ftw. Technical Report

FTW-TR-2006-008
Printed May 2006

Diagnosis of Capacity Bottlenecks via Passive Monitoring in 3G Networks: an Empirical Analysis

Fabio Ricciato, Francesco Vacirca and Philipp Svoboda

ftw. Forschungszentrum Telekommunikation Wien
Donaucitystrasse 1, 1220 Vienna, AUSTRIA
funded by the Austrian Kplus Program

FTW-TR-2006-008
Printed May 2006

Diagnosis of Capacity Bottlenecks via Passive Monitoring in 3G Networks: an Empirical Analysis

Fabio Ricciato, Francesco Vacirca and Philipp Svoboda
ftw. Forschungszentrum Telekommunikation Wien,
Donau-City-Strasse 1, A-1220 Vienna, Austria
{riccato,vacirca,svoboda}@ftw.at

Abstract

In this work we address the problem of inferring the presence of a capacity bottleneck from passive measurements in a 3G network. The study is based on one month of packet traces collected in the UMTS core network of mobilkom austria AG & Co KG, the leading mobile telecommunications provider in Austria, EU. During the measurement period a bottleneck link in the UMTS core network was revealed and removed, therefore the traces enable the accurate analysis and comparison of the traffic behavior in the two network conditions: with and without a capacity bottleneck.

Two approaches to bottleneck detection are investigated. The first one is based on the signal analysis of the marginal rate distribution of the traffic aggregate along one day cycle. Since TCP-controlled traffic dominates the overall traffic mix, the presence of a bottleneck strains the aggregate rate distribution and compresses it against the capacity limit during the peak hour. The second approach is based on the analysis of several TCP performance parameters, e.g. estimated frequency of retransmissions. Such statistics are unstable due to the presence of few top users, but this effect can be counteracted with simple filtering methods. Both approaches are validated via simulations.

Our results show that both approaches can be used to provide early warning about future occurrences of capacity bottlenecks, and can complement other existing monitoring tools in the operation of a production network.

Contents

Appendix

1.1 Introduction

Public wide-area wireless networks are now migrating towards third-generation systems (3G), designed to support packet-switched data services. Several 3G systems are being developed worldwide evolving from different 2G technologies. Europe has adopted the Universal Mobile Telecommunication System (UMTS), developed by 3GPP as an evolution of GSM. A 3G network includes two main sections: a packet-switched Core Network (CN), which is based on IP, and a Radio Access Network (RAN). Along with the UMTS RAN (UTRAN) based on W-CDMA, several operators maintain a parallel GPRS/EDGE RAN evolved from the legacy GSM radio access. A general overview of the GPRS/UMTS network structure can be found in [18]. Several UMTS networks became operational since 2003 while the first deployments of GPRS dates back to 2000. Evolved cell-phones and smart-phones already support a broad range of data services, including traditional Internet applications like e-mail, Web, etc. while 3G interface cards for laptop are available since 2004, often coupled with flat-rate subscriptions. The growing popularity of 3G terminals and services has extended the coverage of Internet access to the geographic area, and 3G networks are becoming key components of the global Internet in Europe. Keshav [34] foresees that cell phones will become the dominant component of the future Internet population.

At present the 3G environment is under evolution: the subscriber population and the traffic volume are still in a growing phase (for some details on the network under study see [16]); the relative distribution of terminal types (e.g. laptops vs. handsets) and their capabilities is changing quickly; the portfolio of services that are offered by the operators evolves rapidly. All these aspects build a potential for changes in the traffic pattern that can occur at the macroscopic scale (network-wide) and in a relatively short time frame. In such a scenario, the ability to accurately and extensively monitor the current network state and to early detect global performance drifts and localized anomalies is a fundamental pillar of the network operation and engineering processes. More specifically, it is important to early detect emerging bottlenecks, i.e. network elements that are under-provisioned and therefore cause performance degradation to some traffic aggregate as its volume increases. A bottleneck in the RAN can be a geographical area that is frequently overloaded because the provisioned radio capacity does not match the peak-hour traffic demand. A bottleneck in the CN is often a link or other network element with insufficient capacity to carry peak-hour traffic. A capacity bottleneck always impacts a certain traffic aggregate rather than isolated flows - e.g. all the traffic directed to a certain radio area, or routed over a certain network element.

Monitoring a wide-area network involves considerable costs: the number of links to be monitored is large, and they are spread over a wide geographical area. For some operators there are several parallel RANs to be monitored that are attached to the same CN: besides UMTS and GPRS/EDGE, also WLAN (see [19] for

3G-WLAN interworking) and perhaps in the future WiMAX [4]. For some monitoring applications it is necessary to access configuration parameters (e.g. provisioned link bandwidth), logs and built-in counters from several network elements, and considerable costs, complexity and complications are found in practice where it comes to extraction, gathering and correlation of such heterogeneous data -in the format and semantics - from different elements, with different software and from different vendors. In summary, installing and maintaining a monitoring infrastructure with the same capillarity of the production network might be very expensive. On the other hand, the structure of a 3G network is highly hierarchical and centralized: the whole traffic is concentrated in the Core Network and there are only a few gateway nodes - called GGSNs - that connect it to external networks like the Internet, therefore the whole traffic can be captured at a few monitored links.

The ultimate goal of our research is to develop methods to infer the presence of performance bottlenecks anywhere in the network from the analysis of the traffic captured at few centralized monitoring points, without direct access to all network links. A possible approach is to exploit TCP behavior characteristics: since TCP is closed-loop controlled, its dynamics and performances are dependent on the state of the whole end-to-end flow path. Consequently, in principle it should be possible to infer the presence of performance bottlenecks by looking at the evolution of the TCP aggregate and/or to individual connections at any point along the path. We devised two possible approaches to diagnose the presence of a bottleneck on some network element from the analysis of the traffic aggregate routed through it but captured at a different point:

- From the analysis of the aggregate traffic rate.
- From the analysis of TCP performance indicators, like frequency of retransmissions, round-trip-time (RTT), etc.

In this paper we explore both approaches. This study is based on several weeks of packet traces collected by passive monitoring the core network links of a large operational 3G network (mobilkom austria AG & Co KG, the leading mobile operator in Austria, EU). In the middle of the measurement period a bottleneck link within the UMTS Core Network was detected and removed by a capacity upgrade. Since the traces are complete - i.e. all packet headers have been captured and timestamped - it is possible to analyze and compare several aspects of the traffic dynamics in the two network conditions: with and without the bottleneck. The goal is to identify those patterns that more clearly discriminate between the two conditions and can be taken as "signatures" of the presence of a bottleneck. This would allow the implementation of intelligent agents to be integrated directly into the on-line monitoring system, watching for future occurrences of similar patterns and reporting early warning about emerging bottlenecks. This approach is similar to the so called "post-mortem" analysis that is commonly performed in the area of

network security in order to learn about the dynamics of past attacks and devise countermeasures and detection mechanisms.

The rest of the paper is organized as follows. In Section 1.2 we position our work to the existing literature. The reference network scenario and the monitoring setting are described in Sections 1.3 and 1.4 respectively. Sections 1.5 and 1.6 carry a comparative analysis of traffic patterns before and after the bottleneck removal and hence suggest methodologies for the early diagnosis of emerging bottlenecks. Specifically, Section 1.5 addresses patterns of aggregate traffic rate, while Section 1.6 focuses on several TCP performance indicators. In Section 1.7 we provide a validation of the proposed schemes based on simulations. Finally in Section 1.8 we conclude and identify directions for progressing this work.

1.2 Related works

The present work neighbors different areas of evolving research in networking. First of all, capacity bottlenecks can be considered as anomalous events in a well-engineered network and the method proposed here can be accounted under the umbrella of "anomaly detection" methods. A number of previous papers have addressed the problem of anomaly detection in network traffic, spanning a broad range of different approaches (signal processing, machine learning, data mining etc.). The approach presented in Section 1.5 complements other techniques based on the analysis of the traffic signal [22]. It is relatively simpler than previously proposed techniques, e.g. wavelet analysis was proposed in [24] and [26]. Note however that our approach is focused only on the specific problem of bottleneck detection, rather than attempting at capturing "anomalies" in the general sense.

Another close area to our research is "capacity estimation". Several methods were proposed to infer the capacity and/or the available bandwidth of network links by means of active and passive measurement techniques (see [30] for an overview). The active measurement approaches do not apply here, since large-scale probing would not be accepted into an operational 3G network. Regarding passive measurement techniques, several tools were developed in previous works, e.g. MultiQ [33], Nettimer [20]. While these could be adapted to work on top of our monitoring system - and this is indeed one of the tasks in our future research agenda - they have some intrinsic limitations that might reduce their appeal for being used as operational tools in a production 3G network. In fact, both tools are based on the analysis of the dispersion in the packet arrival time [6], implicitly assuming that the bottleneck link is *buffering* excess packets. In the real network however capacity bottlenecks might be *dropping* packets. This is typically the case for rate-limiters (L2 or L3) that are configured based on token-bucket parameters. Also, the bottleneck element can be a node instead of a link, e.g. an overloaded network element that discards packets exceeding its processing capacity. The over-

load might be caused by different causes (e.g. a software bug consuming internal resources, an attack, or simply by lack of hardware upgrade) but the ultimate effect is the same: a shortage of capacity for the transit traffic. Other complications appear when considering L3/L2 inter-working, e.g. IP over ATM: in this case the net capacity available to the IP layer depends heavily on the packet size distribution, and a bottleneck can emerge due to an unanticipated increase in the frequency of small packets (e.g. following the quick spreading of some new application, or as a consequence of unwanted traffic and scanning worms [10, 29]). It is not clear how bandwidth estimation tools based on packet dispersion would perform in the above scenarios. Instead the methods proposed in our work do not make any assumption about the detailed nature of the bottleneck link, nor requires an exact definition of "capacity". Such simplification is made possible by the narrower scope of our goal: we are not interested in estimating the available capacity in the general case (as in [33, 20]), but just in detecting events of capacity starvation, by whatever definition of "capacity".

The second approach presented in this paper (Section 1.6) is related to the analysis of TCP connection characteristics in real networks. Many papers focused on TCP measurements for traffic modeling purposes (e.g. [27]), whereas only few works exploited passive TCP measurements to infer the status of the network. In [32] the authors use heuristics based on the TCP congestion control mechanisms and on the estimation of the Re-transmission Time-Out in order to reconstruct the TCP sender behavior, and classify out-of-order packets according to different causes. The same authors use in [31] a passive measurement methodology to infer the TCP sender congestion window and the connection RTT. As pointed out in both papers, their analysis is meant to be a first step towards a better understanding of the correlation between the properties of the traversed path and TCP connection metrics, ultimately aiming at the identification of the Autonomous System that degrades the connection throughput with high packet losses. While the underlying idea is similar to our work, namely exploiting TCP dynamics to infer performance degradation points, the application scenarios are different (Tier-1 backbone vs. 3G Core Network) and require different approaches and methodologies. In [21] the authors present the `Tstat` [5] tool for inferring the status of TCP connections and several associated metrics. Based on such tool they analyze the traffic on the edge of a campus network and estimate the performance from the perspective of individual users. Our goal is different, namely to validate the current network state from the perspective of the operator and produce early warning about large-scale performance bottleneck. In [28] the authors describe a TCP packet loss estimation method and they exploit it to study the effect of an Internet Service Provider (ISP) upgrade on TCP behavior. This approach, in principle similar to our, does not exploit TCP acknowledgment packets (ACK) to bypass asymmetric routing problem. In our network this problem does not apply and ACK packets enable us to obtain more information on TCP connection status.

Earlier parts of this work were presented in [13] and [15]. To the best of our

knowledge no previous work has addressed directly the problem of bottleneck detection in 3G systems, nor has reported empirical observations about a real bottleneck found in an operational network.

1.3 Reference Network Scenario

The reference network scenario is depicted in Figure 1.1. The 3G network has a tree-like deployment: the Mobile Stations (MS) and the Base Stations (BTS) are geographically distributed (nation-wide), but the level of concentration increases when moving towards the boundary of the 3G CN towards the Internet. There are several SGSNs (Serving GPRS Support Nodes) and few GGSNs (Gateway GPRS Support Nodes) and the traffic is concentrated on a small number of Gn/Gi links near the GGSNs, therefore with few probes on these interfaces one can capture the whole traffic. The problem addressed here is how to infer the presence of a bottleneck in the 3G core network from the passive observation of the traffic at a single monitoring point - on Gn in our case. Remarkably, we assume only a minimal information about the structure and settings of the whole network: for instance, *the bandwidth provisioned at each link is not known* to the monitoring agent, nor is known the detailed network structure. This allows a dramatic reduction in the complexity and maintenance effort of the monitoring system itself, as discussed later.

Definition of Sub-Aggregate (SA). The only required information are those enabling the discrimination of different sub-aggregate components inside the total traffic. A sub-aggregate (SA for short) defines the portion of the overall traffic observed at the monitoring point that is routed through a specific part of the network. A SA can be associated to each network element (e.g. SGSN, RNC, cell). For example, one could define a SA for a single SGSN x, meaning that all the traffic routed through SGSN x can be separated by the rest and examined separately. At a coarser granularity one could consider the SA towards a cluster of SGSNs y co-located at a single physical site and sharing the same access bandwidth to the site. At a finer granularity SAs can be associated to individual RNCs (say z). An even higher granularity, e.g. per-cell, might be considered for detecting recurrent bottlenecks or heavy congestion in the radio link, and then trigger a local revision of the radio planning[1]. Due to the typical tree-like structure of the 3G network SAs are hierarchically nested ($z \subset y \subset x$), and the analysis of SAs at different levels might help to individuate the position of a bottleneck, an approach that is in principle similar to network tomography [7]. In the most simple approach, the analysis of each SA is performed in isolation. In other cases the comparison between differ-

[1]Per-cell discrimination requires access to Gb/IuPS traces since such information is not communicated to the GGSNs and therefore is not visible on Gn. While we are currently working on per-cell analysis of TCP indicators (see [14] for preliminary results) this option is regarded as a future extension and is not considered in the present work, which is limited to the analysis of Gn traces.

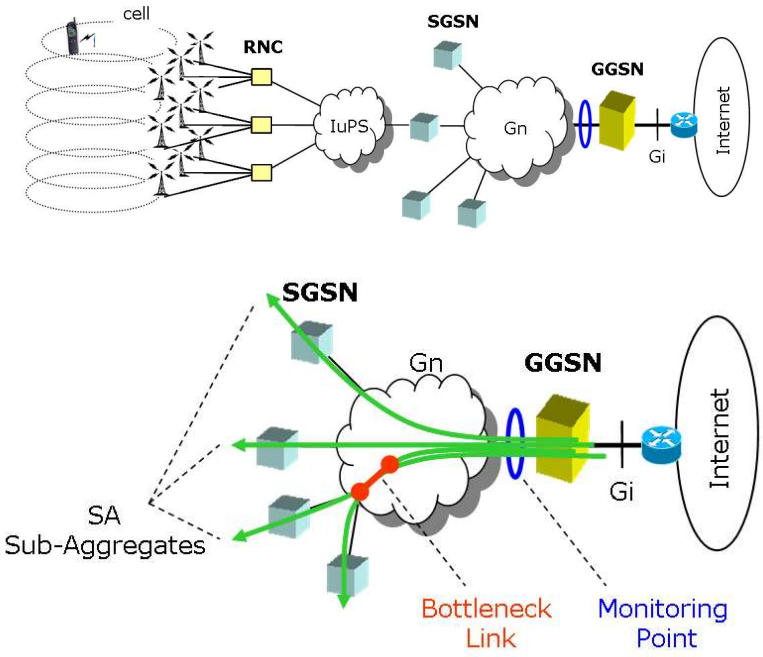

Figure 1.1. Reference network scenario (top) and example of bottleneck on Gn (bottom).

ent SAs at the same hierarchical level will help to pinpoint an abnormal behavior. However the comparison requires some caution, since different SAs might display different behaviors due to different load conditions, independently of the presence of capacity bottleneck or other network anomalies. Consider for example that the traffic to different SGSNs is produced by users in different geographical areas (e.g. urban vs. rural), generating different traffic mix and volumes. Still, if one is able to extract summary indicators that are invariant to different load conditions, then the direct comparison between SAs would provide a precious contribution to the task of bottleneck detection. The identification of synthetic and invariant indicators is one of the ultimate goals of our research.

SA discrimination. Sub-aggregate analysis requires sub-aggregate discrimination, that is the ability to refer each traffic unit (a packet or a connection) to a specific SA. In practice the way that this association is implemented depends on several technical details that are to some extent dependent on the specific configuration of the network. In this respect, it is preferable to set the monitoring point on Gn rather than Gi, since the lower layers of the Gn protocol stack include useful information for this purpose. For instance, the Gn network is IP-based: after GTP encapsulation user packets are encapsulated into UDP/IP packets carrying the IP address of the destination SGSN/GGSN: this allows direct per-SGSN and

per-GGSN discrimination. A higher level of discrimination can be achieved by tracking the cell/routing-area information present in some messages exchanged between the MS and the GSNs (see [14]). Note that the detailed tracking mechanism is different for GPRS and UMTS. In this paper we do not consider further the technicalities involved in the task of SA discrimination, since this would require a thorough analysis of the 3G protocols and some insight into the engineering practice of the real network. As a working hypothesis we assume that the monitoring system is capable of capturing and discriminating all packets belonging to a certain SA and the problem is to diagnose the presence of a bottleneck affecting such specific SA. This should trigger an alarm to the network staff, which should then start an in-depth inspection of the network and specifically of the SA path. The fact that the alarm is raised for a specific SA gives a useful initial hint about the possible location of the bottleneck but it is not sufficient to pointedly localize the bottleneck, nor to discriminate whether it is found upstream or downstream the monitoring point. Nevertheless restricting the troubleshooting process within the scope of the SA path is already a very valuable point in practice.

Obliviousness to the provisioned capacity. We assume that the capacity provisioned on each link is not known to the monitoring system, nor is known the *detailed* network structure. This eliminates the burden of maintaining the monitoring system synchronized with the configuration of the network elements, with clear savings in the complexity and maintenance efforts of the whole monitoring system. To appreciate such gain consider that a real network might have a very complex deployment. For example the connectivity between GGSN x and SGSN y (Gn interface) might involve a combination of physical circuits, L2 virtual circuits (e.g. ATM, Frame Relay) and LAN segments internal to each site. Therefore the path from x to y might include multiple links of different nature, dedicated or shared with other paths, and nodal equipments spanning multiple technologies. Our monitoring system is oblivious to the *detailed* deployment of the Gn links: we only look at the SA traffic flowing between x and y, regardless of its path.

A possible approach to detect congestion would be to compare the measured traffic volume with the provisioned path capacity, i.e. the minimum net capacity between x and y. However there are several complications associated to this approach:

1. Operational complexity: in order to derive the exact capacity of the path $x-y$ one might need to access the configuration files of several nodes along the path.

2. Deviations from nominal behavior: in case of malfunctioning of some network equipment the actual capacity of the link might depart from the nominal one.

3. L2 provisioned capacity: in case of rate-limited Virtual Channels (typically ATM) the net capacity "visible" at the IP layer depends on the actual distri-

bution of IP packet size. The larger the fraction of short packets, the smaller the net available capacity. In this scenario it is possible that a shortage of capacity emerges due to macroscopic changes in the statistics of the packet size (e.g. following new popular applications, or multiplication of undesired traffic resulting from worm infections [10, 29, 12]).

By taking an oblivious approach about the provisioned capacity the above complications are avoided. Furthermore, a number of additional advantages emerge. First, being decoupled from the actual configuration of the network equipments, the monitoring system does not need to be updated upon each reconfiguration or re-provisioning event, which saves costs in terms of system maintenance and communication overhead within the staff. Second, the resulting diagnosis method is more robust, since it will detect shortage of the *actual* bandwidth rather than of the *planned* one. Possible causes of mismatching between the two include human mistakes in the configuration of the network elements, equipment malfunctioning, mismatching between L3 and L2 bandwidth and others. After the signature of a capacity bottleneck has been recognized, the network staff has to locate it exactly, which requires knowledge of the detailed network deployment and eventually inspection of several elements during the troubleshooting process. However, being able to trigger an early alarm without relying on external information about the underlying network configuration greatly improves the robustness of the overall process.

1.4 Monitoring Setting

The development of a large-scale passive monitoring system, including a parser for the whole protocol stack of the 3G core network, and its deployment in the operational network were accomplished within the METAWIN project [23]. Packets are captured with Endace DAG cards [1] and recorded with GPS synchronized timestamps. For privacy requirements traces are anonymized by hashing any fields related to user identity at the lower layers of the 3G stack (IMSI, MSISDN, etc.) and user payload is removed. The traces include TCP/IP headers enabling the analysis of several TCP statistics. While we passively monitor all CN interfaces (Gi, Gn, Gb, IuPS) the results presented in this work are based exclusively on Gn traces. All Gn links were monitored near the GGSNs, covering 100% of UMTS traffic from home subscribers, roaming traffic is not considered.

For this work we collected 4 weeks of traces during December 2004. A capacity bottleneck was in place in the network, affecting a certain portion of the UMTS traffic, and was removed during the measurement period by a capacity upgrade. The bottleneck link was found on a IP-over-ATM Virtual Circuit with a cell-level rate-limiter based on the standard General Cell Rate Algorithm (GCRA). The link

was serving a physical site hosting multiple SGSNs. We were able to discriminate the SA component crossing the bottleneck element out of the total Gn trace and analyze it separately. In the rest of the paper we will therefore refer exclusively to the analysis of this sub-trace. For proprietary reasons we can not disclose several absolute quantitative values like traffic volumes, number of users, number of Gn links, etc. Where possible we provide only relative values, i.e. fractions, or re-scaled values. Similarly we can not provide any further detail about the specific structure of the monitored network and its deployment.

As discussed above, we are interested in looking at TCP behavior because of the end-to-end nature of its dynamics. Our measurements confirmed a large prevalence of TCP traffic in the 3G network. Furthermore it is likely that a large part of the traffic seen as UDP packets in the Core Network is accountable as TCP-controlled: for instance TCP connections tunneled into IPsec VPNs will be seen as UDP at our monitoring point.

1.5 Analysis of the Aggregate Rate

Here we focus on some basic statistical properties of the *instantaneous rate* for the traffic aggregate. The rate is measured in time bins of fixed length τ. The value of τ must be larger than few times the typical round-trip-time (RTT), as for smaller values the notion of "rate" for TCP-controlled flows vanishes. On the other hand too large values for τ will average out short-term fluctuations, which is against the goal of measuring the "instantaneous" rate. Given that the typical RTT measured in the operational UMTS network is in the order of half a second (see [16]) we used τ =10s. The results given below in this section will confirm the goodness of such choice for our purposes. Only the packets belonging to the specific SA under study are considered here in the traffic count. For proprietary reasons the whole time-series, denoted by $s_\tau(t)$, has been rescaled by an arbitrary factor denoted by u, therefore all rate values in the following graphs will be expressed in terms of this arbitrary unit (note that u is not related to the link capacity, therefore a rate value e.g. 0.6 does *not* refer to a link load of 60%).

The measured traffic is highly asymmetric (approximately 60% was Web): the ratio between uplink/downlink (from/to the MS) traffic was approximately 1:3. Only the downlink traffic reached the bottleneck capacity, therefore we will not consider here the uplink rate. Note that all user data packets are considered in the byte count, regardless of the specific payload protocols (TCP, UDP, ICMP, etc.). However due to the strong prevalence of TCP traffic the overall SA rate is shaped by the TCP dynamics.

Figure 1.2 plots the signal s_τ during the entire monitoring period. A bottleneck

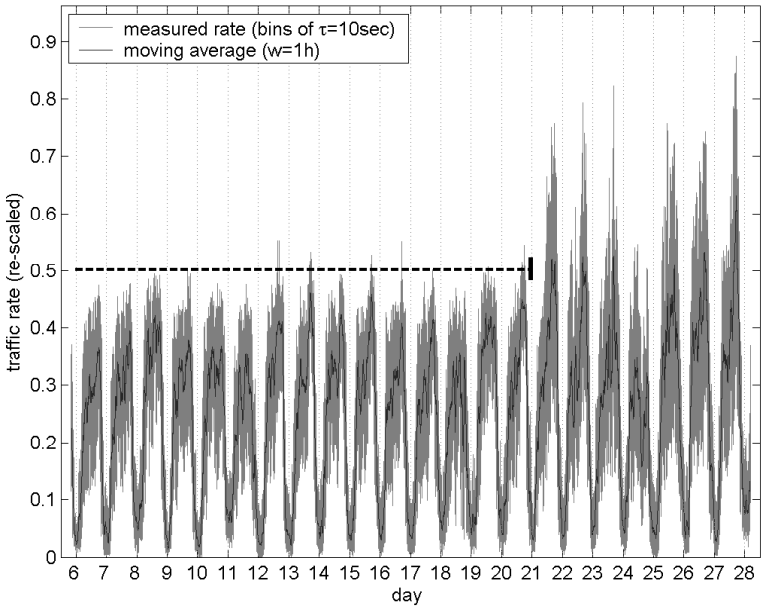

Figure 1.2. SA total rate for the monitored period (10s bins).

with bandwidth limit $0.5u$ was in place during the first part of the period, and was removed in the night between the day 20 and 21.

We detected the bottleneck on day 20 by the visual inspection of the aggregate rate over the last few days, as appeared in Figure 1.3. At a careful look we noticed that the variability range in the peak hour (point "A") was approximately the same as during the night time ("B"). Still the variability range at intermediate traffic levels ("C") was broader. We deemed it strange: we would have expected higher variability in correspondence of higher traffic level. Based on such qualitative observation, we made the hypothesis that some capacity restriction along the path of this SA was acting as a capping barrier for the TCP-controlled aggregate, preventing the maximum downlink rate from exceeding the provisioned bandwidth **which was unknown to us**. Triggered by our report, a quick investigation by the network staff resulted in the identification of a capacity shortage along the path of this SA, that was removed within few hours by a capacity upgrade. More precisely, the bottleneck was found on a IP-over-ATM link with a cell-based rate-limiter. Inspired by this event, we recognized the possibility to implement an automatic method for detecting future occurrences of similar events, by translating the qualitative observation based on the visual inspection into a quantitative indicator.

We analyzed the evolution of some basic properties of the marginal distribution of s_τ - namely variability and symmetry - at different traffic loads. To do so, we

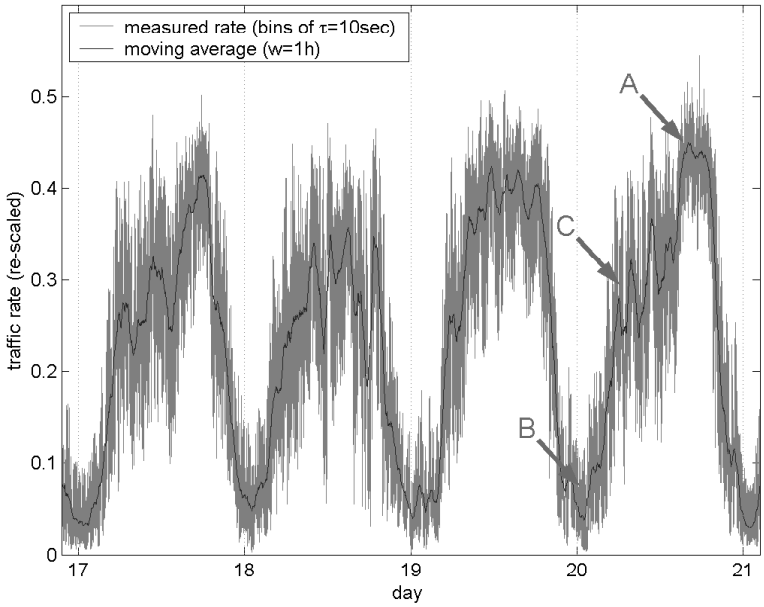

Figure 1.3. SA total rate for days 17-20 (10s bins).

considered a sliding window of length w and analyzed the empirical distribution of the samples in $D_{\tau,w}(t) = \{s_\tau(\theta) : t - w/2 \leq \theta \leq t + w/2\}$ for different values of t. The value of w should be short enough such that the signal s_τ can be considered stationary within this window, on the other hand too short values for w will include too few signal samples to have a robust estimation of the moments. We set w =1h as a compromise between two concurrent needs. We extracted the first moments of the empirical rate distribution $D_{\tau,w}(t)$, specifically the mean $m(t)$, the variance $\sigma^2(t)$ and the skewness $\gamma(t) = \frac{E\left((s_\tau - m)^3\right)}{\sigma^3}$. Recall that the skewness (3rd-order central moment) is a simple measure of the asymmetry of the distribution: positive values $\gamma > 0$ indicate a right-skewed distribution, negative values $\gamma < 0$ indicate left-skewness, symmetric distributions hold $\gamma = 0$. The values of these empirical moments depend on the time-scale parameters τ and w, but for sake of simplicity we will omit them in the notation and simply use e.g. m instead of $m_{\tau,w}$. In addition to the moments, we also computed the median $\mu(t)$ and percentiles $p_{10}(t)$, $p_{90}(t)$ of $D_{\tau,w}(t)$.

Before computing the moments, we pre-filtered the data from the outliers. This avoids that a few large samples bias the estimated moments in their neighborhood (i.e. within w/τ lags). An outlier can be e.g. a large burst of packets coming from the Internet - recall that the aggregate rate is monitored upstream of the bottleneck. A sample value $s_\tau(t)$ is classified as outlier if it falls outside the range $\mu(t) \pm 3\sigma(t)$, with median and standard deviations computed locally in the window $D_{\tau,w}(t)$ cen-

tered around t. This resulted in less than 200 outliers out of 60480 values (0.3%). Generally speaking, for some applications sporadic large values carry useful information about the phenomenon under study and should be considered as an important part of the traffic process rather than being classified as outliers and filtered out [9]. However this is not the case here, since the proposed scheme for bottleneck detection relies on the analysis of the mass and near-tails of the rate distribution.

We are interested in looking at the evolution of the moments and percentiles of the rate distribution with the mean traffic rate. In Figure 1.4 we plot the variance vs. mean trajectory during individual days. Each point represents a pair $\langle \sigma^2(t), m(t) \rangle$ for a specific value of t. The upper graph collects the values for days 17-20 (with the bottleneck in place) while the lower ones refer to day 21-23 (without the bottleneck). Similarly, Figure 1.5 plots the skewness vs. mean trajectory $\langle \gamma(t), m(t) \rangle$. In both Figures 1.4-1.5 the trajectories display different patterns before and after the bottleneck removal.

In order to summarize these data, we fitted the $\langle \sigma^2, m \rangle$ and $\langle \gamma, m \rangle$ trajectories measured in each day with a quadratic curve, i.e. a degree-2 polynomial. The resulting curves are plotted in Figure 1.6 for 22 days. The dashed and continuous lines refer respectively to day 6-20 (when the bottleneck was in place) and day 21-27 (after the bottleneck removal). By comparison with the previous graphs it can be seen that while the measured trajectories (Figures 1.4-1.5) differ from day to day due to random signal fluctuations, the regressed curves are relatively stable. From the upper graph of Figure 1.6 it can be seen that when the bottleneck is in place the variance reaches a maximum when the mean is approximately half of the bottleneck capacity, after which it decreases again. Instead, without the bottleneck the variance keeps increasing steadily with the mean traffic intensity. From the lower graph of Figure 1.6 we learn that the presence of the bottleneck also impacts the skewness of the distribution: in the high traffic region the bottleneck clearly induces left-skewness ($\gamma < 0$), while after the bottleneck removal the distributions tend to be more symmetric, with small positive values of γ.

We applied a similar method to the percentiles $p_{10}(t)$ and $p_{90}(t)$. The resulting regressed trajectories are shown in Figure 1.7, from which it can be easily seen that the trajectory of the inter-percentile distance $\langle p_{90} - p_{10}, m \rangle$ displays a concave behavior similar to the variance.

1.5.1 Interpretation of the results

All the above observations are consistent with the following interpretation. The empirical rate distribution evolves with the mean traffic intensity - therefore with the time-of-day - as sketched in Figure 1.8. For low traffic intensity (night and early-morning) the distribution is right-skewed: since rate can not be negative only

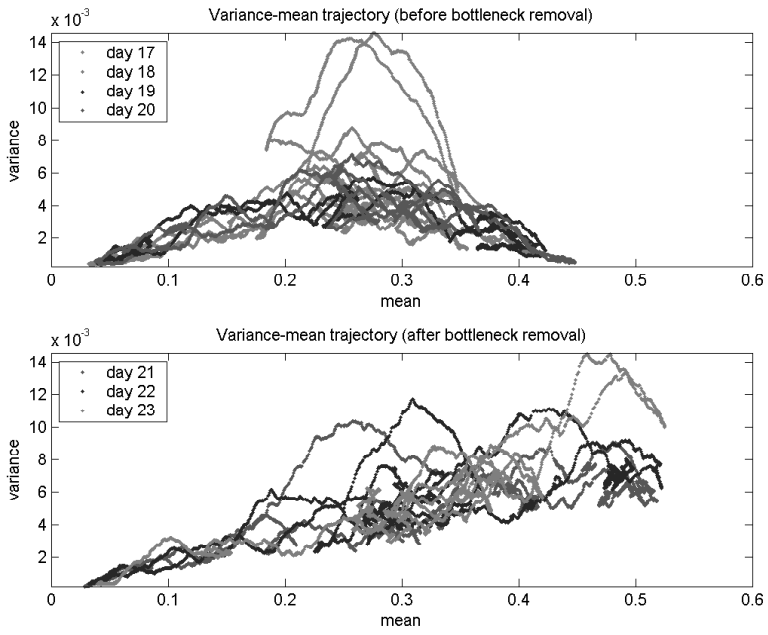

Figure 1.4. Variance vs. Mean trajectory: with (top) and without (bottom) the bottleneck.

positive fluctuations are possible. For higher traffic intensity (morning and afternoon) the rate distribution becomes more dispersed (higher variance) and more symmetric (lower skewness). In fact, the rate of individual connections is limited by the bandwidth on the radio link, but the radio limit acts on individual flows (per-MS channel) or at most on a bundle of few flows belonging to the active MSs in the same cell (per-cell bandwidth). As a result, in the total SA under analysis - covering hundreds of parallel connections spanning several cells - the correlation between the rates of individual connections is very low, and the shape of the total traffic distribution is dominated by the arrival / departure process of the connections, assumed independent at a first approximation. Since higher traffic during the day means essentially more active MSs and more parallel connections, this translates into a marginal rate distribution with increasing mean, variance and symmetry, as is typical with the superposition of many independent ON/OFF sources. Without any common bottleneck, such trend would be maintained also during the peak hour (evening and early-night), with more variance and more symmetry as in Figure 1.8-bottom. Instead, when the bottleneck is in place (see Figure 1.8-top), it introduces an additional limit on the total rate of the whole SA whose effect becomes stronger as the total traffic approaches the bottleneck capacity. At this point two phenomena take place. First, due to the TCP congestion control mechanisms, the bottleneck introduces correlation between the rates of the individual connections. Second, an additional control-loop emerges

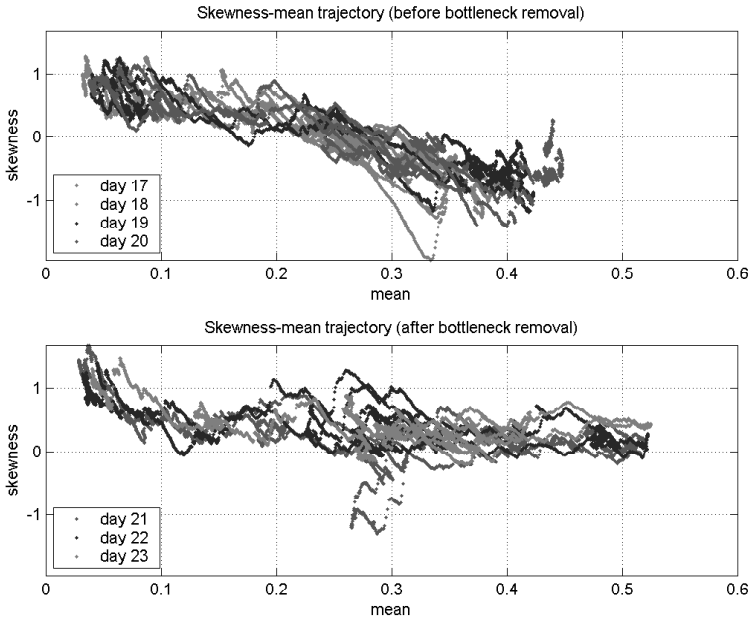

Figure 1.5. Skewness vs. Mean trajectory: with (top) and without (bottom) the bottleneck.

from the user behavior: users will delay the next download until the current one is completed, and arguably some users will abandon the network if the perceived quality of service (e.g. page download time) degrades below what they consider the "acceptable" level. As a result, the user behavior contributes to modulate the number of parallel connections during the congestion period. The cumulative effect of these two control loops (TCP congestion control and user behavior) is that the total marginal rate distribution becomes compressed towards the capacity limit thus reducing its total variance, and the hard-limit on the upper tail - but not on the lower one - leaves the distribution left-skewed.

1.5.2 Diagnosis method

The basic idea of our work is that by simply observing the evolution of the marginal rate distribution during one day cycle, it is possible to infer the presence of a bottleneck, i.e. of a compressing barrier, **without knowing the value of the bottleneck limit**. The goal is to develop one or few empirical indicators telling whether the marginal distribution of the traffic evolves "freely" with the mean traffic intensity, or rather displays signs of strain. Our first proposal is to derive simple indicators based on the coefficients of the regressed trajectories (e.g. variance-mean) along

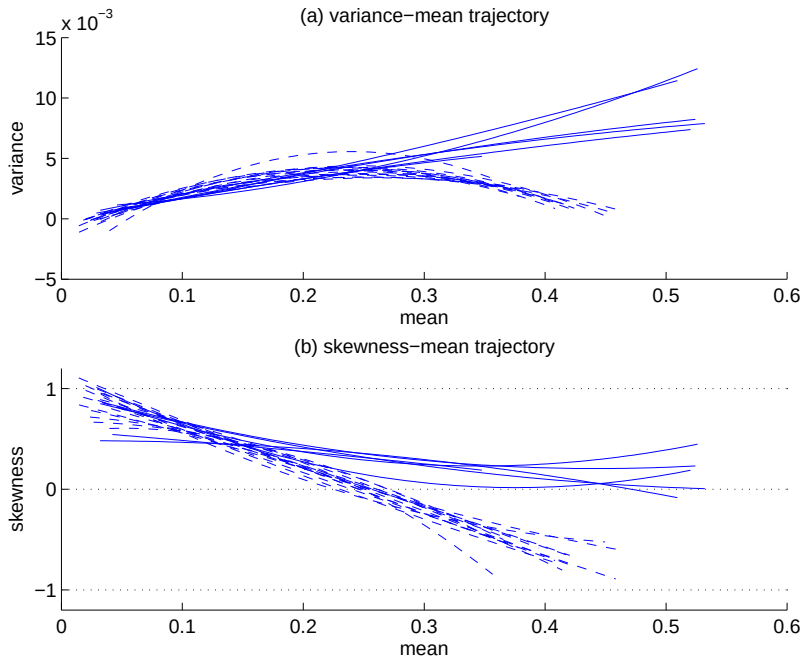

Figure 1.6. Variance-mean (top) and skewness-mean (bottom) fitted trajectories for 22 days. Dashed line: day 6-21, with bottleneck. Solid line: day 21-27, without bottleneck.

one-day cycles, as explained in the following. For a generic day denote by a_j and b_j the coefficients of the regressed polynomial for the variance-mean and skewness-mean trajectory respectively, and by $\overline{m}$ the maximum value of the mean traffic registered during the same day. Denote by $v_1 = 2a_0\overline{m}+a_1$ the slope of the variance-mean trajectory at the extremal point $\overline{m}$. Similarly, denote by $v_2 = b_0\overline{m}^2 + b_1\overline{m} + b_2$ the extremal value of the regressed skewness. In Figure 1.9 we plot the measured values for v_1 and v_2 for the whole monitored period. There is an evident correlation between negative values for both indicators and the presence of the bottleneck. This suggests that a preliminary alarm can be triggered whenever any of such indicator falls below zero, while persistent negative values for 2-3 days should trigger a check intervention by the network staff. In this specific case, by setting the alarming region at $v_1 < 0$ or $v_2 < 0.25$ such simple indicators would have revealed the presence of the bottleneck several days in advance. In practice, the alarming region should be identified based on the historical time-series for the same SA under known bottleneck-free conditions. In fact different SAs at different aggregation levels and/or on different networks might have these indicators fluctuate in different "typical" ranges.

In summary, the proposed algorithm would include the following steps, to be run regularly each day for each SA:

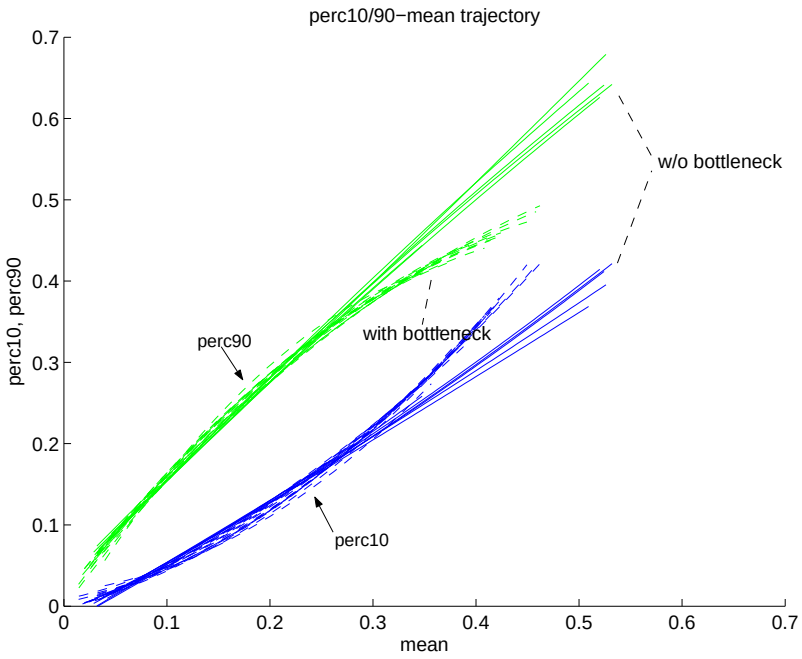

Figure 1.7. $p_{10}(t)$ and $p_{90}(t)$ vs. Mean fitted trajectories. Dashed line: day 6-21, with bottleneck. Solid line: day 21-27, without bottleneck.

1. Collect rate measurements at low time scale (e.g. τ=10s).
2. Remove outlier samples falling outside the range $\mu(t) \pm 3\sigma(t)$, computed in a sliding window of length w (e.g. 1 hour).
3. Compute running moments (mean, variance, skewness) and percentiles ($\mu(t)$, $p_{10}(t)$, $p_{90}(t)$) in sliding window of length w.
4. Apply quadratic fitting to the X-mean trajectories (where X stands for variance, skewness, inter-percentile distance, etc.) to reduce the data to a few coefficients.
5. From these coefficients derive a synthetic vector indicator $\mathbf{V}$.
6. Collect historic records and identify the "typical" region for $\mathbf{V}$.
7. Raise an alarm whenever $\mathbf{V}$ persistently falls outside such region (i.e. for 2-3 consecutive days).

Note that each step is rather easy to implement and requires a limited amount of memory and processing resources: in the end the overall scheme is just a collec-

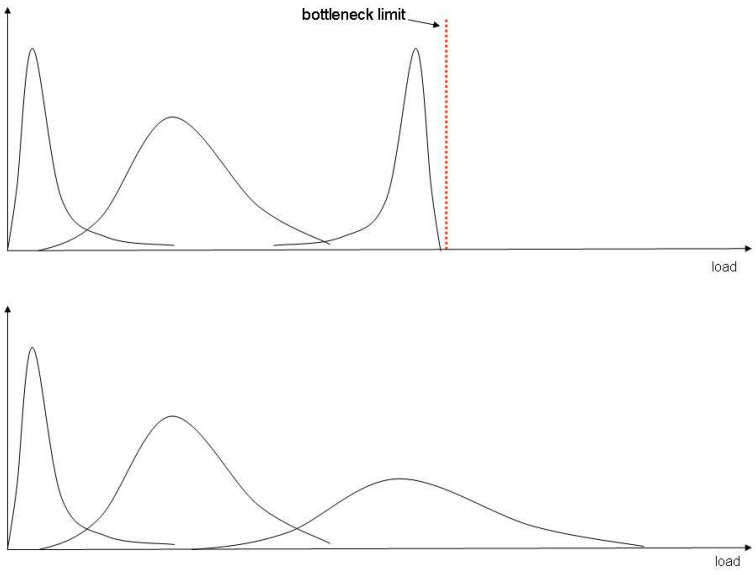

Figure 1.8. Evolution of the marginal rate distribution

tion of simple pieces of time-series analysis, and they are typically found as standard routines in any mathematical library. We remark that the quadratic regression at Step 4 is a key step: by collapsing the set of observations along one day into a few parameters - i.e. the coefficients of the fitted polynomial - it reduces the instability of the indicators and increase the robustness of the whole test. The reason for choosing quadrating fitting is that degree-2 polynomials is the simplest family of curves that includes elements with monotonic behavior (increasing / decreasing in the whole range) as well as elements with non-monotonic behaviors and a local maximum / minimum. We have seen that the presence of the bottleneck has a direct impact on the monotonicity of such trajectories, e.g. the variance-mean trajectory is monotonically increasing under normal "bottleneck-free" conditions, and becomes folded and non-monotonic when the bottleneck is in place. Such patterns can be taken as a signature for the presence of the bottleneck. Degree-1 polynomials are not sufficient to represent non-monotonic behaviors. On the other hand polynomials with degree higher than 2 would not be more helpful for our purposes, and hamper the robustness of the overall scheme by offering more degrees of freedom to reflect secondary details and/or random fluctuations in the trajectory.

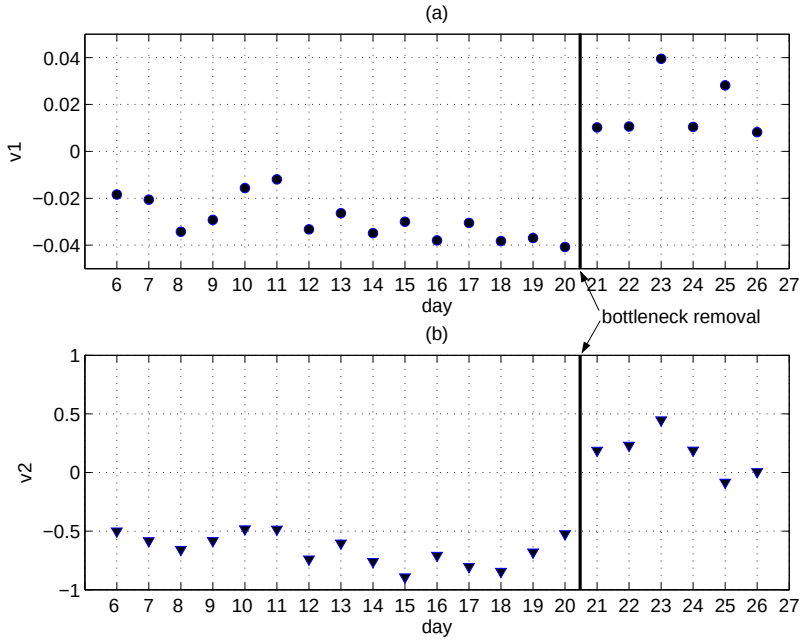

Figure 1.9. Measured values of summary indicators v_1,v_2.

1.6 Analysis of TCP Performance Indicators

In parallel to the rate analysis, we also considered several TCP performance parameters as candidate indicators for the presence of a bottleneck. The first group of potential indicators refers to the frequency of re-transmission events, discriminated into the following categories:

- **FRTX** (Fast-Retransmit Re-transmissions): the number of TCP packet re-transmissions triggered by duplicate ACKs.
- **LRTO** (Loss-induced Retransmission Time-Outs): TCP re-transmission events triggered by the expiration of the TCP Retransmission Time-Out (RTO) caused by packet loss.
- **SRTO** (Spurious Retransmission Time-Outs): TCP re-transmission events due to RTO expiration caused by a large delay, without packet loss.
- **AMB** (Ambiguous Retransmission Time-Outs): TCP packet re-transmissions that we are not able to classify into one of the previous categories. Note that AMB events are always associated to RTO expiration.

All these events were inferred with the procedure proposed in our previous work [17]. Therein the focus was on the estimation of the SRTO events, which in turn required the discrimination of LRTO, FRTX and AMB. We have implemented the estimation algorithms in a modified version of the `tcptrace` tool [2] which was run over the whole trace[2]. For each type of event we measured the relative frequency into each time bin as follows. For each active MS_i we counted the number of occurrences of the specific event (e.g. LRTO) n_i, and the total number of DATA packets N_i. The global frequency of such events is then defined as the ratio $f_{tot} = \frac{\sum_i n_i}{\sum_i N_i}$.

In this work we used the IP address on the MS side as an identifier for the MS. In the 3G network IP addresses are dynamically assigned to MSs when they connect to the network (PDP-context activation [18, p.413]) and released when the connection (PDP-context) is deactivated. In principle the same IP address might be immediately re-assigned to another connecting MS. However we checked off-line that in the network under study the level of address re-usage by different MSs within 1 hour is almost negligible, i.e. only a very small fraction of IP addresses where used by two or more different terminals within each time bin. On such basis we assign each packet to a MS based on the IP address, accepting the small error caused by short-term address re-usage. In the general case it would be desirable to have exact MS discrimination, i.e. to identify the source / destination MS for each packet based e.g. on the IMSI (International Mobile Subscriber Identifier) or any other unique identifier derived from it (for privacy reasons). Recently we improved our monitoring system by implementing direct MS discrimination based on (anonymized) IMSI. This requires stateful tracking of the signaling messages associated to each PDP-context (see [14, 25] for more details), hence each packet can be labeled with a unique and non-ambiguous identifier associated to the MS. Such feature was not yet in place in the dataset from Dec'04, therefore we resorted to (approximate) MS discrimination based on IP address. The results presented below indicate that this approach does not limit the capability of the proposed method to expose the presence of the bottleneck from TCP indicators. This is a good news in practice, since the proposed method can be applied also on "simple" IP-level Gn traces without exact MS discrimination, at least in those networks where the IP address pool is large enough to produce a low level of short-term reusage. Note that the second bottleneck event discussed in Section 1.7.1 was detected by applying the same methodology discussed in this section but with exact MS discrimination.

In addition to the above events we also considered the Round Trip Time (RTT) values. A first exploratory analysis of TCP RTT in UMTS and a comparison with GPRS was reported in our previous work [16]. Instead of the end-to-end RTT, we considered the semi-RTT between the monitored interface (Gn) and the Mobile Station (MS). For sake of simplicity, in the rest of this work we will refer the

[2]The modified `tcptrace` version can be downloaded from http://userver.ftw.at/∼vacirca

Gn-MS-Gn semi-RTT simply as "RTT". An RTT sample is defined as the elapsed time $t_{data} - t_{ack}$, where t_{data} and t_{ack} are the timestamps respectively of a TCP DATA packet arriving from the Internet and of the associated ACK from the MS as "seen" at the monitoring point on Gn. The RTT defined in this way includes three components: the downlink delay (Gn→MS), the uplink delay (MS→Gn) and delay component internal to the MS (e.g. processing and I/O buffering). Only the first two components are accountable as network-dependent, while the latter depends exclusively on the terminal. However, the TCP dynamics, and ultimately the user-perceived performances, will be impacted by the cumulative RTTs. Note that only non-ambiguous DATA-ACK pairs are considered to produce a valid RTT sample: the acknowledgment number of ACK must be at least one byte greater than the last sequence number of the DATA packet; furthermore, it is required that the packet being acknowledged was not retransmitted, and that no packets that came before it in the sequence space were retransmitted after t_{data}. The former condition invalidates RTT samples due to the retransmission ambiguity problem. This is the same procedure that TCP utilizes to estimate the RTT and to set the value of the Retransmission Timeout (Karn's algorithm, see [35]). The latter condition invalidates RTT samples since the ACK could acknowledge cumulatively the retransmitted packet rather than the original DATA packet.

Before the analysis we filtered out all packets on ports tcp:4662 and tcp:445/135. The former is used by popular peer-to-peer file sharing applications: since it typically runs with many parallel TCP connections it is likely to induce self-congestion on the radio channel and/or on the terminal internal resources (e.g., transmission buffer). This would result into poor TCP performances that are application-specific rather than network dependent, therefore do not carry information about the network state. Additionally, during the exploratory analysis we found the presence of a large number of packets directed to ports tcp:445/135, mainly TCP SYN in the uplink direction. This is likely due to some self-propagating worms attached to infected 3G terminals. The presence of such unwanted traffic should be expected since laptops with 3G datacards - often equipped with popular operating system - populate the 3G networks nowadays along with handsets and smartphones. It is well-known that unwanted traffic is a steady component of the traffic in the wired networks since years (see for instance [29]). What it is important here is that most of such packets did not bring any valid contribution for the problem of bottleneck detection while consuming resources in the analysis software (i.e., memory state in `tcptrace`) therefore filtering them out speeds up the analysis process.

1.6.1 Results

In Figure 1.10 we plotted the measured values for each parameter in time bins of 1 hour. The top subgraph shows the cumulative number of TCP DATA packets ($\sum_i N_i$), normalized to the peak value during the entire monitoring period in order

to not disclose the absolute value. The second subgraph reports the average and several percentiles (5%, 50% 95%) of the RTT samples extracted in each time bin. The remaining subgraphs report the measured frequency of FRTX, LRTO, SRTO and AMB events respectively. Recall that the bottleneck was removed in the night between day 20 and 21. From Figure 1.10 it appears that the RTT statistics display a large variability, with average values which are occasionally very large. However, a deeper look would reveal that the most of RTT spikes occur over night, when the traffic volume and the number of active terminals are very low. This suggests that such spikes might be the effect of few MSs generating a large volume of packets and hence RTT samples. In case that such MSs incur large RTT values for some specific reason (e.g. poor local radio condition, intense mobility) they might introduce a bias in the whole RTT statistics. This is more likely to happen when the network load - therefore the number of active MSs - is low.

Regarding the re-transmission indicators, it can be seen that FRTX, SRTO and AMB frequencies display periodic spikes before day 21, particularly at the peak hour, which are clearly an effect of the bottleneck. Among them, the SRTO seems to be the best indicator for this type of bottleneck. In fact, it can be seen that after the bottleneck removal the SRTO frequency stays at a "physiological" level (below 0.1% in UMTS) that is highly stable: no large fluctuations are present, and there is no apparent dependancy on the time-of-day and therefore on the network load. In other words, the "normal" value of SRTO frequency in UMTS is invariant to changes in the network load. As discussed above in Section 1.3, this is a highly desirable characteristic for a parameter that should be used as an indicator for abnormal network conditions.

The behavior of FRTX and AMB shows some correlation with the presence of the bottleneck, with large spikes mostly during peak-load hours before day 21. However they also display some sporadic high values after the bottleneck removal. Regarding the LRTO, there are no evident differences in the behavior before and after day 21. However, similarly to the RTT, we notice that most of the spikes seen for FRTX, LRTO and AMB after the bottleneck removal are located during the off-peak hours, which again suggests the possibility of bias due to few MSs contributing with many samples ("big" N_i) and in bad local conditions ("bad" $\frac{n_i}{N_i}$ frequency). In order to counteract the biasing effect of such few "bad-and-big" MSs we used a simple heuristic approach. The underlying idea is to filter out in each time bin the few "worst" MSs, i.e. those few contributors that inflate the whole statistic. For the RTT we rank the terminals w.r.t. the product of the average measured RTT $\overline{r}_i$ and the number of valid RTT samples K_i (note this is different from the number of DATA packets N_i), we filtered out the 10 MSs with the highest $\overline{r}_i \times K_i$ product in each time bin ("worst-10"), and re-compute all the RTT parameters (average and percentiles) from the residual set.

For the other indicators associated to packet losses and retransmissions we follow a similar approach, namely to filter out the "worst-10" MSs from the global

statistics. However, since the underlying metric is always a relative frequency in the form of a fraction $\frac{\sum_i n_i}{\sum_i N_i}$, the definition of "worst" MSs requires a more sophisticated approach that is discussed separately in Section 1.6.2 for ease of treatment.

In Figure 1.11 we plot the same quantities of Figure 1.10 after the filtering process (note that the respective subgraphs might be in different scales). All the frequencies of re-transmission events now display a more predictable behavior after the bottleneck removal, thus confirming our hypothesis that most of the instability was due to few outlier MSs. The large residual spikes, now regularly located in the peak hours, disappear completely after day 21. This clearly relates with the presence of the bottleneck, and the change in the behavior is so clear that one can immediately pinpoint the bottleneck removal time.

Regarding the RTT statistics, the filtering process had a dramatic effect and almost completely canceled the fluctuations of the average - now firmly anchored to the level of 500ms - and of the lower percentiles. However, there is no evidence of correlation between the RTT values shown in Figure 1.11 and the presence of the bottleneck. The conclusion is that the RTT process, at least as estimated with the methodology described above, is not a good indicator for this type of bottleneck. A likely explication is that the RTT estimation process only considers selected DATA-ACK pairs that do not hold any ambiguity in the RTT estimation. This method filters away "invalid DATA-ACK pairs", that typically emerge in the neighborhood of events like packet losses, retransmissions and timeouts. Now, these are exactly the events that are generated by the bottleneck. In other words, RTT statistics were intrinsically "cleaned-up" due to the way RTT samples are extracted, which explains why they did not react to this type of bottleneck. Still, it can be argued that different types of bottleneck - e.g. those with large buffering space, as a shaper - might be better captured by indicators associated to the packet delay rather than to retransmissions or timeouts.

1.6.2 Counteracting instability

For each type of re-transmission event (e.g. LRTO) we measured the relative frequency of such events into each time bin as follows. For each active MS_i we counted the number of event occurrences n_i and the total number of DATA packets N_i. In other words, we consider a bi-dimensional process $\{n_i, N_i\}$, with $n_i < N_i$. We denote by d_N the probability distribution of the random variable N_i, and by M the total number of active MS in the time bin. The global frequency f_{tot} of such events is then defined as :

$$f_{tot} = \frac{\sum_{i=1..M} n_i}{\sum_{i=1..M} N_i} \tag{1.1}$$

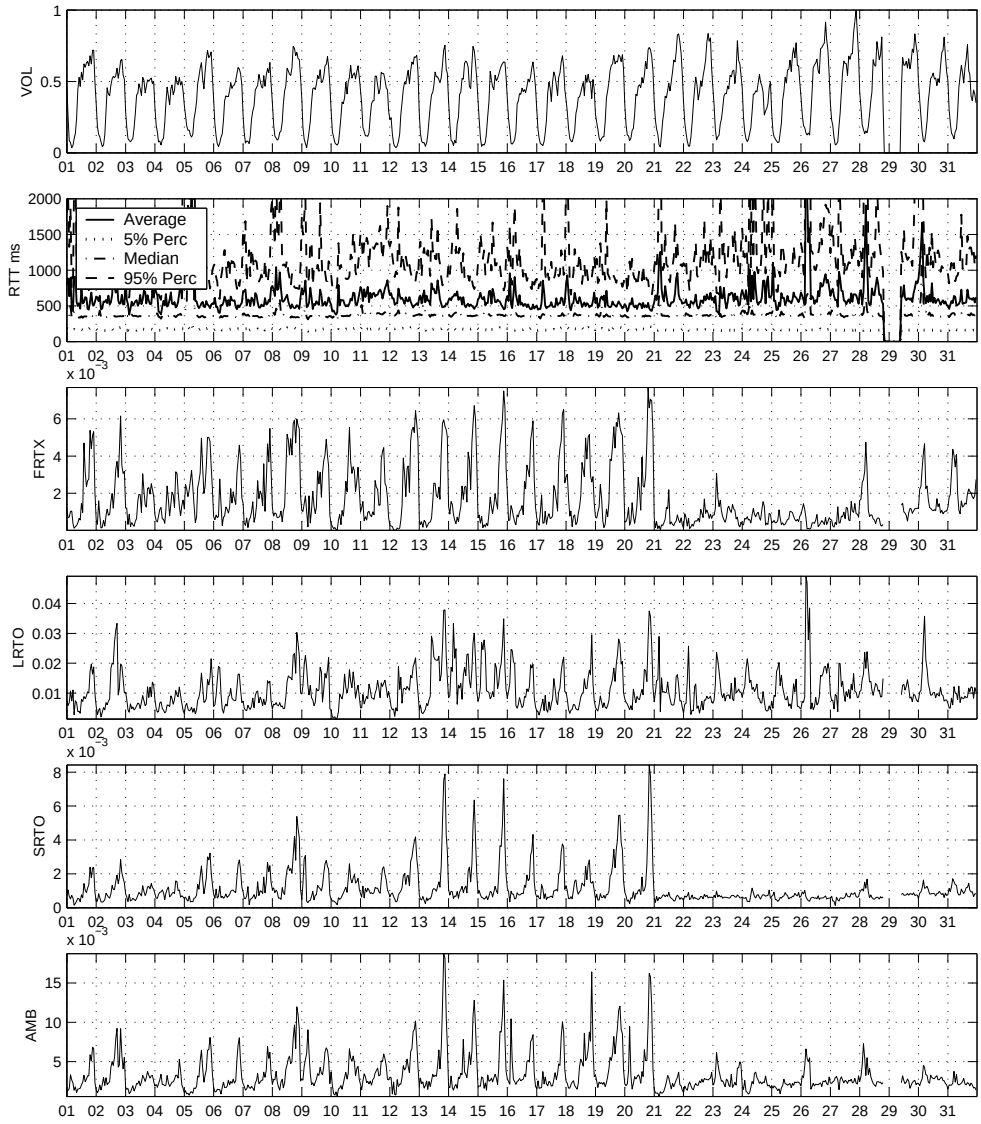

Figure 1.10. TCP performance indicators (1h bins). From top to bottom: total packet count, RTT, FRTX, LRTO, SRTO, AMB.

The basic idea of our work is that the value of f_{tot} can indicate the presence of a bottleneck. In the absence of a performance bottleneck the LRTO events should be sporadic, and the fraction of MSs with $n_i > 0$ in each time bin should be relatively small. This scenario is reflected into a very small value of f_{tot}. On the other hand, if a bottleneck is in place along the SA path the global incidence of retransmission events should be higher - larger values of f_{tot} - and at the same time should span a larger fraction of MSs.

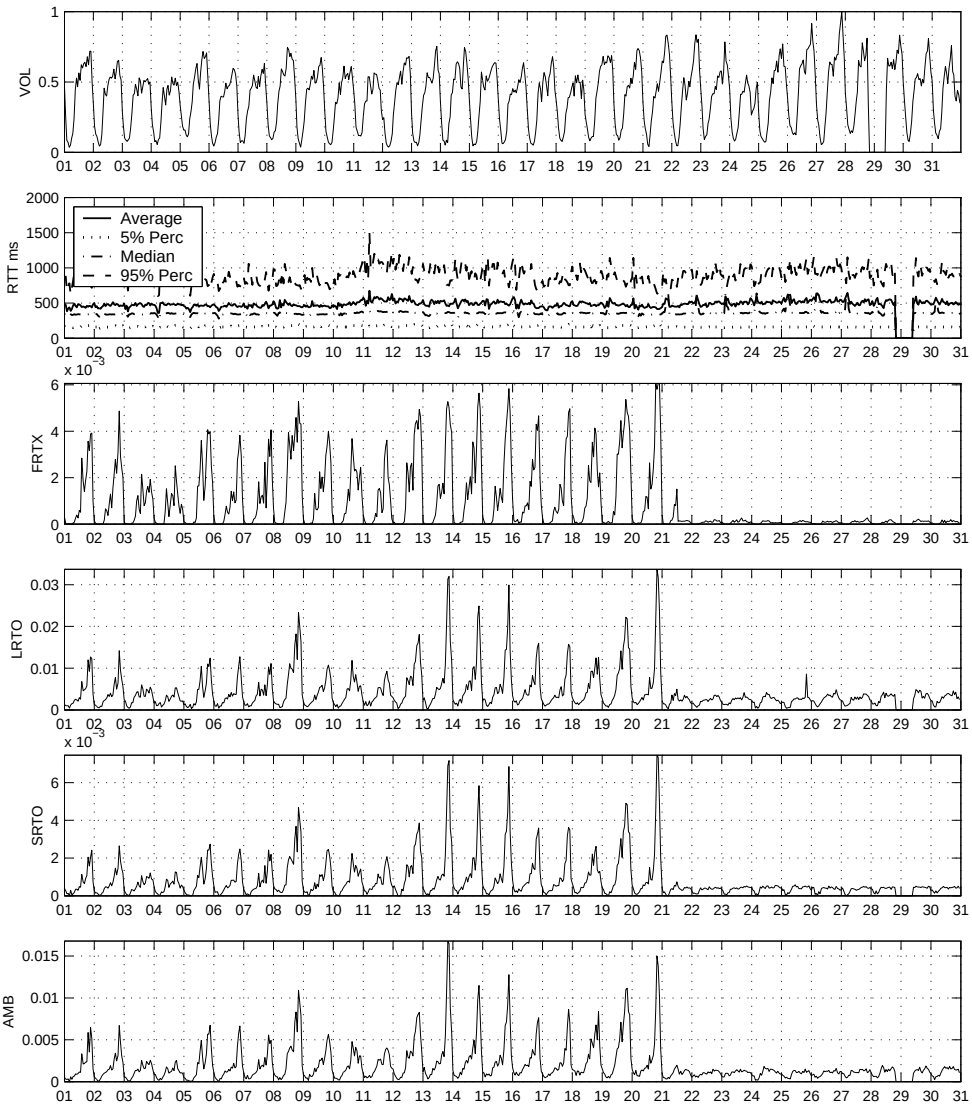

Figure 1.11. TCP performance indicators after filtering the worst-10 MSs in each bin.

Besides these two extreme scenarios, we must consider other possible cases:

1. No bottleneck is in place for the whole SA, but a subset of the MSs are affected by a higher incidence of retransmission events due to some local bottleneck (e.g. congestion in a specific cell or bad radio condition);

2. One or few MSs display a higher incidence of retransmission events due to specific MSs conditions (e.g. poor radio coverage or terminal malfunctioning); in the particular case that these MSs generate a very large number of

packets, they can inflate the overall global frequency f_{tot}; this is more likely to happen when the population of active MSs is relatively small (e.g. during low-traffic periods) or when the distribution of traffic per-MS (therefore of N_i) is heavy-tailed, such that a few "big" MSs can dominate the whole statistics.

In particular, the latter case introduces a large instability in the value of f_{tot}, as a consequence such metric can not be used directly "as such" to trigger an alarm, and some manipulation of the data is needed to discriminate between different cases.

Let us consider a possible ordering function $\psi : i \rightarrow r_i^{\psi}$ for the MS set and denote by $r_i^{\psi} \in \{1..M\}$ the rank of i-th MS. Given a certain ordering function ψ we denote by $f_{\psi}(k), k = 1..M$ the cumulative frequency of retransmission events over the first k MSs, formally:

$$f_{\psi}(k) = \frac{\sum_{i:r_i^{\psi} \leq k} n_i}{\sum_{i:r_i^{\psi} \leq k} N_i} \tag{1.2}$$

Given a specific dataset, each ordering function ψ leads to a different $f_{\psi}(k)$ curve, they all converging to the same value $f_{\psi}(M) = f_{tot}$. Among all the possible ordering possibilities we consider the one that minimizes the function $f_{\psi}(k)$, formally:

$$\overline{\psi} : f_{\overline{\psi}}(k) \leq f_{\psi}(k) \;\; \forall k, \psi \neq \overline{\psi} \tag{1.3}$$

In Appendix .1 we provide an algorithm to find a minimal ordering $\overline{\psi}$ and jointly compute the associated curve $f_{\overline{\psi}}(k)$. From the latter we derive the curve $f(x) = f_{\overline{\psi}}(k/M)$ by normalizing the argument to the total number M of MSs seen in each period, i.e. $x = k/M$. Figure 1.12 provides several sample $f(x)$ curves for LRTO events as computed during the peak-hour periods of different days in the dataset with (upper graph) and without (lower graph) the bottleneck in place. It can be seen that even after the bottleneck removal (lower graph) in some case the value of f_{tot} is inflated by the the contribution of very few MSs. In fact the curve $f(x)$ increases steeply only in the neighborhood of $x \approx 1$ (i.e., $k \approx M$). Also we note that a large fraction of MSs (about 40%) did not experience any LRTO event. At the opposite extreme, before day 20 it is evident that the large value of f_{tot} was not the effect of few biasing MS but rather of a high incidence of LRTO events spanning a broad fraction of MSs (more than 85%).

In summary, in the former case (type A) the high value of f_{tot} is the sporadic effect of one or few "big and bad" MSs, with high traffic volume ("big": high N_i) and high incidence of LRTOs ('bad": high n_i/N_i) - likely due to some local phenomenon like e.g. poor radio conditions - while in the latter case (type B) there

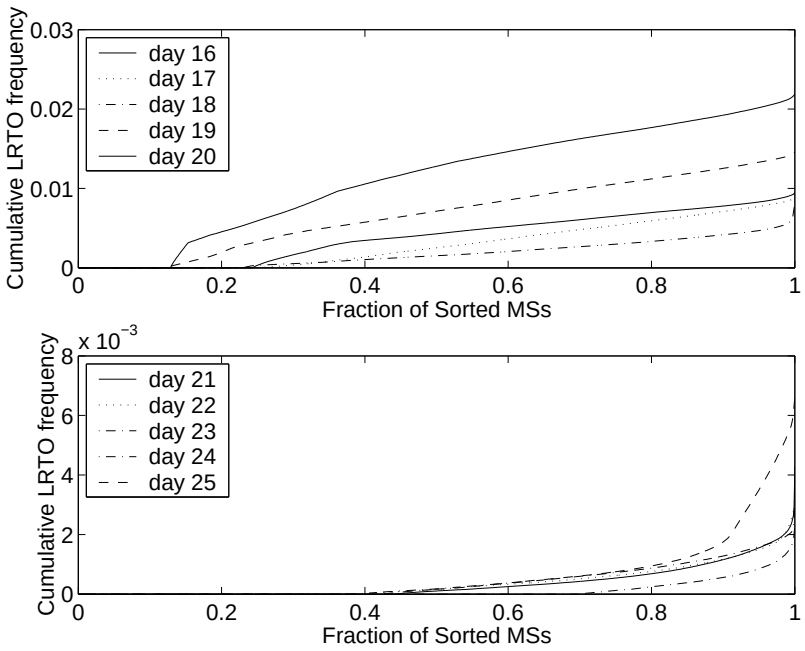

Figure 1.12. $f(x)$ curves in the peak-hour time bins of some monitored days, before (top) and after (bottom) the bottleneck removal.

is evidence of some global phenomenon affecting the whole MS population (i.e. the bottleneck in our case). Clearly, the raw value of f_{tot} as defined in eq. 1.2 is not sufficient to discriminate between the two conditions, and a watermark alarm based on the value of f_{tot} would inevitably cause false alarms when type A events occur (e.g. day 26). Since we are interested in detecting event of type B, we need a more robust indicator, not subject to the instability introduced by few sporadic big-and-bad MSs. One simple possibility would be to simply cut out a small number (say Y) of "worst" MSs, i.e. the right-most tail of the $f(x)$ curve, and then use the value $f^*_{tot} = f_{\overline{\psi}}(\frac{k-Y}{M})$ as the indicator: the previous Figure 1.11 was derived in this way, with $Y = 10$. This approach is similar to the simple heuristic proposed above, where the 10 MSs with the higher value of n_i were filtered out.
An alternative approach is to define a global metric on $f(x)$ that is robust to the impact of few big-and-bad contributors. A possible solution is to use the integral area of the function $f(x)$ as an indicator, formally:

$$A_{\overline{\psi}} = \int_0^1 f(x) = \frac{1}{M} \sum_{k=1}^{M} f_{\overline{\psi}}(k) \tag{1.4}$$

In figure 1.13 we reported the values of the $A_{\overline{\psi}}$ metric for the different types of events.

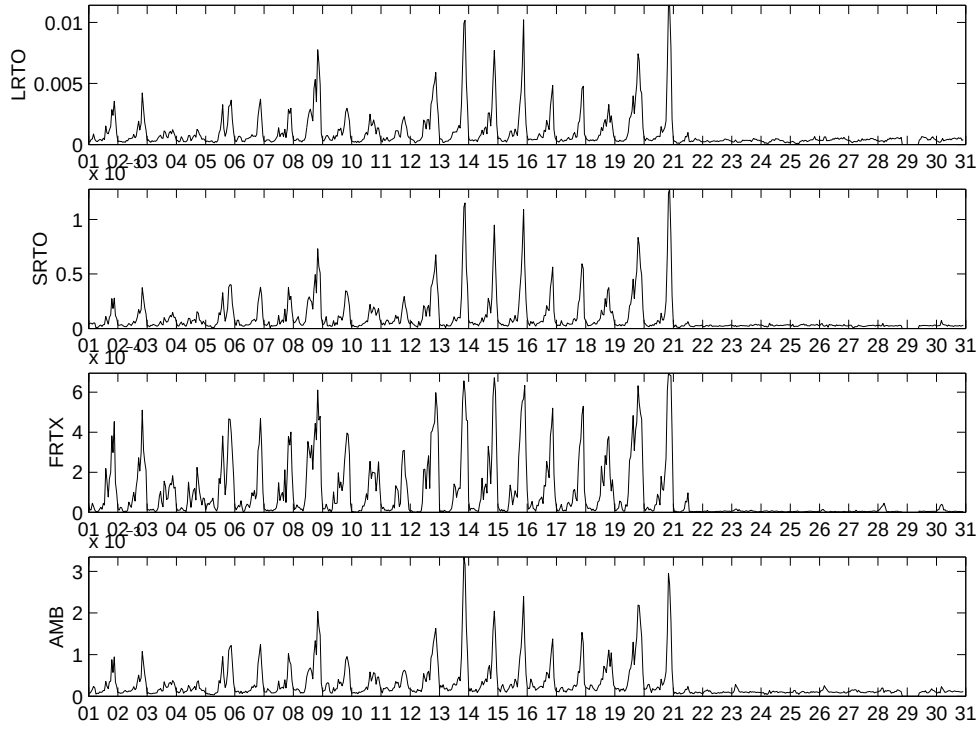

Figure 1.13. Measured values of the $A_{\overline{\psi}}$ indicator

1.6.3 Considerations

The results carried by the above analysis show that it is possible to build powerful bottleneck indicators from performance parameters estimated by passive monitoring TCP traffic. The goodness of an indicator depends on its **predictability** under nominal operational conditions, which in turn requires good stability, and on its **responsiveness** to abnormal conditions. Once that a "good" indicator of the network state is defined, statistical testing methods can be applied to provide automatic alarms when an abnormal condition occurs. From a general point of view, the "better" is the indicator signal in terms of predictability and responsiveness, the simpler can be the statistical testing technique. In the specific case considered here, we found that all the proposed indicators associated to re-transmission events are so effective that even the most simple testing method, i.e. a fixed threshold placed adequately, would have provided early warning about the presence of the bottleneck several days in advance. However this is true only for indicators that have been adequately filtered to counteract the instability and the bias effect of few "big-and-bad" MSs. The approaches proposed here - namely worst-Y filtering and area-based indicators, both based on the minimal ordering function $\overline{\psi}$ -

appear to be very effective for practical applications. However in the end they are heuristic in nature and largely based on intuition. Further refinements are possible in the direction of hypothesis testing methods on the $\{n_i, N_i\}$ process that are more theoretically based, a point that has been left for future work.

Another important performance metric for the practical applicability of the proposed method is its scalability with the traffic volume - number of packets and number of MSs - and the required processing / memory resources. While a formal analysis is beyond the scope of this work, we can report that the overall algorithm could be run in real-time for the total traffic aggregate in the whole network (absolute values can not be disclosed). A complete task includes a `tcptrace` run and post-processing of the output with UNIX scripting language routines (`bash / awk`). The ratio between processing time and real-time is below 1:4 on a high-end power PC (3 GHz processor, 1 GB RAM). Note that the modified version of `tcptrace` includes a patch for optimized memory management. We expect large improvements to be achievable with optimized C code, while distributed processing can be used for very-large networks (e.g. using a separate PC for each GGSN link). The current figures show that these algorithms can be successfully implemented in real-time, and this is indeed one of the future tasks in our project [23].

1.7 Validation

All the graphs reported in the previous sections and specifically Figures 1.6, 1.9 and 1.13 indicate that the identified patterns were stable over many days in both regimes, i.e. with and without the bottleneck. However both the approaches described in Sections 1.5 and 1.6 were based on the empirical analysis of a single bottleneck. In principle, it might be possible that those patterns were due to specific conditions of the bottleneck under analysis, and would not apply in other cases with e.g. different traffic mix, aggregation level, L1/L2 technologies, etc. In this section we present additional results that support the validity and generality of both approaches.

1.7.1 Another bottleneck in the real network

The power of the proposed approaches was corroborated "in the field" by the discovery of other bottlenecks. Since 2004 the prototype of the METAWIN monitoring system has been collecting complete packet-level traffic traces on the Gn links near the GGSN. The traces are completely anonymized as described in Section 1.4 and are kept stored for few days, during which we can perform different types of analysis and the inspection of the global traffic patterns as part of an ongoing research project (DARWIN project [23]). The availability of such data allows the sporadic

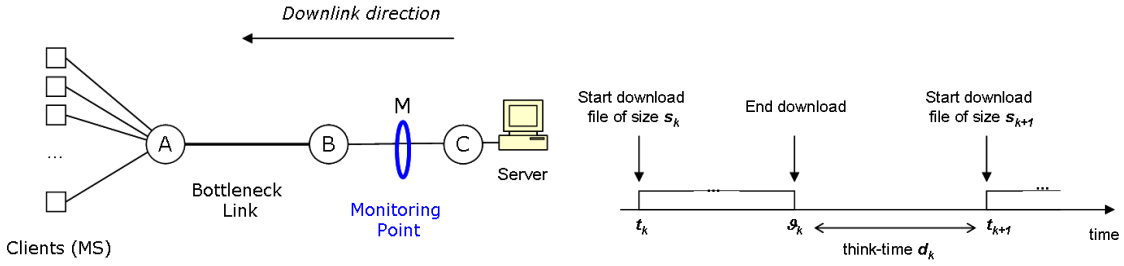

Figure 1.14. Simulation scenario: topology (left) and user model (right).

inspection of the traffic patterns discussed in the previous sections, and specifically of the indicators reported in Figures 1.6, 1.9 and 1.11-1.13. Exact MS discrimination was implemented in the monitoring system, and the filtering mechanisms for TCP indicators were run using unique MS identifiers rather than IP address, as was done in Section 1.6 for the Dec'04 trace. During one of such inspections at the beginning of 2006 we identified the presence of another bottleneck in the Gn network, which was readily removed by the network staff within few hours. The 2006 bottleneck was found on a different link than its 2004 predecessor and at a different aggregation level. After more than one year, the traffic mix and volume has grown and important changes have occurred in the network (e.g. introduction of HSDPA in the UMTS Radio Access Network). Despite these macroscopic differences it is remarkable that the 2006 bottleneck yields very similar patterns (not shown here) to those presented in the Figures 1.6, 1.9 and 1.13 for its predecessor. Note however that in both cases the bottleneck was found on IP-over-ATM rate-limited links.

1.7.2 Simulations

In this section we present simple simulation results in order to validate the applicability of the proposed methods to a more general network scenario. We remark that the goal of these simulations is *not* to reproduce the traffic and network settings in a "realistic" scenario matching all the details observed in the real network. Rather on the contrary, we use the simulation tool to build-up an "abstract" scenario that is purposely different from the observed one, where we retain only few fundamental ingredients that we believe are keys to the emergence of the patterns identified in the previous sections. In other words, the distance between the two different scenarios, namely the real network and an ultra-simplified simulated network, measures the generality of the proposed methodologies as they apply equally well in both cases. All simulations where performed in *ns-2* [3].

1.7.2.1 Simulation scenario

The simulated topology is depicted in Figure 1.14. It consists of a simple bell network, with a bottleneck link AB and the monitoring point M placed upstream on link BC. Each client i is connected to the A node via a dedicated link. For each link the value of the downlink bandwidth is picked randomly among the values 64, 128 and 384 kbps. The value of uplink bandwidth is fixed to 64 kbps. Similarly the one-way propagation delay on each link is extracted uniformly in the range $[100ms, 200ms]$. The choice of such parameter is based on the typical radio channel bandwidth used in UMTS cells, and on the typical RTT measured in the same network (see Figure 1.10). Note that the bottleneck link implements a simple FIFO drop-tail discipline: this is a major point of difference with the two bottlenecks found in the real network, which were instead cell-based rate-limiters at the ATM layer.

The user model is very simple: each user i starts downloading a series of files of size $s_{i,k}$ ($k = 1, 2...$) at times $t_{i,k}$. All files are transferred via TCP, and the $s_{i,k}$ variables are i.i.d. with mean 100 kbytes. Two types of file size distributions are considered: exp-neg and Pareto with tail parameter $\alpha = 1.6$. The completion time of the k-th download, denoted by $\theta_{i,k}$, depends on the network conditions and on the interplay with the other concurrent flows, and increases dramatically in case of congestion. The next download is started at time $t_{i,k+1} = \theta_{i,k} + d$, with d ("think-time") being a random variable extracted from an exp-neg distribution with mean 75 sec. Note that each users waits until the completion of the previous transfer before starting a new download, therefore the overall traffic offered to the network (i.e. the rate of transfer requests) depends on the current network state through the variables $\theta_{i,k}$. Following [8] we denote such user model as "closed-loop".

The virtual simulation time is 24h. During this period the number of concurrent users is varied between 10 and 150 according to a triangular profile obtained as follows. During the first (second) half of the simulation 140 users are activated (deactivated) at random points uniformly distributed in the interval $[3h, 12h]$ ($[12h, 21h]$). We performed different simulations with different values of the bottleneck bandwidth b for the link AB, and for each experiments we applied the analysis presented in the previous sections.

1.7.2.2 Results of aggregate-rate analysis

In order to validate the rate-based approach presented in Section 1.5 we extracted the signal s_τ at the monitoring point M (ref. Figure 1.14) at the granularity of $\tau = 10s$ for different values of the bottleneck bandwidth b. The signals for two sample experiments (b =100 Mbps and b =1 Mbps) are depicted in Figure 1.15. Already the visual comparison between the two signals clearly shows that the pres-

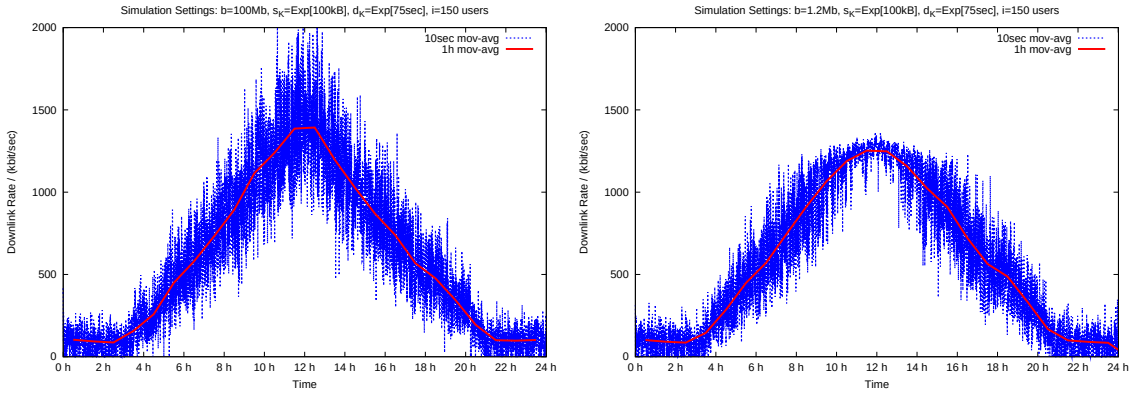

Figure 1.15. Aggregate rate in simulations for different time-granularity (1s, 10s and 1h). Left: no bottleneck (b=100 Mbps); Right: with bottleneck (b=1.2 Mbps)

ence of the bottleneck induces a sharp variance reduction in the signal s_τ at 10s granularity. We applied to the simulated signals the method proposed in Section 1.5 and extracted the Variance-Mean and Skewness-Mean trajectories for different values of b. The resulting curves are depicted in Figure 1.16. They are fully consistent with the patterns found in Figures 1.6. The effect of the bottleneck on the rate trajectories starts to be visible already at b =1.8 Mbps, suggesting that such indicators have a pretty good sensitivity and are able to detect also "light" bottleneck for which the gap between the offered traffic and the available capacity is not large. This is comfortant about the possibility of detecting new bottleneck rather early as they start to appear, i.e. when the growth in offered traffic starts to approach the link capacity.

1.7.2.3 Results of TCP indicators

In order to validate the method based on TCP indicators presented in Section 1.6 we extracted from the *ns-2* simulator the packet-level trace captured at the monitoring point M, and fed them into the modified version of `tcptrace`. In Figure 1.17 we plot the measured indicators in time-bin of 1h and for different values of b. It can be seen that all the RTO indicators, including the SRTO frequency, yield high values during the peak-hour when the bottleneck is in place. The effect of the bottleneck on the TCP indicator starts to be visible for bottleneck bandwidth lower than b =1.5 Mbps.

1.8 Conclusions and Ongoing Work

In this work we have suggested two possible approaches to bottleneck detection based on passive traffic monitoring. The first method is based on the statistical analysis of the aggregate rate. The second method is based on TCP performance indicators. Both methods are relatively simple to implement.

A point of caution is due regarding the generalization of the results: it can not be expected that the detection mechanism developed out of few specific events will necessarily capture *all* possible cases. Different types of resource limitations (e.g. cell-level rate-limiter, packet buffering shaper, FIFO drop-tail) can expose different "signatures" on TCP performance indicators. Different network environments might have different network-level parameters (e.g. per-user bandwidth, RTT) and support different user populations using different application mix, which might have an impact on the choice of the time-scales τ and w for the rate-based scheme. Nevertheless the fact that the same patterns found in two different real-world bottlenecks were reproduced by an ultra-simplified simulation scenario, with different user model and different types of resource limitation (ATM rate-limiter vs. FIFO drop-tail) is pretty comfortant about the robustness of the proposed approach to different network conditions.

From a practical point of view, we believe that bottleneck detection schemes to be used in production networks should be based on a bank of parallel tests on different indicator signals and parameters, in order to increase the likelihood that future events are not missed. Also, as new types of events are found during the network operation and their signature are investigated ("post-mortem analysis"), the set of detection algorithms can be extended with new tests, in a sort of continuous learning process that is principle similar to the way Intrusion Detection Systems (IDS) and Antivirus software evolved. The prerequisite for this kind of studies is the availability of complete TCP/IP header information from packet traces. We are currently implementing these algorithms on top of the on-line monitoring system developed in the METAWIN project [23], and they are being used experimentally in the production network of mobilkom austria AG & Co KG.

On the theoretical side, the natural continuation of this work is to devise a framework for locating the performance bottlenecks from the comparative analysis of the SAs at different hierarchical levels. This approach holds some similarities to network tomography [7], since the common idea is to infer the internal state of the network from external measurements ("black-box" approach). Note that in all the considered cases - in the real traces as well as in the simulation scenario - the whole SA crossing the bottleneck can be observed at the the monitoring point. This is a direct consequence of the tree-like topology of the 3G Core Network. As a future work we will investigate the applicability of the proposed scheme in more general network topologies, where the monitoring point captures only a fraction of the SA traffic that eventually crosses congested link. We are also investigating

the applicability of the proposed schemes to detect bottleneck in the 3G Radio Access Network. We expect that for lower-granularity SAs (per-cell) bottleneck diagnosis based on the aggregate rate becomes more challenging due to the lower level of flow aggregation, which makes the case for preferring TCP performance indicators (for a preliminary report on this activity see [14]).

This work was made possible by the availability of complete traces under different network conditions, which allows the exploration and comparison of the *traffic* behavior with and without the bottleneck. An intriguing direction for further study would be to analyze the behavior of the *users* in the two scenarios. For instance a comparison of the connection interruption rate and other parameters related to user patience (see [11]) would enable a more concrete assessment of the real impact of the bottleneck as perceived by the users.

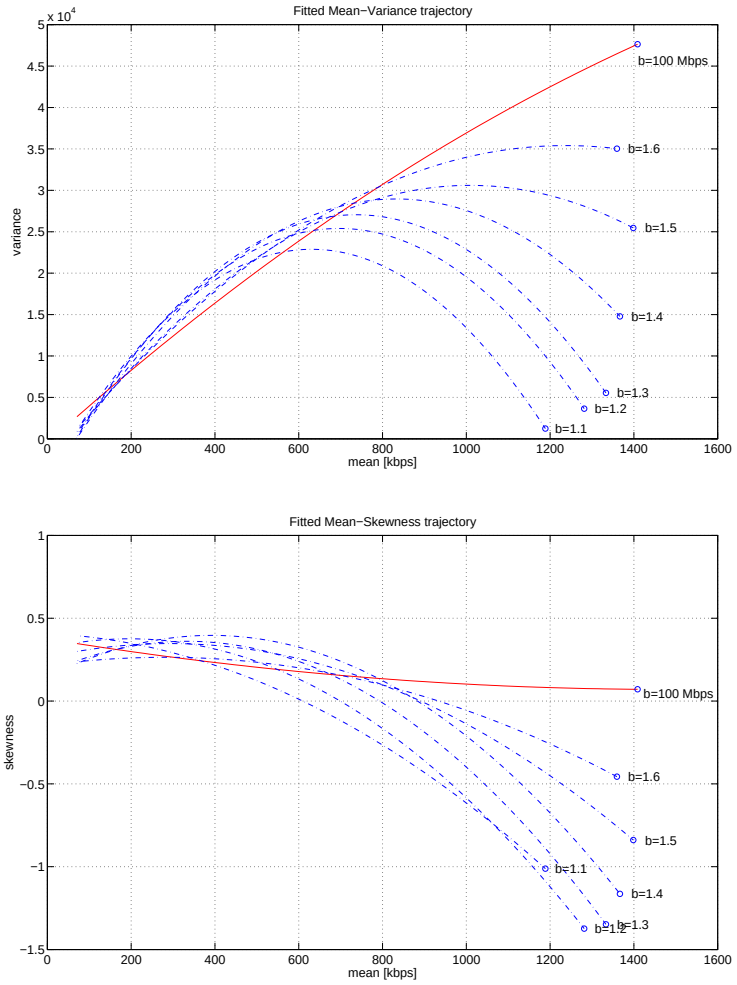

Figure 1.16. Fitted trajectories in simulations: Variance-Mean (left) and Skewness-Mean (right). The upper curve (b =100 Mbps) is the reference case (without bottleneck).

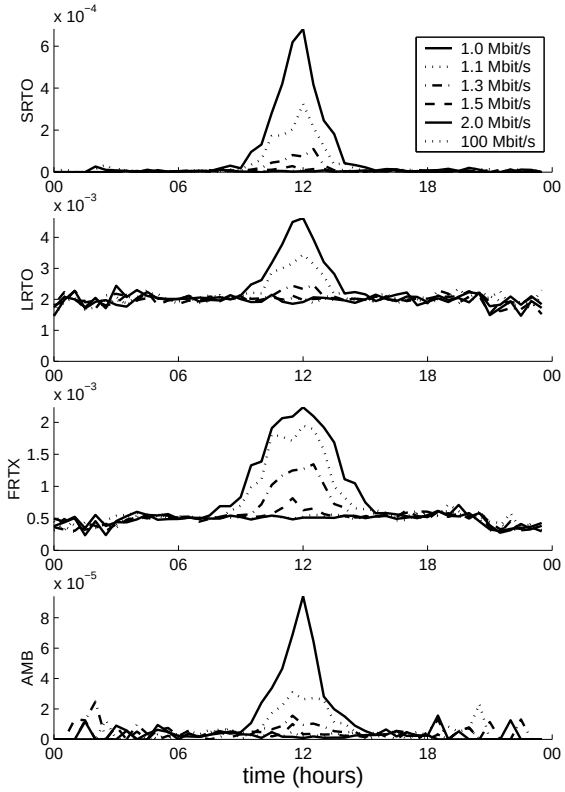

Figure 1.17. TCP indicators extracted from simulations.

Acknowledgment

This work is part of the DARWIN project [23] supported by the Austrian research program Kplus and co-financed by mobilkom austria AG & Co KG and Kapsch CarrierCom. The authors would like to acknowledge the developers of the monitoring system, particularly E. Hasenleithner, R. Pilz and P. Krueger: without their work this research would have not been possible.

References

[1] Endace Measurememt Systems http://www.endace.com.

[2] Tcptrace 6.6.1 home page *http://www.tcptrace.org*. The modified version is available at *http://userver.ftw.at/~/vacirca* .

[3] The Network Simulator homepage. *http://nsnam.isi.edu/nsnam*.

[4] The WiMAX forum *http://www.wimaxforum.org* .

[5] Tstat web page. *httcp://www.tlc-networks.polito.it/Tstat* (2001).

[6] C. DOVROLIS, P. RAMANATHAN, D. MOORE. Packet dispersion techniques and capacity estimation. *IEEE/ACM Trans. on Networking 12*, 6 (December 2004).

[7] R. CASTRO, M. COATES, G. LIANG, R. NOWAK, BIN YU. Network tomography : Recent developments. *Statistical Science 19*, 3 (2004).

[8] A. DHAMDHERE, C. DOVROLIS. Open issues in router buffer sizing. *ACM Computer Communication Review 36*, 1 (January 2006).

[9] BENOIT B. MANDELBROT. *Fractals and Scaling in Finance: Discontinuity, Concentration, Risk.* Springer, 1997.

[10] C.C. ZOU, W. GONG, D. TOWSLEY. Code Red Worm Propagation Modeling and Analysis. *Proc. of the 9th ACM conference on Computer and Communications Security (CCS'02), Washington, DC, USA.* (November 2002).

[11] D. ROSSI, M. MELLIA, C. CASETTI. User patience and the web: a hands-on investigation. *Globecom* (2003).

[12] F. RICCIATO. Unwanted Traffic in 3G. *Editorial for ACM Computer Communication Review 36*, 2 (April 2006).

[13] F. RICCIATO, F. VACIRCA, M. KARNER. Bottleneck Detection in UMTS via TCP Passive Monitoring : A Real Case. *Proc. of CoNEXT conference, Tolouse, France* (October 2005).

[14] F. RICCIATO, F. VACIRCA, W. FLEISCHER, J. MOTZ, M. RUPP. Passive Tomography of a 3G Network: Challenges and Opportunities. *Poster at IEEE INFOCOM'06* (April 2006).

[15] F. RICCIATO, W. FLEISCHER. Bottleneck Detection via Aggregate Rate Analysis : A Real Case in a 3G Network. *Short paper at IEEE/IFIP NOMS'06, Vancouver* (April 2006).

[16] F. VACIRCA, F. RICCIATO, R. PILZ. Large-Scale RTT Measurements from an Operational UMTS/GPRS Network. *1st Int'l Conference on Wireless Internet (WICON'05), Budapest* (July 2005).

[17] F. VACIRCA, T. ZIEGLER, E. HASENLEITHNER. An Algorithm to Detect TCP Spurious Timeouts and its Application to Operational UMTS/GPRS Networks. *To appear in Computer Networks*.

[18] J. BANNISTER, P. MATHER, S. COOPE. *Convergence Technologies for 3G Networks*. Wiley, 2004.

[19] K. AHMAVAARA, H. HAVERINEN, R. PICHNA. Interworking Architecture Between 3GPP and WLAN systems. *IEEE Communications Magazine* (November 2003).

[20] K. LAI, M. BAKER. Nettimer: A Tool for Measuring Bottleneck Link Bandwidth. *Usenix* (2001).

[21] M. MELLIA, A. CARPANI AND R. LO CIGNO. Measuring IP and TCP behavior on Edge Nodes. *IEEE Globecom, Taipei* (Nov 2002).

[22] M. THOTTAN, C. JI. Anomaly Detection in IP Networks. *IEEE Trans. on Signal Processing 51*, 8 (August 2003).

[23] METAWIN AND DARWIN PROJECTS. *http://userver.ftw.at/~ricciato/darwin*.

[24] P. HUANG, A. FELDMANN, W. WILLINGER. A nonintrusive, wavelet based approach to detecting network performance problems. *Proc. of the Int'l Measurements Workshop (IMW'01)* (2001).

[25] P. SVOBODA, F. RICCIATO, E. HASENLEITHNER, R. PILZ. Composition of GPRS/UMTS traffic : snapshots from a live network. *4th Int'l Workshop on Internet Performance, Simulation, Monitoring and Measurement, Salzburg (IPS-MOME'06)* (February 2006).

[26] PAUL BARFORD, JEFFERY KLINE, DAVID PLONKA AND AMOS RON. A signal analysis of network traffic anomalies. *Proc. of the 2nd ACM SIGCOMM Workshop on Internet Measurement Conference* (Nov 2002).

[27] PAXSON, V., AND FLOYD, S. Wide area traffic: the failure of Poisson modeling. *IEEE/ACM Transactions on Networking 3*, 3 (1995).

[28] PETER BENKO AND ANDRAS VERES. A Passive Method for Estimating End-to-End TCP Packet Loss. *IEEE Globecom, Taipei* (Nov 2002).

[29] R. PANG, V. YEGNESWARAN, P. BARFORD, V. PAXSON, L. PETERSON. Characteristics of Internet Background Radiation. *Proc. of the Int'l Measurements Conference (IMC'04), Taormina, Italy* (October 2004).

[30] R. S. PRASAD, M. MURRAY, C. DOVROLIS, K. CLAFFY. Bandwidth Estimation: metrics, measurement techniques and tools. *IEEE Network* (Nov/Dec 2003).

[31] S. JAISWAL ET AL. Inferring TCP Connection Characteristics Through Passive Measurements. *IEEE INFOCOM 2003* (Apr 2003).

[32] S. JAISWAL ET AL. Measurement and classification of out-of-sequence packets in a tier-1 ip backbone. *IEEE INFOCOM 2003* (Apr 2003).

[33] S. KATTI, D. KATABI, C. BLAKE, E. KOHLER, J. STRAUSS. MultiQ: automated detection of multiple bottleneck capacities along a path. *Proc. of the Int'l Measurements Conference (IMC'04), Taormina, Italy* (October 2004).

[34] S. KESHAV. Why Cell Phones Will Dominate the Future Internet. *Editorial for ACM Computer Communication Review 35*, 2 (April 2005).

[35] W. RICHARD STEVENS. *TCP/IP Illustrated - Volume 1: The Protocols*. Addison-Wesley, 1994.

.1 APPENDIX: Minimal ordering algorithm

Given a process $\{n_i, N_i\}$ the problem is to find the minimal ordering function that minimizes the function $f_\psi(k) = \frac{\sum_{i:r_i^\psi \leq k} n_i}{\sum_{i:r_i^\psi \leq k} N_i}$ (see eq. 1.2) for each value of k. We propose an algorithm that find the minimal ordering and at the same time computes the minimal function $f_\psi(k)$ iteratively. The algorithm uses the following data structures:

- R_j: the set of MSs with an assigned rank r_i after j iterations;
- V_j: the residual set of MSs that have not yet been assigned a rank;
- H: the current set of candidate elements with $n_H := n_{h_1} = n_{h_2}$, $N_H := N_{h_1} = N_{h_2}$, $\forall h_1, h_2 \in H$.

At bootstrap (step 0), the set R_0 is initialized with all those MSs with $n_i = 0$ and f_0 set to 0. At each step j, we have to select the set of elements $H \subseteq V_j$ that minimizes $f_{j+1} = \frac{\sum_{i \in R_j} n_i + n_H}{\sum_{i \in R} N_i + N_H}$ and that will be included into R_{j+1}. A brute-force approach

would simply try all possible elements of V_j, i.e. compute all possible values of f_{j+1} and pick the minimum one. A more effective strategy can be implemented on the basis of the observation that each generic element $x \in V_j$ implicitly identifies a subset of elements in V_j, specifically those having $n_i \geq n_x$ AND $N_i \leq N_x$, that would certainly hold a larger value of f_{j+1}. The algorithm will then exploit such property to immediately exclude such elements from the set of candidate elements. At each iteration j the following steps are performed:

1: $H := \{y \in \{x \in V_j \parallel N_x \text{ is maximum } \} \parallel n_y \text{ is minimum } \}$[3];
2: $V_{tmp} := \{x \in V_j \parallel N_x < N_H n_x < n_H\}$;
3: $f_{tmp} = \frac{\sum_{i \in R_j} n_i + n_H}{\sum_{i \in R_j} N_i + N_H}$;
4: **while** $(V_{tmp} \neq \emptyset)$ **do**
5: $\quad C := \{y \in \{x \in V_{tmp} \parallel N_x \text{ is maximum } \} \parallel n_y \text{ is minimum } \}$;
6: $\quad V_{tmp} := \{x \in V_{tmp} \parallel N_x < N_C n_x < n_C\}$;
7: $\quad f_{new} = \frac{\sum_{i \in R_j} n_i + n_C}{\sum_{i \in R_j} N_i + N_C}$;
8: $\quad$ **if** $(f_{new} < f_{tmp})$ **then**
9: $\quad\quad H := C$
10: $\quad\quad f_{tmp} = f_{new}$
11: $\quad$ **end if**
12: **end while**
13: $V_{j+1} := \{x \in V_j \parallel x \notin H\}$
14: $R_{j+i} := R_j \bigcup H$
15: $f_{j+1} = \frac{\sum_{i \in R_j} n_i + z \cdot n_H}{\sum_{i \in R} N_i + z \cdot N_H}$, where z is the cardinality of H.

The algorithm stops at the n-th step when V_n is empty.

[3]The $\parallel$ symbol means: "such that".