Network performance assessment using adaptive traffic sampling*

René Serral-Gracià, Albert Cabellos-Aparicio, Jordi Domingo-Pascual
{rserral, acabello, jordid}@ac.upc.edu

Advanced Broadband Communications Centre, Technical University of Catalunya (UPC), Spain

Abstract. Multimedia and real-time services are spreading all over the Internet. The delivery quality of such contents is closely related to its network performance, for example in terms such as low latency or few packet losses.
Both customers and operators want some feedback from the network in order to assess the real provided quality. There are proposals about distributed infrastructures for the on-line reporting of these quality metrics. The issue all these infrastructures must face is the high resource requirements to keep up with accurate and live reporting.
This paper proposes an adaptive sampling methodology to control the resources needed for the network performance reporting. Moreover, the solution keeps up with accurate live reporting of the main network performance metrics. The solution is tested in a European wide testbed with real traffic which permits to assess the accuracy and to study the resources consumption of the system.

Key words: Traffic sampling, Adaptive sampling, Quality assessment, Network Metrics, Distributed traffic monitoring

1 Introduction

Since the apparition of multimedia tools such as *Skype* or *Ekiga* the Internet became a multimedia aware system. Using these services usually comes with an associated fee. Hence, the clients require some guarantees in the Service Level Agreement (SLA). This raises the need of providing some feedback to the customers about the performance given by the network, nowadays achieved through the so called network quality metrics.

Therefore on-line traffic monitoring infrastructures for network performance assessment are quickly spreading. The main goal of such systems is to efficiently provide a distributed intra and inter-domain infrastructure where to evaluate the network metrics.

Distributed network monitoring infrastructures can be divided into two main groups: high and low level. On the one hand, high level systems deliver information about the network's status by focusing on the study of its capacity and availability, while providing means to visually monitor the status of the traffic. On the other hand, low level approaches focus on direct traffic observation for assessing the network performance.

* This work was partially funded by IST under contract 6FP-004503 (*IST-EuQoS*), MCyT (Spanish Ministry of Science and Technology) under contract TSI 2005-07520-C03-02 and the CIRIT (Catalan Research Council) under contract 2005 SGR 00481.

These systems may use a distributed infrastructure in order to gather the network metrics. This requires of certain control traffic to maintain the structures for the live reporting.

Our solution uses a low-level approach, which instead of providing information about the availability of the network or gathering information for off-line processing, it provides real-time information about the following network metrics: One-Way Delay (RFC-2679), Packet Loss Ratio (RFC-2680) and bandwidth usage. These metrics are among the most used, as stated on ITU-T Rec. Y.1540, and help operators to monitor and verify the status of their network. In order to achieve this we proposed the *Network Parameter Acquisition System* (NPAS) [1], a distributed monitoring infrastructure that collects and analyses the traffic on-line in an intra-domain environment. NPAS collects the traffic in all the different ingress and egress points of the domain under analysis, monitors the desired flows and sends the collected information to an analysis unit. The analysis unit computes the metrics by using the collected information through timestamps and packet sizes.

Gathering this information in a per packet basis is very expensive in terms of bandwidth and processing resources needed by the control traffic. In our previous work we proposed mechanisms that used traffic aggregation [1] and traffic static sampling techniques [2] in order to reduce the resource consumption. We plan to further reduce the used resources with an adaptive sampling technique.

The novel proposed distributed adaptive sampling dynamically adapts the sampling rate depending on the traffic conditions, reducing the resources needed by the control traffic down to 4% compared to the case of per packet reporting. Specifically our contribution uses the IP Delay Variation in order to estimate the accuracy of the sampled results, adjusting the sampling rate accordingly with minimum loss in accuracy. Moreover, the technique efficiently controls the use of resources, scheduling the sampling rates as required by the system.

The system is tested in a European wide scenario by using 12 different locations with different traffic profiles. The results show that the enhancements permit a tight control over the used resources for the reporting and an increase in the accuracy compared to the case of static sampling.

The rest of the paper is structured as follows, next section focuses on the explanation of the already existing solutions to perform distributed network performance assessment, centring on the most similar to our work. The paper continues by describing the state of the art in network parameter acquisition system and the current static sampling solution. After this the paper centres on the main proposal, which is an adaptive sampling solution to schedule the resources used for the live network performance assessment. This approach is validated in section 5, where tests in a real scenario are performed. The paper finishes with the conclusions and the work left for further study.

2 Related Work

Traffic sampling is a broadly deployed technique used to reduce the resources required for traffic analysis [3,4]. In general the sampling schemes are static [5], where a sampling rate is decided in advance and applied during all the study. Others use adap-

tive methods [6] which decide dynamically the best sampling rate over time, adapting to the changes suffered by the system.

Given the distributed nature of our scenario, the applicable sampling schemes are limited. Duffield et. al developed a sound methodology, namely trajectory sampling [7], based on hash sampling, which was used to infer packet paths simultaneously with different collection points. This schema has been used several times in other scenarios [2, 4, 8] using static sampling rates. As it has been said before, we propose a novel adaptive sampling technique based on the trajectory sampling, but relying on the IP Delay Variations (IPDV). We apply this technique on a distributed scenario used to monitor specific network performance metrics.

Regarding distributed network performance assessment proposals, at the best of our knowledge very few tools have been published. Firstly, perfSONAR [9] is a tool intended to monitor any network metric on a distributed fashion. Specifically the authors present a methodology that provides meaningful network performance indicators, which can be used to visually monitor the status of the traffic. The main difference between this tool and our proposal is that while perfSONAR limits the study to the link status by active traffic generation, NPAS analyses directly the network metrics by passively collecting the traffic. Secondly InterMON [10] is based on interaction and coordination of different kinds of tools with access to a common database. The system's main goal is to gather network metrics for off-line processing, analysis and data-mining. In this case, the main difference between InterMON and our proposal is that we are focused on providing live values for some network metrics not just gathering information for off-line processing.

3 Description of Network Parameter Acquisition System

Network Parameter Acquisition System as proposed in [1] is a distributed network performance assessment infrastructure. It computes the network metrics suitable to quantify the network performance over sensible traffic (e.g. VoIP or Videoconferencing) on-line.

The system collects the traffic on each ingress and egress points in the network, and only analysing the traffic selected by the configured filters. It computes the following network metrics: One-Way Delay (OWD), Packet Loss Ratio (PLR) and used bandwidth.

3.1 Basic system

This infrastructure presents a framework which reports the intra-domain traffic metrics to higher layer entities to assess whether the specified performance constraints (policies) are fulfilled or not. This mechanism can also be used for triggering alarms to give feedback to other system units in case that the network is not providing the contracted quality . Both alarms and policies management are out of the scope of this specification.

Figure 1 shows a generic network scenario where the system is deployed. It is composed by the following entities:

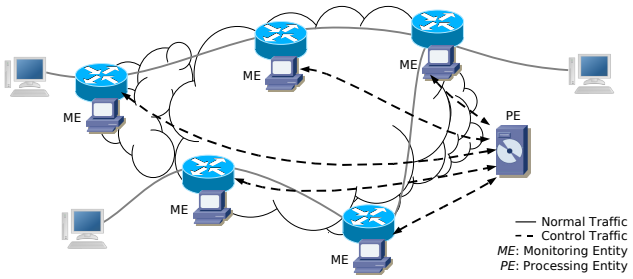


Fig. 1. Example Scenario

The Monitoring Entity (ME) is in charge of the traffic collection via selection filters. The traffic selection policy aggregates the data in a per flow or per Class of Service (CoS) basis, but it can be easily extended to others.

The traffic information collected by MEs is sent to a per-domain analyser entity *(Processing Entity - PE)* which computes the network metrics of all the traffic under analysis.

The Processing Entity (PE) is the gathering point within the domain. It configures the ME policies, performs most of the processing, matches the information of the packets coming from the MEs and computes the network metrics on-line. The results are published as a network management service to monitor the status of the traffic under analysis in real-time.

Depending on the information sent by the ME more control traffic is required to perform the monitoring. In general this can be a problem for under provisioned domains without dedicated links for network management traffic, we believe that this is the case in most ISPs. NPAS controls this issue by having three different modes of operation: *i)* per packet reporting, where the traffic information is sent in a per packet basis. *ii)* using time aggregation, where a time sliding window is used to optimise the network usage. *iii)* by using traffic sampling, as further discussed in the section 3.2.

More detail in the specification of the first two modes and the analysis of the bandwidth usage can be found at [1]. In this paper we focus on the traffic sampling mode of operation.

3.2 Static distributed sampling

The complexity of applying sampling in this distributed scenario is that centralised techniques (e.g. the techniques studied in [11]) are not well suited for this task, since they do not guarantee that all the ME capture exactly the same set of packets. Thus, accurately determining packet losses or one way delays is not feasible. We addressed this issue in [2] by using a deterministic sampling technique. It was used to match exactly the same packets all over the different ME to compute the network metrics. This

permits to all the different ME to collect traffic independently, and only the selection function at start up time has to be shared by the MEs.

This distributed sampling technique computes a hash function over a set of fields on the packet's header. For this to work in our environment, using only one hash table is not sufficient, since the monitoring framework has to guarantee that all the flows under analysis are considered. So, the sampling must be applied within individual flows, not directly to all the collected traffic. Hence, the packets are identified by using two different keys, *flow* and *packet* with the corresponding two level hash table. Thus, a packet is only analysed if it falls within specified positions in the hash table (A), of which only the first r packets are considered. Changing r gives different sampling rates $(\frac{r}{A})$. The hash table is sent from each ME to the PE when A packets have arrived or after a timeout t. This time interval t is known as a bin and it limits the upper bound for the live reporting interval. This permits to efficiently control the resources and the applied sampling rate.

4 Adaptive Distributed Sampling

Using traffic sampling in order to reduce the required resources is usually a good option. But choosing the optimal sampling rate with minimum loss of accuracy is not straight-forward.

Moreover, knowing that control traffic (the reporting information from MEs to PEs) must be limited on the network, it might force the per flow sampling rate to be further reduced. Hence, selecting the right flows to reduce the sampling rate leads to better results than just equally sharing the sampling rate among the flows.

The methodology presented here permits to efficiently estimate the One-Way Delay and Packet Loss Ratio using adaptive sampling.

4.1 Problem Formulation

Given a domain $\mathcal{D}$ with $\mathcal{R} = \{r_1, \ldots, r_n\}$ ingress and egress points of the network, and $\mathcal{F}_\mathcal{D} = \{f_1, \ldots, f_m\}$ the list of active flows within the domain. Then $R_{f_i}(t)$ (R_i from now on) is the rate of the i^{th} flow at time t, being n the number of ME and m the number of flows.

For simplicity we consider through out the paper that the resources for a ME is the bandwidth required to send information to the PE.

The resources required to monitor all the existing flows from ME i to ME j are

$$x'_{ij} = \mathcal{S} \sum_{k=1}^{m} \delta_{ijk} R_k \tag{1}$$

where $\mathcal{S}$ is a constant defining the amount of resources needed to process (and send) a single control packet, for example the number of bytes. As it is a constant, for the ease of exposition we will assume $\mathcal{S} = 1$. And $\delta_{ijk} = 1$ if f_k goes from r_i to r_j and 0 otherwise. From this we derive that the resources required for each ME (X'_i) are $X'_i = \sum_{j=1}^{n} \left(x'_{ij} + x'_{ji} \right)$. Hence, reporting in a per packet basis implies that the total amount of resources (X) required for on-line monitoring in domain $\mathcal{D}$ is: $X_\mathcal{D} = \sum_{i=1}^{n} X'_i$.

In general per packet reporting requires too much resources to be feasible. We ease this requirement by applying adaptive traffic sampling to the solution. We consider that $X_{\mathcal{D}}$ are the global resources available to perform the collection and performance assessment. We need a fair share of the resources among all the ME, considering that ME with more load (e.g. with more number of monitored flows) deserve more of the resources. Hence, we need to adapt the per flow sampling rate (ρ) in order to comply with this restriction:

$$X_{\mathcal{D}} \geq X \geq \mathcal{S} \sum_{i=1}^{m} \rho_i R_i \tag{2}$$

where ρ_i is the sampling rate $(0 \leq \rho_i \leq 1)$ of flow i, R_i the rate in packets per second and X the maximum resources reserved for the monitoring. Note that R_i (in packets/sec) is known since the ME initially must collect all the traffic to decide whether it falls within the specified threshold set in the hash table as described in 3.2.

We need a lower bound of the sampling rate, namely $\rho_{i_{min}}$, which guarantees that it is adjusted depending on the rate in a per flow basis, since low rate flows at small time scales are very sensible to sampling rates. This lower bound determines the minimum required resources for the whole monitoring task. Thus,

$$X_{min} = \sum_{i=1}^{m} \rho_{i_{min}} R_i \tag{3}$$

Where $\rho_{i_{min}}$ is the minimum ρ_i for flow i. This works well if the traffic has constant rates, but since traffic in general is variable these values need to be updated periodically. Moreover, new and expired flows are bound to appear over time. Both the flow management and the update strategy will be discussed later.

The minimum applicable sampling rate has to take into account the rate of the flows, limited by expression $\rho_{i_{min}} = \frac{1}{R_i} \quad \forall i \setminus 1 \leq i \leq m$.

Where $\rho_{i_{min}}$ is the minimum applicable sampling rate for all f_i. This guarantees at least that one packet per flow is captured. If c_{min} is the number of samples per f_i then,

$$\begin{aligned} \rho_{i_{min}} &= 1 \quad \forall i \setminus c_{min} \geq R_i \\ \rho_{i_{min}} &= \frac{c_{min}}{R_i} \quad , otherwise \end{aligned} \tag{4}$$

when more precision is required or more resources are available, assuring that at least c_{min} packets per flow are considered.

In the same way we can define the maximum sampling rate to

$$\begin{aligned} \rho_{i_{max}} &= 1 \quad \forall i \setminus c_{max} \geq R_i \\ \rho_{i_{max}} &= \frac{c_{max}}{R_i} \quad , otherwise \end{aligned} \tag{5}$$

where c_{max} determines the maximum number of per flow packets to be collected. Then $X_{max} = \sum_{i=1}^{m} \rho_{i_{max}} R_i$ and $X_{max} \leq X \leq X_{\mathcal{D}}$.

In order to ease the exposition we define $x_{i_{min}}$ and $x_{i_{max}}$ which refer to the minimum and maximum resources for a particular flow respectively.

4.2 Resource Sharing

With equations 2 and 3 we know that after X_{min} are used, the remaining available resources (ΔX) to share among the flows are

$$\Delta X = X - X_{min} \tag{6}$$

The goal now is to share the ΔX resources. Let's define ω_i as the weight assigned to the flow i where $\sum_{i=1}^{m} \omega_i = 1$, then the resources assigned to the flow follow equation 7.

$$x_i = \min\{\lfloor \Delta X \omega_i \rfloor + x_{i_{min}}, x_{i_{max}}\} \tag{7}$$

It can be noted that the resources assigned to the flow are bounded by the X_{max} as assigned previously. Now the weights (ω) may be assigned in different ways, in this work as a proof of concept we study IPDV driven sharing.

IPDV driven The ΔX resources need to be shared smartly. Obviously it is much more useful to increase ρ_i for flows with less accuracy on the metric estimation. But the estimated error is unknown in advance, so it is not possible to determine the sampling rate deterministically. The proposed mechanism uses $\widehat{IPDV}$ as a measure of the accuracy of the estimator.

Before detailing the methodology, it is important to define $IPDV$ in our context. $IPDV$ usually is defined as $OWD_i - OWD_{i-1}$, excluding lost packets (See RFC-3393), or as $OWD_{99.9^{th}prc} - OWD_{min}$ in Rec. ITU-T 1541. In our work the first definition is not applicable since the packet sequence is lost during the sampling. Regarding the second option, obtaining the 99.9^{th} percentile for low rate flows is not possible because there are too few packets in the bin. As an alternative, RFC-3393 states that it is possible to specify a custom selection function to compute $IPDV$. Hence we use $IPDV_i = |\widehat{OWD} - OWD_i|$, for all the i packets within the bin. Where the estimator $\widehat{OWD} = E[OWD]$ for all the sampled packets.

Using the $IPDV$ as an accuracy estimator is based on the following rationale:

1. Constant OWD leads to null $IPDV$, where both $\widehat{OWD}$ and $\widehat{IPDV}$ can be perfectly estimated using $\rho_{i_{min}}$.
2. High $IPDV$ with low ρ_i might lead to inaccurate $\widehat{OWD}$ since the subset of packets selected by ρ_i may not be representative enough. Hence it is worth to increase ρ_i for high variable flows.
3. But, in some cases, with ρ_i we can incorrectly estimate small $\widehat{IPDV}$ due to the bias of the selected subset. This false negative will be corrected in later bins as $\widehat{IPDV}$ converges.
4. The other case is incorrectly estimating high $\widehat{IPDV}$ when in reality it is low. That being a false positive and forcing a needless increase of ρ_i for the flow. But not leading to incorrect estimation.

The probability of false positives or false negatives decreases as ρ_i increases. Then the system automatically tunes ρ_i reducing these mistakes in the estimation. Therefore, the different weights (ω_i) are distributed using:

$$\omega_i = \frac{\widehat{IPDV}_i}{\sum_{j=1}^{m} \widehat{IPDV}_j} \tag{8}$$

Fairly scheduling the ΔX resources proportionally to its delay variation.

4.3 Update strategy

Flow rates vary over time, ideally in our approach we need the rate for all flows in advance. Since this is not possible in a distributed scenario, our proposal uses a periodic updating mechanism divided into these steps:

1. *Warm-up phase:* initially the PE assigns x_{min} resources to the system.
2. *Working phase:* in each step (bin of size t) the PE collects the rates and the estimated network metrics and computes the corresponding c_i for each flow f_i as explained before. This c_i will be effective on the next step.

Since the rate (r_i) might vary from one bin to the other, the estimated effective rate considered in the analysis is $\hat{R} = \frac{\sum_{i=1}^{k} R_i}{k}$, where k is the history of the system.

4.4 New and expired flows

Since users connect to and disconnect from many services, continuously new flows arrive and others end in a domain. In our system the effects of such behaviour are that expired flows release resources for the reporting, while new flows force a rescheduling of the already shared ones.

In the case that a flow does not have any active packets on the bin, its resources are not used in the current step. Note that increasing the ρ for other flows would lead to incorrect estimation as the other ME receiving (or sending) the flow are not aware of the situation. In the next bin, the flow without active packets will be assigned c_{min} resources and will enter in the *Warm-up phase* again. If a flow stays in *Warm-up phase* during several consecutive bins it is considered as expired, hence it is removed from the resource scheduling.

Another condition to be considered is the apparition of new flows on the system. When this occurs, there is no time for the ME to ask the PE for a rescheduling of the resources, so the ME enters in *local rescheduling mode*.

We know that each ME has X_i resources dedicated to it. Therefore to accommodate the new flow (x_j) the ME has the following alternatives:

1. There are flows without active packets in the bin: in this situation c_{min} resources from that inactive flow are used for the new. Equation 7 guarantees that the resources are present.
2. All the flows have active packets: ME (r_j) searches for the $max(x_i)\ \forall i\ x_{1\ldots m}$ and shares its resources in the following way: $x_{i_{new}} = x_i - c_{min}$ and $x_{j_{new}} = c_{min}$. Forcing $x_{j_{new}}$ to enter in *Warm-up phase*.

The PE can fairly reallocate the resources to fit the new situation in the next step.

5 Tests and Results

The adaptive sampling methodology proposed in this paper has been validated by a set of more than 500 experimental tests performed during 2006 using twelve different testbeds across Europe. The testbeds were provided by EuQoS [12] partners, covering a total of 5 countries and 4 different access technologies (LAN, xDSL, UMTS and WiFi) [13] with an overlay architecture on the Géant research network. For this work the testbed is configured to act as a single domain, with one ME deployed on each testbed, amounting to 12 ME and 1 PE on this scenario.

5.1 Validation Tests

The tests were performed by actively generating synthetic periodic traffic, emulating different applications, namely VoIP and Videoconferencing; with a broad set of combinations of different packet rates and packet sizes, from 16 up to 900 packets per second, with a random duration between 1 and 10 minutes. A wide range of packet loss and delay variation conditions were encountered given the different cross-traffic found on the network at different days, hours and physical location of each testbed.

The goal of the evaluation is twofold, in the one hand we compute the accuracy in the estimation of OWD and PLR. On the other hand, we analyse the used resources for the task.

For each test the full trace was collected. With the traces we simulated a network with the flows entering and leaving the system following an uniform randomly distribution up to a maximum of 100 simultaneous flows.

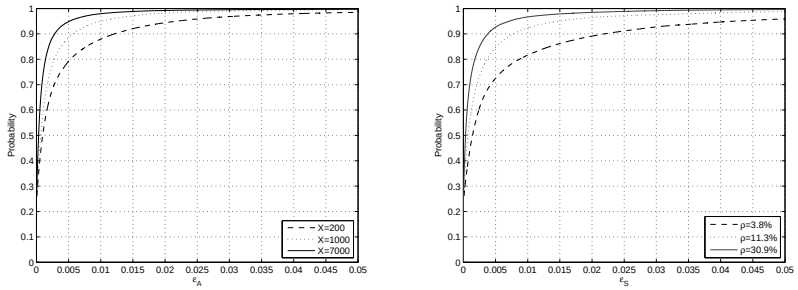
This way we were able to evaluate the results for different resource reservations (X) by taking as reference the complete original trace. In the experiments the used X were: $7000, 5000, 3000, 2000, 1000, 500, 300$ and 200 with the constant $\mathcal{S} = 1$ as detailed before. We also consider a bin size of $t = 175ms$ which provides a good tradeoff between the interval for live reporting and collection time to apply the proper sampling rate. We used previously this bin size with success in [1].

Summarising the methodology used for obtaining a trace per different resources is:

1. Generate the test traffic and monitor all the traces in the MEs.
2. Simulate the scenario with multiple simultaneous flows using the real traces.
3. Split the results into bins of size t.
4. Apply off-line the adaptive sampling for the different resources (X).
5. For each bin and X the $\widehat{PLR}$ and the average $\widehat{OWD}$ are computed.
6. Finally the estimated OWD for the sampled bin is compared with the real value.

5.2 Evaluation Results

In order to evaluate the system we compare the obtained accuracy using adaptive sampling with the application of a fairly equivalent static sampling. Since the sampling rate (ρ) depends directly of c and R_i not all values of ρ within a bin lead to a feasible situation. For example, for low rate flows, (e.g. 16 packets per second), using the bin size $t = 175ms$ leads to $\simeq 3$ packets per bin, then $\rho_{i_{min}} = \frac{2}{3}$ since with IPDV we need at least 2 packets to compute it. But in the case of flows with 200 packets per bin then $\rho_{i_{min}} = \frac{2}{200}$.



(a) Adaptive sampling (b) Static Sampling

Fig. 2. CDF of OWD accuracy comparison between adaptive and static sampling

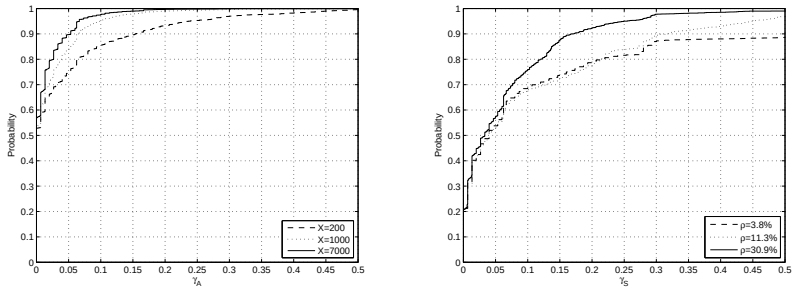
Required resources: The row labelled as (ρ_{eff}) of Table 1 shows the amount of resources effectively needed for each value of X. It details the global sampling rate applied taking into account all the flows and their durations. This value is obtained by aggregating all the considered samples out of the total amount of packets sent by all the tests. As it can be noted for low X values the sampling rate is very small, which means that only 3.8% (in the case of $X = 200$) of the resources needed to monitor $X_{\mathcal{D}}$ are required by our proposal.

X	200	300	500	700	1000	2000	3000	5000	7000
ρ_{eff}	3.8%	4.5%	7.3%	9.0%	11.3%	17.6%	23.4%	27.3%	30.9%
ε_{delay}	0.025	0.018	0.013	0.012	0.010	0.008	0.007	0.005	0.005
γ_{loss}	0.22	0.17	0.15	0.11	0.09	0.07	0.06	0.05	0.05

Table 1. Effective global sampling and Delay and Loss errors for 95^{th} percentile per X

OWD Estimation: In order to evaluate the accuracy of the OWD estimation we compared the results obtained per X with the real traces by using relative delay error values: $\varepsilon = \left|1 - \frac{\hat{d}}{D}\right|$. Where D is the average OWD per bin taken from the complete trace and $\hat{d}$ stands for the estimate obtained by our solution for different X. Figure 2 shows the comparison between adaptive sampling (left) and static sampling (right). The figures show the Cumulative Distribution Function (CDF) for various X for the adaptive and equivalent sampling rates for the static case. The X-axis of the figures holds the relative error (ε) while Y-axis is the error probability.

The results clearly show the gain obtained by our adaptive solution. Where adaptive sampling outperforms static sampling by around 50% for low values of X. For example in 95% of the tests, for $X = 200$ the $\varepsilon_A \leq 0.020$ while $\varepsilon_S \leq 0.040$. When the resources are increased the difference is reduced down to 5% and below for $X \geq 1000$. Moreover, adaptive sampling gives a more robust and reliable mechanism to control and schedule



(a) Adaptive sampling (b) Static Sampling

Fig. 3. CDF of PLR accuracy comparison between adaptive and static sampling

the used resources. This sensible gain in accuracy is further enhanced when dealing with packet losses as shown later.

From the accuracy point of view the figure 2 shows that increasing the resources for the reporting improves greatly $\widehat{OWD}$, in fact 95% of the tests had an error smaller than 0.025 as summarised in the row labelled as ε_{delay} in Table 1.

PLR Estimation: Given that many flows did not have any packet loss and the system always estimates this case correctly, we decided to remove from the study such flows for not biasing the results. In this case PLR comparison is performed by using absolute errors. The results show that the accuracy is lower than for OWD as shown in Figure 3, but much higher than in the case of static sampling, whose error is up to $\sim 25\%$ in the best considered case $(30.9\%$ sampling$)$ for 95% of the tests. Compared to the worst obtained with adaptive sampling which is $\sim 20\%$ for the minimum resources.

Eventhough for reasonable sampling rates the results show an error lower than 10% for 95% of the cases, the full details are in the row labelled as γ_{loss} in Table 1. A way to overcome this lower accuracy in the estimation of PLR, is to increase the bin size (t) in order to gather more information per bin. Hence improving the sampling estimation of the number of lost packets, which error is bound by $\varepsilon \leq 1.96\sqrt{\frac{1}{c}}$ where c is the number of lost samples detected. A deep analysis of this is left for further study.

6 Conclusions and Future Work

In this paper we proposed a new adaptive sampling methodology for distributed network performance assessment. Our contribution is focused on the resource scheduling and the optimisation of the required resources with minimum loss of accuracy in the live reporting. As a proof of concept, we use IPDV as an accuracy estimator to set the the sampling distribution.

Our adaptive approach outperforms static sampling solutions by smartly scheduling more resources to the flows with poorer estimation in the performance assessment. In

order to evaluate the accuracy of the estimation of OWD and PLR, we performed a series of tests in a European-wide scenario. The results show a negligible relative error in the OWD estimation (lower than 2.5% for 95% of the cases). In the case of PLR the accuracy is lower but kept within bearable thresholds.

PLR estimation can be improved by enlarging the bin size. Hence we leave as an important part of our future work the use of dynamic bin sizes. This way, low rate flows report more detailed information and the sampling rate could be better adjusted.

Another detail left as future work is the treatment of X as a distributed resource. This suits better a real scenario where the resources assigned to each ME can vary within the same domain.

Finally with the current system design, it always uses all the available resources regardless of the conditions. A further optimisation would be to focus on the minimisation of such used resources when possible.

References

1. René Serral-Gracià, Pere Barlet-Ros, and Jordi Domingo-Pascual. Coping with Distributed Monitoring of QoS-enabled Heterogeneous Networks. In *IT-News (QoSIP) - 2008*, pages 142–147, 2008.
2. René Serral-Gracià et al. Distributed sampling for on-line qos reporting. In *Submitted and pending acceptance*, 2008.
3. Nick Duffeld. Sampling for Passive Internet Measurement: A Review. *Statistical Science*, 19(3):472–498, 2004.
4. Tanja Zseby. Comparison of Sampling Methods for Non-Intrusive SLA Validation. In *E2EMon*, 2004.
5. Irene Cozzani and Stefano Giordano. Traffic sampling methods for end-to-end QoS evaluation in large heterogeneous networks. In *Computer Networks and ISDN Systems 30*, pages 1697–1706, 1998.
6. Wenhong Ma, Changcheng Huang, and J. Yan. Adaptive sampling for network performance measurement under voice traffic. In *IEEE-ICC*, 2004.
7. N. G. Duffield and Matthias Grossglauser. Trajectory sampling for direct traffic observation. *IEEE/ACM Trans. Netw.*, 9(3):280–292, 2001.
8. Tanja Zseby. Stratification Strategies for Sampling-based Non-intrusive Measurements of One-way Delay. In *Passive and Active Measurements*, 2003.
9. A. Hanemann, J. W. Boote, and et. al. PerfSONAR: A Service Oriented Architecture for Multi-Domain Network Monitoring. In *Third International Conference on Service Oriented Computing, LNCS 3826, ACM SIGsoft and SIGweb*, pages 241–254, 2005.
10. I. Miloucheva, P.A. Gutierrez, and et. al. Intermon architecture for complex QoS analysis in inter-domain environment based on discovery of topology and traffic impact. In *Inter-domain Performance and Simulation Workshop, Budapest*, March 2004.
11. C. Veciana-Nogués, A. Cabellos-Aparicio, J. Domingo-Pascual, and J. Solé-Pareta. Verifying IP Meters from Sampled Measurements. In *Kluwer Academic Publishers. Testing of Communicating Systems XIV. Application to Internet Technologies and Services. IFIP TC6/WG6.1. TestCom*, pages 39–54, 2002.
12. [IST] EuQoS - End-to-end Quality of Service support over heterogeneous networks - http://www.euqos.eu/, September 2004.
13. D5.1.1: Technical requirements for trials, tasks scheduling - http://www.euqos.eu/documents/md_1216_d-511.pdf, September 2004.