End-to-end quality of service seen by applications: a statistical learning approach

Pablo Belzarena*, Laura Aspirot

Facultad de Ingeniería, Universidad de la República, Montevideo, Uruguay

Abstract

The focus of this work is on the estimation of quality of service (QoS) parameters seen by an application. Our proposal is based on end-to-end active measurements and statistical learning tools. We propose a methodology where the system is trained during short periods with application flows and probe packets bursts. We learn the relation between QoS parameters seen by the application and the state of the network path, which is inferred from the interarrival times of the probe packets bursts. We obtain a continuous non intrusive QoS monitoring methodology. We propose two different estimators of the network state and analyze them using Nadaraya-Watson estimator and Support Vector Machines (SVM) for regression. We compare these approaches and we show results obtained by simulations and by measures in operational networks.

Key words: End-to-end active measurements, statistical learning, Nadaraya-Watson, Support Vector Machines, QoS

1. Introduction

Most multimedia services in packet switched networks have some end-to-end quality of service (QoS) constraints that should be enforced like delay, jitter and loss rate. However, end-to-end QoS parameters cannot in general be estimated from data obtained in each isolated router. Therefore, methodologies to perform end-to-end active measurements and estimations have been developed during the last ten years. Examples of such methodologies can be found in [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]

Some Internet service operators are offering now one or more "premium" services like video on demand, high quality video conferences, high definition IPTV, telematic services with real time requirements, etc.. However, the rate at which these services have grown is smaller than initially expected. One of the main reasons behind this slowness is probably the difficulties that exist in the current Internet architecture to guarantee end-to-end QoS. The different proposals developed during the last 15 years in order to assure QoS (like IntServ, DiffServ, etc.) were not broadly deployed by the operators.

These difficulties are only exacerbated when the service provider offers a service spanning multiple domains. In this case, nodes of the path are under the administration of different network operators. Yet another aspect that further complicates the problem is the increasing heterogeneity of the access technologies (xdsl, cablemodem, wifi, wimax, 2G, 3G, mesh networks, etc.).

In the previous context, an important issue in order to control the end-to-end QoS is admission control. In a premium services network, an admission control mechanism based on the end-to-end performance helps the operator to control the end-to-end QoS. This constitutes one of the main motivations of this work.

An end-to-end admission control tool involves many different tasks. In this work we focus specifically in how to monitor the network in order to predict the end-to-end QoS seen by a premium service. With this information an admission control tool can decide whether it accepts a new service request or not. Although this problem is our main motivation, the tools proposed in this work can be applied to many other operation and management problems; e.g. continuous monitoring of a Service Level Agreement (SLA).

A possible measurement technique for such monitoring tool is to send the application traffic (a video for example) and to measure then its QoS parameters at the receiver. However, in many cases these application flows have bandwidth requirements that are not negligible compared with links capacity. This technique could

*Corresponding author

Email addresses: belza@fing.edu.uy (Pablo Belzarena), aspirot@fing.edu.uy (Laura Aspirot)

overload a congested link, degrading the QoS perceived by clients using the system. In addition, it may only be used then if the measurements are infrequent, and sporadic QoS degradations are tolerated. However, this is clearly not the case if the operator requires a permanent or frequent network monitoring.

In order to avoid the possibility of being themselves the cause of congestion, some measuring techniques estimate the QoS parameters seen by an application using light probe packets. However, these methodologies do not consider the particular characteristics of the application. In this sense, they implicitly assume, for instance, that the delay of a specific application can be approximated by the probe packets delay. This assumption is naturally not always true since QoS parameters depend on the statistical behavior of each type of traffic. Therefore, in many cases, this kind of estimation yields inaccurate results.

We propose a methodology that is an intermediate point between both approaches (to send a multimedia flow during long periods or to send light probe packets during short periods) and provides an accurate estimation of QoS parameters seen by an application without overloading the network during long periods.

The basic idea is to learn the relation between the probe packets interarrival times statistic and the QoS parameters seen by an application. We will assume that the former characterizes the state of the network. Once the relation has been learned, we may predict the QoS parameters just by sending light probe packets.

More formally, we consider the regression model

$$Y = \Phi(X) + \varepsilon$$

where X, Y and ε are random variables. The random variable X is an estimation of the state of the network, the response Y is the QoS parameter seen by the application (delay, jitter, loss rate, etc.) and ε is a centered random variable which represents possible errors (e.g. in modelling, measurements, etc.) where ε and X are assumed independent.

The previous formulation evidences two problems. First, it is necessary to find an accurate estimation of the state of the network (the variable X). Second, it is necessary to estimate the function Φ. We estimate this function learning Φ from samples of the random variables Y and X.

Concerning the estimation of the state of the network, this work presents two contributions. The first one is a functional approach, where the estimation considers the empirical distribution function of probe packets interarrival times. This allows us to take into account other features that we cannot capture for example with the mean of interarrival times. However, this approach presents some drawbacks, in particular that it does not take into account time correlations. The second contribution addresses this point, by considering another estimator of the state of the network that captures information about time correlations. This estimator is presented in detail in section 5. As we shall see, it obtains very good results and in some cases it is necessary to consider only a very small subset of parameters related with this estimator to have accurate estimations of the QoS.

On the other hand in order to estimate the function Φ we propose a statistical learning approach based on two different tools: Nadaraya-Watson estimator and Support Vector Machines (SVM). Nadaraya-Watson, a method which was originally devised for real data [13], is used in this work mainly for functional regression [14]. In particular we will use SVM in its regression variant, called Support Vector Regression (SVR). SVR has been extensively used for many applications since the nineties [15, 16]. However, networking researchers started to apply SVR only a few years ago [17, 18, 19]. The nonparametric approach considered in this work differs from others in the literature, as will be discussed in section 8). Our main contribution in this aspect consists in analyzing the use of these two estimators in the case of nonstationary data. In particular, we provide theoretical insight for applying a functional Nadaraya-Watson estimator in a nonstationary context. Moreover, we study the impact of nonstationary data when applying SVM techniques. In this case we present an implementation of SVM that leads to accurate estimates even in the presence of nonstationarity.

We organize this paper in the following way. In section 2 we detail our approach to monitor the QoS parameters seen by an application. In section 3 there is a brief introduction to the statistical learning tools used in this work: Nadaraya-Watson and Support Vector Machines. In section 4 we characterize the state of the network by the empirical distribution of the probe packets interarrival times and the QoS estimation is based on functional Nadaraya-Watson. We analyze some preliminary experimental results for video applications. In section 5 we propose an alternative estimator of the state of the network, one which is related with the length of the queues. We also discuss the advantages of this estimator when applying SVM tools. In section 6 we analyze the impact of video characteristics in video QoS estimations. In section 7 we show results from experiments in three different operational networks. In section 8 we analyze other works that estimates QoS seen by applications. Finally, in section 9, we discuss the main conclusions of this work .

2. Problem formulation and proposed solution

We first consider the case of a path with a single-link. The multi-link case is discussed later. We assume that the cross traffic, the link capacity and the buffer size are unknown. The QoS parameter seen by the application is called Y and it is a function of the link and traffic characteristics:

$$Y = F(X_t, V_t, C, B)$$

where X_t is the cross traffic stochastic process, V_t is the stochastic process corresponding to the application traffic, C is the link capacity and B is the buffer size.

The link capacity C and the buffer size B are not known but are assumed to have constant values during the monitoring process. As the goal is to estimate a QoS parameter over the known process V_t (a video sequence for example), V_t can be considered as an input to our problem. Taking into account the previous considerations, we can say that $Y = F(X_t)$. At the end of this section we further discuss these assumptions on C, B, and V_t.

Let us now discuss how we addressed the problem of estimating the cross traffic X_t. Since we will work both with the single and the multi-link cases, we shall refer to the X_t estimator as "estimator of the state of the network path".

The methodology for estimating cross traffic consists on sending packets from the user equipment and measuring the interarrival times between these packets at the end of the link. The interdeparture times of the probe packets have a constant value. K is the probe packets size and C is the link capacity. In order to avoid loading the link, K should be as small as possible.

When two consecutive probe packets are queued in the same busy period at the link queue, as shown in figure 1, the interarrival time is equal to $\frac{X_i}{C} + \frac{K}{C}$, where X_i is the amount of cross traffic that arrived at the queue between probe packets i and $i+1$. Then, during the busy periods, the interarrival times are proportional to the cross traffic volume at least up to a constant.

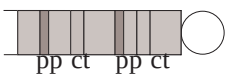

Figure 1: Probe packets (pp) and cross traffic (ct) in one queue busy period.

If two consecutive probe packets are not queued during the same busy period the estimation of the cross traffic can be inaccurate. This problem is analyzed by Machiraju et al. [21], that present a rigorous probabilistic approach to active probing methods for cross traffic estimation. They analyze the system identifiability and show that different cross traffic types can give rise to the same sequence of observed probe delays. Therefore, it is not always possible to determine the distribution of any desired aspect of the cross traffic by using probes.

However, we are not looking for an accurate estimator of the cross traffic. Ultimately, we want to estimate Y. Therefore, our interest is to find an estimator (represented by the variable X) that is a function of the probe packets interarrival times and, most importantly, that allows us to distinguish between possible states of the network . Later we will discuss some possible options for X, i.e. some possible functions of the probe packets interarrival times.

In order to estimate Φ and Y we consider two different phases. The first phase is called the learning phase. In the learning phase we send a burst of probe packets with fixed interdeparture time and fixed size K. Immediately after the probe packets we send a video sequence sample during a short period. This procedure is repeated periodically sending the probe packets and the video sequence alternatively as shown in figure 2.

Figure 2: Probe packets (pp) and probe video (video) are sent together in order to map the state of the network into the QoS metric measured over the video sequence.

We build the variable X_j by measuring for each test j the interarrival times of the probe packets burst. We also measure the QoS parameter Y_j of the corresponding video sequence resulting in a pair (X_j, Y_j) for each test. The problem is how to estimate the function $\Phi : \mathcal{D} \to \mathbb{R}$ by $\widehat{\Phi}$ from these observations, where $X \in \mathcal{D}$ and $\mathbb{R}$ is the real line.

The second phase is called the monitoring phase. During the monitoring phase we send only the probe packets. We build the variable X in the same way as in the learning phase. The QoS parameter $\widehat{Y}$ of the video sequence is estimated using the function $\widehat{\Phi}$, built in the learning phase, as $\widehat{Y} = \widehat{\Phi}(X)$. It should be noted that this procedure does not load the network because it avoids sending the video sequence during the monitoring phase.

Remark 1. *The previous discussion is based on the single-link case. We discuss now some considerations*

about the multi-link case. First, we must highlight that the multi-link case can be reduced to the single-link one in many important scenarios. For example, when the application service is offered on a server located in the ISP backbone (for example a video on demand server) and the user access is a cellular link or an ADSL link. In these cases the only bottleneck is normally the access link (backbones are generally overprovisioned), which is equivalent to the single-link case.

However, there are cases where the packets must wait in more than one queue. In these cases the different queues modify the variable X that we use to characterize the cross traffic. This means that we estimate a variable X where the influence of all queues are accumulated. Nevertheless, even in this case our method will work fine if it is possible to distinguish with this variable between different cross traffic processes observed in the path.

Remark 2. *Another assumption was that the network path, the link capacities and the buffer sizes are fixed. For the last two, this "constant value" assumption is reasonable. However, the route between two points on the network can change over time. This problem can be solved by periodically verifying the route between two points, using for example an application like traceroute. If a new route is detected two circumstances can arise. If the system has learned information about the new route, this information can be used for the estimation. If the system has not learned information about the new route, it is necessary to trigger a learning phase. However, let us highlight that in some cases a change in the route does not affect the measures, for example when the bottleneck is in the access link and the backbone is overprovisioned.*

Remark 3. *In this section we assumed that the system is trained with a single kind of video (we assume that V_t is a fixed sequence). As the video QoS parameters depend on a set of characteristics like encoding, bitrate, frame-rate or motion level, we can train the system with a set of video sequences that represent the different classes of videos. Later the system will use the corresponding training samples depending on the specific video that we want to monitor. This point will be further discussed in section 6.*

3. Statistical Learning

In this section we discuss the mathematical tools selected to estimate Φ in the model

$$Y = \Phi(X) + \varepsilon.$$

We present a brief review of current results about Nadaraya-Watson estimations in subsection 3.1 and about Support Vector Regression in subsection 3.2.

3.1. The Nadaraya-Watson estimator

This is a nonparametric estimator, i.e. it does not assume an explicit form for the function Φ nor any particular probability distribution for the random variables involved in the model.

Several results on nonparametric regression for real as well as multi-dimensional random variables have appeared since the works of Nadaraya and Watson [22, 23]. The Nadaraya-Watson estimator for the real case is defined as:

$$\widehat{\Phi}_n(x) = \frac{\sum_{i=1}^{n} Y_i K\left(\frac{\|x-X_i\|}{h_n}\right)}{\sum_{i=1}^{n} K\left(\frac{\|x-X_i\|}{h_n}\right)} = \frac{\sum_{i=1}^{n} Y_i K_n(X_i)}{\sum_{i=1}^{n} K_n(X_i)} \qquad (1)$$

$K(\cdot)$ is a Kernel, which is a positive function whose integral over its support is one, $K_n(X_i) = K\left(\frac{\|x-X_i\|}{h_n}\right)$ (where $\|\cdot\|$ is the euclidian norm) and h_n is a sequence that tends to zero with n (called the kernel bandwidth). This estimator may then be interpreted as a weighted average of the samples $Y_1, \ldots, Y_n$. The weights are given by $K_n(X_i)$ taking into account the distance between x and each point of the sample $X_1, \ldots, X_n$.

3.2. Regression with SVM (SVR)

Let us now discuss the alternative method to estimate Φ that we shall consider in this paper. Arguably, the simplest form we could assume is a linear regressor as follows:

$$\widehat{\Phi}(x) = \langle x, \widetilde{\beta} \rangle + \beta_0$$

In this case, the objective of SVR is to find an hyperplane $\langle x, \widetilde{\beta} \rangle + \beta_0$ such that all points (X_i, Y_i) belong to an "ε-tube" centered at the hyperplane, i.e. the distance of (X_i, Y_i) to the hyperplane should be less or equal than ε for all $i = 1, \ldots, n$. Moreover, the hyperplane should be as "flat" as possible where the flatness is related with the smoothness in the nonlinear case. The regression problem leads to the following optimization:

$$\min_{\widetilde{\beta}} \frac{1}{2}\|\widetilde{\beta}\|^2$$

s.t.:

$$Y_i - \langle X_i, \widetilde{\beta} \rangle + \beta_0 \leq \varepsilon \ , \ \forall i = 1, \ldots, n$$
$$\langle X_i, \widetilde{\beta} \rangle + \beta_0 - Y_i \leq \varepsilon \ , \ \forall i = 1, \ldots, n$$

A key assumption in the previous optimization problem is that a feasible solution exists. Obviously this is

not always the case. To find feasible solutions, we must allow some error, allowing the possibility that some points do not belong to the "ε-tube". Then, the problem can be reformulated in the following way:

$$\min_{\widetilde{\beta}} \frac{1}{2}\|\widetilde{\beta}\|^2 + \gamma \sum_{i=1}^{n}(\zeta_i + \zeta_i^*) \qquad (2)$$

s.t.:

$$Y_i - \langle X_i, \widetilde{\beta}\rangle + \beta_0 \leq \varepsilon + \zeta_i \ , \forall i = 1, \ldots, n$$
$$\langle X_i, \widetilde{\beta}\rangle + \beta_0 - Y_i \leq \varepsilon + \zeta_i^* \ , \forall i = 1, \ldots, n$$
$$\zeta_i, \zeta_i^* \geq 0$$

This problem can be solved more easily through the dual formulation. Using the Lagrangian and the Karush-Kuhn-Tucker (KKT) conditions, it can be shown that:

$$\widehat{\Phi}(x) = \sum_{i=1}^{n}\left(\alpha_i^0 - \alpha_i^{0*}\right)\langle X_i, x\rangle + \beta_0^* \qquad (3)$$

where α_i^0 , α_i^{0*} are the Lagrange multipliers corresponding to the two sets of constraints. Moreover, if $\alpha_i^0, \alpha_i^{0*} \neq \gamma$ and $|\widehat{\Phi}(X_i) - Y_i| < \varepsilon$, from the KKT conditions, then $\alpha_i^0, \alpha_i^{0*}$ must be zero. This means that the points that are inside the "ε-tube" do not contribute to the solution: i.e. these points are not needed to calculate $\widehat{\Phi}(x)$. Only the points with Lagrange multipliers $\alpha_i^0, \alpha_i^{0*} \neq 0$ are needed. These points are called the "support vectors".

It should be noted that the solution of the regression problem using SVR depends only on the dot product between the data (as it is shown in equation (3)). In this sense, in order to apply SVR in a general space $\mathcal{F}$ it is only necessary to know the form of the dot product in such space. Therefore, the idea for the nonlinear case is to map the input data into a higher dimensional space $\mathcal{F}$ by a function $\varphi : \mathbb{R}^d \rightarrow \mathcal{F}$. Then, we can apply the linear regression procedure explained above but in the space $\mathcal{F}$ (called the feature space). This technique generalizes SVR, effectively making it a nonparametric method. The mapping lead us to the following solution:

$$\widehat{\Phi}(x) = \sum_{i=1}^{N_s}(\alpha_i^0 - \alpha_i^{0*})K(X_i, x) + \beta_0^*$$

where N_s is the number of support vectors and $K(X_i, x)$ is a kernel function. A function is called a kernel if it represents a dot product in the feature space:

$$K(x, x') = \langle\varphi(x), \varphi(x')\rangle.$$

In particular, we use a radial basis function (rbf) kernel due to the good performance shown in different applications:

$$K(x, x') = e^{-\frac{1}{\sigma}\|x - x'\|^2} \qquad (4)$$

Finally, let us highlight that in [24, 25] it is shown that choosing the flattest function in the feature space leads to a smooth function in the input space. This is an important reason to look for a flat solution in the linear case.

4. The empirical distribution of the probe packets interarrival times

4.1. Why functional regression?

In this section we introduce, justify and discuss our first option for variable X: the empirical distribution of the probe packets interarrival times. When considering this first proposal, one could argue that it is not actually necessary to consider the whole distribution of probe packets interarrival times, but rather a simplified or shorter description of it. For instance, the mean and/or the variance.

Let us now evaluate this simpler definition of X through a concrete example simulated in ns-2 [37]. We simulate a link fed with a video trace, cross traffic and probe packets. The cross traffic corresponds to different Markovian ON-OFF sources. We alternate periodically between two types of ON-OFF sources, corresponding to high and low cross traffic levels.

Each test consists firstly on a probe packet burst with fixed interdeparture time. After this burst we send a simulated video traffic (a video traffic trace). For each test j we compute from the probe packets the mean and the variance of the interarrival times (the variable X_j in our model) and we measure the average delay of the video packets (the variable Y_j in our model).

In figure 3 it can be observed the measured Y (video delay) and the estimated Y using these possible choices of X. It should be noted that the estimation does not distinguish between busy and light periods. This means that the chosen X is not enough to discern between them. Let us remark that in other unreported experiments, which used simulated data and data taken from operational networks, the estimations of Y using these descriptors were always inaccurate.

On the other hand, in figure 4 we show four empirical distribution functions for simulated data. Two of them were obtained in the presence of high cross traffic and the others with low cross traffic. In this case, we may clearly appreciate the impact of the two regimes on X, and distinguish them through it. The next sections analyze how to estimate the QoS parameters through these empirical distribution functions.

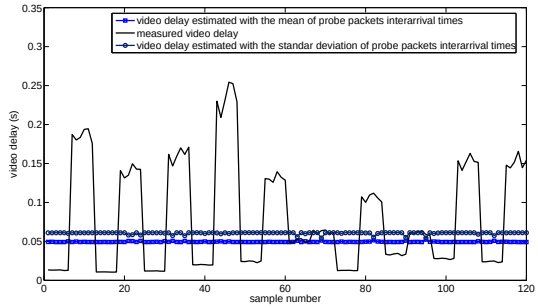

Figure 3: Measured and estimated video delay for simulated data using the mean and variance of the probe packets interarrival times

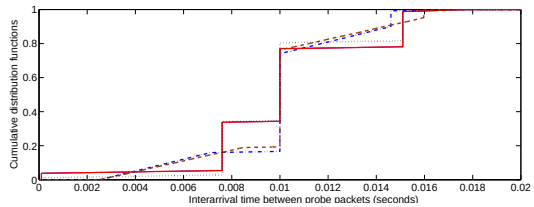

Figure 4: Empirical distributions of probe packets interarrival times

4.2. Functional Nadaraya-Watson estimator

For functional random variables (i.e. when the regressor X is a random function) Ferraty, Goia and Vieu [26] introduced a Nadaraya-Watson type estimator for Φ, defined by equation (1), where the difference with the real case is that $\|\cdot\|$ is a seminorm on a functional space $\mathcal{D}$. The estimation $\widehat{\Phi}_n(x)$ will be accurate if there are enough training samples near x. One of the main issues in the functional approach is the "curse of dimensionality", i.e. the number of samples needed to accurately estimate increases with the dimension of the space. This issue becomes crucial when the observations come from an infinite dimensional vector space. This problem is addressed in the literature and we refer for example to [14, 26, 27, 28, 29] for different approaches. These works state the convergence and the asymptotic distribution of the estimator for stationary and weakly dependent (for example mixing) functional random variables.

4.3. Extensions to the nonstationary case

The cross traffic X_t on the Internet is a dependent and nonstationary process. This topic has been studied by many authors during the last ten years. Zhang et al. [30, 31] show that many processes on the Internet (losses for example) can be well modelled as independent and identical distributed (i.i.d.) random variables within a "change free region", where stationarity can be assumed. They describe the overall network behavior as a series of piecewise-stationary intervals. Karagiannis et al. [32] have found nonstationarity at different time scales analyzing the traffic of a link belonging to a Tier 1 ISP. They found that the traffic can be considered stationary at small time scales with events that change its stationarity at a multi-second scale o larger.

The nonstationarity has different causes, but in any case it is very important to have estimators that can be used with nonstationary traffic. As our data comes from the Internet and it is typically nonstationary, we extend previous results on functional nonparametric regression to this case. Instead of considering equally distributed random variables X we consider a model introduced by Perera in [33] defined by

$$X_i = \varphi(\xi_i, Z_i)$$

where ξ_i takes values in a seminormed vector space with a seminorm $\|\cdot\|$, and Z_i is a real random variable that takes values in a finite set $\{z_1, z_2, \ldots, z_m\}$. For each $k = 1, \ldots, m$ the sequence $(\varphi(\xi_i, z_k))_{i\geq 1}$ is weakly dependent and equally distributed, but the sequence Z_i may be nonstationary as in [33]. The model represents a mixture of weakly dependent stationary processes, but the mixture is nonstationary and dependent. Here ξ represents the usual variations of the traffic, and the variable Z selects between different traffic regimes, and represents types of network traffic.

With this model two main theoretical issues appear: the convergence and the asymptotic distribution of the estimator. A preliminary result about the estimator convergence was published in [34] and an enlarged version in [35]. The asymptotic distribution of the estimator for this model is discussed in [36]. We prove in this work the almost sure convergence of the estimator, i.e. that with probability one the estimator converges to the real value when the number of samples goes to infinity. The statement and proof of this theorem is in Appendix A.

4.4. First application to simulated data

In this section we analyze the accuracy of estimations with functional Nadaraya-Watson applied to simulated data. We analyze the estimation procedure by simulations using the ns-2 simulator. We simulate a link fed with a video trace, a simulated cross traffic and probe packets. The cross traffic corresponds to a model $X = \varphi(\xi, Z)$. We have two Markovian ON-OFF sources and Z is a random variable that takes values in $\{0, 1\}$ selecting periodically between this two sources. Fixing

the value of Z we obtain stationary processes $\varphi(\xi, 0)$ and $\varphi(\xi, 1)$.

The first source (source 0) generates Markovian ON-OFF traffic corresponding to $\varphi(\xi, 0)$ with average bit rate varying from 150 Kbps to 450 Kbps and average time Ton in the ON state and Toff in the OFF state varying from 100 to 300 ms. The second source (source 1) generates Markovian ON-OFF traffic corresponding to $\varphi(\xi, 1)$ with average bit rate varying from 600 Kbps to 900 Kbps and average time Ton in the ON state and Toff in the OFF state varying from 200 to 500 ms. For each period an independent random variable is sampled to select the average bit rate. The payload of probe packets is 20 bytes and for the video packets is 1400 bytes. The video sequence has an average bit rate of 480 Kbps. The link capacity is 1.6 Mbps.

We send this cross traffic to a network link together with the probe packets and the simulated video sequence. Each test consists on a probe packet burst with fixed interdeparture time. After this burst we send a simulated video traffic (a video traffic trace). For each test j we compute from the probe packets the empirical distribution function of interarrival times X_j and we measure the average delay Y_j of the video packets.

In order to compute the Nadaraya-Watson estimator the kernel is

$$K(x) = \begin{cases} (x^2-1)^2 & \text{if } x \in [-1,1] \\ 0 & \text{if } x \notin [-1,1] \end{cases}$$

and we use the L^1 norm for the distance between the empirical distribution functions.

Concerning the time scales in our experiment the probe traffic is sent with fixed time t between consecutive probe packets. The aim is to find some criterion for choosing the best time scale in order to have accurate estimates. We consider different sequences of observations for a finite set of time scales $\{t_1, t_2, \ldots, t_r\}$. In practice, as we send bursts of probe traffic with fixed time t between packets we have observations with time scales in the set $\{t, 2t, \ldots, rt\}$. Consider $n+m$ observations for each time scale

$$\left\{\left(X_i^{t_j}, Y_i^{t_j}\right) : 1 \leqslant i \leqslant n+m,\ 1 \leqslant j \leqslant r\right\}$$

By dividing the sequence for a fixed time scale in two we can estimate the function Φ^{t_j} (for the time scale t_j) by $\widehat{\Phi}_n^{t_j}$ with the first n samples. We then compute the difference

$$\sigma^2{}_{t_j}(n,m) = \frac{1}{m}\sum_{i=1}^{m}\left(\widehat{\Phi}_n^{t_j}\left(X_{n+i}^{t_j}\right) - Y_{n+i}^{t_j}\right)^2$$

that gives a measure of the estimator performance for the time scale t_j. By computing $\sigma^2{}_{t_j}(n,m)$ for $1 \leqslant j \leqslant r$ we will choose $t^*_{n,m}$ such that

$$\sigma^2{}_{t^*_{n,m}}(n,m) = \min\left\{\sigma^2{}_{t_j}(n,m) : 1 \leqslant j \leqslant r\right\}$$

In our simulations $t = 10$ ms and we obtained that $\sigma^2{}_{t_j}(n,m)$ was minimized and almost the same in the case of $t_j = 10$ ms and $t_j = 20$ ms. Then we select the interdeparture time 20 ms in order to minimize the probe packet load.

The kernel bandwidth is selected with a similar procedure. For a training sample $\{(X_i, Y_i) : 1 \leqslant i \leqslant n\}$ we consider the estimation of Y_i defined by $\widehat{Y}_i = \widehat{\Phi}_{n,i}(X_i)$ where $\widehat{\Phi}_{n,i}$ is obtained from the training sample without considering (X_i, Y_i). Next we compute

$$\sigma^2(h_n) = \frac{1}{n}\sum_{i=1}^{n}\left(\widehat{\Phi}_{n,i}(X_i) - Y_i\right)^2$$

for a finite set of kernel bandwidths H and we select the kernel bandwidth h_n^* such that

$$\sigma^2(h_n^*) = \min\{\sigma^2(h_n) : h_n \in H\}.$$

The performance of the estimator is shown in figures 5 and 6. More precisely, in the simulations we obtained 360 values of (X, Y), which were divided in two subsets. The estimation of Φ is obtained from the last 300 samples and the accuracy of the estimation is evaluated over the first 60 samples by comparing $\Phi_n(X_j)$ with the measured average delay Y_j for $j = 1, \ldots, 60$. The relative error in each point j is computed as $\frac{|\widehat{\Phi}_n(X_j) - Y_j|}{Y_j}$. Note the precision of the estimation, specially when compared with figure 3.

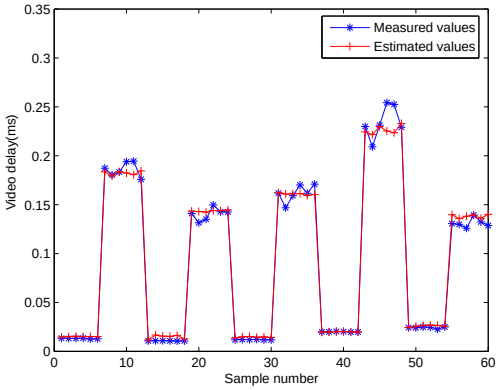

Figure 5: Measured and estimated video delay for simulated data.

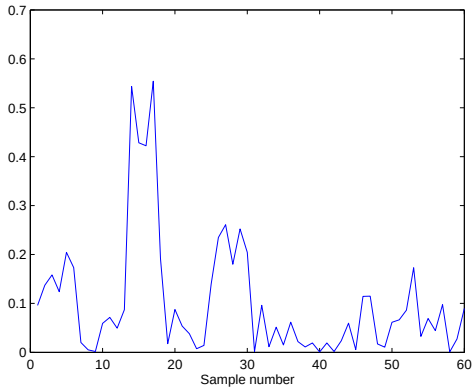

Figure 6: Relative error in the estimation of figure 5.

4.5. Weakness of this approach

Despite the very good results shown by the estimation in the previous section, it presents certain drawbacks we now discuss. Firstly, the empirical distribution of the probe packets interarrival times does not capture well the time correlation of the cross traffic process. This information is important for estimations in operational networks because the way the queue is filled depends on the cross traffic correlation.

Moreover, and from a more practical point of view, using an empirical distribution may have a high cost in storage space and computing time. In fact, to develop an online system it is important to represent the state of the network with the minimum number of features. However, we have already seen that considering a "summary" of this distribution (like the mean or the variance) does not yield good results.

We will then look for another function of the interarrival times in order to estimate Y with a small set of features. The use of SVR rather than Nadaraya-Watson has an advantage in this context. Nadaraya-Watson is a nonparametric estimator where the model is represented by all training points. For each new point the estimator must be calculated using all training data. However, SVR only needs to store the support vectors that are fewer than the whole training sample.

Therefore, the goal of the next section is to develop an accurate estimation procedure using SVR with a small set of features taken from the interarrival times.

5. The estimator of the queues state.

In this section we present another estimator of the network path state taking into account the analysis of the previous section. This new estimator will address many of the drawbacks considered before. In order to present it we first describe the single-link case in subsection 5.1, whereas the general network path case is discussed in subsection 5.2. Experimental results using the new estimator combined with the Nadaraya-Watson and SVM techniques are presented in subsections 5.3 and 5.4 respectively.

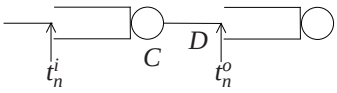

Figure 7: The single-link case.

5.1. The single-link case

Consider a probe packet n that arrives to a link queue at time t_n^i and leaves the link at time t_n^o (see figure 7). If the capacity of the link is C, the difference between t_n^o and t_n^i is such that:

$$\frac{q\left(t_n^i\right)}{C} + K = \left(t_n^o - t_n^i\right). \tag{5}$$

$q\left(t_n^i\right)$ is the queue length when packet n arrives to the queue, $K = \frac{P}{C} + D$ is a constant where P is the fixed probe packets size and D is the link propagation delay (i.e. the time it takes to transmit one bit over the physical medium). Even if (5) may be used directly to estimate the queue size, it has an important drawback: the queue size depends on the clocks in two different places, as the time t_n^i should be stamped at the sender and the time t_n^o should be stamped at the receiver. Consider instead the following equation:

$$\frac{q\left(t_n^i\right)}{C} = \frac{q\left(t_{n-1}^i\right)}{C} + \left(t_n^o - t_{n-1}^o\right) - \left(t_n^i - t_{n-1}^i\right). \tag{6}$$

Equation (6) is obtained by considering two consecutive packets, $n-1$ and n, and the difference between the following equations:

$$\frac{q\left(t_n^i\right)}{C} + K = \left(t_n^o - t_n^i\right),$$

$$\frac{q\left(t_{n-1}^i\right)}{C} + K = \left(t_{n-1}^o - t_{n-1}^i\right).$$

Applying equation (6) recursively and assuming that there exists a time t_0^i where the queue is empty we have that:

$$\frac{q\left(t_n^i\right)}{C} = \sum_{j=1}^{n}\left[\left(t_j^o - t_{j-1}^o\right) - \left(t_j^i - t_{j-1}^i\right)\right] \tag{7}$$

This equation allows us to estimate the size of the queue seen by the probe packets with the important advantage that it depends only on times taken with the same clock, as interdeparture time $t_j^i - t_{j-1}^i$ is known and $t_j^o - t_{j-1}^o$ is the interarrival time for consecutive packets.

Remark 4. *The time when the queue is empty can be estimated looking for interarrival times equal to interdeparture times.*

Remark 5. *Equation (7) is also valid in the case where the interdeparture times are not fixed and for example they follow some probability distribution.*

Remark 6. *In equation* (5) *we assume that K is constant. In order to consider a variable delay we should assume* $K_n = R + R_n$ *where R is the minimum delay and* R_n *is a random variable that represents the variable component of the delay experimented by packet n. In this case*

$$\frac{q\left(t_n^i\right)}{C} - \frac{q\left(t_0^i\right)}{C} + R_n - R_0 = \sum_{j=1}^{n}\left[(t_j^o - t_{j-1}^o) - (t_j^i - t_{j-1}^i)\right].$$

As in equation (7) *we can assume that there is a time* $t = 0$ *such that the queue is empty and the variable component of the delay is zero, so* $q\left(t_0^i\right) = R_0 = 0$. *The estimator, i.e. the sum in equation* (7), *takes into account the size of the queue and the variable component of the delay seen by packet n. In this case, even if it is not a precise estimator of the queue size, it is a suitable estimator of the network state, which is our actual objective.*

Remark 7. *Active measurements are affected by time stamping errors. Two main problems with time stamping are clock resolution and clock drift.*

We will address first the clock resolution issue. The real departure time t_n^i *for probe packet n can be expressed as* $t_n^i = \hat{t}_n^i + \varepsilon_n^i$, *where* $\hat{t}_n^i$ *is the time stamp and* ε_n^i *is the error due to clock resolution, a random variable with values in* $[0, \Delta^i]$, *with* Δ^i *the clock resolution at the departure node. The same occurs with arrival times,* $t_n^o = \hat{t}_n^o + \varepsilon_n^o$, *with values in* $[0, \Delta^o]$, *with* Δ^o *the clock resolution at the arrival node. In this case equation* (7) *can be written as*

$$\begin{aligned}\frac{q(t_n^i)}{C} &= \sum_{j=1}^{n}\left[\left(\hat{t}_j^o - \hat{t}_{j-1}^o\right) - \left(\hat{t}_j^i - \hat{t}_{j-1}^i\right)\right] \\ &\quad + \sum_{j=1}^{n}\left[\left(\varepsilon_j^o - \varepsilon_{j-1}^o\right) - \left(\varepsilon_j^i - \varepsilon_{j-1}^i\right)\right] \qquad (8)\end{aligned}$$

Please note that the mean of equation (8) *is precisely* (7) *since* $\left(\varepsilon_j^o - \varepsilon_{j-1}^o\right)$ *and* $\left(\varepsilon_j^i - \varepsilon_{j-1}^i\right)$ *are centered random variables with values in* $[-\Delta^i, \Delta^i]$, $[-\Delta^o, \Delta^o]$. *However, the variance of* $\frac{q(t_n^i)}{C}$ *has an additional term* $nV^i + nV^o$, *where* V^i *and* V^o *are the variances for* $\varepsilon_j^i - \varepsilon_{j-1}^i$ *and* $\varepsilon_j^o - \varepsilon_{j-1}^o$ *and n is the packet number. In the experiment the clock should be such that its resolution error variance is small compared with the variance in the queue size, so that the term* $nV^i + nV^o$ *becomes negligible. In addition, we can reduce the error variance due to the the number of packets (as for packet n the error variance is* $nV^i + nV^o$*) by setting to zero the packet number and* $\frac{q(t_n^i)}{C}$ *each time we find an empty queue.*

On the other hand, as the time between probe packets is in the order of milliseconds or less, the clock drift error is generally negligible. Anyway we are assuming for both problems that the clocks at the arrival and departure nodes are accurate enough compared with the target times that we want to analyze.

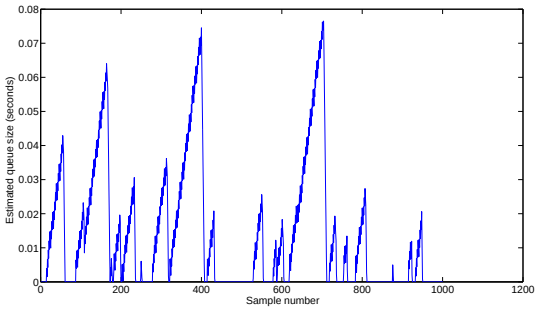

Figure 8: Queue size estimated by probe packets for one single-link in the presence of heavy cross traffic.

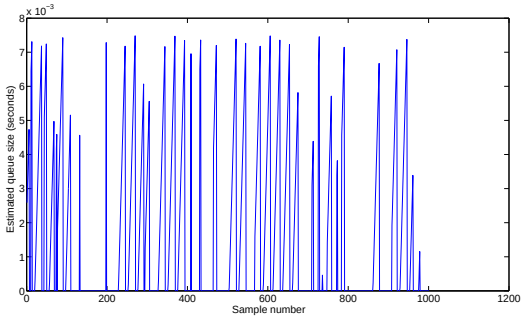

Figure 9: Queue size estimated by probe packets for one single-link in the presence of light cross traffic.

Finally, we show in figures 8 and 9 the estimation of the queue size computing equation (7) in simulations for

two different cross traffic processes, obtained with ns-2 in one single-link. We send a burst of probe packets and we generate Markovian ON-OFF cross traffic in order to show the behavior of $q\left(t_n^i\right)$ under different traffic loads.

5.2. Extension to a path with many links

In this section we analyze a path with N links, where each link has capacity C_l. Let $q_l(t)$ be the queue size of link l at time t. Packet n arrives to the link at time $t_n^{i,l}$ and leaves it at time $t_n^{o,l}$. Taking into account that

$$t_j^{o,l-1} = t_j^{i,l},$$

and defining

$$\Delta_{l,j} = t_j^{i,l} - t_{j-1}^{i,l} = t_j^{o,l-1} - t_{j-1}^{o,l-1},$$

for each link l in the path, equation (7) can be rewritten as:

$$\begin{aligned} q_l\left(t_n^{i,l}\right) &= C_l \sum_{j=1}^{n} \left[(t_j^{o,l} - t_{j-1}^{o,l}) - (t_j^{i,l} - t_{j-1}^{i,l})\right] \\ &= C_l \sum_{j=1}^{n} \left(\Delta_{l+1,j} - \Delta_{l,j}\right), \quad \forall l \in \{1,\ldots,N\} \end{aligned}$$

The total queue size over the path is then:

$$\sum_{l=1}^{N} q_l\left(t_n^{i,l}\right) = \sum_{l=1}^{N} C_l \sum_{j=1}^{n} \left(\Delta_{l+1,j} - \Delta_{l,j}\right)$$

We add to each of the first $N-1$ terms of the sum $C_N \sum_{j=1}^{n}\left(\Delta_{l,j} - \Delta_{l+1,j}\right)$ and to the last term we add $C_N \sum_{j=1}^{n}\left(\Delta_{N,j} - \Delta_{1,j}\right)$. Since the total sum of these terms $C_N \sum_{j=1}^{n}\left(\sum_{l=1}^{N-1}\left(\Delta_{l,j} - \Delta_{l+1,j}\right) + \Delta_{N,j} - \Delta_{1,j}\right)$ is zero then: $\sum_{l=1}^{N} q_l\left(t_n^{i,l}\right) = \sum_{l=1}^{N-1}(C_l - C_N)\sum_{j=1}^{n}\left(\Delta_{l+1,j} - \Delta_{l,j}\right) + C_N \sum_{j=1}^{n}\left(\Delta_{N+1,j} - \Delta_{1,j}\right)$. Finally, taking into account that $\sum_{j=1}^{n}\left(\Delta_{l+1,j} - \Delta_{l,j}\right) = \frac{q_l\left(t_n^{i,l}\right)}{C_l}$ we obtain that $\sum_{l=1}^{N} q_l\left(t_n^{i,l}\right)\left(1 - \frac{C_l - C_N}{C_l}\right) = C_N \sum_{j=1}^{n}\left(\Delta_{N+1,j} - \Delta_{1,j}\right)$ and then:

$$\sum_{l=1}^{N} \frac{q_l\left(t_n^{i,l}\right)}{C_l} = \sum_{j=1}^{n}\left(\Delta_{N+1,j} - \Delta_{1,j}\right). \tag{9}$$

Equation (9) means that the sum of the interarrival times minus the sum of the interdeparture times of the probe packets until one specific probe packet, is equal to a linear combination of the sizes of the queues seen by this probe packet when it arrives at each queue in the path. This quantity is strongly correlated with the state of the path.

Finally, for a sequence of probe packets we define a sequence of estimators, where $\widehat{q}_n$ is the estimator computed for packet n, by:

$$\widehat{q}_n = \sum_{j=1}^{n}\left[\left(t_j^{o,N} - t_{j-1}^{o,N}\right) - \left(t_j^{i,1} - t_{j-1}^{i,1}\right)\right] \tag{10}$$

From this sequence of estimators $\widehat{q}_n$ we compute several statistics (e.g., the empirical distribution, the mean, the variance, etc.) that we use as the state of network path X in order to estimate Y.

5.3. Regression using the empirical distribution of $\widehat{q}_n$

The use of $\widehat{q}_n$ rather than the use of the empirical distribution of the interarrival times has at least one important advantage: the estimator $\widehat{q}_n$ captures more information about the correlation of the cross traffic process than the distribution of the interarrival times.

In order to analyze this estimator we will use the same simulated data as in section 4. These simulated data correspond to the case of one single-link, but it is presented here in order to discuss some properties of $\widehat{q}_n$. In section 7 the estimator $\widehat{q}_n$ is used with real data in a path with many links. We consider 360 (X_i, Y_i) samples obtained with simulated traffic. This samples are divided in two sets: the validation set $\left\{(X_i^1, Y_i^1) : i = 1,\ldots,120\right\}$ and the training set $\left\{(X_i^2, Y_i^2) : i = 121,\ldots,360\right\}$. The estimator $\Phi_n\left(X_i^1\right)$ will be calculated for each point of the validation sample, using the other 240 points for training.

Let us first estimate the QoS parameter using the empirical distribution of $\widehat{q}_n$ and the functional Nadaraya-Watson estimator. The estimation was done using the kernel bandwidth that minimizes the mean square error. The result is shown in figure 10 where we may appreciate that this procedure obtains a good estimation when the delay is high, and that its accuracy decreases significantly when the delay is low.

In order to get insight into the previous problem we consider the empirical distribution of the estimator $\widehat{q}_n$. Figure 11 shows several empirical distribution of the simulated data. The three functions of the right side of the picture correspond to high values of Y and the other distributions correspond to low values of Y (a zoom on these last curves is shown in figure 12). It should be clear from these graphs that the L^1 distance between any two empirical distributions is much smaller when Y is low than for greater values of it. Actually, if we divide the sample in two, depending on the value of Y, we would obtain very interesting results regarding the optimal kernel bandwidth. In the case of the estimator $\widehat{q}_n$

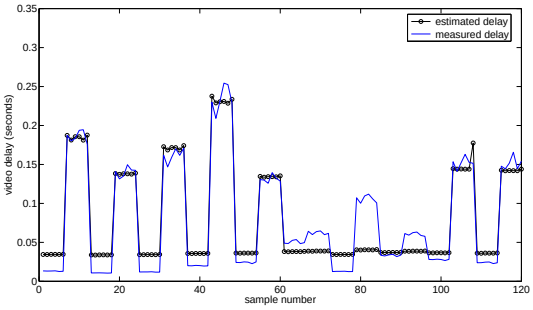

Figure 10: Estimated and measured video delay using the empirical distribution of $\widehat{q}_n$ (equation (10)) and Nadaraya-Watson.

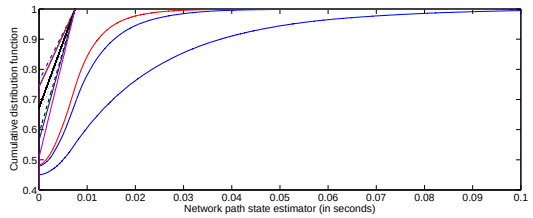

Figure 11: Empirical distribution of the network path state estimator $\widehat{q}_n$ for simulated data with different cross traffic levels.

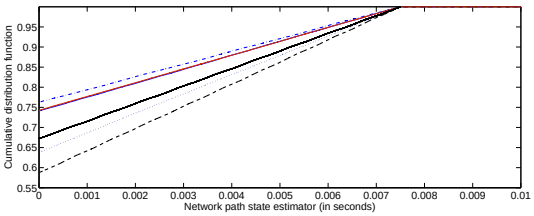

Figure 12: Empirical distribution of the network path state estimator $\widehat{q}_n$ for simulated data with low and medium cross traffic levels.

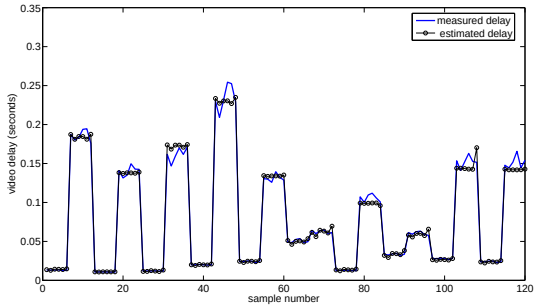

Figure 13: Measured and estimated video delay using the empirical distribution function of $\widehat{q}_n$ and Nadaraya-Watson with four different kernel bandwidths.

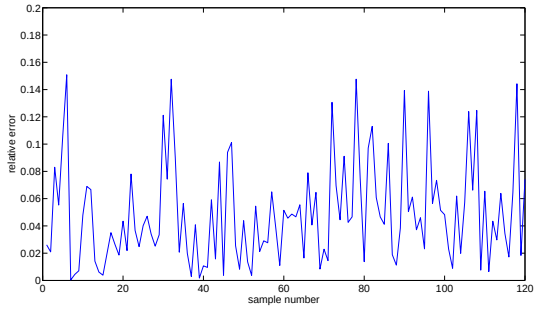

Figure 14: Relative error in the estimations of figure 13.

for the low delay values zone the optimal kernel bandwidth is 8×10^{-5} and in the high delay values zone is 2×10^{-3}. The optimum using only one kernel bandwidth is 1.4×10^{-3} because in the mean square error high values have more influence. This is the reason why in the low delay values zone the estimation is less accurate.

A possible solution to this problem would then be to use different values of the kernel bandwidth depending on the value of Y. For instance in figures 13 and 14 we show the results obtained by using four different kernel bandwidths. The estimation is accurate and the relative error is in the worst case around 15 %.

It should be noted that we did not find these problems when using the empirical distribution of the interarrival times (figure 5). In that case, the optimal kernel bandwidth for low and high values of Y are relatively similar. We have shown in figure 4 four empirical distributions of the interarrival times. Two of them are for cases of high Y values and the other two are for low Y values. In that case, the L^1 norm was a good metric to distinguish among them. In this sense, an alternative solution to considering several bandwidths is to consider a different norm altogether.

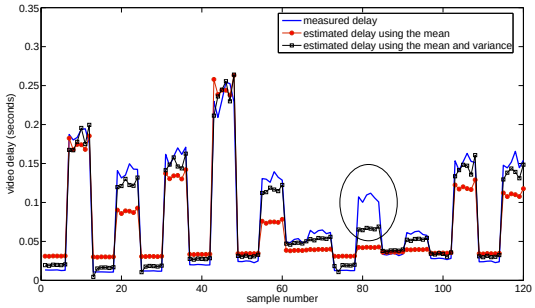

Figure 15: Measured and estimated video delay using the mean and the variance of $\widehat{q}_n$ and SVR.

Instead of using several kernel bandwidths or looking for another norm, we will see in the following subsection that the simpler alternative of considering a small set of parameters from $\widehat{q}_n$ together with SVR also provides excellent results.

5.4. *Regression using parameters obtained from the sequence $\widehat{q}_n$*

As mentioned in section 1, ideally we would like to design a framework that may be applied online. In order to achieve this, it is important to consider as few as possible parameters when making the estimation.

We will first consider, as earlier, the case in which X is the mean and the variance of $\widehat{q}_n$. Before, let us briefly discuss how we chose the parameters of SVR. The parameter γ (see equation (2)) controls the tradeoff between the error minimization and the "flatness" of the solution. Moreover, the rbf kernel (equation (4)) is controlled by δ. The method used to optimize these parameters is to build a grid and to find the point that minimizes the mean square error over the validation sample.

Figure 15 shows the estimation when X is the mean of $\widehat{q}_n$. In this case, even if significantly better than the results shown in figure 3, the estimation is not particularly accurate. The same figure shows that the estimation is improved when jointly considering the mean and the variance. However, there are certain areas in which the precision is unacceptably low (e.g. the circle in figure 15).

To find features that further distinguish operation points, let us take a more detailed look at figures 11 and 12. For low load situation (figure 12), we may see that the difference lies mainly on the point at which the distribution curve intersects the y-axis. This value may be interpreted as the percentage of time the queue is empty. On the other hand, we may observe that the queue size distribution has a heavier tail as load increases (see figure 11). In this sense, operation points are easily distinguishable through a high percentile of the corresponding distribution (e.g. 90 %).

Taking into account the previous considerations we add the percentage of time with empty queue and the 90 % percentile to X. Figure 16 shows the estimation using these four parameters, and figure 17 shows the estimation relative error. Note that the precision in this case is very good. The worst relative error is obtained for low values of Y, which is not very important for QoS considerations.

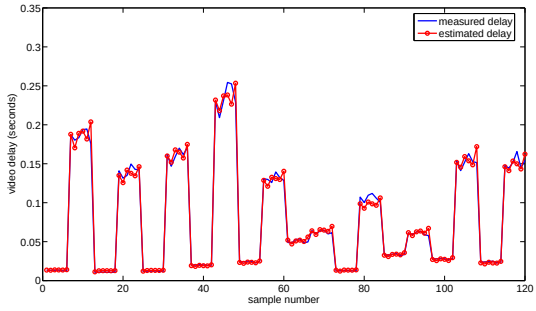

Figure 16: Measured and estimated video delay using mean, variance, percentage of time with empty queue and percentile 90 % of $\widehat{q}_n$ and SVR.

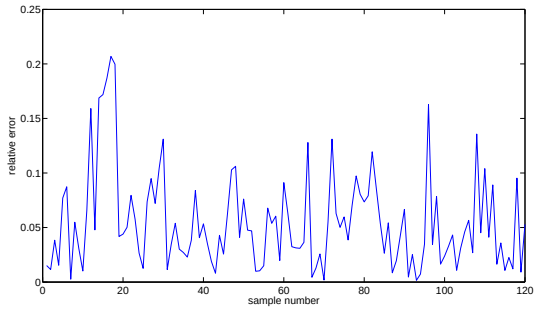

Figure 17: Relative error in the estimations of figure 16.

6. QoS estimation with different video sequences

The goal of this section is to verify by simulations that it is not necessary to train the system with each specific video sequence. We will verify that the estimation of the QoS parameters of a video depends only on some characteristics like codec type, frame rate, movement level, etc. The results are accurate if the estimation is

done with a system trained with another video sequence with the same characteristics.

We use the same simulation scenario as in the previous sections (one single-link with ON-OFF cross traffic) but here we feed the link with different real traces of a set of 30 seconds video sequences. These traces are real video sequences that we send between two computers and we capture the video packets (departure time and packet size) at the sender LAN interface.

We simulate, for each video sequence, 200 samples (X, Y) where X is a vector with mean, variance, percentage of time with empty queue, and 90 % percentile, obtained from the sequence $\widehat{q}_n$. Y is the delay measured over the video sequence.

The 200 pairs (X, Y) of each video sequence were divided in two sets. The first 120 experiments were used as training samples and the other 80 as validation samples. Using two different training sets we estimate the delay for each video sequence. On the one hand we estimate the delay training and validating with the same video sequence. On the other hand we estimate the delay training and validating with different video sequences.

In table 1 we show some characteristics of four particular video sequences. The first three video sequences were encoded with the same codec and with the same frame rate. The three videos are part of different films: video 1 is about an electric storm, video 2 is about a jump from a bridge and video 3 is about a car race. Video 4 is encoded with another codec and with another frame rate.

Table 1: Video characteristics

Video	Codec	frame rate
1	MPEG 1	25 fps
2	MPEG 1	25 fps
3	MPEG 1	25 fps
4	MPEG 4	30 fps

Figure 18 shows the estimated delay of the validation samples of video 2 using as training sample either video 2 or video 1. Figure 19 shows the same results but using video 3 and video 1 respectively. In both cases the performance of the estimator is very good. On the other hand, figure 20 shows the obtained results when using videos 4 and 1. In this case the estimation done with the training sample of video 4 has good accuracy, but this is not the case when the training is performed with video 1.

The previous results show that it is unnecessary to train the system for each particular video sequence. However, it may be necessary to train the system for the different codecs, frame rates, resolutions and quality levels of the videos. At first sight, this would result in a huge number of measurements, which should consider all the combinations of possible codecs, frame rates,

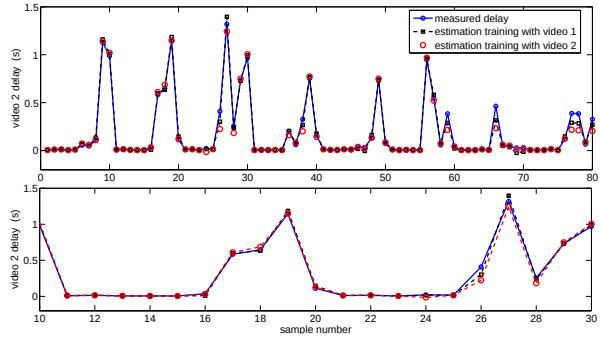

Figure 18: Video 2 estimated and measured delay using video 1 and video 2 as training samples. (The bottom figure is a zoom of the upper figure in a time interval.)

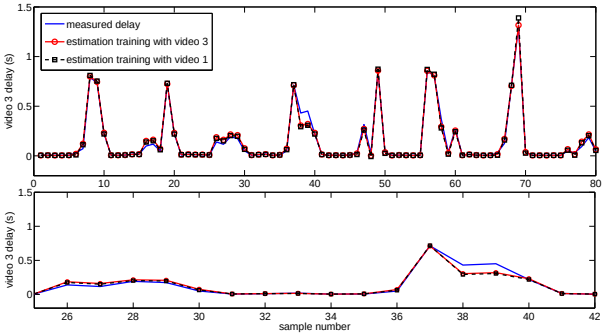

Figure 19: Video 3 estimated and measured delay using video 1 and video 3 as training samples. (The bottom figure is a zoom of the upper figure in a time interval.)

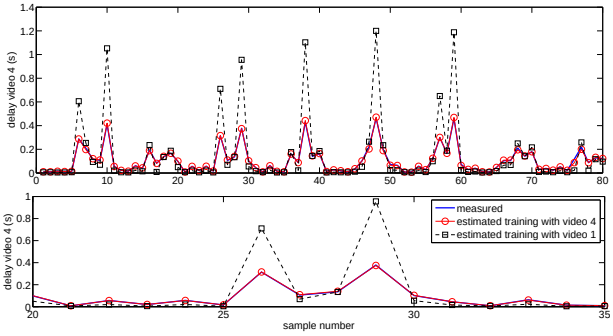

Figure 20: Video 4 estimated and measured delay using video 1 and video 4 as training samples. (The bottom figure is a zoom of the upper figure in a time interval.)

quality levels and resolutions. This raises some considerations on the practical application of this methodology, which we now discuss.

First, let us recall that although one of the main motivations of this work is an admission control system for a video on demand service or other video services, the methodology developed in this work is applicable for other multimedia services.

Second, we remark that each video service provider uses only a small set of the encoding parameters. The ITU recommendation H264 [38] that is being adopted by different systems form 3G to HDTV systems has 16 levels (combinations of resolutions, frame rates, etc.). However, for each specific service only a few levels are used. For example, a video service for a 3G network normally uses a MPEG2 or MPEG4-part 10 video codec with a video resolution adapted for a cell phone and with a bit rate adapted to its network. A high quality on-demand video service provider or a TV service provider uses another set of parameters.

The following are examples of typical sets of parameters recommended for different services by a video system provider [39]:

1. Scenario: mobile content (3G), resolution: 176×144, frame rate: 10-24 fps
2. Scenario: internet standard definition, resolution: 640×480, frame rate 24 fps
3. Scenario: high definition, resolution: 1280×720, frame rate 24 fps

Taking into account the previous considerations, it is clear that our monitoring system must be trained only for the small set of encoding parameters that the service provider uses for its specific network service.

Finally, we note that different video encoding parameters may have different relevance for our learning system. An interesting research topic, that escapes the scope of the present paper, is the influence of these different parameters in the training system.

7. Experimental Results.

In this section we present the results obtained by our techniques when applied to different operational networks. The experiments were performed with a measurement software tool specially developed for this purpose. In order to evaluate the practical limits of our methodologies we analyze three scenarios that have different levels of complexity. In section 7.1 we present measurements obtained using an ADSL access. In section 7.2 we present some multidomain measurements done between a server located at Universidad de la República in Uruguay an a server located in Canada. Finally, in section 7.3 we present measurements using a cellular access network.

7.1. ADSL tests

The first experiment we present was carried out in a host connected to the internet through an ADSL access. In this case the bottleneck link is precisely the access and cross traffic was relatively controlled. The estimated parameter Y was the video delay. Figure 21 shows the first analysis.

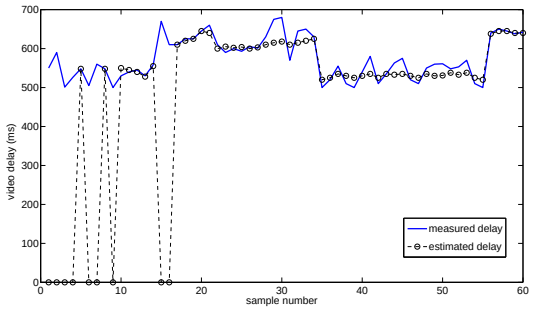

Figure 21: Estimated and measured video delay in an ADSL link in the presence of peer to peer cross traffic using Nadaraya-Watson.

We monitor video sequences streamed from an Internet server. Each estimation is calculated using the previous data as training samples. For this reason, the first estimations are not accurate and they improve as the number of samples grows. In this case we use functional Nadaraya-Watson estimator and the empirical distribution of interarrival times as X. Note how several of the first samples are estimated as zero, indicating that no data is available to estimate them. Between samples 10 and 14, when the estimation seems to be stable, we start a peer to peer connection that generates a step change in the cross traffic. In the new conditions there are not enough points in the training sample in order to estimate the video delay. This situation can be seen in sample 15 where the estimation is zero . In sample 35 we finish the peer to peer connection. We come back to the same state as at the beginning of the test. In sample 55 we restart the peer to peer connection. In this case the system has training data learned between samples 14 to 34 and the estimations are accurate.

In the second experiment we monitor a video streamed from an Internet server to a PC located in a LAN connected through an ADSL link to the Internet. We monitor the video during a long period of time and

in this case the user does not generate cross traffic. The cross traffic is generated only by other users of the LAN or the Internet. As a consequence, and as opposed to the previous experiment, we have no control over the cross traffic. We have done 50 experiments. We use the functional Nadaraya-Watson estimator and the variable X is the empirical distribution of the probe packets interarrival times. (As an example Figure 22 shows 30 values of X, i.e. 30 empirical distributions of the probe packets interarrival times.) In this case each point is estimated with the rest of the data. Figure 23 shows the estimation. The accuracy of the estimation is good, except on some peak values. This could be improved with a larger training sample.

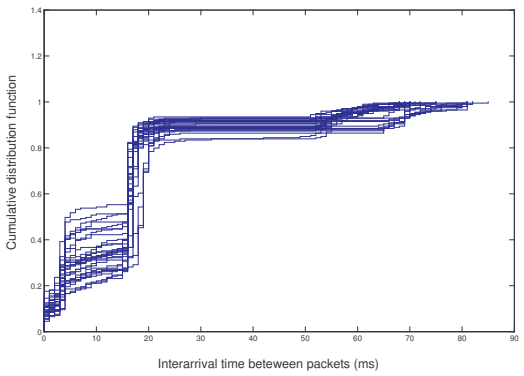

Figure 22: 30 empirical distributions of the probe packets interarrival times in an ADSL link.

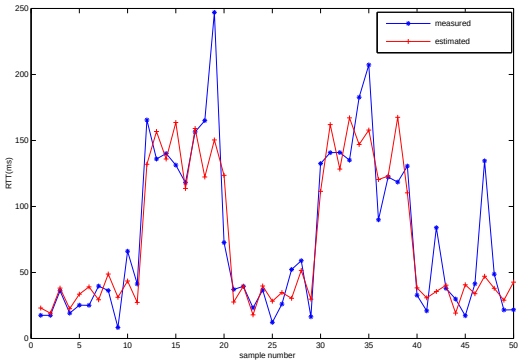

Figure 23: Estimated and measured video delay in an ADSL link using Nadaraya-Watson.

7.2. Multidomain tests

In this section we analyze the estimation of the QoS parameters of a video downloaded through multiple domains. We study three different parameters: delay, losses and jitter. The experiments were done between a server located at Universidad de la República (Uruguay) and a server located in Canada (69.77.X.X).

We have sent video sequences encoded at 382 kbps and probe packets of 80 bytes on wire at a fixed interdeparture time of 15 ms. The probe packets represent a constant bit rate traffic of around 40 kbps (i.e. approximately 10 times lower load than the video traffic). The path between both servers is shown in table 2 and has more than twenty links. It should be noted that hop number 13 implements a load balancing mechanism. This situation is an additional complication for the estimation.

hp number	interface	ip	interface 2
2	192.168.240.1	(192.168.240.1)	
3	69.77.183.13	(69.77.183.13)	
4	69.77.175.90	(69.77.175.90)	
5	69.77.175.94	(69.77.175.94)	
6	69.77.175.66	(69.77.175.66)	
7	69.77.175.38	(69.77.175.38)	
8	142.47.135.1	(142.47.135.1)	
9	142.46.0.14	(142.46.0.14)	
10	ge-5-2-110.hsa2.Detroit1.Level3.net	(64.152.144.1)	
11	ae-8-8.ebr2.Chicago1.Level3.net	(4.69.133.242)	
12	ae-2-54.edge3.Chicago3.Level3.net	(4.68.101.116)	
13	sl-st20-chi-5-0.sprintlink.net	(144.232.19.173)	(144.232.8.113)
14	sl-crs2-chi-0-11-5-0.sprintlink.net	(144.232.8.218)	
15	sl-crs2-chi-0-11-3-0.sprintlink.net	(144.232.20.53)	
16	sl-crs1-mia-0-11-2-0.sprintlink.net	(144.232.18.217)	
17	sl-bb20-mia-13-0-0.sprintlink.net	(144.232.2.253)	
18	sl-st21-mia-2-0.sprintlink.net	(144.232.9.199)	
19	sl-antel1-1-0.sprintlink.net	(144.223.245.162)	
20	ibb2uni1-p1.antel.net.uy	(200.40.22.37)	
21	ibb2cen1-1-3.antel.net.uy	(200.40.16.89)	
22	iem2cen1-0-1.antel.net.uy	(200.40.17.50)	
23	seciu-ibgp.adinet.com.uy	(200.40.160.9)	
24	r3.rau.edu.uy	(164.73.128.129)	
25	eth-fing.rau.edu.uy	(164.73.253.34)	

Table 2: Traceroute between the servers

The experiments were carried out during 15 days in November 2008. During these days some route changes were observed. However, the route in table 2 corresponds to the most stable one (more that the 80 % of the experiment time). In many cases, the route changed for a couple of hours and then came back to the stable one. The time between experiments was 15 minutes. The bandwidth of the access links of both servers are 4 Mbps and 10 Mbps respectively. We use 7 days of experiments as training samples and estimate two other days with this training sample. The first estimation was done using Nadaraya-Watson. In this case the variable X was the empirical distribution of the estimator $\widehat{q}_n$ and Y was the video sequence mean delay. We have used four different zones and the corresponding optimal kernel bandwidths, as explained in subsection 5.3. Mean delays vary between 220 and 250 ms. This estimation has a mean square error (MSE) of 8.6 over the two days validation sample. We have then performed the same estimation but using SVR. For the variable X we have

used different features of the estimator $\widehat{q}_n$. In table 3 we list the MSE for each individual feature.

Feature	MSE
mean of $\widehat{q}_n$	10.8
variance of $\widehat{q}_n$	12.8
percentile 10	36
percentile 20	36
percentile 30	25
percentile 40	14
percentile 50	12.8
percentile 60	11.8
percentile 70	11.8
percentile 80	12.3
percentile 90	13.6
percentile 95	13
percentile 99	26
percentage of empty queue	16
probe packet losses	23

Table 3: MSE of each individual feature

We have evaluated different combinations of these features. We found that using as X a vector containing the mean value, the percentile 70 of $\widehat{q}_n$ and the percentage of time with empty queue, the MSE is almost the same than the one obtained using the empirical distribution of $\widehat{q}_n$ and Nadaraya-Watson. Figure 24 shows the measured and estimated values of the video sequence mean delay. Note how the estimation has a good accuracy.

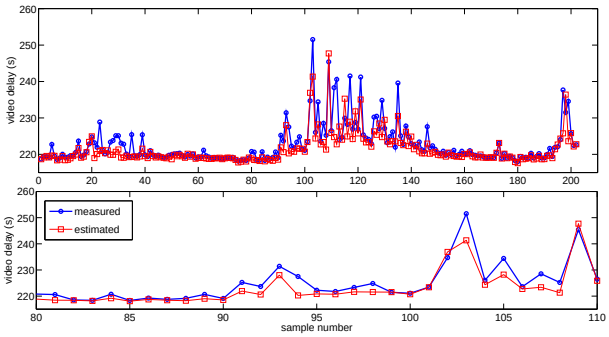

Figure 24: Estimated and measured video delay in a multidomain network using SVR. (The bottom figure is a zoom of the upper figure in a time interval.)

The next case is the estimation of the video sequence jitter. In this case in order to have similar MSE than in the functional case with Nadaraya-Watson it was necessary to consider nine features: mean value, variance, percentiles 50, 60, 70, 80, 90, 95 of $\widehat{q}_n$ and the percentage of time with empty queue. Figure 25 shows the measured and estimated value of the video jitter.

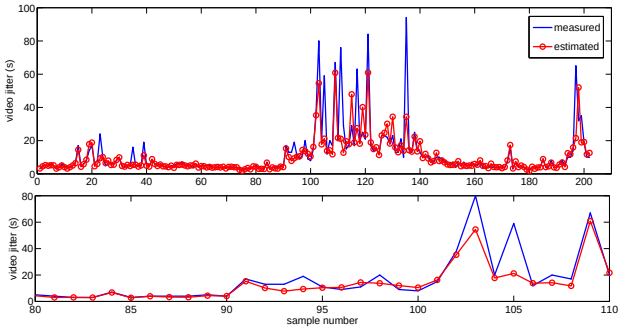

Figure 25: Estimated and measured video jitter in a multidomain network using SVR. (The bottom figure is a zoom of the upper figure in a time interval.)

Finally, we analyze the losses seen by the video sequence. In this case we analyze three different sets of features as variable X:

Case 1: the empirical distribution of $\widehat{q}_n$.

Case 2: the probe packet losses.

Case 3: the mean value, the variance, the percentiles 50, 60, 70, 80, 90, 95 of $\widehat{q}_n$, the percentage of time with empty queue and the probe packet losses.

The MSE in these cases is shown in Table 4, whereas figure 26 shows the measured and estimated value of the video packet losses using SVR.

Case	MSE
1	2×10^{-4}
2	1.6×10^{-4}
3	1.2×10^{-4}

Table 4: MSE for losses estimated with three different set of features

The last case scenario we shall consider is shown in figure 27. In this case we trained with the same 7 days as before, but the prediction is performed for a different day than the two previously considered. At the beginning and at the end of the plot the estimation seems to be accurate. However, in the middle zone the estimation follows the curve of the measured delay but with a difference of around 5 ms. Analyzing the information collected during the experiments we found that during this day the route changed. Table 5 shows part of the new route detected. We may conclude then that the new path latency is different from the stable one, but that the queues that affect the packets delay have essentially the same behavior.

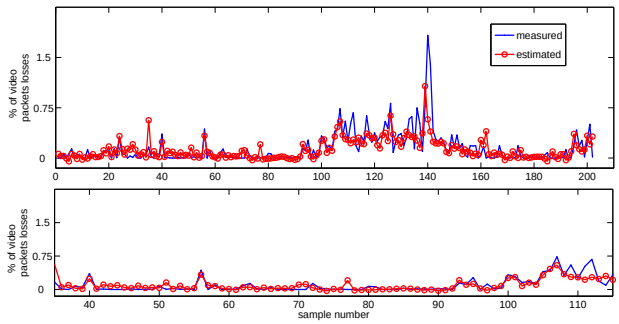

Figure 26: Estimated and measured video packet losses in a multidomain network using SVR.

hop number	interface	ip
9	142.46.128.90	(142.46.128.90)
10	if-1-150.mcore3.MTT-Montreal.as6453.net	(216.6.111.13)
11	if-13-0.mcore4.NQT-NewYork.as6453.net	(216.6.87.21)
12	if-5-0.mcore3.NYY-NewYork.as6453.net	(216.6.87.54)
13	if-12-0-0-723.core4.AEQ-Ashburn.as6453.net	(216.6.42.61)
14	sl-st20-ash-14-0-1.sprintlink.net	(144.223.246.105)
15	sl-bb21-dc-10-0-0.sprintlink.net	(144.232.20.151)
16	sl-bb21-mia-6-0-0.sprintlink.net	(144.232.9.26)
17	sl-st21-mia-3-0-0.sprintlink.net	(144.232.2.240)
18	sl-st21-mia-2-0.sprintlink.net	(144.232.9.199)

Table 5: part of the new route

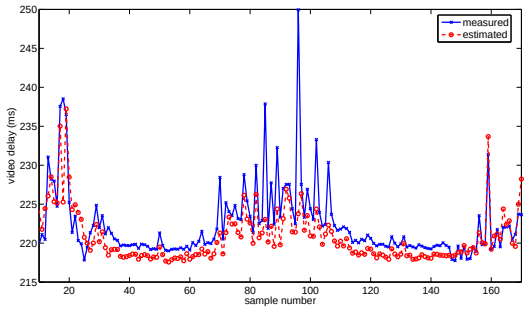

Figure 27: Estimated and measured video delay in a multidomain network using SVR under a route change with respect to the training phase.

Finally, let us highlight that the methods underestimate some peak values of the different Y parameters (delay, jitter, losses). However, it must be taken into account that these peaks are less frequent than the lower values of Y. Therefore, more training days are needed in order to estimate them with the same accuracy.

7.3. Cellular access tests

In this case a GPRS/EDGE connection is used with a PC and an EDGE modem. The video sequences are streamed from a server located at Facultad de Ingeniería, Universidad de la República to the PC. In this case the videos were encoded at an average rate of 96 kbps.

We have done 65 tests sending in each test a video sequence and a probe packets burst. In this case the delays observed are high and have strong variability. When, as in this case, the number of samples is relatively low, a technique usually applied to select or to validate the model is "one left out". For instance, to choose the parameters of SVR we have used the following procedure: for each pair of fixed parameters (δ, γ) we estimate each point with the rest of the samples (the 64 remaining points). The chosen (δ, γ) is the one that minimizes the MSE over the 65 estimated points.

We observe during the validation that the estimation was not accurate. More in particular, the resulting (δ, γ) results in a good estimation for the points in the zone with low delay or in the zone with high delay but not in both zones. Analyzing this result, we concluded that the nonstationarity of the data could be the cause of this problem. The nonstationarity was also present in the multidomain case. However, the number of samples there was larger than in this case, making its impact less noticeable.

Nonstationarity was previously observed and analyzed by other authors [40, 41, 42, 43], who applied SVM to other kinds of nonstationary data, for example to biological data. In their works they propose different solutions. The main idea behind them is to use a set of different kernels or a set of different parameters in different zones or time-scales of the samples. In this paper we follow this main idea, but propose a specific solution to our problem.

The method is as follows. We take the first 30 samples in order to select the (δ, γ) parameters. We divide these 30 samples in three classes, of 10 samples each, according to the value of Y. The classes correspond to the higher, lower and intermediate values of Y. We calculate the pair of parameters (γ_j, δ_j) that minimizes the MSE in each class j, using the one left out procedure described before. In addition, we calculate the barycentre of the points X_j of each class.

Next, we take the other 35 points not used to select the model in order to validate it. We classify each point X_k of the validation sample according to the distance between this point and the barycentre of each class. Once the class j_1 of each point is selected, we select the parameters $(\gamma_{j_1}, \delta_{j_1})$ of the corresponding class. Finally, we estimate each of the 35 Y_k using these parameters and the other 64 points.

Figures 28 and 29 show the video losses and its mean delay for the 35 points of the validation sample. The accuracy of the estimation is very good taking into ac-

count the variability of the data.

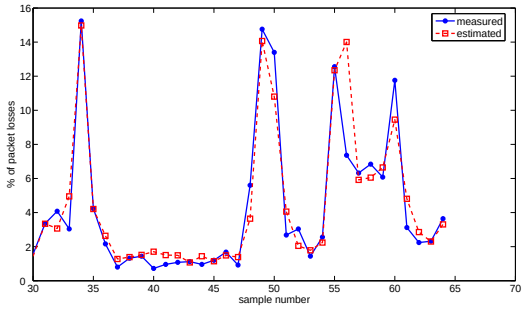

Figure 28: Estimated and measured video packet losses in a GPRS/EDGE network using SVR.

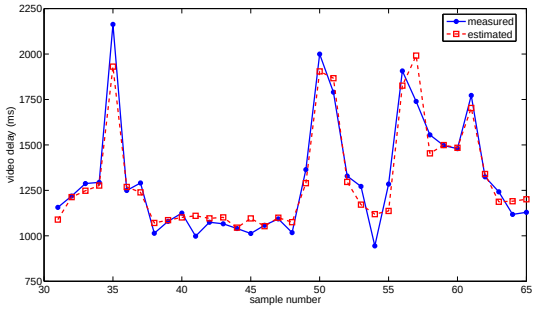

Figure 29: Estimated and measured video delay in a GPRS/EDGE network using SVR.

Finally we want to state some considerations on the algorithm performance for online applications like an admission control tool. The phase of the algorithm that is the most important in terms of performance is the estimation phase, and not the training one. Consequently, in order to analyze the performance we measure the time consumed for the estimation of a new end-to-end QoS parameter value Y for a system previously trained with 1500 training samples. Our monitoring system continuously sends the probe packets, and when a new video is required by the user, the system makes the QoS estimation by calling the libsvm tool [44]. It must be taken into account that the libsvm tool is a general propose library and is not optimized for online applications.

When the video request arrives, using the last 30 seconds of the probe packets interarrival times, the system calculates the average, variance, the percentile 90% of the queue estimator and percentage of time with empty queue. These four parameters were the same used for the system training. After the calculation of these four parameters the libsvm tool is called in order to estimate Y for this new point. The time consumed to estimate a new point with the previous procedure in a typical PC is between 200-300 ms. This time is very reasonable for a session level online applications like an admission control system.

8. Related work

The aim of this section is to compare our work with other approaches proposed to estimate the QoS seen by applications. We do not claim that the list of works discussed here is exhaustive. We will remark only the proposals that, in our opinion, are the most representative of different possible approaches that can be used for estimating QoS seen by applications.

One commonly used methodology infers the QoS seen by an application directly by the performance seen by a simple sequence of probe packets (using a ping for example) [45]. This methodology can lead to very important errors. For example, figure 20 shows that it is not possible to infer the performance of a video through the one obtained by another video with different traffic statistics. Only worse results can be expected if a ping is used to infer the video QoS performance.

To avoid this problem, some of the proposals to estimate QoS parameters were designed for specific applications or for a specific QoS metric. For example [46] predicts the packet losses from packets end-to-end delay variation modelling the correlation between both for a specific application. In this work they test their method in different ns-2 simulations and the experiments show a 60% – 90% accuracy. The main differences with our work are that they analyze only the estimation of packets losses and that in their methodology it is necessary to send the application traffic to measure the delay variations over the application packets in order to predict the packet losses. As they are motivated by VoIP applications, sending the application traffic is not a problem because this traffic generally does not overload the network. However, their methodology is not suitable for monitoring when the application traffic rate is not negligible compared with the network capacity, as for example in a high definition network or a video application over a cellular network.

Parlos [47] describes a predictor for a multi-step-ahead estimation of the end-to-end delay or round trip time. The methodology requires sending the application traffic during some time in order to collect the end-to-end delay data. With this data a neural network is trained, which is used in order to predict the end-to-end delay (or RTT) for some steps ahead in the future.

The author presents some experiments using TCP traffic. The main motivation of the work is the prediction of the TCP Round Trip Time. In the experiments Parlos uses a TCP source located in USA and a destination point located in Europe reporting an average error of around 10% during the experiments. As in the previous referenced work, this method focuses only in a specific QoS parameter (end-to-end delay or RTT in this case) and it is necessary to send the application traffic to train the neural network in the present conditions of the network in order to predict the end-to-end delay or RTT in the near future.

A different approach but with similar drawbacks can be found in the paper of Backhouse and Gu [48]. They propose a video packet loss prediction technique based on the relation between the time-varying cross traffic rate and the packet losses. This relationship is derived by studying the queuing properties in the network. The cross traffic is estimated from the interarrivals times of the video packets and applying a Bayesian model. They compare their model by simulations with other packet losses models like a two state Markovian model. Their goal is different than ours: they want to incorporate their predictor to a video codec whose encoding rate is adapted to the predicted packet loss probabilities.

Ohsaki et al. [49] model the relation between the packets interdeparture time (the system input) and the end-to-end packet delay variation (the output of the system) using a system identification technique. They use an ARX (Auto Regresive eXogenous) model in order to estimate the system transfer function. The cross traffic is modelled as white noise. Taking into account the aim of our work, the main drawback of their method is that the ARX model is a linear and time invariant model so it can predict only in the near future of the system where the linear approximation is valid.

Another tool to perform active measurements without loading the network can be found in [50]. In this paper the authors propose a new protocol called MGRP (Measurement Manager Protocol) an in-kernel service that schedules and transmits probes on behalf of active measurement tools. MGRP permits measurement algorithms to be written as if they were active but implemented as if they were passive. The active tools specify an entire train of probes and MRGP treats each probe as a vessel that can carry useful payload of other applications that sends data to the same destination. By filling most or all probes with useful data MGRP allows active algorithms to approach the low overhead of passive ones.

Tao and Guerin [20] infer the packet losses using a parametric model (a Hidden Markov Model). They assume a two state Markov model for the losses (periods with losses and periods without losses). They estimate the parameters of their loss model by sending probe packets. Later, in order to estimate the losses seen by the application knowing the application traffic rate they "sample" the losses Markov model. They report mean errors of around $1-3\%$ in the loss rate. Their estimations are done measuring the loss rate during 15 minutes between two Universities in the USA with synthetic ON-OFF traffic or a CBR-like video. The main differences with our work is that their method can only be applied to the packet losses process and that their model is parametric in the sense that the packet losses process is assumed to be a two-state markov model and the application traffic is modelled as a CBR or as an ON-OFF. They also have some specific assumptions in their model, for example they assume that the losses are not influenced by the application traffic. This assumption is not verified in our experiments where there are evidence of losses influenced by the application traffic. Although the reported errors are very low, in their experiments the loss rate is estimated over a 15 minutes period that is a long period compared with our estimations over video sequences of about 20 or 30 seconds. We have used these short sequences because for admission control applications we need to take decisions with present information. During long periods some videos could end, others can gain access to the network and the present network state can be very different than the last 15 minutes average.

9. Conclusions

This work estimates QoS parameters seen by applications. We propose a non intrusive procedure based on end-to-end active measurements and statistical learning tools. We compute two different estimators of the state of the network: the empirical distribution of probe packets interarrival times and an estimator of the queues in the path. The statistical learning approach gives accurate results, both in simulations and in different operational networks. With the empirical distribution of probe packets interarrival times and the empirical distribution of the queues estimator we obtain good results applying functional Nadaraya-Watson estimation. We also include some theoretical results that justify its application in this context. However, this approach has some drawbacks: the computational cost and the need to store all the data in order to define a model. With the estimator of the queues in the path we can extract a few characteristics and analyze the data with Support Vector Machines. We also study the problems that arise using

this tool when the data is nonstationary. We propose a solution for this problem allowing accurate results that may be applied for online estimations.

10. Acknowledgments

This work was partially supported by project CSIC-UdelaR "Modelado y evaluación del desempeño de redes inalámbricas estructuradas y mesh". We would like to thank the anonymous referees for their detailed and valuable comments, and to Federico Larroca for helpful discussions.

A. Proofs of consistency for the functional Nadaraya-Watson estimator in a nonstationary case

The Nadaraya-Watson estimator is

$$\widehat{\Phi}_n(x) = \frac{\sum_{i=1}^{n} Y_i K\left(\frac{\|x-X_i\|}{h_n}\right)}{\sum_{i=1}^{n} K\left(\frac{\|x-X_i\|}{h_n}\right)} = \frac{\sum_{i=1}^{n} Y_i K_n(X_i)}{\sum_{i=1}^{n} K_n(X_i)} \tag{11}$$

We consider

$$\widehat{\Phi}_n(x) = \frac{g_n(x)}{f_n(x)},$$

where

$$g_n(x) = \frac{1}{n\psi(h_n)} \sum_{i=1}^{n} Y_i K_n(X_i),$$

$$f_n(x) = \frac{1}{n\psi(h_n)} \sum_{i=1}^{n} K_n(X_i),$$

with $\psi(h_n)$ a normalization that depends on the concentration of samples around x and is defined later.

Lemma 1. *Let $X = (X_n)_{n\geq 1}$, $Y = \Phi(X) + \varepsilon$, with ε a centered real random variable independent from X that satisfy*

H 1. *There are two independent processes $\xi = (\xi_n)_{n\in\mathbb{N}}$, $Z = (Z_n)_{n\in\mathbb{N}}$, such that ξ is stationary, with values in a function space, Z is real with values in $\{z_1, \ldots, z_m\}$ and there exists a function φ that takes values in a function space $\mathcal{D}$ such that*

$$X_n = \varphi(\xi_n, Z_n).$$

H 2. *There exist positive functions $\psi, \psi_1, \ldots, \psi_m$ defined en $\mathbb{R}^+ \times \mathcal{D}$, $c_1, \ldots, c_k$ defined in $\mathcal{D}$ and a subset $\Delta \subset \{1, \ldots, m\}$ such that for all $h > 0$*

$$P\left[\|\varphi(\xi_1, z_k) - x\| \leq h\right] = c_k(x)\psi_k(h, x),$$

with $\lim_{h\to 0} \frac{\psi_k(h,x)}{\psi(h,x)} = 1$ if $k \in \Delta$, and $\lim_{h\to 0} \frac{\psi_k(h,x)}{\psi(h,x)} = 0$ if $k \in \Delta^c$, where Δ^c is the complement of subset Δ. In what follows we write $\psi(h_n)$ instead of $\psi(h_n, x)$, in order to simplify notation.

H 3. *The functions $u \mapsto \psi_k(u, x)$ are differentiable in $\mathbb{R}^+$, with derivative $\psi'_k(u, x)$ and*

$$\lim_{h\to 0} \frac{h}{\psi_k(h, x)} \int_0^1 K(u)\psi'_k(uh, x)du = d_k(x),$$

where d_k are functions defined in $\mathcal{D}$.

H 4. *For all $k \in \{1, \ldots, m\}$ the following limit exists*

$$\lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} P(Z_i = z_k)$$

and we denote it by p_k.

H 5. *Φ is a continuous function.*

H 6. *K is positive with support in $[0, 1]$.*

H 7. *The kernel bandwidth h_n satisfies that $\lim_{n\to\infty} h_n = 0$ and $\lim_{n\to\infty} n\psi(h_n) = 0$.*

Then,

$$\lim_{n\to\infty} E(f_n(x)) = f(x),$$

where $f(x) > 0$ and f is defined for all $u \in \mathcal{D}$ by

$$f(u) = \sum_{k\in\Delta} p_k d_k(u) c_k(u),$$

and

$$\lim_{n\to\infty} E(g_n(x)) = \Phi(x) f(x).$$

Remark 8. *Hypothesis H 2, as in [29] is about concentration of random variables $\varphi(\xi_n, z_k)$, represented by $\psi_k(h)$, in a ball centered at x with radius h. In the real case, for variables in $\mathbb{R}^d$ with a density function, the distribution in a ball centered in 0 and with radius h is proportional to h^d. The components of the mixture that finally determine the normalization ψ are those with indexes in Δ that correspond to the most concentrated variables around x. Hypothesis H 3 implies that $\|\varphi(\xi, z_k) - x\|$ has density $c_k(x)\psi'_k$. For variables in $\mathbb{R}^d$ with a density function, d_k is a constant for all $x \in \mathbb{R}^d$. Hypothesis H 4 guarantees some kind of stationarity "in mean". This hypothesis is verified for example by periodic random variables. A counterexample can be constructed as in example 2.2 in [33]. Hypotheses H 5, H 6, H 7 are usual for kernel estimation.*

In order to show almost sure convergence we consider the estimator conditioned to the values of Z, and we work with a random vector in $\mathbb{R}^{2m}$ as follows.

We consider the variable $\tilde{X}^n = (\tilde{X}^{1,n}, \ldots, \tilde{X}^{2m,n})$ with values in $\mathbb{R}^{2m}$ defined for $i \geq 1$ by

- for $l \in \{1, \ldots, m\}$

$$\begin{aligned}\tilde{X}_i^{l,n} &= \frac{1}{\sqrt{\psi(h_n)}} K_n\left(\varphi(\xi_i, z_l)\right)\left(\Phi(\varphi(\xi_i, z_l)) + \varepsilon_i\right) \\ &\quad - \frac{1}{\sqrt{\psi(h_n)}} E\left[K_n\left(\varphi(\xi_i, z_l)\right)\Phi(\varphi(\xi_i, z_l))\right],\end{aligned}$$

- for $l \in \{m+1, \ldots, 2m\}$

$$\begin{aligned}\tilde{X}_i^{l,n} &= \frac{1}{\sqrt{\psi(h_n)}} K_n\left(\varphi(\xi_i, z_{l-m})\right) \\ &\quad - \frac{1}{\sqrt{\psi(h_n)}} E\left[K_n\left(\varphi(\xi_i, z_{l-m})\right)\right]\end{aligned}$$

Theorem 8. *We assume hypotheses of Lemma 1 and the following additional hypotheses*

H 9. *For all $B \subset \mathbb{N}$, $n \in \mathbb{N}$, $i \in \{1, \ldots, 2m\}$*

$$E\left[S_n\left(B, \tilde{X}^{i,n}\right)^4\right] \leq C\left(\frac{card(B_n)}{n}\right)^2$$

where $B_n = B \cap [1, n]$ and $S_n\left(B, \tilde{X}^{i,n}\right) = \frac{1}{\sqrt{n}} \sum_{k \in B_n} \tilde{X}_k^{i,n}$

H 10. *The kernel bandwidth h_n satisfies*

$$\lim_{n\to\infty} h_n = 0$$

and for some $0 < \beta < 1$

$$\lim_{n\to\infty} \psi(h_n) n^{\beta/2} = 0.$$

Then,

$$\lim_{n\to\infty} \widehat{\Phi}_n(x) = \Phi(x)$$

almost surely.

Remark 9. *Hypothesis H 9 is about weakly dependence, and it may be obtained for example from α-mixing assumptions. Hypothesis H 10 gives a kernel bandwidth convergence speed to prove the theorem.*

A.1. Proof of lemma 1

$$E\left(f_n(x)\right) = \frac{1}{n\psi(h_n)} \sum_{i=1}^{n} E\left(K_n(X_i)\right)$$

As ξ and Z are independent

$$\begin{aligned}E(K_n(X_i)) &= E\{E(K_n(X_i)|Z_i)\} \\ &= \sum_{k=1}^{m} E\{K_n(\varphi(\xi_i, z_k))\} P(Z_i = z_k)\end{aligned}$$

For each k $(\varphi(\xi_i, z_k))_{i\geq 1}$ is a stationary sequence, then

$$E(f_n(x)) = \sum_{k=1}^{m}\left(\frac{1}{\psi(h_n)} E\left(K_n(\varphi(\xi_1, z_k))\right)\frac{1}{n}\sum_{i=1}^{n} P(Z_i = z_k)\right).$$

Hypothesis H 2 implies that the density of $\|\varphi(\xi_1, z_k) - x\|$ is the function $u \mapsto c_k(x)\psi_k'(u, x)$ and then

$$E\left[K_n(\varphi(\xi_1, z_k))\right] = h_n c_k(x) \int_0^1 K(u)\psi_k'(uh_n, x)du.$$

By hypotheses H 2, H 3, H 7 we have that

$$\lim_{n\to\infty} \frac{1}{\psi(h_n)} E\left[K_n(\varphi(\xi_1, z_k))\right] = d_k(x)c_k(x)1_{\{k\in\Delta\}},$$

and by H 4

$$\begin{aligned}\lim_{n\to\infty} E(f_n(x)) &= \lim_{n\to\infty} \sum_{k=1}^{m}\left(\frac{1}{\psi(h_n)} E\left(K_n(\varphi(\xi_1, z_k))\right)\right. \\ &\quad \left.\frac{1}{n}\sum_{i=1}^{n} P(Z_i = z_k)\right) \\ &= \sum_{k\in\Delta} p_k d_k(x) c_k(x)\end{aligned}$$

Analogously

$$\begin{aligned}E(g_n(x)) &= \sum_{k=1}^{m}\left(\frac{E\left(\phi\left(\varphi(\xi_1, z_k)\right)K_n(\varphi(\xi_1, z_k))\right)}{\psi(h_n)}\right. \\ &\quad \left.\frac{1}{n}\sum_{i=1}^{n} P(Z_i = z_k)\right).\end{aligned}$$

Then

$$E(g_n(x)) = \phi(x)E(f_n(x)) + R_n,$$

where

$$\begin{aligned}R_n &= \sum_{k=1}^{m}\left(\frac{E\left[\{\phi\left(\varphi(\xi_1, z_k)\right) - \phi(x)\}K_n(\varphi(\xi_1, z_k))\right]}{\psi(h_n)}\right. \\ &\quad \left.\frac{1}{n}\sum_{i=1}^{n} P(Z_i = z_k)\right) \\ &\leq \sup_{u:\|x-u\|\leq h_n} |\phi(u) - \phi(x)|\, E(f_n(x))\end{aligned}$$

and $\lim_{n\to\infty} E(g_n(x)) = \phi(x)f(x)$ comes from continuity of ϕ and $\lim_{n\to\infty} E(f_n(x)) = f(x)$.

A.2. Proof of theorem 8

As $\phi_n(x) = \frac{g_n(x)}{f_n(x)}$ it is sufficient to prove that $f_n(x)$ converges almost surely to $f(x)$ and that $g_n(x)$ converges almost surely to $\phi(x)f(x)$. We have that

$$f_n(x) - f(x) = f_n(x) - E(f_n(x)) + E(f_n(x)) - f(x)$$

$$g_n(x) - \phi(x)f(x) = g_n(x) - E(g_n(x)) + E(g_n(x)) - \phi(x)f(x)$$

and from proposition 1

$$\lim_{n\to\infty} E(f_n(x)) - f(x) = 0,$$

$$\lim_{n\to\infty} E(g_n(x)) - f(x)\phi(x) = 0.$$

Then we must prove that $f_n(x) - E(f_n(x))$ and $g_n(x) - E(g_n(x))$ converge to zero almost surely. In order to prove that $f_n - E(f_n(x))$ converges a zero almost surely it is sufficient to prove complete convergence, that is for all $\varepsilon > 0$ the series $\sum P(|f_n(x) - E(f_n(x))| > \varepsilon)$ is convergent. $f_n(x) - E(f_n(x)) = \frac{1}{n\psi(h_n)} \sum_{i=1}^{n} [K_n(X_i) - E(K_n(X_i))] = \frac{1}{\sqrt{n\psi(h_n)}} S_n$ where

$$S_n = \frac{1}{\sqrt{n\psi(h_n)}} \sum_{i=1}^{n} [K_n(X_i) - E(K_n(X_i))]$$

Applying Markov inequality we have that $P(|f_n(x) - E(f_n(x))| > \varepsilon) \leq E\left(\frac{[f_n(x) - E(f_n(x))]^4}{\varepsilon^4}\right) = \frac{E(S_n^4)}{(n\psi(h_n))^2\varepsilon^4}$ We compute $E\left(S_n^4\right)$ conditioned to the trajectory of Z, that is Z^∞ that verifies $Z^\infty = z^\infty$ if $Z_i = z_{j_i}$ for all $i \geq 1$. A trajectory is a random variable that takes values in the space of sequences with values in $\{z_1, \ldots, z_m\}$.

$$\begin{aligned} E\left(S_n^4\right) &= E\left[E\left(S_n^4 | Z^\infty\right)\right] \\ &= \int_T E\left(S_n^4 | Z^\infty = z^\infty\right) d\mu \end{aligned}$$

where T is the space of trajectories. To prove complete convergence we will prove that for a subset of trajectories with probability one it is verified that $E(S_n | Z^\infty = z^\infty) \leq a_n$ where $\sum \frac{a_n}{(n\psi(h_n))^2}$ converges. For each trajectory we compute

$$\begin{aligned} &E\left(S_n^4 | Z^\infty = z^\infty\right) \\ &= E\left(\left[\frac{1}{\sqrt{n}} \sum_{i=1}^{n} \frac{(K_n(\varphi(\xi_i, z_{l_i})) - E[K_n(\varphi(\xi_i, z_{l_i}))])}{\sqrt{\psi(h_n)}}\right]^4\right) \\ &= E\left(\left[\frac{1}{\sqrt{n}} \sum_{i=1}^{n} \tilde{X}_i^{l_i,n}\right]^4\right) \end{aligned}$$

with $l_i \in \{m+1, \ldots, 2m\}$. We group terms depending on the values of Z and we obtain

$$\sum_{i=1}^{n} \tilde{X}_i^{l_i,n} = \sum_{k=m+1}^{2m} \sum_{i \in A_n^k} \tilde{X}_i^{k,n}$$

where $A_n^k = \{i : Z_i = z_k\} \cap [1, n]$ and

$$\begin{aligned} \frac{1}{\sqrt{n}} \sum_{i=1}^{n} \tilde{X}_i^{l_i,n} &= \sum_{k=m+1}^{2m} \frac{1}{\sqrt{n}} \sum_{i \in A_n^k} \tilde{X}_i^{k,n} \\ &= \sum_{k=m+1}^{2m} S_n\left(A^k, \tilde{X}^{k,n}\right) \end{aligned}$$

To compute $E\left(\left[\sum_{k=m+1}^{2m} S_n\left(A^k, \tilde{X}^{k,n}\right)\right]^4\right)$ we have that

$$\begin{aligned} &\left[\sum_{k=m+1}^{2m} S_n\left(A^k, \tilde{X}^{k,n}\right)\right]^4 \\ &= \left[\sum_{\substack{k_1;\ldots;k_m \\ k_1+\ldots+k_m=4}} \frac{4! S_n^{k_1}\left(A^{m+1}, \tilde{X}^{m+1,n}\right) \ldots S_n^{k_m}\left(A^{2m}, \tilde{X}^{2m,n}\right)}{k_1! \ldots k_m!}\right] \end{aligned}$$

Taking expectation and applying Holder inequality and hypothesis H 9,

$$\begin{aligned} &E\left[S_n^{k_1}\left(A^{m+1}, \tilde{X}^{m+1,n}\right) \ldots S_n^{k_m}\left(A^{2m}, \tilde{X}^{2m,n}\right)\right] \\ &\leq E\left(\left[S_n^4\left(A^{m+1}, \tilde{X}^{m+1,n}\right)\right]^{\frac{k_1}{4}}\right) \ldots E\left(\left[S_n^4\left(A^{2m}, \tilde{X}^{2m,n}\right)\right]^{\frac{k_m}{4}}\right) \\ &\leq C(k_1, \ldots, k_m)\left(\frac{card\left(A_n^{m+1}\right)}{n}\right)^{\frac{k_1}{2}} \ldots \left(\frac{card\left(A_n^{2m}\right)}{n}\right)^{\frac{k_m}{2}} \end{aligned}$$

Then, as $card\left(A_n^k\right) \leq n$, we have that $E\left(S_n^4 | Z^\infty = z^\infty\right) \leq C$. The proof of $E\left(S_n^4 | Z^\infty = z^\infty\right) \leq C$ is independent of the trajectory. To prove complete convergence is sufficient to choose h_n such that

$$\sum_{n=1}^{\infty} \frac{1}{(n\psi(h_n))^2}$$

converges, for example $(n\psi(h_n))^2 \geq n^\alpha$ with $\alpha > 1$. Considering $\psi(h_n) \geq \frac{1}{n^{\beta/2}}$, with $0 < \beta < 1$ and hypothesis H 10 we obtain complete convergence to zero for $f_n(x) - E[f_n(x)]$. The proof for $g_n(x) - E[g_n(x)]$ is analogous considering $\tilde{X}^{k,n}$ with $k \in \{1, \ldots, m\}$.

References

[1] M. Adler, T. Bbu, R. Sitaraman, D. Towsley, Tree layout for internal network characterizations in multicast networks, in: J.

Crowcroft, M. Hofmann (Eds.), Proceedings of the Third international Cost264 Workshop on Networked Group Communication, Lecture Notes In Computer Science 2233, Springer-Verlag, London, 2001, pp. 189–204.

[2] R. Cáceres, N. Duffield, G. Horowitz, D. Towsley, Multicast-based inference of networkinternal loss characteristics, IEEE Transactions on Information Theory 45(7) (1999) 2462–2480.

[3] V. Jacobson, Pathchar, a tool to infer characteristics of internet path, presented at the Mathematical Sciences Research Institute, 1997.

[4] M. Jain, C. Dovrolis, Pathload: A measurement tool for end-to-end available bandwidth, Proceedings of Passive and Active Measurements (PAM), (2002) 14–25.

[5] M. Jain, C. Dovrolis, End-to-end available bandwidth: measurement methodology, dynamics, and relation with TCP throughput, IEEE/ACM Transactions in Networking, 11(4) (2003) 537–549.

[6] M. Jain, C. Dovrolis, End-to-End Estimation of the Available Bandwidth Variation Range, Proceedings of the 2005 ACM SIGMETRICS international conference on Measurement and modelling of computer systems (2005) 265–276.

[7] K. Lai, M. Baker, Nettimer: A Tool for Measuring Bottleneck Link Bandwidth. Proceedings of the USENIX Symposium on Internet Technologies and Systems (2001) 123–134.

[8] E. Lawrence, G. Michailidis, V.N. Nair, Inference of Network Delay Distributions Using the EM Algorithm, Technical Report, University of Michigan, 2003.

[9] F. Lo Presti, N.G. Duffield, J. Horowitz, D. Towsley, Multicast-Based Inference of Network-Internal Delay Distributions, ACM/IEEE Transactions on Networking 10 (2002) 761–775.

[10] V. Ribeiro, R. Riedi, R. Baraniuk, J. Navratil, L. Cotrell, PathChirp: Efficient Available Bandwidth Estimation for Network Paths. Passive and Active Measurement Workshop (2003).

[11] J. Strauss, D. Katabi, F. Kaashoek, A measurement study of available bandwidth estimation tools, Internet Measurement Workshop, Proceedings of the 2003 ACM SIGCOMM conference on Internet measurement (2003) 39–44.

[12] Y. Tsang, M. Coates, R. Nowak, Network Delay Tomography, IEEE Transactions on Signal Processing 51 (2003) 2125–2136.

[13] E.A. Nadaraya, Nonparametric estimation of probability densities and regression curves, Mathematics and its Applications (Soviet Series), 20. Dordrecht: Kluwer Academic Publishers Group, 1989.

[14] F. Ferraty, P. Vieu, Nonparametric Functional data analysis: Theory and Practice, Springer Series in Statistics. New York: Springer, 2006.

[15] B.E. Boser, I.M. Guyon , V.N. Vapnik, A training algorithm for optimal margin classifiers, Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory, ACM Press (1992) 144–152.

[16] V.N. Vapnik, The nature of statistical learning theory, Springer NY, 1995.

[17] R. Beverly, K. Sollins, A. Berger, SVM Learning of IP Address Structure for Latency Prediction, ACM MineNet06, Pisa, Italy (2006) 299–304.

[18] P. Bermolen, D.Rossi, Support Vector Regression for Link Load Forecast, Proc. of IEEE IT-NEWS08, Venice, Italy (2008) 191–201.

[19] M. Mirza, J. Sommers, P. Bardford, X. Zhu, A machine learning approach to TCP throughput prediction, In ACM SIGMETRICS07, San Diego, CA (2007) 97–108.

[20] S. Tao, R. Guerin, On-line estimation of internet path performance: an application perspective, INFOCOM, Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies, 3 (2004) 1774–1785.

[21] S. Machiraju, D. Veitch, F. Baccelli, A. Nucci, J. Bolot, Theory and practice of cross-traffic estimation, SIGMETRICS (2005) 400–401.

[22] E.A. Nadaraya, On estimating regression, Theory of Probability and its Applications 9(1) (1961) 141–142.

[23] G.S. Watson, Smooth regression analysis, Sankhy$\bar{a}$ Ser. A, 26 (1964) 359–372.

[24] S.M. Clarke, J.H. Griebsch, T.W. Simpson, Analysis of Support Vector Regression for approximation of Complex Engineering Analyses, Journal of mechanical design (2005).

[25] A.J. Smola, B. Schölkopf, K.R. Müller, The Connection Between Regularization Operators and Support Vector Kernels, Neural Networks, 11(4) (1998) 637-649.

[26] F. Ferraty, A. Goia, P. Vieu, Functional nonparametric model for time series: a fractal approach for dimension reduction. Test, 11 (2002) 317–344.

[27] F. Ferraty, P. Vieu, Nonparametric models for functional data, with application in regression, time-series prediction and curve discrimination. J. Nonparametr. Stat., 16 (2004) 111–125.

[28] F. Ferraty, A. Mas, P. Vieu, Nonparametric regression on functional data: inference and pratical aspects. Australian and New Zealand Journal of Statistics, 11 (2007).

[29] E. Masry, Nonparametric regression estimation for dependent functional data: asymptotic normality. Stochastic Process. Appl., 115 (2005) 155–177.

[30] Y. Zhang, V. Paxson, S. Shenker, S., The Stationarity of Internet Path Properties: Routing, Loss, and Throughput. ACIRI Technical Report (2000).

[31] Y. Zhang, N. Duffield, On the constancy of internet path properties, IMW '01: Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement (2001) 197–211.

[32] T. Karagiannis, M. Molle, M. Faloutsos, A. Broido, A nonstationary Poisson view of Internet traffic, Proc. of INFOCOM 2004. Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies, 3 (2004) 1558–1569.

[33] G. Perera, Irregular sets and central limit theorems. Bernoulli, 8 (2002) 627–642.

[34] L. Aspirot, P. Belzarena, B. Bazzano, G. Perera, End-To-End Quality of Service Prediction Based On Functional Regression. Proc. Third International Working Conference on Performance Modelling and Evaluation of Heterogeneous Networks (HET-NETs 2005), Ilkley, UK (2005).

[35] L. Aspirot, P. Belzarena, B. Bazzano, G. Perera, End-To-End Quality of Service Prediction Based On Functional Regression, Performance Modelling and Analysis of Heterogeneous Networks. Demetres Kouvatsos (Eds.), River Publishers, 8 (2009) 153–168.

[36] L. Aspirot, K. Bertin, G. Perera, Asymptotic normality of the Nadaraya-Watson estimator for nonstationary data. Journal of Nonparametric Statistics, 21 (2009) 535–551.

[37] McCanne S., Floyd S., ns network simulator, url:http://www.isi.edu/nsnam/ns/.

[38] International Telecommunications Union, ITU-T Recommendation H.264: Advanced video coding for generic audiovisual services, url:http://www.itu.int/rec/T-REC-H.264-200711-I/en, (2007).

[39] Apple, Quicktime software, url:http://images.apple.com/quicktime/pdf/H264_Technology_Brief.pdf

[40] G.R. Lanckriet, N. Cristianini, P. Barlett, L. El Ghaoui, M.I. Jordan, Learning the kernel matrix with semi-definite programming, Proceedings of the 19th International Conference on Machine Learning, Sydney, Australia, Morgan Kauffman (2002).

[41] D.P. Lewis, T. Jebara, W.S. Noble, Nonstationary kernel combination, Proceedings of the International Conference on Machine

Learning, Pittsburgh, PA (2006).
[42] C.S. Ong, A.J. Smola, R.C. Williamson, Learning the kernel with hyperkernels, Journal of Machine Learning Research, 6 (2005) 1043–1071.
[43] P. Pavlidis, J. Weston, J. Cai, W.N. Grundy, Gene functional classification from heterogeneous data, Proceedings of the Fifth Annual International Conference on Computational Molecular Biology (2001) 242–248.
[44] C-C. Chang, C-J. Lin, LIBSVM: a library for support vector machines,
url:http://www.csie.ntu.edu.tw/ cjlin/libsvm, (2001).
[45] CISCO, IP SLA Monitoring,
url:http://www.cisco.com/en/US/docs/ios/ipsla/configuration/guide/sla-redirect.html
[46] L. Roychoudhuri, E.S. Al-Shaer, Real-time packet loss prediction based on end-to-end delay variation Network and Service Management, IEEE Transactions on Network and Service Management 2(1) (2005) 29–38.
[47] A.G. Parlos, Identification of the Internet end-to-end delay dynamics using multi-step neuro-predictors, IJCNN '02, Proceedings of the 2002 International Joint Conference on Neural Networks 3, (2002) 2460–2465.
[48] A. G. Backhouse, I. Y. H. Gu, Bayesian traffic dynamics and packet loss prediction for video over IP networks. Multimedia Systems 11(5) (2006) 468–479.
[49] H. Ohsaki, M. Morita, M. Murata, On modelling Round-Trip Time Dynamics of the Internet Using System Identification, Lecture Notes on Computer Science (LNCS) 2343, Springer, (2002) 359–371.
[50] P. Papageorge , J. McCann , M. Hicks, Passive aggressive measurement with MGRP. ACM SIGCOMM Computer Communication Review 39(4), (2009) 241–250.