduced [13]. A specific problem of mobile connectivity consists in the fact that Temporary Block Flows have to be set up [2]. The definition an equivalent bottleneck based on end-to-end throughput measurements and a subsequent application of causality arguments allows deriving approximations for such initial delays beyond the unavoidable minimal one-way transit time.

The remainder of the paper is organized as follows. Section 2 places the method amongst related work, focusing on scaling studies and end-to-end measurements as roots for the throughput modeling used in this work. The statistical parameters used to summarize and compare the perceived throughput processes at server (= sender) and client (= receiver) are presented in Section 3. Section 4 provides some background information to the different generations of mobile communications with focus on their nominal throughput offerings. The setup and implementation of the measurements is described in Section 5, including considerations of the above-mentioned initial delay and of warm-up phases. We then investigate the selected summary statistics and estimations of initial delays obtained by active measurements of application-

perceived throughput for UMTS and GPRS in Section 6. There, we also illustrate the impact of the size of the throughput averaging interval. Finally, Section 7 presents conclusions and outlook.

2. Related Work

2.1. Traditional View on Throughput

Traditionally, the notion of throughput (also called bandwidth) is considered as per session, *i.e.* the *averaging interval* ΔT matches the *observation interval* ΔW. This happens with the end of determining file transmission times in rate-shared scenarios such as TCP-based file transfer [33] and the related degree of user satisfaction [6]. In [17, 34], average throughputs per session are used for investigating and classifying the performance of different sessions in wireless scenarios. In general, such studies do not consider temporary variations of throughput *during* the sessions.

2.2. Scaling Studies

The consideration of smaller time scales ($\Delta T \ll \Delta W$) and higher-order statistics offers added value for traffic characterization, which will be exemplified in this section.

The use of throughput averaging intervals of

$$\Delta T_i = \Delta T_0 \, k^i, i \in \mathcal{N} \tag{1}$$

with typical scaling factors $k = 2$ or $k = 10$ became popular in the beginning of the 1990's. The investigation of higher-order statistics (such as variance and autocorrelation) of throughput averaged over ΔT_i allowed the discovery and analysis of scaling phenomena such as self-similarity, multi-fractality and long-range dependence [19, 22, 32]. Long-range dependence is for instance seen from the lag-k autocorrelation coefficient of a random variable having the form

$$\rho(k) \sim c_k \cdot k^{2H-2}\,, \tag{2}$$

where H is the *Hurst factor* that reflects the degree of self-similarity. One typical consequence of long-range dependence is a slowly decaying variance

$$\mathbf{Var}[R_s(\Delta T_i)] \sim c_i \cdot \Delta T_i^{2H-2}\,, \tag{3}$$

where $R_s(\Delta T)$ is a throughput time series built upon averaging intervals of ΔT. Plotting the variance of the throughput versus ΔT on a double-logarithmic scale allows for the determination of H. The existence of areas with different gradients in such log-log plots might point at multi-fractal properties.

In a recent study, Juva *et al.* [18] analyzed data from the Finnish university network (Funet) using averaging intervals between $\Delta T = 1$ s and 5 min. Their analysis is mainly focused on the mean-variance relationship regarding traffic volumes X_n in consecutive intervals:

$$\mathbf{Var}[X_n] = \phi \cdot \mathbf{E}[X_n]^c. \tag{4}$$

The exponent c turns out to be scale-invariant, which is shown by double-logarithmic scatter plots of throughput variance versus average throughput. Depending on the data measured, type of link, and measured interval, $c \in [0.5, 4.0]$. If the traffic was Poissonian, then both ϕ and c were equal to one.

Since the times of ATM, quantiles of throughput distributions have been used for dimensioning purposes [9, 14, 31]. The latter publication focuses on the impact of the time scale on the capacity

required to keep the probability of capacity overrun below a given level (*e.g.* $\epsilon = 1$ %). Assuming a normal distribution, we obtain the required capacity [31]

$$C(\Delta T, \epsilon) = \mathbf{E}[R_s] + \frac{1}{\Delta T}\sqrt{(-2log\epsilon - log2\pi) \cdot \mathbf{Var}[R_s(\Delta T)]}\,. \tag{5}$$

Another interesting observation in [18] is the fact that even with a considerably large averaging interval ΔT, the autocorrelation does not vanish. Thus, even if long-range dependence might not exist, the observed throughput process seems to have a quite distinct after-effect.

2.3. The Network Management View

Interestingly enough, using $\Delta T \ll \Delta W$ has been common practice in network management for quite a long time, but with the limitations that (i) the corresponding time plots are visually inspected, but not analyzed beyond minimal, maximal and average values, and (ii) typical averaging intervals have been rather long ($\Delta T \geq 5$ min).

The 5 min time scale is also of interest in the context of demand modeling and provisioning. For instance, [1] is using a $\Delta T = 5$ min interval when measuring the point-to-point traffic matrix in the IP backbone. Data are analyzed *e.g.* regarding the mean-variance relationship (4) for different demands in the European and American subnetworks. Recently, shorter averaging intervals have attracted interest. The popular open-source network management tool MRTG [23] that originally employed 5 min intervals comes now with a time resolution of 10 s, which matches the capabilities of SNMP-capable networking equipment [5]. Modern network management tools such as InfoSim StableNet [16] and the RMON tool NI Observer [21] support $\Delta T = 1$ s. Reference [3] investigates throughput monitoring for $\Delta T \in$ [50 ms, 2 s], and [20] is using $\Delta T = 100$ ms interval for studies of web traffic variability as Internet routers appear to have corresponding buffering capabilities. Also, [13] identified $\Delta T = 100$ ms as a useful time scale for describing the throughput of a video conference application. As mentioned above, reference [31] considers link dimensioning as a function of the averaging interval of interest *cf.* (5).

Most studies consider *passive measurements* at one point of reference in a fixed network, *e.g.* on a backbone link, which reflects the typical viewpoint of an operator.

2.4. End-to-End Considerations

In order to reflect the viewpoint of the user, a comparison between a sent and a received packet stream needs to be carried out. *Active measurements* sending probing traffic into a network and deducing the available throughput or bandwidth from these measurements (*cf.* [34]) reveal this view, but in general, merely measurement session averages throughput ($\Delta T = \Delta W$) are reported.

Reference [13] combined the passive observation approach using comparably short averaging intervals with the recently described end-to-end view and used higher-order throughput statistics for the identification and classification of bottlenecks. From throughput histogram difference plots, it can be seen whether the network acts as a shared or shaping bottleneck, *i.e.* increases or decreases the burstiness of a packet stream.

The method can be used for both passive and active measurements. The current tutorial extends the concept presented in [13] as follows:

1. mobile systems are focused;
2. the throughput process is observed on application instead of link level;
3. active measurements on hardly loaded systems are performed in order to reveal basic properties of the systems under study; and
4. additional statistical parameters like standard deviation and autocorrelation are considered.

3. Application-Perceived Throughput Statistics

A typical starting point for traffic characterization purposes are *traces*, *i.e.* lists of packet-related information such as

- T_p: the time when packet p was observed at a point of reference;
- L_p: the length of packet p (payload) at a point of reference;
- any other information such as IP addresses, port numbers, *etc.*

In general, the raw data $\{T_p, L_p\}_{p=0}^{k-1}$ need some kind of post-processing such some further condensation of the information and the calculation of statistical parameters in order to extract and highlight effects of interest. In the following, we perform both steps.

As we are particularly interested the traffic flow properties, we focus on a *discrete-time fluid flow traffic model*. To this aim (in a first step) we collect the contributions of packets observed during short averaging intervals ΔT. We treat the first packet ($p = 0$) of the trace as synchronization packet both at sender and receiver, which is observed at T_0, respectively. This is motivated as the receiving application begins to act upon reception of this packet. Then, we calculate the corresponding throughput time series

$$R_{\mathrm{A},s} = \frac{\sum_{\forall p: T_p \in]T_0+(s-1)\Delta T, T_0+s\Delta T]} L_p}{\Delta T} \tag{6}$$

containing $n = \Delta W/\Delta T$ values. As point of reference, we use the application level (index $_{\mathrm{A}}$). On this level, L_p reflects the payload sent by a server application (index$^{\mathrm{in}}$) or received by a receiver application (index$^{\mathrm{out}}$). The time stamp is taken just before a packet is sent or just upon reception. A detailed description of the corresponding procedure is found in Section 5.

The second step consists in calculating selected summary statistics such as average, standard deviation, throughput histograms and autocorrelation coefficients, which is detailed in the following subsections.

3.1. Average application-perceived throughput

Definition:

$$\bar{R}_{\mathrm{A}} = \frac{1}{n} \sum_{s=1}^{n} R_{\mathrm{A},s} \tag{7}$$

A change of this parameter between server ($\bar{R}_{\mathrm{A}}^{\mathrm{in}}$) and client ($\bar{R}_{\mathrm{A}}^{\mathrm{out}}$) reflects missing traffic at the end of the observation interval:

$$L = \max\left\{(\bar{R}_{\mathrm{A}}^{\mathrm{in}} - \bar{R}_{\mathrm{A}}^{\mathrm{out}})\,\Delta W, 0\right\} \tag{8}$$

That share of traffic might be overdue (*i.e.* appear in the next observation interval) or might have been lost. The use of the max-operator is motivated by the fact that there might be overdue traffic from an earlier interval reaching the receiver in the current observation interval, yielding $\bar{R}_{\mathrm{A}}^{\mathrm{out}} > \bar{R}_{\mathrm{A}}^{\mathrm{in}}$. The corresponding loss ratio is obtained as

$$\ell = \frac{L}{\bar{R}_{\mathrm{A}}^{\mathrm{in}}}\,. \tag{9}$$

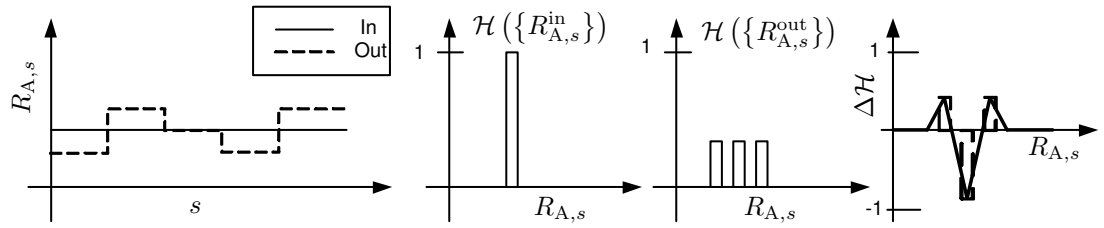


Figure 1. Anticipated time plot, throughput histograms at input and output and throughput histogram difference plot (from left to right) in case of a shared bottleneck [13].

3.2. Standard deviation of the application-perceived throughput

Definition:

$$\sigma_{R_{\mathrm{A}}} = \sqrt{\frac{1}{n-1}\sum_{s=1}^{n}\left(R_{\mathrm{A},s} - \bar{R}_{\mathrm{A}}\right)^2} \tag{10}$$

A rising standard deviation ($\sigma^{\mathrm{out}}_{R_{\mathrm{A}}} > \sigma^{\mathrm{in}}_{R_{\mathrm{A}}}$) reflects a growing burstiness of the traffic between sender and receiver, while a sinking standard deviation ($\sigma^{\mathrm{out}}_{R_{\mathrm{A}}} < \sigma^{\mathrm{in}}_{R_{\mathrm{A}}}$) means a reduction of burstiness. The latter case is typical for a shaper [11].

3.3. Application-perceived throughput histogram

Definition:

$$h_{R_{\mathrm{A}}}(i) = \frac{\text{number of } R_{\mathrm{A},s} \in](i-1)\Delta R, i\Delta R]}{n} \tag{11}$$

If the throughput histogram at the receiver $\mathcal{H}\left(\left\{R^{\mathrm{out}}_{\mathrm{A},s}\right\}\right)$ is broader – in terms of non-vanishing values $h_{R_{\mathrm{A}}}(i)$ when plotted versus $i\Delta R$ – than the one at the sender $\mathcal{H}\left(\left\{R^{\mathrm{in}}_{\mathrm{A},s}\right\}\right)$, the burstiness has increased due to interfering traffic [12]. In the other case, the traffic has been shaped, yielding a more sharp throughput distribution at the receiver $\mathcal{H}\left(\left\{R^{\mathrm{out}}_{\mathrm{A},s}\right\}\right)$. As shown in figures 1 and 2, the *throughput histogram difference plot* $\Delta\mathcal{H}$ with

$$\Delta h_{R_{\mathrm{A}}}(i) = h^{\mathrm{out}}_{R_{\mathrm{A}}}(i) - h^{\mathrm{in}}_{R_{\mathrm{A}}}(i) \tag{12}$$

originally defined in [13] and serving as a *bottleneck indicator* helps to visualize these changes perceived by traffic on its way through a network as follows:

- Shared bottleneck $\Leftrightarrow$ W shape of the bottleneck indicator;
- Shaping bottleneck $\Leftrightarrow$ M shape of the bottleneck indicator.

Compared to standard deviation values, the throughput histograms contain more detailed information about the impact of the bottleneck.

3.4. Lag-j autocorrelation coefficient of the application-perceived throughput

Definition:

$$\hat{\rho}_{R_{\mathrm{A}}}(j) = \frac{\sum_{s=1}^{n-j}(R_{\mathrm{A},s} - \bar{R}_{\mathrm{A}})(R_{\mathrm{A},s+j} - \bar{R}_{\mathrm{A}})}{(n-j)\,\sigma^2_{R_{\mathrm{A}}}} \tag{13}$$

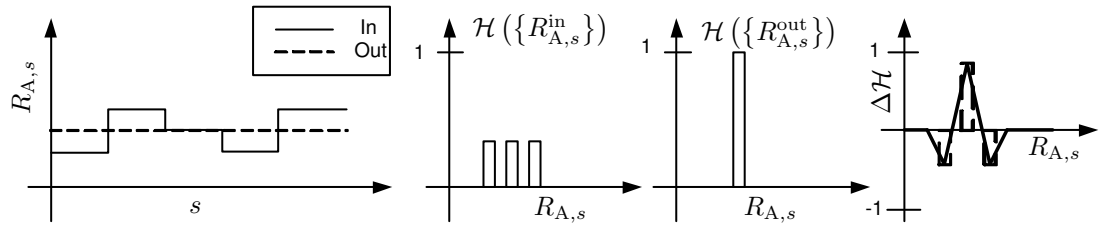


Figure 2. Anticipated time plot, throughput histograms at input and output and throughput histogram difference plot (from left to right) in case of a shaping bottleneck [13].

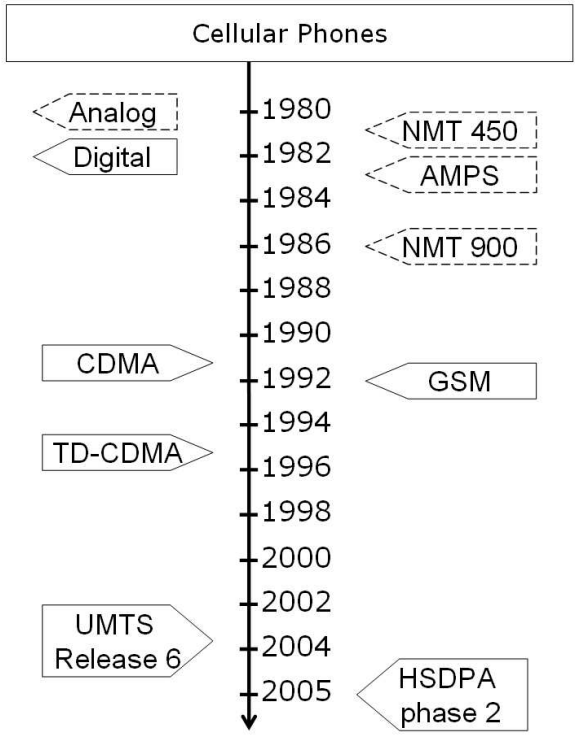


Figure 3. Cellular phones history.

The autocorrelation coefficients allow to detect and compare degrees of after-effects and periodicities within the throughput processes at the server's ($\hat{\rho}^{in}_{R_A}$) and client's ($\hat{\rho}^{out}_{R_A}$) side. Periodicies are revealed by positive spikes of $\hat{\rho}_{R_A}(j)$ when plotted versus j. Changes of the autocorrelation coefficients (from $\hat{\rho}^{in}_{R_A}$ to $\hat{\rho}^{out}_{R_A}$) reflect changes of after-effects and periodicities within the throughput process imposed by the network.

4. Mobile Communication Technologies and Generations

In this section, an overview of mobile communication technologies is given with particular focus on their throughput offerings [7, 8, 25, 27].

4.1 1G (First Generation)

In the 1960s, the Improved Mobile Telephone System (IMTS) was the first 1G mobile phone system with an analogue voice (analogue circuit-switched) technology. This system uses a trans-

mitter of ∼200 W using Frequency Division Multiple Access (FDMA) in the 800–900 MHz frequency band. In 1982 the Advanced Mobile Phone System (AMPS), Total Access Communications System (TACS), Extended Total Access Communications System (ETACS), Nordic Mobile Telephone System (NMT 450), *cf.* Figure 3, and Nippon Telegraph and Telephone System (NTT) were invented with different features, and still some features are used within today's technology.

One of the key features was that a geographic area is divided into so-called cells. Each cell is using a sets of channels/frequencies and the neighbor cells are using an others sets of channels/frequencies to avoid collisions, *cf.* Figure 4 (a). A typical AMPS system uses 832 full-duplex channels/frequencies. This means that ∼45 channels/frequencies are used by each cell. A group contains four cells (A–D), *cf.* Figure 4 (b), or seven cells (A–G), *cf.* Figure 4 (c), each with different channels/frequencies. Several groups are put together so each cell is two cells away to avoid interference between each other, *cf.* Figure 4 (d). A Base Station (BS) is located at the center of each cell. Each BS is connected to a Mobile Switching Centre (MSC) or a Mobile Telephone Switching Office (MTSO).

4.2 2G–2.5G (Second Generation)

The analogue system was changed to a digital system. With the digital technology, better quality and quantity are achieved as compared to the 1G analogue system. New features are added to the 2G digital system: roaming, higher voice quality, several security/encryption levels, and services like Short Message Services (SMS). Different systems are developed for each region: Global System for Mobile Communications (GSM) in Europe, IS-95A CDMA standard (CDMAone) and IS-136 (D-AMPS / TDMA) in the USA, and Personal Digital Cellular (PDC) in Japan. The CDMAone increases the capacity by 8 to 10 times as compared to the AMPS analog system and 4 to 5 times as compared to the GSM system. The digital systems use: Time Division Multiplexing Access (TDMA), Code Division Multiplexing Access (CDMA), or Frequency Division Multiplexing (FDM).

The General Packet Radio Service (GPRS) bandwidth consists of three components; coding schemes (CS), time slots, and GSM bands (1800/1900 MHz or 900 MHz). The generation between the 2G and 3G is called 2G-Enhanced. The 2.5G extends the 2G system with a packet switched system called General Packet Radio Service (GPRS) which provides a data speed of 56–144 kbps, and enhanced data rates for High Speed Circuit Switched Data (HSCSD) and GSM Evolution (EDGE), while the IS-95B is the 2.5G technology for CDMAone.

First, there exist four types of *Coding Schemes* CS $\in \{1, 2, 3, 4\}$ with corresponding error corrections {high, medium, low, none}. Today only the first two are usually implemented due to the implementation cost, *cf.* Table 1. At the Air Interface, the data rates are maximal, *cf.* Table 2. Second, there are 8 slots available at maximum. Given these two conditions, Table 1 provides an overview of nominal throughput values.

Besides the selection of CS and the number of time slots, GPRS standards have stated 29 handset classes. Two of the handset classes are typically implemented, class 4 and class 10. A class 4 handset can only use a maximum of 4 slots, 3 slots for the downlink (3D) and 1 slot for the uplink (1U). A class 10 device can use maximal 5 slots, with the following combinations: 4D + 1U or 3D + 2U, *cf.* Table 3. Classes 13 to 18 have more than 5 active slots. Classes 19 to 29 have up to 8 active slots in half-duplex mode.

There are also three handset classes for devices. Class A handsets are able to send or receive data and voice at the same time. Class B handsets are able to send or receive data and voice but not at the same time. Class C handsets have only one of the two features implemented.

Delay in GPRS is a matter of concern [26]. GPRS does not buffer data as store-and-forward networks do. Instead, GPRS is forwarding the packet as fast as possible. Still the round trip

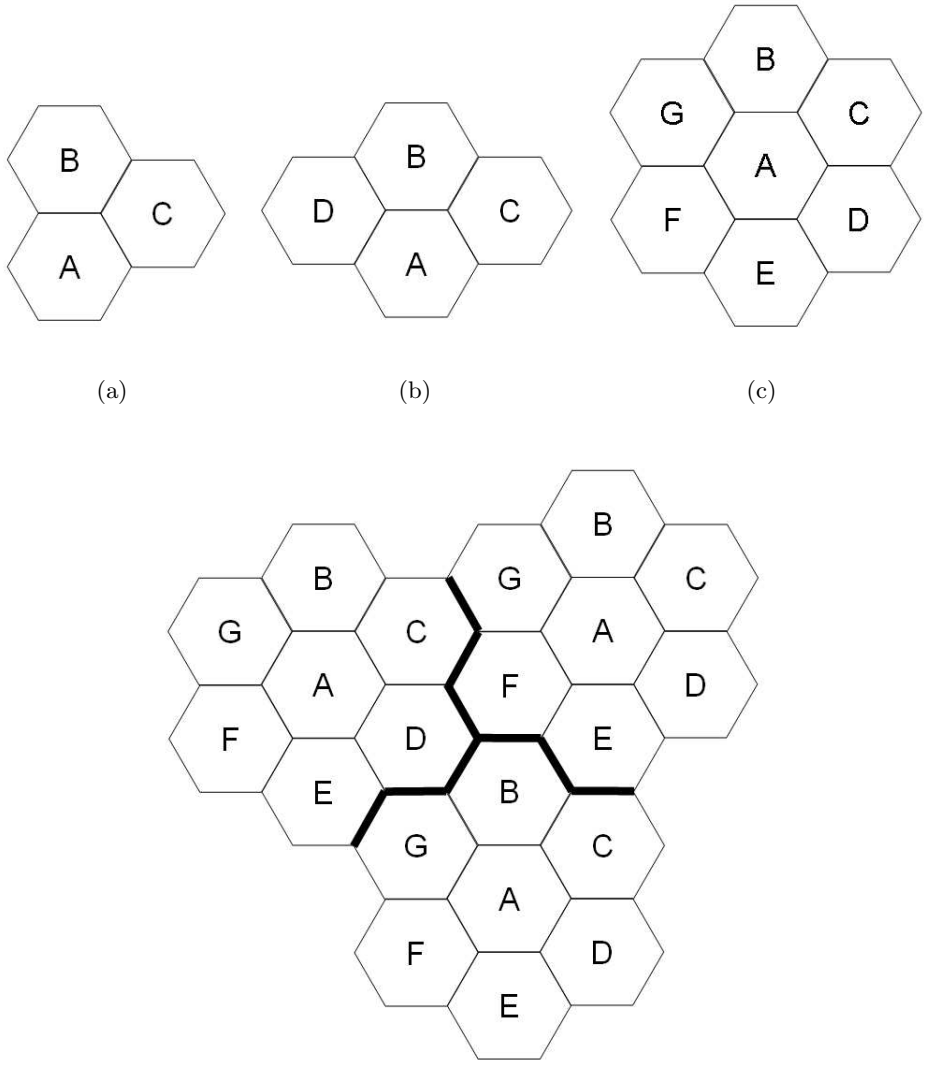


Figure 4. (a) 3-group of cells. (b) 4-group of cells. (c) 7-group of cells. (d) Several 7-groups of cells.

Table 1. Nominal throughput for GPRS at link level.

Coding Scheme	CS1	CS2	CS3	CS4
# Slots	[kbps]	[kbps]	[kbps]	[kbps]
1	9.05	13.40	15.60	21.40
2	18.10	26.80	31.20	42.80
3	27.15	40.20	46.80	64.20
4	36.20	53.60	62.40	85.60
5	45.25	67.00	78.00	107.00
6	54.30	80.40	93.60	128.40
7	63.35	93.80	109.20	149.80
8	72.40	107.20	124.80	171.20

Table 2. Available Air Interface User Rates (AIUR).

AIUR [kbps]
4.8
9.6
14.4
19.2
28.8
38.4
43.2
57.6

Table 3. GPRS handset classes.

Class	# slots downlink	# slots uplink	Max. # slots
1	1	1	2
2	2	1	3
3	2	2	3
4	3	1	4
5	2	2	4
6	3	2	4
7	3	3	5
8	4	1	5
9	3	2	5
10	4	2	5
11	4	3	5
12	4	4	5

time (RTT) is a magnitude higher than in an ordinary fixed network. For delay class 1, a 95 % delay quantile of up to 1.5 s is to be expected. This behavior has to be taken seriously when implementing higher layer protocol or applications. Additionally, GPRS has a jitter problem much worse than in the fixed network. Jitter together with high delay is usually felt quite annoying by an end user.

4.3 3G (Third Generation)

This generation provides more features for the multimedia compared to previous generations together with the important global roaming capability. Two main systems are used: Universal Mobile Telecommunication Service (UMTS) with Wideband CDMA (W-CDMA) in Europe, and CDMA2000 with Multi-Carrier CDMA (MC-CDMA) in the USA. The 3G system is using the 2 GHz band using a data speed up to 2 Mbps, *cf.* Table 4. The Radio Network Subsystem (RNS) is also referred to as UTRAN and consists of the RNC and Node B. These three kinds of operation modes for UTRAN depend upon the duplex technique used. It can be Frequency Division Duplex (UTRA-FDD), Time Division Duplex (UTRA-TDD) and the Dual-mode using both FDD and TDD modes.

The time is divided into radio frames (0–71) of 720 ms in total, and each frame of 10 ms (38400 chip/slot) is divided into slots (0–14). Thus, each slot takes 0.667 ms and includes the Dedicated Physical Channel (DPCH) for the downlink and Dedicated Physical Control Channel (DPCCH) together with the Dedicated Physical Data Channel (DPDCH) for the uplink [28, 29].

The Dedicated Traffic Channel (DTCH) and its channel coding, *cf.* Figure 5, starts at the physical layer with a bit rate of 960 kbps and a spreading factor of 4. Several frames are used with 9600 bits/frame. Each frame is divided into 15 slots which has 640 bits/slot. Each slot is put together and split up into two parts, the DTCH (9525 bits) and the DCCH (75 bits). Finally with turbo coding and CRC the information data ends up with 3840 bits with a data rate of 384 kbps.

FDD allocates two frequencies simultaneously, one for the downlink and one for the uplink.

Table 4. UMTS data rates in different cells.

Cells	Data Rate
Pico cell	2.048 Mbps
Medium size cell	384 kbps
Large macro cells	144 kbps and 64 kbps
Very large cells	14.4 kbps
Speech	4.75 kbps - 12.2 kbps
Satellite	9.6 kbps

The big advantage is that this is full duplex, data can be sent and received simultaneously. TDD does not need to use any guard slots and thus there is no need for time-critical functions like synchronizations between sender and receiver. A drawback is the additional cost which is related to the technique. Also, it's hard to alternate between the size of different bandwidth for a special QoS if this is required. For the FDD the spreading factors are from 256 (15 kbps at the physical channel) to 4 (960 kbps at the physical channel) when using the uplink and 512 to 4 when using the downlink.

TDD allocates only one frequency for both downlink and uplink. The slots used could be allocated dynamically to follow the bandwidth required. This technique requires special equipment to maintain the time synchronizations needed for the frame and slot split. The TDD has two additional options, the 3.84 Mcps and the 1.28 Mcps option. For TDD the spreading factors range from 16 to 1 when using both the uplink and downlink.

The main interest is the QoS perceived by the user. This stretches between the User Equipment (UE) and Core Network (CN), *cf.* Figure 6, which symbolizes the end-to-end service. Different interfaces are connected together to create the UNTRAN network. The Air (Uu) interface uses two different modulation methods Quadrature Phase Shift Keying (QPSK) for the downlink and Offset Quadrature Phase Shift Keying (OQPSK) for the uplink. The difference is that OQPSK applies a 0.5 bit delay in the modulation.

The latest releases are the UMTS phase 6 and the upgraded W-CDMA High Speed Downlink Packet Access (HSDPA) phase 2.

4.4 4G (Forth Generation)

Though higher throughputs as compared to 3G are expected, particular focus lies on the integration of different kinds of wired, wireless and mobile networks. An important enabler is the fact that this generation will be entirely based on the Internet protocol (IP), first on IPv4 and later on IPv6. Particular attention is paid to Voice over IP (VoIP), streaming video and data. This generation is expected to emerge between 2006 and 2010, and probably even later of because the late implementation of the 3G system [30].

5. Measurement Setup

5.1. Parameter Settings

According to the prerequisites of the fluid flow model [12, 13], several packets should be captured in one averaging interval in order to get a differentiated view on throughput variations. As GPRS allows only for a couple of packets to be sent or received per second, an averaging

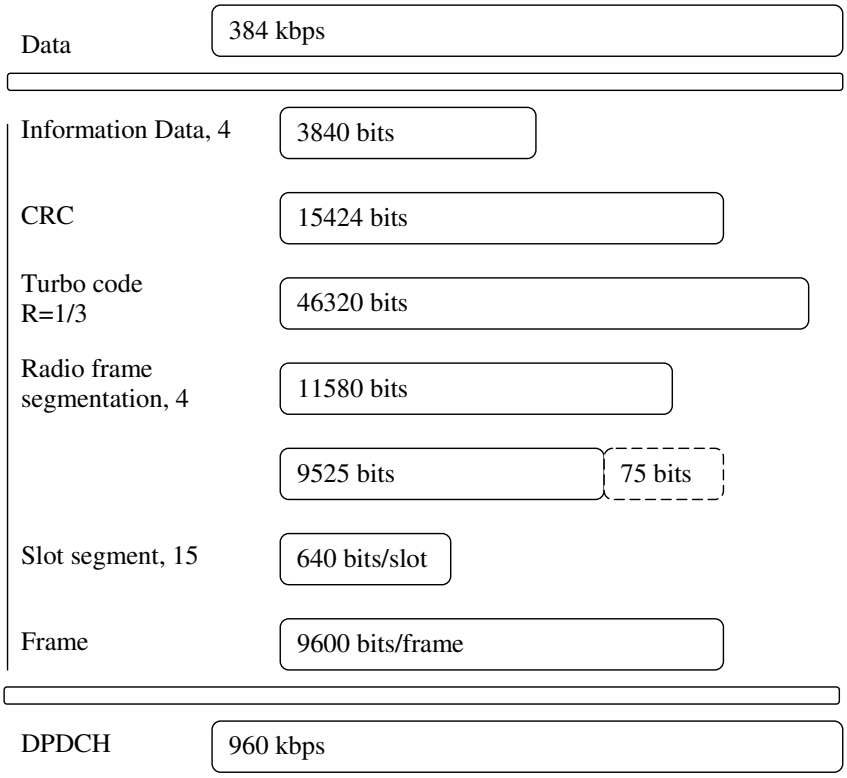


Figure 5. Channel coding (384 kbps).

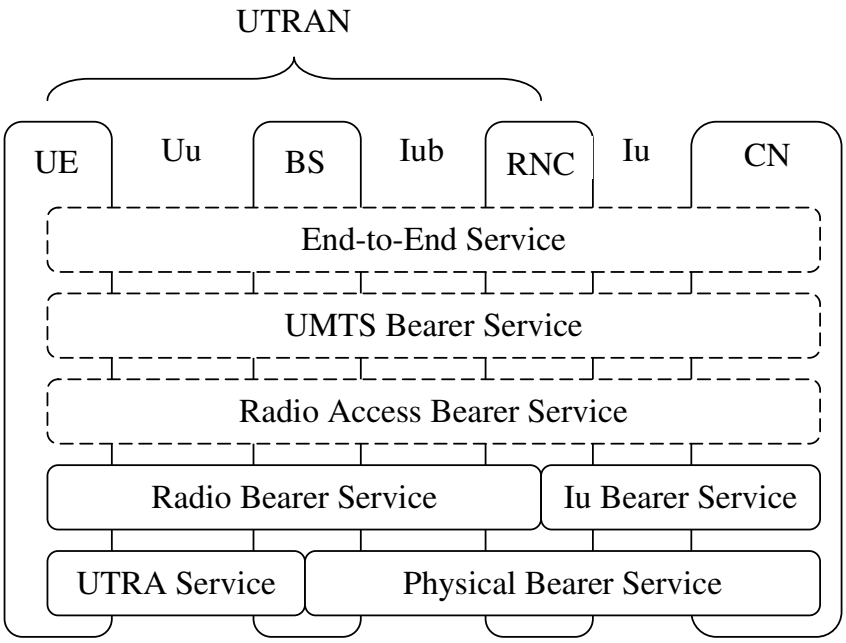


Figure 6. UTRAN architecture.

interval of $\Delta T = 1$ s was chosen. With a typical observation window of duration $\Delta W = 1$ min, the throughput time series to be analyzed consist of $n = 60$ values. In some particular cases, different ΔT values around one second and longer observation windows $\Delta W \leq 5$ min will be applied. Due to the limited resolution in time, time stamping inaccuracies due to the limited exactness of the computer clock should be a minor issue, while the quite short observation interval should damp the potential effect of clock drift.

All throughput histograms in Section 6 will be presented with a throughput resolution of $\Delta R = 1$ kbps and the autocorrelation plots with a confidence interval of 95 %.

5.2. Layer of Interest

When it comes to speed and capacity issues, vendors and providers in general specify the available bit rate on link or physical layer (OSI layers 2 and 1, respectively). However, applications perceive throughput at application layer (OSI layer 7) or even above, which is considered in this work. Due to the overhead introduced by the layers in-between (at least OSI layer 3 and 4 in the Internet context), the average application-perceived throughput is exclusives upper-bounded by the link-layer capacity C_{Link}. Denote Ω_i as overhead introduced by layer i and $\bar{L}$ as average packet length on application layer, we arrive at

$$\frac{\bar{R}_{\mathrm{A}}}{C_{\mathrm{Link}}} = \frac{\bar{L}}{\bar{L} + \Omega_3 + \Omega_4}\,. \tag{14}$$

For UDP/IP, $\Omega_4 = 8$ B and $\Omega_3 = 20$ B, respectively. For an average packet length of $\bar{L} = 128$ B, the average application-perceived throughput is $\frac{128}{128+20+8} \simeq 82$ % of the link capacity; for $\bar{L} = 480$ B, this share rises to 94 %.

In case the (average) offered traffic on application level $\bar{R}_{\mathrm{A}}^{\mathrm{off}}$ exceeds the throughput indicated by Equation 14, a share of

$$\ell = \frac{\bar{R}_{\mathrm{A}}^{\mathrm{off}}\,\frac{\bar{L}+\Omega_3+\Omega_4}{\bar{L}} - C_{\mathrm{Link}}}{\bar{R}_{\mathrm{A}}^{\mathrm{off}}\,\frac{\bar{L}+\Omega_3+\Omega_4}{\bar{L}}} = 1 - \frac{\bar{L}}{\bar{L} + \Omega_3 + \Omega_4} \cdot \frac{C_{\mathrm{Link}}}{\bar{R}_{\mathrm{A}}^{\mathrm{off}}} \tag{15}$$

is lost. In the above mentioned case of $\bar{L} = 480$ B, loss of $\ell \simeq 6$ % is to be expected when the offered traffic on application level matches the link capacity, *cf.* Table 7.

5.3. Initial Delay

One specific property of mobile networks consists in the fact that the first packet experiences an extraordinarily high delay beyond the usual one-way delay due to the need to set up a so-called Temporary Block Flow [2, 15]. As the measurements synchronize on the first observed packet, the receiver starts capturing of the time series $\left\{R_{\mathrm{A},s}^{\mathrm{out}}\right\}_{s=1}^{n}$ too late, yielding quite high throughput values $R_{\mathrm{A},s}^{\mathrm{out}}$ during the first intervals. In the following, we describe how to cope with this problem.

Let us define the network as an equivalent fluid-flow bottleneck as proposed in [13]. The buffer content at the end of interval s is described by

$$X_s = X_{s-1} + (R_{\mathrm{A},s}^{\mathrm{in}} - R_{\mathrm{A},s}^{\mathrm{out}})\,\Delta T\,;\;\; X_0 := 0\,. \tag{16}$$

Obviously, X_s cannot be negative due to reasons of causality: a certain packet cannot be received in an interval prior to when it was sent. Taking this into account and applying (16) in a recursive manner, we arrive at the *causality condition*

$$\sum_{s=1}^{n} R_{\mathrm{A},s}^{\mathrm{in}} \geq \sum_{s=1}^{n} R_{\mathrm{A},s}^{\mathrm{out}} \quad \forall n\,. \tag{17}$$

In case of a late first packet, $R^{\text{out}}_{\text{A},s}$ is typically larger than $R^{\text{in}}_{\text{A},s}$ during the first interval(s), which means that condition (17) is not met. To compensate for this problem, we can delay the time series $\left\{R^{\text{out}}_{\text{A},s}\right\}_{s=1}^{n}$ artificially by $q\Delta T$ by including zero samples in the beginning so that

$$q^* = \min\left\{q : \sum_{s=1}^{n} R^{\text{in}}_{\text{A},s} \geq \sum_{s=1}^{n} R^{\text{out}}_{\text{A},s+q} \quad \forall n\right\}, \; R^{\text{out}}_{\text{A},0}, \ldots, R^{\text{out}}_{\text{A},q^*-1} := 0 \tag{18}$$

From Equation 18, we derive an estimation of the extra initial delay of the first packet of

$$\tau_0^R = q^* \, \Delta T \, . \tag{19}$$

In case the original traces are available, such an estimation can as well be constructed by comparing the cumulative inter-arrival times of the packets in $\left\{T_p^{\text{in}}, L_p^{\text{in}}\right\}_{p=0}^{k-1}$ and $\left\{T_p^{\text{out}}, L_p^{\text{out}}\right\}_{p=0}^{k-1}$, respectively. The corresponding causality condition reads as follows:

$$T_p^{\text{in}} - T_0^{\text{in}} \leq T_p^{\text{out}} - T_0^{\text{out}} \quad \forall p \, . \tag{20}$$

We obtain an estimation of the extra initial delay of

$$\tau_0^T = \min\left\{\tau : T_p^{\text{in}} - T_0^{\text{in}} \leq T_p^{\text{out}} - T_0^{\text{out}} + \tau \quad \forall p\right\} . \tag{21}$$

Comparisons of (19) and (21) showed that in case of considerable loss (8) at the beginning of the observation interval, q^* becomes too small. The reason for this is obvious: the loss reduces the output flow $R^{\text{out}}_{\text{A},s+q}$, and thus, (18) is met "too early" when q is increased. For this reason, we construct another bound by taking $\ell > 0$ into account:

$$q^{**} = \min\left\{q : \sum_{s=1}^{n} R^{\text{in}}_{\text{A},s} \geq \sum_{s=1}^{n} R^{\text{out}}_{\text{A},s+q} + \ell \quad \forall n\right\}, \; R^{\text{out}}_{\text{A},0}, \ldots, R^{\text{out}}_{\text{A},q^{**}-1} := 0 \, ; \tag{22}$$

$$\tau_0^L = q^{**} \, \Delta T \, . \tag{23}$$

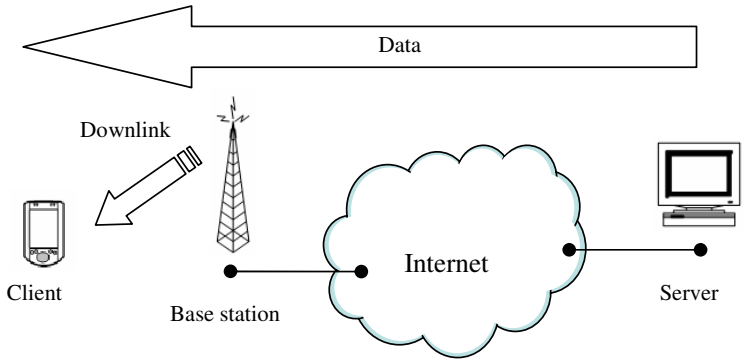
Please note that ℓ is calculated over the whole observation window in order to ensure that this portion of traffic is very likely to be lost. Estimations of τ_0^R and τ_0^L will be compared with τ_0^T in Section 6.5.

5.4. Warm-Up Phase

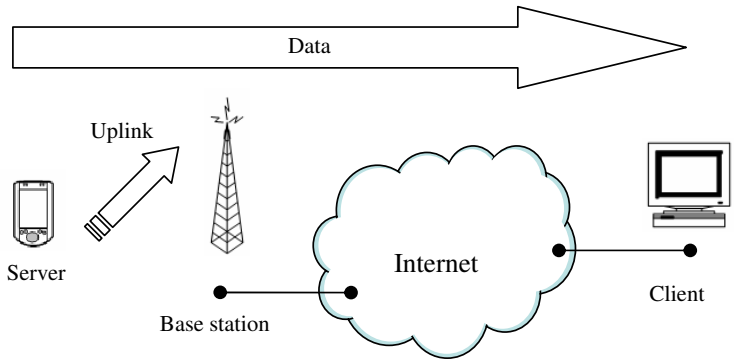
A practicable work-around of the problem of the delayed first packet consists in not considering the first k' (say 100) packets of the trace both at sender and receiver, yielding the traces $\left\{T_p^{\text{in}}, L_p^{\text{in}}\right\}_{p=k'}^{k-1}$ and $\left\{T_p^{\text{out}}, L_p^{\text{out}}\right\}_{p=k'}^{k-1}$ as a basis for analysis, respectively. This way of treating the problem is comparable to the elimination of a warm-up phase in a simulation by not taking the first samples into account. Thus, we observe the some kind of steady-state thoughput process. This work-around was applied except for the study of the initial delay found in Section 6.5.

5.5. UDP Generator

For the measurements of application-perceived throughput, two scenarios were considered: One is called the *downlink scenario*, *cf.* Figure 7 (a), in which the client is connected to a base station (BS) and the server to the Internet via 100 Mbps Ethernet. The other one is called the *uplink scenario*, Figure 7 (b), in which the server is connected to an BS and the client to the Internet via 100 Mbps Ethernet. These names are based on the way the data traffic is directed regarding BS.



(a) Downlink scenario.



(b) Uplink scenario.

Figure 7. Mobile scenarios.

Table 5. Query Performance Parameters in Server and Client Code.

Line	Code
01	[Dllimport("kernel32.dll")]
02	extern static ulong QueryPerformanceCounter(ref ulong x);
03	[Dllimport("kernel32.dll")]
04	extern static ulong QueryPerformanceFrequency(ref ulong x);

The measurements are produced by using a User Datagram Protocol (UDP) generator. The generator is trying to send UDP datagrams of constant length as regularly as possible with a sequence number inside each datagram. The datagrams are not sent *back-to-back*, but spaced in order to yield a certain load. The minimal time the packets are spaced is hereafter called *inter-packet delay*. The software tries to keep this value as closely as possible. However, as blocking calls are used, sending packets too fast as compared to the capacity of the link close to the sender can increase the effective inter-packet delay, *cf.* Sections 6.2 and 6.4.

The software was developed in C# running on both server and client, and these are producing and monitoring the datagram stream. The original time stamp resolution is limited to 10 milliseconds, which is not good enough for our purposes. Thus, specific coding was necessary to improve the time stamp resolution to one thousand of a millisecond. This was achieved by using *performance counters* in conjunction with the system time to provide smaller time increments. To this aim the `kernel32.dll` functions `QueryPerformanceCounter` and `QueryPerformanceFrequency` were used, *cf.* Table 5. For each measurement, the process for each run was set to *realtime* in the Microsoft Windows operating system.

All data are saved in a static vector to create the necessary space at the start-up of the program. If with was neglected memory allocations or hard disc access could jeopardize the timestamping, which should be avoided at all cost. At the end of executions all time stamps in the static vector are saved, *cf.* Figure 8 (right-hand side).

The UDP generator starts with saving the starting time in variable $ctr0$ to be used as an absolute value to every other time stamp in the code, *cf.* Figure 8 (TS 0) and Table 6 (line 01), using the `QueryPerformanceCounter`. All parameters are using *ulong* because the amount of numbers represented by the hardware except for i which has to be done by type-casting.

Before and after each the *inter-packet delay* function a time stamp (TS) is executed, *cf.* Figure 8 (TS 0 to 4) and Table 6 (line 05, 06 and 09). To maintain the nominal inter-packet delay, a while-loop containing an absolute measurement is used for all packets to be sent, *cf.* Figure 8 (TS 0, 2a and 2b) and Table 6 ($ctr2 - ctr0 < delayTimeLong \cdot i$). Together, the *while loop* is being hold until the accumulated inter-packet delay time is reached, *cf.* Figure 8 (TS 2a and 2b). Before and after each *send* function a time stamp is executed, *cf.* Figure 8 (TS 3 and 4) and Table 6 (line 15 and 17). The parameters *rndFile, rndFile.Length, ipAddress, remotePort* are used inside each packet. Finally the datagram is sent out. The time stamp 3 ($ctr3$) is used at the sender side.

6. Measurements and Results

6.1. UMTS Downlink

The first UMTS downlink case presented addresses a modestly loaded link with the nominal inter-packet delay time set to 90 ms, *cf.* Figure 9. In this case, the impact of the network is hardly visible; the average throughput is the same on server and client side, while the standard deviation has grown slightly, still, it is small as compared to the average both at sender and receiver. The throughput histograms are almost identical, which is also shown by the throughput

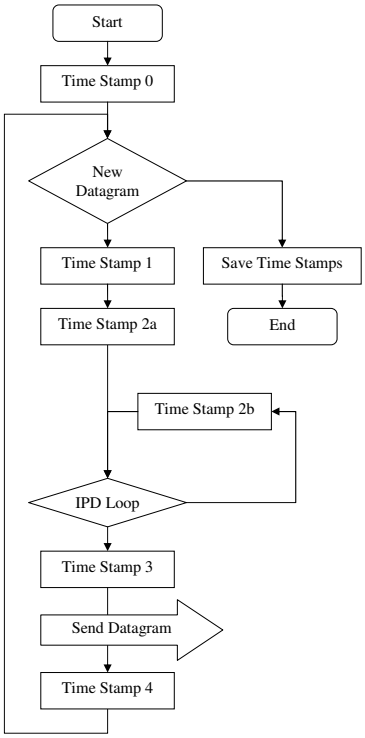
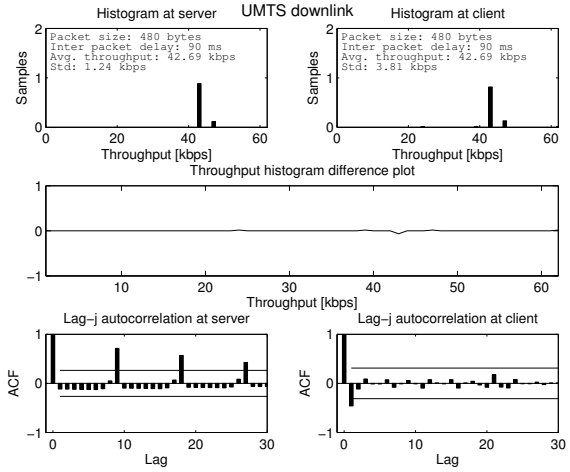


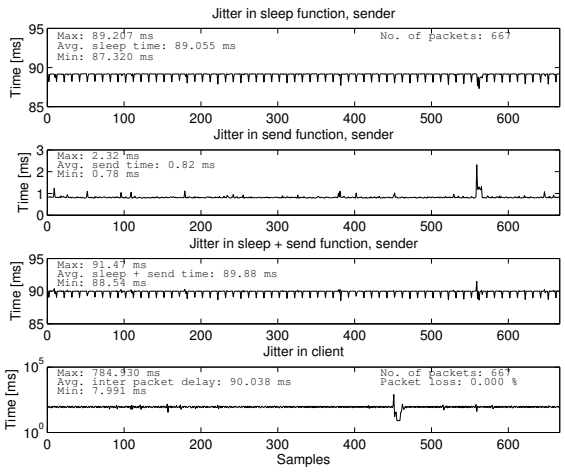
Figure 8. UDP Generator with Time Stamps.

Table 6. Inter-Packet Delay Algorithm in Server.

Line	Code
01	QueryPerformanceCounter(ref ctr0); // Perform this line once at start up
...	...
05	QueryPerformanceCounter(ref ctr1);
06	QueryPerformanceCounter(ref ctr2);
07	while (((ctr2 · 1000000 / freq) − (ctr0 · 1000000 / freq)) < (delayTimeLong · 1000 · (ulong)i))
08	{
09	QueryPerformanceCounter(ref ctr2);
10	}
...	...
15	QueryPerformanceCounter(ref ctr3);
16	s.Send(rndFile,rndFile.Length,ipAddress,remotePort);
17	QueryPerformanceCounter(ref ctr4);



(a) Throughput histograms, difference plot and ACF plots.



(b) Time sample plots.

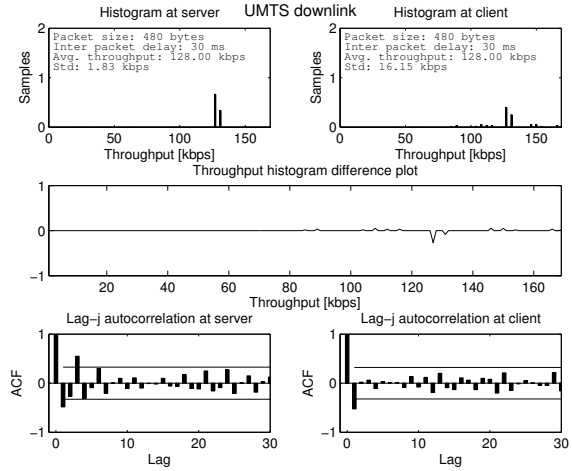
Figure 9. UMTS downlink scenario, 90 ms inter-packet delay.

histograms different plot that is close to zero, *cf.* Figure 9 (a) (middle). At the server side, the autocorrelation plot, *cf.* Figure 9 (a) (bottom-left), reveals a throughput periodicity of about 9 s, which stems from the fact that the averaging interval is not an integer multiple of the inter-packet delay. However, at the client side, this periodicity is less pronounced, *cf.* Figure 9 (a) (bottom-right). Obviously, the network has destroyed some of the original throughput autocorrelation and its structure. The plots from the time domain, *cf.* Figure 9 (b) time stamp 1, shows some rather small amount of jitter. The jitter indicated from the first time stamp is compensated to sustain the throughput. In this case the true IPD is 89.88 ms (second from bottom). Figure 9 (b) (bottom) show the time plot perceived by the receiver. The mean IPD at the client side is slightly larger than the mean IPD at the server, displaying a distinct burst deviation around sample 450. The client does not indicate any packet loss.

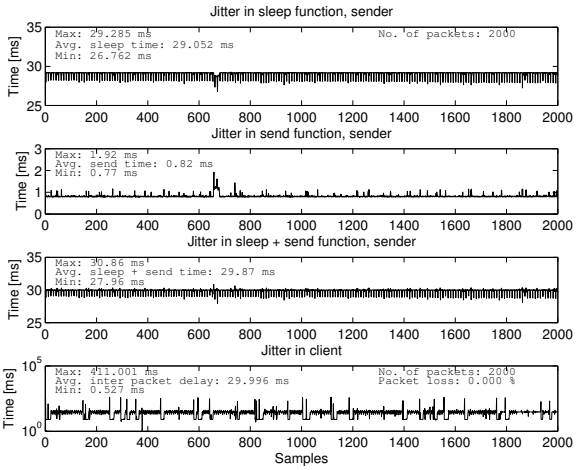
In second case, the nominal inter-packet delay time is set to 30 ms, *cf.* Figure 10. Still the average throughput is the same on both the server and client side, but the standard deviation is much higher at the client side as compared to the server side. The client also starts to perceive more jitter compared to the previous example, *cf.* Figure 10 (b) (bottom). Still, the client does not indicate any packet loss. The autocorrelation structure is again changed significantly by the network.

In the third case, the offered throughput was increased to 384 kbps by reducing the inter-packet delay to 10 ms, *cf.* Figure 10. The server produces a stream at exactly this speed (ΔT is an integer multiple of the nominal inter-packet delay). Consequently, the standard deviation is zero and the autocorrelation undefined. Following the argumentation in Section 5.2, we obviously face an overloaded bottleneck, and data loss amounts to $\ell = (1 - 359.68/384) \simeq 6$ %. However, according to standard deviation and histogram, *cf.* Figure 11 (a) (top-right), the datagrams arrive much more scattered at the client's side, which receives data at rates varying from roughly 175 to 475 kbps. The bottleneck indicator seen in Figure 10 (a) (middle) reveals a shared bottleneck. The autocorrelation, *cf.* Figure 11 (a) (bottom-right), is rather small but irregular. The jitter perceived by the client is also very high together with some packet loss, *cf.* Figure 11 (b) (bottom).

Table 7 contains an overview of all UMTS downlink measurements that have been carried out.

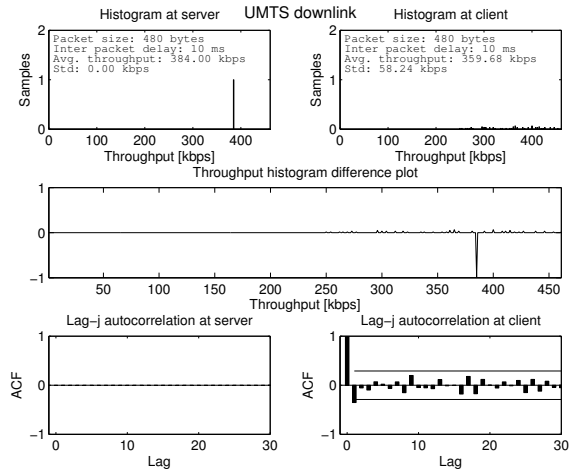


(a) Throughput histograms, difference plot and ACF plots.

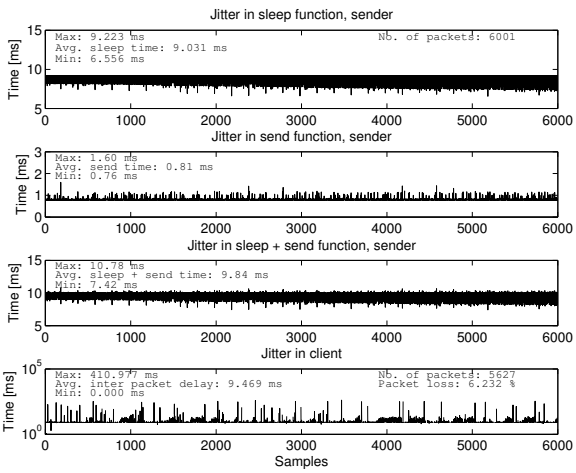


(b) Time sample plots.

Figure 10. UMTS downlink scenario, 30 ms inter-packet delay.



(a) Throughput histograms, difference plot and ACF plots.



(b) Time sample plots.

Figure 11. UMTS downlink scenario, 10 ms inter-packet delay.

Table 7. UMTS Downlink with packet size of 480 bytes.

Nominal IPD [ms]	Average IPD [ms]	$\bar{R}_{\mathrm{A}}^{\mathrm{in}}$ [kbps]	$\bar{R}_{\mathrm{A}}^{\mathrm{out}}$ [kbps]	$\sigma_{R_{\mathrm{A}}}^{\mathrm{in}}$ [kbps]	$\sigma_{R_{\mathrm{A}}}^{\mathrm{out}}$ [kbps]	ℓ [%]
10	9.84	384.00	359.68	0.00	58.24	6.2
11	10.85	349.12	348.42	1.07	68.19	0.2
12	11.85	320.00	319.94	1.83	95.05	0.0
15	14.87	256.00	254.98	1.83	67.30	0.4
20	19.87	192.00	192.00	0.00	55.64	0.0
30	29.87	128.00	128.00	1.83	16.15	0.0
40	39.88	96.06	95.81	2.06	56.80	0.2
50	49.87	76.80	76.80	2.34	4.90	0.0
60	59.89	64.00	63.94	1.83	6.12	0.1
80	79.88	48.00	47.94	1.94	3.71	0.1
90	89.88	42.69	42.69	1.24	3.81	0.0
100	99.89	38.40	38.40	0.00	1.00	0.0

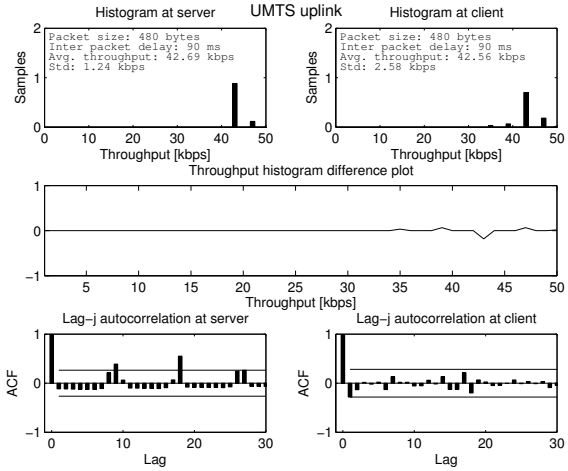
There is a tendency that throughput jitter increases as the offered load approaches the capacity of the downlink (384 kbps).

6.2. UMTS Uplink

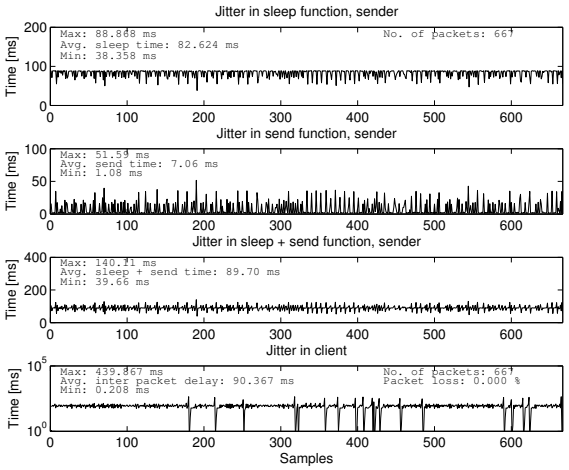
We now apply the same type of overload simulation to the UMTS uplink. The nominal inter-packet delay was set to 90 ms, *cf.* Figure 12. In this case, the impact of the network is visible but small; the average throughput is the same on server and client side, while the standard deviation has grown slightly; still, it is small as compared to the average both at sender and receiver. At the server side, the autocorrelation plot, *cf.* Figure 12 (a) (bottom-left), reveals a throughput periodicity of about 9 s. However, at the client side, this periodicity is less pronounced, *cf.* Figure 12 (a) (bottom-right). No packet loss is perceived. Figure 12 (b) shows that the sender's jitter is hardly damped by the adaptable sleep function. This indication comes from the burstiness of the data sent thought the link.

In the second case, the offered traffic amounts to 480 × 8 bit / 60 ms = 64 kbps. However, the average throughput at the server's side reached only 59.26 kbps, which means an effective average inter-packet delay of 65 ms. A strange peak at 0 kbps occurs in the throughput histogram at the server's side, *cf.* Figure 13 (a) (top-left). Obviously, datagrams were buffered and sending was delayed until there is capacity available on the UMTS link. The server was sending at 0 to 112 kbps; the standard deviation is almost as large as the average throughput. At the client side the average throughput also amounts to 59.26 kbps, which means that there is no missing traffic at the end of the observation interval (0 % loss). The throughput histogram is much more compact, displaying throughputs between 35 and 66 kbps. The standard deviation is reduced almost by factor eight, the bottleneck indicator (Figure 13 (a) (middle)) displays a shaping bottleneck. The autocorrelation is quite small, *cf.* Figure 13 (a) (bottom). Another indication is the time spent in the sleep function is very low, *cf.* Figure 13 (b) (top), which indicates that sleeping time is already consumed by the send function.

Table 8 summarizes results from the UMTS uplink measurements. Again, we recognize the trend that the receiver perceives more throughput variation as the sender approaches the nominal capacity of 64 kbps. However, as soon as this capacity is surpassed, the average inter-packet delay stays below the nominal inter-packet delay, and the throughput at the sender starts to jitter.

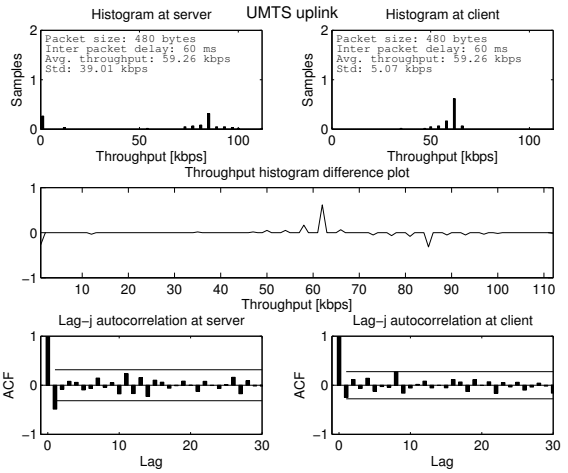


(a) Throughput histograms, difference plot and ACF plots.

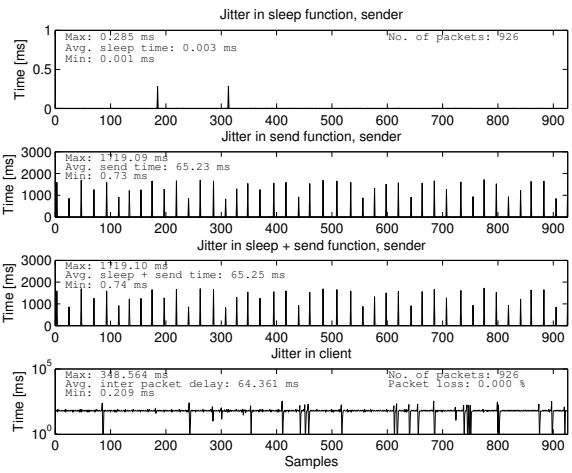


(b) Time sample plots.

Figure 12. UMTS uplink scenario, 90 ms inter-packet delay.



(a) Throughput histograms, difference plot and ACF plots.

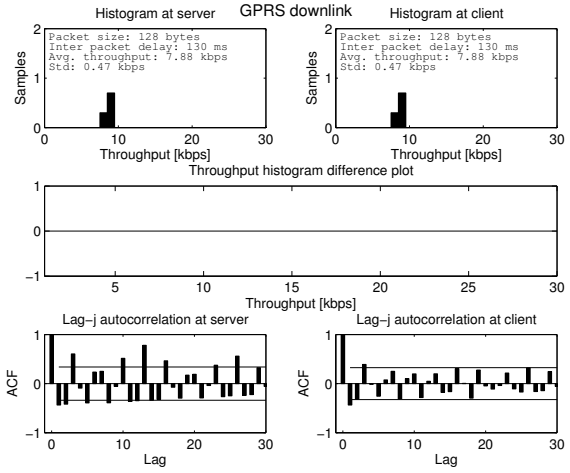


(b) Time sample plots.

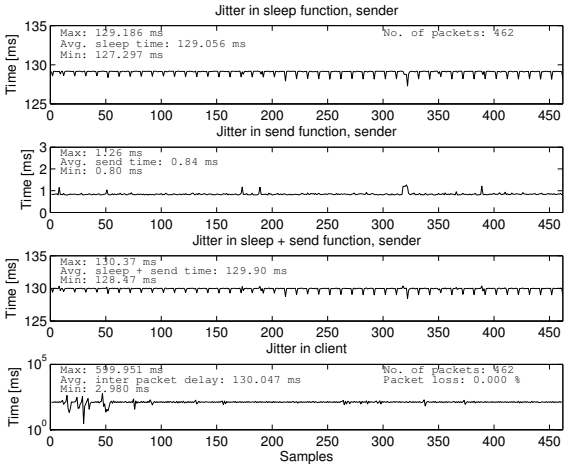
Figure 13. UMTS uplink scenario, 60 ms inter-packet delay.

Table 8. UMTS Uplink with packet size of 480 bytes.

Nominal IPD [ms]	Average IPD [ms]	$\bar{R}_{\mathrm{A}}^{\mathrm{in}}$ [kbps]	$\bar{R}_{\mathrm{A}}^{\mathrm{out}}$ [kbps]	$\sigma_{R_{\mathrm{A}}}^{\mathrm{in}}$ [kbps]	$\sigma_{R_{\mathrm{A}}}^{\mathrm{out}}$ [kbps]	ℓ [%]
50	65.61	58.62	58.62	41.13	6.92	0
60	65.25	59.26	59.26	39.01	5.07	0
70	69.70	54.91	54.91	1.77	11.71	0
80	79.70	48.00	48.00	1.94	5.27	0
90	89.70	42.69	42.56	1.24	2.58	0
100	99.67	38.40	38.40	0.00	4.30	0



(a) Throughput histograms, difference plot and ACF plots.



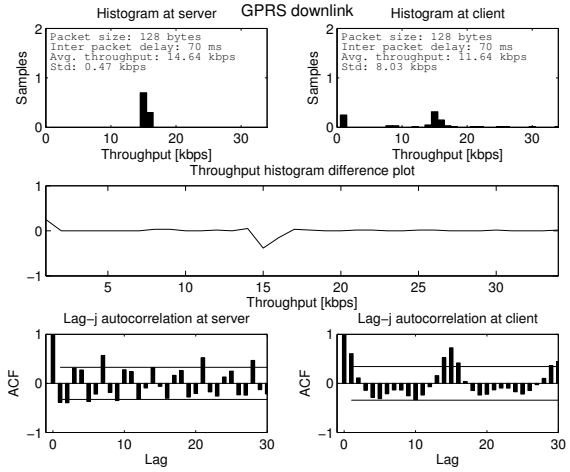
(b) Time sample plots.

Figure 14. GPRS downlink scenario, 130 ms inter-packet delay.

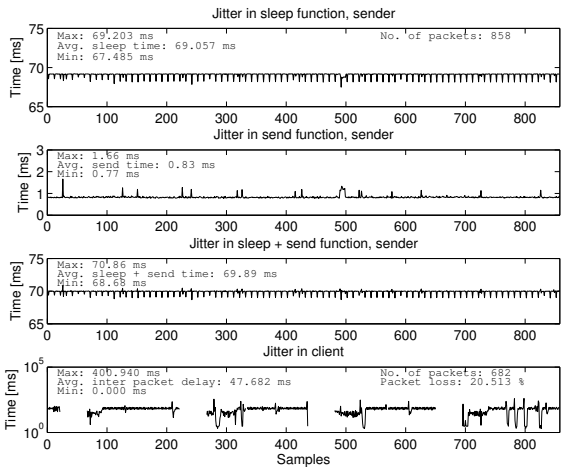
6.3. GPRS Downlink

The first GPRS downlink case addresses a modestly loaded link with the nominal inter-packet delay time set to 130 ms, *cf.* Figure 14. The throughput histograms and the standard deviations on server and client side are equal, *cf.* Figure 14 (a) (top). The autocorrelation at the sender, *cf.* Figure 14 (a) (bottom-left), reflects the non-integer ratio of ΔT and the nominal inter-packet delay. It is slightly damped by the network, but the original correlation structure is preserved. According to Figure 14 (b) the packet loss is zero.

The second case addresses a GPRS downlink with the nominal inter-packet delay time set to 70 ms, *cf.* Figure 15, the server sent with an almost constant throughput of 14 to 16 kbps, yielding an average of 14.64 kbps. However, the client received the datagrams in a scattered way at throughputs varying from 1 to 33 kbps which an average of 11.64 kbps. As in the UMTS downlink case, *cf.* Figure 11, the network introduces additional burstiness. From Figure 15 (a) (top), we can see that the standard deviation grows roughly by factor 16. Figure 15 (a) (middle) reveals a shared bottleneck. Data loss amounts to about 20 % with a period of 9 s, *cf.* Figure 15 (b) (bottom). The autocorrelation (Figure 15 (a) (bottom)) at the server side displays a periodic



(a) Throughput histograms, difference plot and ACF plots.



(b) Time sample plots.

Figure 15. GPRS downlink scenario, 70 ms inter-packet delay.

behavior of 3.5 s, while the client perceives a longer period of about 15 s which is probably due to the overload situation with regular throughput breakdowns. Reference [15] reports on abortive file downloads via GPRS, which might have its roots in the here-described problems.

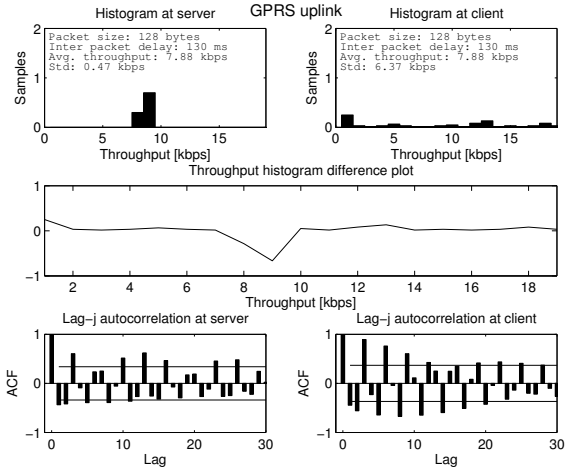
Table 9 provides an overview of different measurements. Quite high losses occur quite frequently. However, no real trend can be seen. The only loss-less case in which the network behaves transparently is the case of a nominal IPD of 130 ms. In all other cases, there is a considerable growth of the standard deviation at the receiver.

6.4. GPRS Uplink

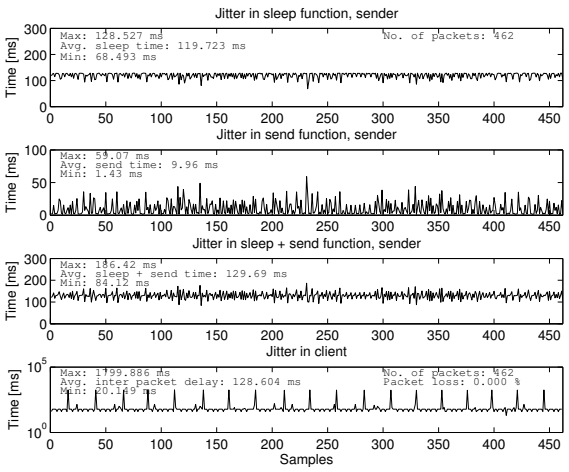
The first GPRS uplink case addresses a modestly loaded link with the nominal inter-packet delay set to 130 ms, *cf.* Figure 16. Already at this load, the packets are received in a bursty way at the client side, *cf.* Figure 16 (a) (top-right); Figure 16 (a) (middle) indicates a shared

Table 9. GPRS Downlink with packet size of 128 bytes.

Nominal IPD [ms]	Average IPD [ms]	$\bar{R}_{\mathrm{A}}^{\mathrm{in}}$ [kbps]	$\bar{R}_{\mathrm{A}}^{\mathrm{out}}$ [kbps]	$\sigma_{R_{\mathrm{A}}}^{\mathrm{in}}$ [kbps]	$\sigma_{R_{\mathrm{A}}}^{\mathrm{out}}$ [kbps]	ℓ [%]
50	49.89	20.50	16.93	0.58	9.67	15.2
60	59.89	17.07	14.47	0.49	10.37	15.3
70	69.89	14.64	11.64	0.47	8.03	20.5
80	79.89	12.80	10.22	0.52	7.01	20.2
90	89.89	11.38	10.80	0.33	3.19	5.1
100	99.87	10.26	9.73	0.61	4.44	5.0
110	109.89	9.32	8.26	0.31	4.75	11.4
120	119.89	8.53	7.22	0.49	5.89	15.4
130	129.90	7.88	7.88	0.47	0.47	0.0



(a) Throughput histograms, difference plot and ACF plots.



(b) Time sample plots.

Figure 16. GPRS uplink scenario, 130 ms inter-packet delay.

bottleneck. The autocorrelation at the server side displays a periodic behavior of 3 s with highly correlated values (Figure 16 (a) (bottom)), data loss is zero.

In the second GPRS uplink case, the nominal inter-packet delay was set to 80 ms, yielding a server transmission rate of 12.82 kbps, *cf.* Figure 17 (a) (top-left). This which is almost matched by the measured averaged throughput, which amounts to 12.77 kbps. At the server and client side datagrams appeared as a scattered stream with throughputs of 1 to 48 kbps at the server and 1 to 20 kbps at the client. The standards deviation is slightly reduced. We observe a shared bottleneck in the histogram difference plot, *cf.* Figure 17 (a) (middle). The channel destroys the 4 s throughput periodicity but introduces some extra low-term correlation, *cf.* Figure 17 (a) (bottom). From Figure 17 (b), we see that there exist situations in which sleeping is not an option. Between samples 125–195 and 230–260, such non-sleeping situations are indicated. Packet loss occurs close to sample 150, with a total packet loss ratio of 0.4 %.

Table 10 summarizes the results of the GPRS uplink measurements. From the rise of the standard deviation at the sender $\sigma^{\text{in}}_{R_\text{A}}$, we can deduct the cases in which an overload situation is given (nominal inter-packet delay between 50 and 80 ms). For longer inter-packet delays, the standard deviation at the receiver $\sigma^{\text{out}}_{R_\text{A}}$ is similar in size. Loss is neglectible.

6.5. Impact of the Initial Delay

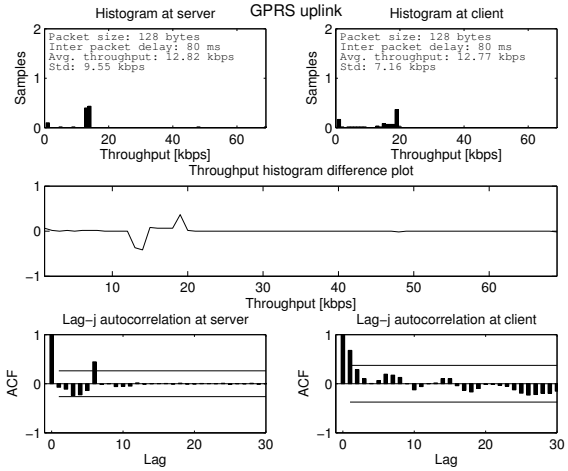
This section visualizes the impact of the initial delay and evaluates the throughput-based criteria presented in Section 5.3.

Figure 18 (a) plots the unshifted throughput time series $R^{\text{in}}_{\text{A},s}$ and $R^{\text{out}}_{\text{A},s}$ (*i.e.* $q = 0$) for the UMTS downlink with 90 ms inter-packet delay and $\Delta T = 0.5$ s. Excessive throughput values are observed at the output in the beginning. Figure 19 (b) plots the content of the equivalent bottleneck

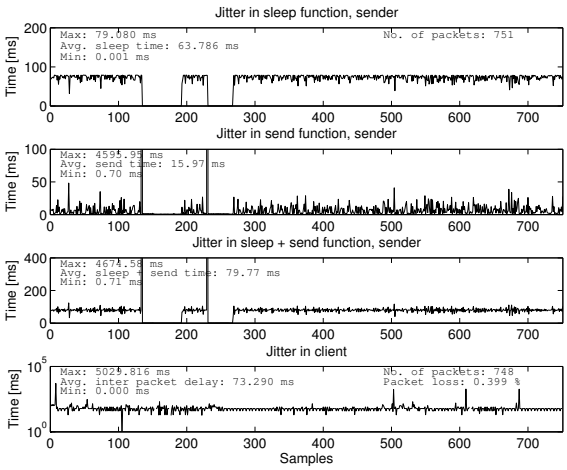
$$X_s = X_{s-1} + (R^{\text{in}}_{\text{A},s} - R^{\text{out}}_{\text{A},s+q})\,\Delta T\,;\ X_0 := 0, q \in \mathcal{N}^0 \tag{24}$$

for different values of q. Obviously, $X_s < 0$ for $q < 3$, while a shift by $q^* = 3$ yields causality ($X_s \geq 0\ \forall s$), which leads us to the estimation $\tau^R_0 = 3 \cdot 0.5$ s $= 1.5$ s (19).

We now look at a case involving loss. Figure 19 (a) plots the unshifted throughput time series



(a) Throughput histograms, difference plot and ACF plots.

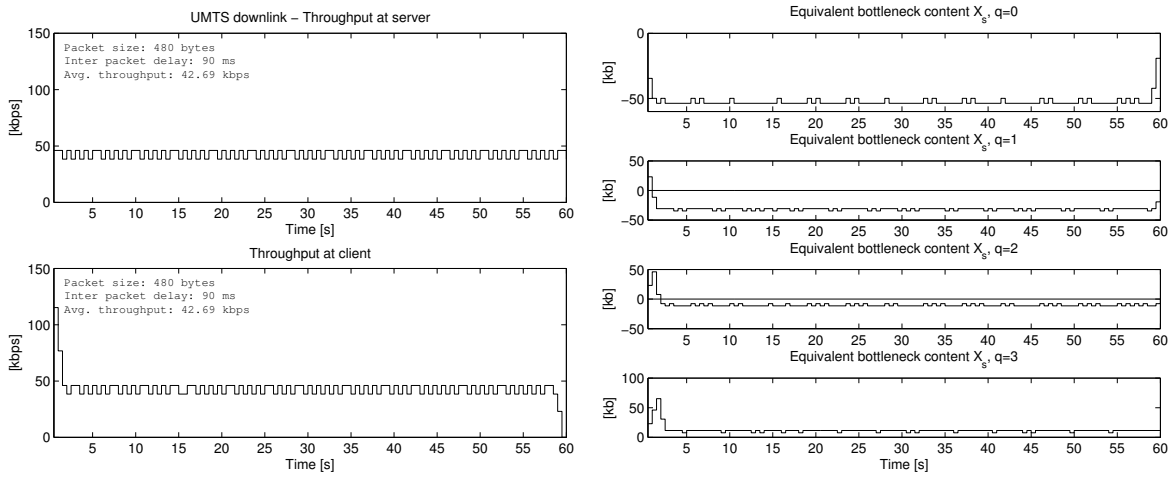


(b) Time sample plots.

Figure 17. GPRS uplink scenario, 80 ms inter-packet delay.

Table 10. GPRS Uplink with packet size of 128 bytes.

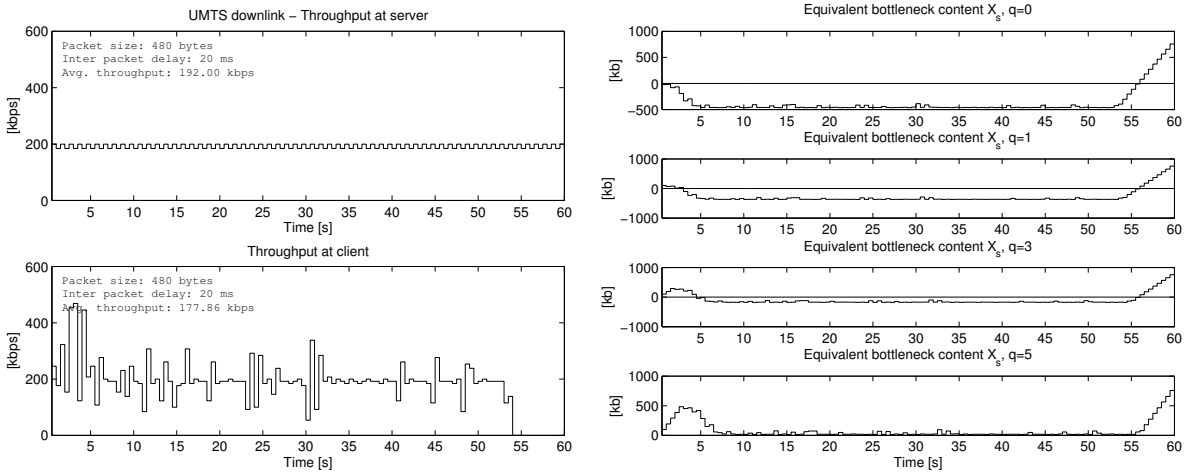
Nominal IPD [ms]	Average IPD [ms]	$\bar{R}_{\mathrm{A}}^{\mathrm{in}}$ [kbps]	$\bar{R}_{\mathrm{A}}^{\mathrm{out}}$ [kbps]	$\sigma_{R_{\mathrm{A}}}^{\mathrm{in}}$ [kbps]	$\sigma_{R_{\mathrm{A}}}^{\mathrm{out}}$ [kbps]	ℓ [%]
50	130.24	8.58	8.58	24.64	2.13	0.0
60	91.24	11.95	11.91	26.46	3.99	0.3
70	104.15	10.02	10.00	24.53	4.55	0.2
80	79.77	12.82	12.77	9.55	7.16	0.4
90	89.67	11.38	11.38	0.33	6.88	0.0
100	99.71	10.24	10.07	0.42	6.40	0.3
105	104.74	9.76	9.52	0.52	6.77	0.2
110	109.72	9.32	9.32	0.31	6.74	0.0
115	114.72	8.91	8.91	0.47	6.62	0.0
120	119.74	8.53	8.38	0.49	6.70	0.0
130	129.69	7.88	7.88	0.47	6.37	0.0



(a) Throughput time series.

(b) Buffer content of the equivalent bottleneck.

Figure 18. UMTS downlink scenario, 90 ms inter-packet delay.



(a) Throughput time series.

(b) Buffer content of the equivalent bottleneck.

Figure 19. UMTS downlink scenario, 20 ms inter-packet delay.

Table 11. UMTS with packet size of 480 bytes.

	Downlink				Uplink			
IPD Nominal [ms]	τ_0^T [ms]	τ_0^R [s]	τ_0^L [s]	Packet Loss [%]	τ_0^T [ms]	τ_0^R [s]	τ_0^L [s]	Packet Loss [%]
10	N/A	N/A	5.0	8.7				
11	4933	2.0	5.0	5.2				
12	5862	2.0	6.0	6.5				
15	4375	2.0	4.5	4.3				
20	6854	2.5	7.0	7.3				
30	6221	3.0	6.5	6.1				
40	5658	3.5	6.0	4.1				
50	5805	3.5	6.0	4.7	N/A	N/A	N/A	0.0
60	6164	3.0	6.0	5.4	3074	3.0	3.0	0.0
70	1051	1.5	1.5	0.0	4115	4.5	4.5	0.0
80	6558	4.0	7.0	5.1	4288	4.5	4.5	0.0
90	1260	1.5	1.5	0.0	3990	4.5	4.5	0.0
100	5320	4.0	5.5	2.4	4088	4.5	4.5	0.0

Table 12. GPRS with packet size of 128 bytes.

	Downlink				Uplink			
IPD Nominal [ms]	τ_0^T [ms]	τ_0^R [s]	τ_0^L [s]	Packet Loss [%]	τ_0^T [ms]	τ_0^R [s]	τ_0^L [s]	Packet Loss [%]
80	1526	2.0	11.0	15.1	181	0.5	0.5	0.2
90	808	1.0	1.0	0.1	1493	1.5	1.5	0.1
100	2074	2.5	8.5	9.9	321	0.5	1.0	0.6
110	1870	2.0	8.0	9.8	345	0.5	0.5	0.0
120	1899	0.5	12.5	19.9	305	0.5	0.5	0.0
130	1187	1.5	1.5	0.0	402	0.5	0.5	0.2

$R_{\mathrm{A},s}^{\mathrm{in}}$ and $R_{\mathrm{A},s}^{\mathrm{out}}$ (*i.e.* $q = 0$) for the UMTS downlink with 20 ms inter-packet delay and $\Delta T = 0.5$ s. Again, quite high throughput values are observed at the client in the beginning, while in the end there is no traffic at all. Figure 19 (b) plots the content of the equivalent bottleneck (24) for different values of q. Finally, a shift by $q^* = 5$ yields causality, which gives $\tau_0^R = 2.5$ s. Please observe the final "staircase" for $q = 5$ visualizing the loss L. Comparing with Table 11, we find that τ_0^R underestimates the time-based initial delay estimation $\tau_0^T \simeq 6.85$ s. This is due to the initial loss. Using the loss-compensated criterion (22), we in fact obtain $\tau_0^L = 7$ s, *i.e.* an upper bound for τ_0^T.

Let us now look at the UMTS-related results presented in Table 11, where N/A stands for cases when the minimum in (18), (21) or (22) could not be found due to overload situations. If there is no loss, $\tau_0^R = \tau_0^L$. Except for some few cases, τ_0^L provides a tight upper bound of the extra initial delay. For UMTS, the anticipation of having the losses collected at the beginning of the interval seems to be accurate.

Some results for GPRS are presented in Table 12. Here, we observe that τ_0^R in most cases provides a good bound for τ_0^T, while τ_0^L overestimates the initial delay especially in cases of heavy

loss ($\ell \simeq 0.1 \ldots 0.2$). This has its origin in a loss process rather different from the UMTS case: Loss is not concentrated to the beginning but appears rather bursty during ΔW, *cf.* Sections 6.3 and 6.4. Still, τ_0^T can serve as a (conservative) upper bound if no further information on the nature of the loss process is available. Again, in case of negligible loss ($\ell \to 0$), both estimations yield the same value $\tau_0^R = \tau_0^L$.

6.6. Impact of the Averaging Interval ΔT

This section highlights the impact of the averaging interval ΔT on the performance measures introduced in Section 3. Figure 20 shows throughput histograms, corresponding difference plots and plots of the lag-j autocorrelation coefficients for the UMTS downlink with an inter-packet delay of 20 ms for different values of ΔT as observed during $\Delta W = 5$ min. Figure 21 illustrates the standard deviations at sender and receiver together with the number of samples as functions of ΔT.

In general, standard deviation and autocorrelation at the receiver get smaller as the averaging interval grows, which means that high-frequent jitter components are averaged out at least to some extent. Also, the throughput difference plots get more narrow as ΔT grows. Interestingly enough, they do not vanish, which means that the network cannot be considered to be transparent on these time scales. In [18], considerable autocorrelation has been observed on a backbone link for $\Delta T = 5$ min. We can conclude that the network can have a distinct long-term after-effect on the throughput of a stream of interest.

7. Conclusions and Outlook

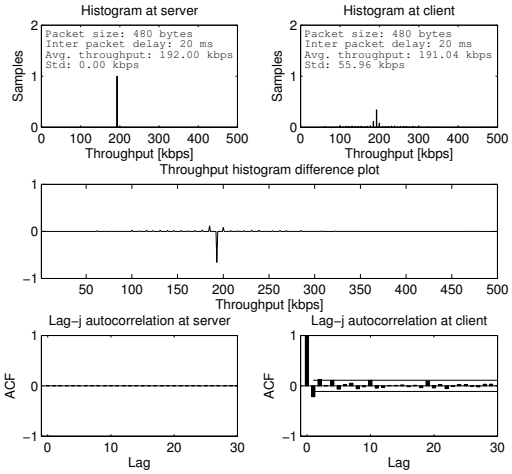
This paper focuses on measurements of application-perceived throughput on rather short averaging intervals. The influence of GPRS, and UMTS networks in both uplink and downlink direction on relevant summary statistics is illustrated and discussed. The throughput-related statistics have shown to be capable of visualizing critical network impacts on application perforrmance. Using the concept of an equivalent bottleneck, it is also possible to derive bounds for the extra delay of the first packet due to the need of setting up a Temporary Block Flow via caulasility arguments. Thus, we have been able to describe these kind of delays by throughput measurements.

In general, both UMTS and GPRS networks are capable of introducing enormous amounts of jitter, which is seen from throughput deviations based on one-second averages. The impact of these mobile channels on throughput histograms and autocorrelation functions are clearly visible. While UMTS is almost transparent in terms of throughput as long as the application uses only a small share of the nominal capacity, GPRS hardly does without jitter.

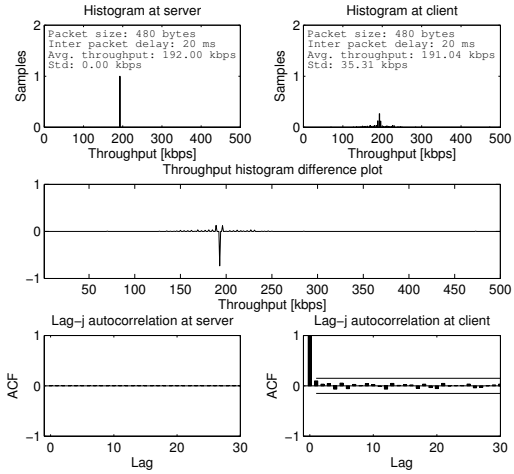
In the *downlink* direction, there is a certain risk of data loss even if the nominal capacity of the mobile link is not reached yet. It was observed that the GPRS network in use had considerable problems in delivering packets in downstream direction. Loss ratios of 10 to 20 % were not uncommon, which efficiently jeopardizes the performance of streaming applications.

In the *uplink* direction, the burstiness also rises as long as the nominal capacity of the wireless link is not reached. This effect is reversed as soon as the nominal capacity is surpassed: In case the sending application transmits datagrams too fast, the send function itself acts as a shaper by holding packets until they can be sent, which can be considered as some kind of “force feed-back”. Loss is avoided that way, but timing relationships within the stream can differ substantially from the ones imposed by the streaming application.

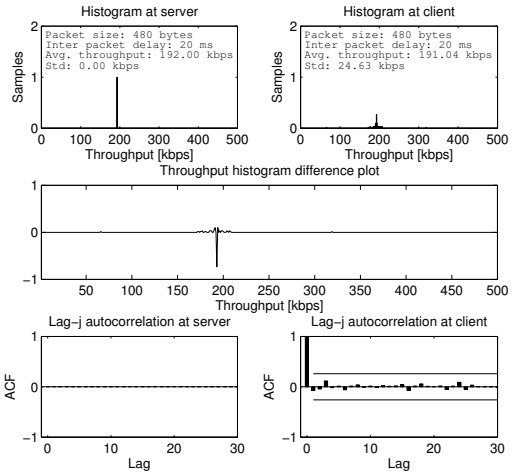
Summarizing, UMTS seems to be suitable for streaming services as long as the service uses merely a part of the nominal link capacity. GPRS, on the contrary, does not seem to be feasible



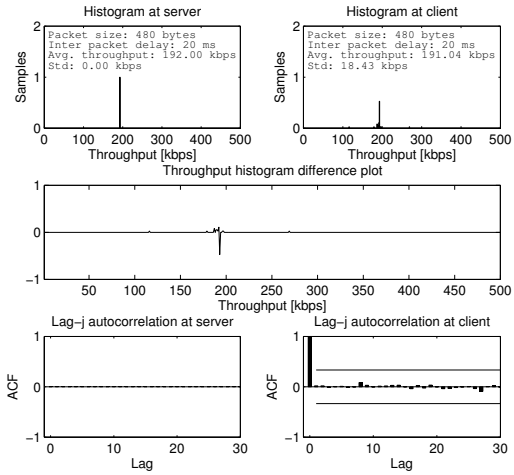
(a) $\Delta T = 0.5$ s



(b) $\Delta T = 1.0$ s



(c) $\Delta T = 3.0$ s



(d) $\Delta T = 5.0$ s

Figure 20. Impact of ΔT on throughput histograms, difference plots and autocorrelation functions.

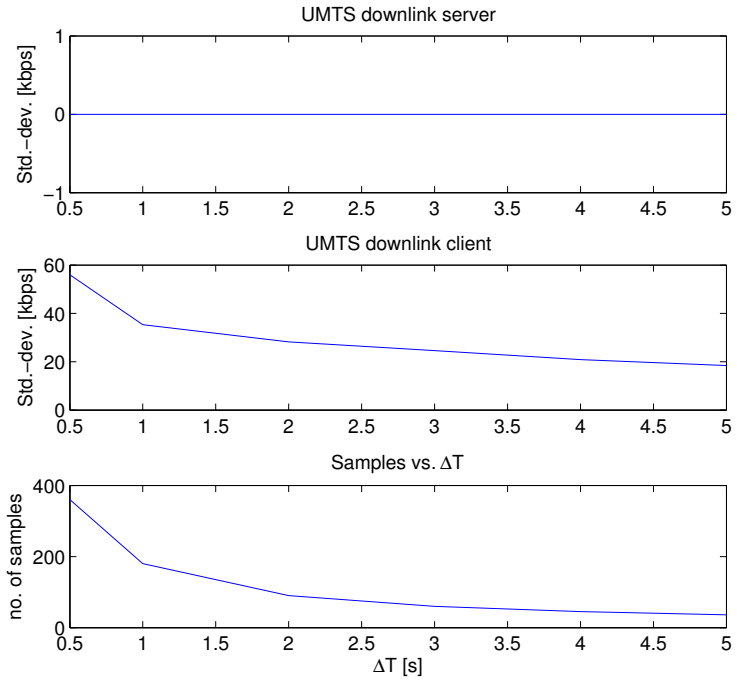


Figure 21. Impact of the averaging interval ΔT on standard deviation and number of time intervals n.

for streaming services at all and might even fail supporting elastic services such as file transfers because of the large amounts of loss and jitter.

Based on the above analysis and knowledge gained, an application could gain more throughput and being more efficient if the datagrams are sent with some inter-packet delay instead of sending all datagrams as fast as possible. This inter-packet delay, *i.e.* the nominal throughput, should be well-adapted to the throughput supported by the weakest communication link. As the available capacity could be changed during an ongoing connection due to cross traffic, it might be important to adaptively change the inter-packet delay to the new conditions. In general, the application programmer considering the use of mobile channels should be conscious of these throughput variations. Moreover, these results are aimed at optimizing network selection based on requirements provided by users and applications.

Future work will compare the offerings by different providers and will take cross traffic into account. Also, other time scales as well as the relationship between network behavior and user perception needs to be investigated further.

References

[1] G. Anders, M. Johansson, and T. Telkamp. Traffic Matrix Estimation on a Large IP Backbone - A Comparison on Real Data. In *Proceedings of the 2004 ACM SIGCOMM Internet Measurement Conference*, pages 149–160, Taormina, Sicily, Italy, 2004.

[2] C. Bettstetter, H.-J. Vogel, and J. Eberspächer. GSM Phase 2+, General Packet Radio Service GPRS: architecture, protocols and air interface. *IEEE Communications Surveys*, 2, 1999.

[3] J. Bolliger and T. Gross. Bandwidth monitoring for network-aware applications. In *Proceedings of the 10th IEEE International Symposium on High Performance Distributed Computing (HPDC-10'01), 7–9 August, San Francisco, USA*, 2001.

[4] P. Carlsson, D. Constantinescu, A. Popescu, M. Fiedler, and A. Nilsson. Delay performance in IP routers. In *Proceedings of HET-NETs '04 Performance Modelling and Evaluation of Heterogeneous Networks*, Ilkley, UK, July 2004.

[5] P. Carlsson, M. Fiedler, K. Tutschku, S. Chevul, and A. A. Nilsson. Obtaining reliable bit rate measurements in SNMP-managed networks. In *Proceedings of the 15th ITC Specialists Seminar on Traffic Engineering and Traffic Management*, pages 114–123, Würzburg, Germany, July 2002.

[6] J. Charzinski. Fun Factor Dimensioning for Elastic Traffic. In *Proceedings of the ITC Specialist Seminar on Internet Traffic Measurement, Modeling and Management, Sept. 18–20, Monterey, USA*, 2000.

[7] D. Comer. *Internetworking with TCP/IP Vol I: Principles, Protocols, and Architectures (fourth edition)*, pages 377–386. Prentice Hall, 1993.

[8] M. Feuerstein and T. Rappaport. *Wireless Personal Communications*, pages 200–201. Kluwer Academic Publishers, 1993.

[9] M. Fiedler and R. G. Addie. Verification and application of a second-order scale symmetry for queueing systems. In P. Key and D. Smith, editors, *Teletraffic Engineering in a Competitive World. Proceedings of the 16th International Teletraffic Congress (ITC-16)*, pages 807–816, Edinburgh, UK, June 1999.

[10] M. Fiedler, S. Chevul, L. Isaksson, P. Lindberg, and J. Karlsson. Generic Communication Requirements of ITS-Related Mobile Services as Basis for Seamless Communications. In *Proceedings of the First EuroNGI Conference on Traffic Engineering (NGI 2005)*, Rome, Italy, April 2005.

[11] M. Fiedler, S. Chevul, O. Radtke, K. Tutschku, and A. Binzenhöfer. The network utility function: A practicable concept for assessing network impact on distributed systems. Technical Report 355, Universität Würzburg, April 2005. To be presented at ITC-19, Beijing, August–September 2005.

[12] M. Fiedler and K. Tutschku. Application of the stochastic fluid flow model for bottleneck identification and classification. In *Proceedings of the SCS Conference on Design, Analysis, and Simulation of Distributed Systems 2003, 30 March – 3 April, Orlando, USA*, 2003.

[13] M. Fiedler, K. Tutschku, P. Carlsson, and A. Nilsson. Identification of performance degradation in IP networks using throughput statistics. In J. Charzinski, R. Lehnert, and P. Tran Gia, editors, *Providing Quality of Service in Heterogeneous Environments. Proceedings of the 18th International Teletraffic Congress (ITC-18)*, pages 399–407, Berlin, Germany, September 2003.

[14] G. Haßlinger, F. Hartleb, and M. Fiedler. The relevance of the bufferless analysis for traffic management in telecommunication networks. In *Proceedings of the IEEE European Conference on Universal Multiservice Networks*, Colmar, France, October 2000.

[15] T. Hoßfeld, K. Tutschku, and F.-U. Andersen. Mapping of file-sharing onto mobile environments: Enhancements by UMTS. Technical Report 343, University of Würzburg, November 2004.

[16] InfoSim GmbH & Co. KG. Homepage, 2005. URL: http://www.infosim.net/ (checked 2005-07-01).

[17] Z. Ji, J. Zhou, M. Takai, and R. Bagrodia. Scalable simulation of large-scale wireless networks with bounded inaccuracies. In *Proceedings of ACM MSWIM 2004*, Venice, IT, October 2004.

[18] I. Juva, R. Susitaival, M. Peuhkuri, and S. Aalto. Traffic characterization for traffic engineering purpose: Analysis of Funet data. In *Proceedings of the First EuroNGI Conference on Traffic Engineering (NGI 2005)*, Rome, Italy, April 2005.

[19] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson. On the self-similar nature of Ethernet traffic (extended version). *IEEE/ACM Transactions on Networking*, 2(1):1–15, February 1994.

[20] R. Morris and D. Lin. Variance of aggregated web traffic. In *Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Cat. No.00CH37064)*, volume 1, pages 360–366, Tel Aviv, Israel, 2000.

[21] Network Instruments – Protocol Analyzer, Network Protocol Analysis, Monitoring, Management, and Troubleshooting. URL: http://www.networkinstruments.com/ (checked 2005-07-01).

[22] I. Norros. On the use of fractional Brownian motion in the theory of connectionless networks. *IEEE Journal on Selected Areas in Communications*, 13(6):953–962, 1995.

[23] T. Oetiker. The multi router traffic grapher (MRTG). URL: http://www.mrtg.org/ (checked 2005-07-01).

[24] Personal Information for Intelligent Transport Systems through Seamless communications and Autonomous decisions. URL: http://www.aerotechtelub.se/piitsa/ (checked 2005-07-01).

[25] S. Qingguo. Handover in Packet-Domain UMTS. URL: http://www.techonline.com/ (checked 2005-07-01).

[26] J. Schiller. *Mobile Communications, second edition*, pages 126–127; 146–150. Addison-Wesley, 2003.
[27] A. Tanenbaum. *Computer Networks, fourth edition*, pages 153–156. Pearson Education, Inc., 2003.
[28] Technical Specification Group Radio Access Network; Base Station (BS) radio transmission and reception (FDD), 3GPP TS 25.104 version 7.0.0 Release 2005-06.
[29] Technical Specification Group Radio Access Network; Physical channels and mapping of transport channels onto physical channels (FDD), 3GPP TS 25.211 version 6.5.0 Release 2005-06.
[30] TechOnLine – Educational Resources for Electronics Engineers. URL: http:/www.techonline.com/community/ (checked 2005-07-01).
[31] R. van de Meent and M. Mandjes. Evaluation of 'user-oriented' and 'black-box' traffic models for link provisioning. In *Proceedings of NGI 2005*, Rome, Italy, April 2005.
[32] A. Veres, Z. Kenesi, S. Molnár, and G. Vattay. TCP's role in the propagation of self-similarity in the Internet. *Computer Communications, Special Issue on Performance Evaluation of IP Networks*, 26(8):899–913, May 2003.
[33] R. Vranken, R. van der Mei, R. Kooij, and J. van den Berg. Flow-level performance models for the TCP with QoS differentiation. In *Proceedings of the International Seminar on Telecommunication Networks and Teletraffic Theory*, pages 78–87, St. Petersburg, Russia, 2002.
[34] C. C. Z. Yang and H. Luo. Bandwidth measurement in wireless mesh networks. Course Project Report, URL: http://www.crhc.uiuc.edu/∼cchered2/pubs.html (checked 2005-05-23).