This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE INFOCOM 2010 proceedings
This paper was presented as part of the main Technical Program at IEEE INFOCOM 2010.

DiffProbe: Detecting ISP Service Discrimination

Partha Kanuparthy, Constantine Dovrolis
School of Computer Science, Georgia Institute of Technology
{partha,dovrolis}@cc.gatech.edu

***Abstract*—We propose an active probing method, called Differential Probing or DiffProbe, to detect whether an access ISP is deploying forwarding mechanisms such as priority scheduling, variations of WFQ, or WRED to discriminate against some of its customer flows. DiffProbe aims to detect if the ISP is doing one or both of delay discrimination and loss discrimination. The basic idea in DiffProbe is to compare the delays and packet losses experienced by two flows: an Application flow A and a Probing flow P. The paper describes the statistical methods that DiffProbe uses, a novel method for distinguishing between Strict Priority and WFQ-variant packet scheduling, simulation and emulation experiments, and a few real-world tests at major access ISPs.**

I. Introduction

There is significant interest recently about the so-called "Network Neutrality" debate [9]. Users are concerned that their access ISPs will soon start degrading the network performance that is offered to certain applications, such as peer-to-peer file sharing, or "over-the-top" services (such as Skype, Vonage or Joost) that may be competing with similar services offered by the ISP. There is already evidence that some ISPs are discriminating against BitTorrent traffic by rate limiting or blocking such flows [8].

In this paper, we propose an active probing method, referred to as *Differential Probing* or DiffProbe, that can be used to detect delay and/or loss discrimination of given traffic flows. Such discrimination can be easily performed by ISPs today, given that most routers allow real-time classification of traffic and provide packet scheduling and buffer management mechanisms that can be used for service discrimination (such as Strict Priority (SP) scheduling, Weighted Fair Queueing (WFQ), or Weighted RED packet dropping). We also show how to distinguish between SP and WFQ scheduling variants.

The detection problem that we focus on can also be viewed as an instance of a new class of *network tomography* [7] problems. Instead of estimating internal link delays or losses or the topology of the network, in this class of tomography problems the objective is to identify the type of forwarding modules that a packet flow goes through. In this paper, we consider two packet scheduling forwarding modules (SP and WFQ) as well as discriminatory packet dropping schemes such as WRED. Our objective is to design and evaluate the probing and statistical methods for the detection of these forwarding modules. A large-scale measurement study using these methods is the subject of our ongoing work, and it will be described in a follow-up paper.

The paper is organized as follows. Section II presents our model for ISP service discrimination. Section III gives the basic idea of DiffProbe and describes the probing pattern. Sections IV and V focus on the delay and loss discrimination detection problems. We have implemented and tested our tool on several ISPs, as described in Section VI. Section VII evaluates the detection accuracy with simulation and controlled emulation experiments. Section VIII presents related work, while Section IX concludes.

II. Basic Model and Definitions

Our basic model is illustrated in Figure 1. A number of users are connected to an ISP I through access links. The user traffic goes through a classifier M, which marks flows as High (H) priority or Low (L) priority. We assume that the classification is done at the granularity of IP flows, even though our method is agnostic to the exact classification scheme (whether it is payload-based, port-based, a behavioral method like BLINC [10], etc).

If the ISP discriminates against low priority traffic, the classified traffic then goes through a *discriminatory ISP forwarding module* that applies different packet scheduling and buffer management policies to the two classes of traffic. Most routers today implement at least two discriminatory schedulers: Strict Priority (SP), and variants of Weighted Fair Queueing (referred to as WFQ in the rest of this paper). SP services a packet from the L queue only if the H queue is empty when the link becomes available. A WFQ scheduler guarantees a minimum bandwidth share to each class[1]. Even though many other scheduling algorithms have been proposed in the literature, SP and WFQ variants are the main schedulers that are available in routers today.

In terms of buffer management and loss discrimination mechanisms, most routers today support Weighted RED (WRED) [2], allowing incoming L packets to be dropped with a higher probability than incoming H packets. Another form of loss discrimination can be performed using the Drop-Longest-Queue policy, which removes a potentially backlogged packet from the longest queue when there is no buffer for an incoming packet.

On the other hand, if the ISP does not perform discrimination, we expect that the scheduling discipline will be First-Come First-Served (FCFS), there will be a single queue for all traffic, and the buffer management policy will be Drop-Tail (DT) (i.e., drop an arriving packet if there is no space to store it). Note that in some cases, an ISP may conduct loss discrimination but not delay discrimination (e.g., to use

[1]The many variants of WFQ differ in how accurate this allocation is across flows in small timescales, an issue that is not important in our context [19].

978-1-4244-5837-0/10/$26.00 ©2010 IEEE

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE INFOCOM 2010 proceedings
This paper was presented as part of the main Technical Program at IEEE INFOCOM 2010.

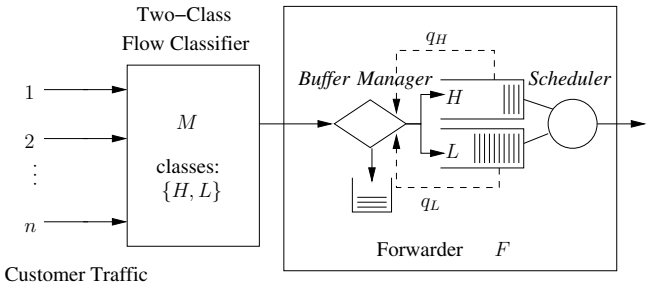

Fig. 1: Model of access ISP discrimination.

FCFS scheduling and WRED on a single queue), or delay discrimination but not loss discrimination (e.g., to use SP scheduling and DT buffer management with both queues sharing the same pool of packet buffers). Our high-level objective is to enable a user of ISP *I* to detect whether *I* performs any type of service discrimination on her traffic using an active probing methodology.

Note that it is possible that an ISP deploys discriminatory mechanisms, but without really affecting the user traffic. All previously mentioned forwarding mechanisms (SP, WFQ, WRED, etc) are identical to FCFS-DT when there are no backlogged packets at the discriminatory link, which is often the case under low load conditions. Obviously, we are not interested in such low load conditions because *there is no effective discrimination* in such cases. Besides, the ISP would not have the incentive to deploy discriminatory mechanisms if they would remain idle. Instead, we aim to detect discrimination when it actually affects user traffic. This would typically happen during time periods, potentially short, of high load at the discriminatory link. This does not mean that we assume that the ISP network is heavily loaded. The more relevant question is whether a user observes any service discrimination even when she receives (or sends) traffic at the maximum possible rate that her access link allows. If not, for all practical purposes the ISP does not deploy service discrimination on her traffic.

III. Differential Probing

The basic idea in Differential Probing is to generate two flows: an *Application (A) flow* that may be classified by the ISP as low priority, and a *Probing (P) flow* that should be classified as normal traffic and thus will not be discriminated against. The user first sends (and then receives) these two flows through the network simultaneously, and then compares their delay and loss statistical characteristics. Discrimination is detected when the two flows have experienced statistically significant queueing delays and/or loss rate.[2] In this paper, we assume that A and P flows are generated between the same end-points. In future work, we plan to extend the proposed architecture to cover cases where the A and P flows traverse different paths but share a discriminatory link.

The A flow can be generated by an actual application or it can be an application packet trace that the user replays. It represents traffic that the user is suspecting the ISP may be discriminating (e.g., BitTorrent or Skype traffic). The P traffic is a synthetic flow that is created by DiffProbe under two constraints. First, it should be sufficiently different than the A flow so that it does not get classified in the same way. Second, it should be sufficiently similar with A so that it observes the same network performance, statistically speaking, in terms of delays or packet losses if there were no discrimination. For example, if the ISP classifies traffic based on port numbers, the P flow can be identical with the A flow but it should use different ports. If the classification is based on packet payload information (e.g., specific HTTP strings), the P flow can have randomized payloads. If the classification is behavioral-based, focusing on specific flow features such as packet sizes, packet interarrivals, port numbers, etc, the P flow can be created in principle as a sufficiently randomized version of the A flow (e.g., with distorted packet sizes, average rate, rate fluctuations, etc). Wright et al. [20] show that it is possible to defeat statistical traffic classification by changing flow characteristics. In practice, our DiffProbe implementation generates P flows from Skype and Vonage A flows using a combination of port, payload, packet size and rate randomization. We expect that this combined randomization would be sufficient for most traffic classifiers today.[3]

To ensure that the two flows see similar network performance when the ISP does not perform discrimination, we rely on the following two techniques. First, we consider only those P packets that have been sent very close in time with a corresponding A packet. Thus, even if the P flow can include many more packets than the A flow, with different sizes and interarrivals, we rely on *paired statistics* and consider (A, P) packet pairs that have "sampled" the ISP discriminatory link at about the same time. Second, a P packet that is sent shortly after an A packet has the same size as the latter. This ensures that the network transmission delays of the (A, P) packet pairs that we consider are equal.

A. Probing pattern

Differential Probing works by sending the A and P flows through the ISP network simultaneously, and then comparing their delay variations and packet losses. Specifically, DiffProbe creates the following probing pattern that consists of two phases, a Balanced Load Period (BLP) followed by a Load Increase Period (LIP). In the following, we denote by $\lambda_A(t)$ the nominal rate of the A flow and by $\lambda_P(t)$ the nominal rate of the P flow, at time t. The flows are of variable bitrate, and so these rates vary with time.

- BLP: we send both flows at their nominal rates.
- LIP: we scale up the rate $\lambda_P(t)$ (by scaling down the packet interarrivals) of the P flow by a factor $g(t) > 1$.

[2]When we refer to "delays" in this paper we mean the end-to-end delays of a flow, after subtracting the minimum observed measurement from the raw end-to-end delay measurements. The presence of a clock offset does not influence these measurements because we focus on relative delays and not absolute delays.

[3]Most commercial classifiers are based on port numbers and payload information. Behavioral classification is viewed as not sufficiently accurate.

The reason we generate a LIP period is explained next. Our objective is to maximize the chances that there is some queueing in the (potentially discriminatory) ISP network. As previously discussed, without having some queueing in the ISP's network it is not important whether a delay and/or loss discrimination mechanism is deployed. Given that the user's access link is probably of lower capacity than the ISP's links, the highest rate that the user can generate is that of her access link. We cannot modify the rate of the A flow, however, as that may affect its classification by the ISP. Consequently, during a LIP period, we artificially increase the rate of the P flow to the point that, together with the A flow, the two flows almost saturate the user's access link. Specifically, we dynamically adjust the P flow rate so that: $\lambda_A(t)+g(t)\lambda_P(t) \approx C_a\,(1-\epsilon)$, where C_a is the user's access link capacity (in the upstream or downstream direction, depending on the direction of probing), and $\epsilon > 0$. Thus, our goal is to *not* introduce queueing at the user's access link, and focus instead on the delays/losses that take place at the ISP's network. In practice, we use $\epsilon = 0.1$ and calculate $g(t)$ with a sliding window estimate of $\lambda_A(t)$. To avoid probing intrusiveness, we increase ϵ if we observe significant losses in the P flow. The LIP duration is chosen so that we have a sufficiently large sample of (A, P) packet pairs to detect loss discrimination (see Section V). The reason behind the BLP period is described next.

Unidentifiability: The BLP is used to identify cases in which we cannot detect whether the ISP deploys service discrimination mechanisms. We compare the higher delays of P packets during the LIP period with the average delays of P packets during the BLP period. If the former are not significantly larger, the stimulus that we generated during the LIP period was not sufficiently high to trigger a significant increase in the queueing delays of the P flow. We view these cases as *unidentifiable*, given that even if the ISP deploys some discriminating mechanisms, those mechanisms would have no significant effect on the user's traffic.

Specifically, we compare the 90th percentile of P flow delay distribution during the LIP period ($\mathcal{D}_{0.9}(P)$) with median delay of the same flow during the BLP period ($\mathcal{D}_{0.5}^{BLP}(P)$):

$$\mathcal{D}_{0.9}(P) > (1+\delta)\mathcal{D}_{0.5}^{BLP}(P) \tag{1}$$

where $\delta > 0$. We choose $\delta = 0.1$ based on empirical observations. We say that delay discrimination is unidentifiable if the above condition is *not* true. We account for any clock skew that may exist between the BLP and LIP periods. The BLP duration is chosen reasonably large (at least 10s) to ensure a sufficiently large number of samples.

IV. Delay Discrimination Detection

Under FCFS scheduling, the two flows will be serviced by the same queue. So, at least in statistical terms, the A and P flows would observe similar delay distributions. On the other hand, a Strict Priority scheduler will provide lower delays to the P flow packets as long as there is some backlog in the discriminatory link. The WFQ scheduler can be used to provide delay discrimination only if the bandwidth share that is provided to the high priority class is larger than the bandwidth share provided to the low priority class, relative to the traffic load of the two classes. Specifically, suppose that the WFQ weights ϕ_H and ϕ_L are assigned to high priority and low priority traffic, respectively (with $\phi_H + \phi_L = 1$). Let λ_i be the offered load in class i. To achieve delay discrimination in favor of the high priority, the ISP should make sure that

$$\alpha_H > \alpha_L \tag{2}$$

where

$$\alpha_i = \frac{\phi_i}{\lambda_i}, i \in \{H, L\}$$

Note that if $\alpha_H \gg \alpha_L$, a WFQ scheduler would exhibit a behavior similar to SP, i.e., it would service low priority packets only when there are no backlogged high priority packets.

We detect delay discrimination as follows. Recall that the A flow is classified as low priority, while P is classified as high priority. We observe the empirical distribution of delays of the A and P flow packets during the LIP period; call them $\mathcal{D}(A)$ and $\mathcal{D}(P)$, respectively. We detect delay discrimination (SP or WFQ) when:

$$\mathcal{D}(A) \gg \mathcal{D}(P) \tag{3}$$

On the other hand, we detect no delay discrimination, i.e., FCFS scheduling, when:

$$\mathcal{D}(A) \approx \mathcal{D}(P) \tag{4}$$

Test for Equality of Delay Distributions: We first perform the test of equality of distributions (4) as follows. Our test is based on the non-parametric *Kullback-Leibler (KL) divergence*, and it is motivated by the test presented in [17]. The KL-test does not assume any priors about the input delay distributions. We have not used the well-known Kolmogorov-Smirnov (KS) test, since that test can be inaccurate when the underlying distributions exhibit *discontinuities*. The delays of Internet paths usually include a large number of samples close to the sum of the propagation and transmission delays, causing significant discontinuities.

The empirical distributions $\mathcal{D}(A)$ and $\mathcal{D}(P)$ are constructed from the measured delay timeseries $\{t_i^A, d_i^A\}$ and $\{t_j^P, d_j^P\}$ of A and P flows respectively. Timestamps t^A and t^P are taken at the sender. We first pre-process the two timeseries to form a *paired* sample $\mathcal{D}$ as follows. For each delay sample (t_i^A, d_i^A), we find the nearest sample (t_j^P, d_j^P) in time, such that $|t_i^A - t_j^P| \leq \tau$ for a threshold τ defined as the transmission time of an MTU-sized packet in the bottleneck link. If there exists no such sample (t_j^P, d_j^P), we discard (t_i^A, d_i^A). Otherwise, we add the delay tuple (d_i^A, d_j^P) to $\mathcal{D}$ and continue with the next sample (t_{i+1}^A, d_{i+1}^A). After pairing, $\mathcal{D}$ will consist of sample pairs that form $\mathcal{D}(A)$ and $\mathcal{D}(P)$. We also discard from $\mathcal{D}$ delay values close to the propagation delay, that is those values that are less than τ time units above the propagation delay. We then subtract the propagation delay (computed as the minimum of all delay samples for that flow) from each delay sample in

$\mathcal{D}$. Thus, our statistical analysis focuses on queueing delays, not absolute end-to-end delays. Note that clock skew does not affect this test because the difference in one-way delays of a paired sample, even with delay discrimination, would be small compared to the timescales in which clock skew is significant (many seconds).

The next step is to construct a non-parametric hypothesis test for the null hypothesis that $\mathcal{D}(A)$ and $\mathcal{D}(P)$ come from the same underlying distribution. For two discrete probability distributions X and Y, the KL-divergence of Y from X is defined as:

$$D\left(X\|Y\right) = \sum_i X(i) \log_2 \frac{X(i)}{Y(i)}$$

The KL-test on $\mathcal{D}(A)$ and $\mathcal{D}(P)$ proceeds as follows:

1) Estimate the probability mass functions X^A and X^P from the samples $\mathcal{D}(A)$ and $\mathcal{D}(P)$ defined on the same set of bins. The bin width $w = 2n^{-1/3}I$ is determined using the inter-quartile range I of the joint sample $\mathcal{D}(A) \cup \mathcal{D}(P)$, where n is the length of the joint sample. We merge those bins with their neighbors if the number of measurements from both samples is less than 1%.
2) Calculate the KL-divergence $D\left(X^A\|X^P\right)$.
3) *Bootstrapping:* randomly partition without replacement $\mathcal{D}(A)$ into two samples $\mathcal{S}_i^1$ and $\bar{\mathcal{S}}_i^1$. Estimate their probability mass functions X_i^1 and $\bar{X}_i^1$ using the binning procedure in (1). Calculate the KL-divergence $D\left(X_i^1\|\bar{X}_i^1\right)$. Repeat this a number of times (we use 200) to estimate the distribution of $D\left(X_i^1\|\bar{X}_i^1\right)$; call it $\mathcal{D}\left(X_i^1\|\bar{X}_i^1\right)$.
4) Reject the null hypothesis if $D\left(X^A\|X^P\right)$ is *large* compared to the distribution $\mathcal{D}\left(X_i^1\|\bar{X}_i^1\right)$. More precisely, define the p-value as:

$$p = Prob\left[D\left(X^A\|X^P\right) \leq \mathcal{D}\left(X_i^1\|\bar{X}_i^1\right)\right]$$

and reject the null hypothesis with this p-value if $p < 0.05$.

Inequality of Delay Distributions: We detect delay discrimination using a test that checks whether one of the two distributions is consistently larger than the other (equation 3). Specifically, our test of inequality is as follows. We first require that the KL-test rejects the null hypothesis that $\mathcal{D}(A) \approx \mathcal{D}(P)$; otherwise we say that the distributions are equal and we detect FCFS. Then, we consider several p-percentiles $\mathcal{D}_p(A)$ and $\mathcal{D}_p(P)$ of the distributions $\mathcal{D}(A)$ and $\mathcal{D}(P)$ using their empirical Cumulative Distribution Function (CDF) estimates. We say that $\mathcal{D}(A) \gg \mathcal{D}(P)$ if:

$$\mathcal{D}_p(A) > \mathcal{D}_p(P) \text{ for all } p \in [0.5, 0.95] \tag{5}$$

We choose the above range for p since the lower percentiles may not be affected by queueing delays, and they can be close to zero for both flows. The percentiles p are determined from the empirical CDF of $\mathcal{D}(A)$, and for each such p we use nearest-neighbor interpolation to find $\mathcal{D}_p(P)$. We also give the user a measure of the *delay difference* between the two flows as $\mathcal{D}_{0.75}(A) - \mathcal{D}_{0.75}(P)$.

Note that there could be a case where the ISP prioritizes the A flow over the P flow. We run our inequality test by swapping A and P as inputs to detect if the P delays are higher than the A delays.

A. Distinguishing WFQ from SP

After we have detected a discriminatory scheduler that treats A as low priority and P as high priority, we examine whether that scheduler is SP or a WFQ variant. The intuition behind this method follows. Consider a two-class discriminating link that services high priority and low priority packets. A packet experiences propagation, transmission, and potentially queueing delays at that link. We distinguish between the queueing delay due to a packet that is currently being transmitted from queueing delays due to other backlogged packets; the former is referred to as the *non-preemption delay* and it can affect all packets irrespective of their priority. The basic idea of the proposed method is that an arriving P packet at an SP scheduler *may* experience non-preemption delay if another packet is currently being transmitted, but it will *never* experience queueing delays due to backlogged low priority packets. In the WFQ scheduler, on the other hand, an arriving P packet may also experience queueing delays due to backlogged low priority packets. At the same time however, we need to consider that P packets may experience queueing delays at both schedulers when they are backlogged behind other high priority packets. So, if we had a way to identify those P packets that were not backlogged behind other high priority packets, we expect that their queueing delays would be bounded by the non-preemption delay at the link, while those delays may be much higher in the case of a WFQ scheduler.

In practice, we have no way to know which P packets are backlogged behind high priority packets from other sources. We can identify, however, bursts of P packets sent by DiffProbe. From such bursts, we only consider the first P packet because that packet is more likely to *not* be backlogged behind other high priority packets.

Further, we limit our sample to those P packets that were sent shortly after A packets. The reason is that those P packets are more likely to arrive at the link during a busy period. Otherwise, if a P packet arrives at an idle scheduler, it will experience zero queueing delay independent of what the scheduler is. Such packets would not help us detect the scheduler's type.

After we have identified the subset of P packets as previously described, we examine the variability of their delay distribution. The basic idea is that, with SP scheduling, the selected P packets will have very small delay variability. In the discriminatory link, their queueing delays will be practically zero (at most the non-preemption delay, which is the MTU divided by the capacity of that link). In practice, of course, we need to use a larger threshold because of possible queueing delays at other links. The selection of that threshold is not critical, however, because the corresponding delay variability with a WFQ scheduler will be significantly higher. Figure 2 shows the distribution of one-way delays of the selected P

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE INFOCOM 2010 proceedings
This paper was presented as part of the main Technical Program at IEEE INFOCOM 2010.

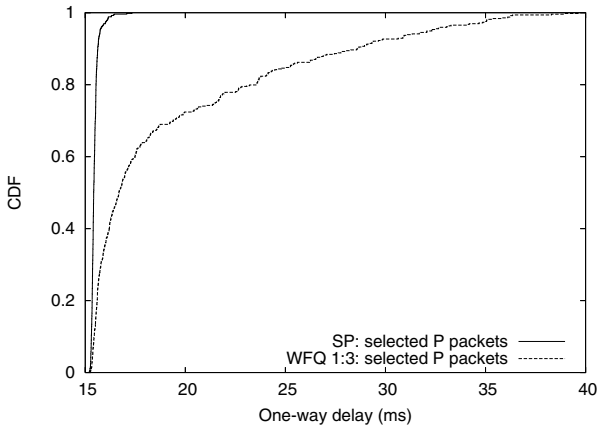

Fig. 2: SP vs. WFQ: distribution of selected P packet delays.

packets for simulated SP and WFQ schedulers. Notice that, with SP, the delay variability of the selected P packets is practically zero; on the other hand, WFQ leads to significant delay variability. In DiffProbe, we measure the delay variation of the selected P packets as the 95th-5th percentile difference. Denote the pth percentile delay of the selected P packets as $\check{\mathcal{D}}_p(P)$. We declare that the scheduler is SP if:

$$\check{\mathcal{D}}_{0.95}(P) - \check{\mathcal{D}}_{0.05}(P) < \kappa$$

The threshold κ is estimated as the MTU packet size divided by the capacity of the access link (given that we do not know the capacity of the discriminatory link). In Section VII we show that the accuracy of this detection method does not depend critically on the value of κ.

V. Loss Discrimination Detection

Discriminatory buffer managers drop packets, already backlogged or arriving, considering the class of those packets. This is different than DropTail or RED, which do not consider the class of the packets they drop. Consider the WRED discriminatory buffer manager in the case of two traffic classes H and L. In practice the ISP would configure the queue thresholds in order to differentiate between traffic classes [1], such that:

$$\hat{q}_{\min}(H) \gg \hat{q}_{\min}(L)$$

Hence, after the average queue length has exceeded the lower threshold for the low priority class, $\bar{q} > \hat{q}_{\min}(L)$, packets of that class will be dropped with a higher probability than packets of the high priority class. In the case of the Drop-Longest-Queue policy, backlogged packets from the longest queue are dropped when needed. Thus, if the low priority class has a longer queue, due to a lower service priority or rate, it will also tend to see a higher loss rate.

We detect *loss discrimination* during the LIP period as follows. We estimate the loss rates of A and P flows (as fraction of packets lost) during the LIP period; denote them as $\ell(A)$ and $\ell(P)$ respectively. We declare that there is loss discrimination when the following condition is satisfied:

$$\ell(A) \gg \ell(P) \tag{6}$$

Conversely, for a non-discriminatory buffer manager, we would have:

$$\ell(A) \approx \ell(P) \tag{7}$$

We describe the specific statistical tests to perform loss rate comparisons next.

Equality of Sampled Loss Rates: We first pre-process the sender-side sequence number timeseries to create a *paired* sample of A and P packets sent almost simultaneously, as done in the pairing procedure for the delay timeseries described in Section IV. Consider the measured loss rates ℓ_A and ℓ_P in the paired samples of the A and P flow, respectively. Let us denote the number of samples sent by the two flows as s_A and s_P respectively.

We use a two-tailed version of the well-known *two-proportion z-test* for equal population proportions of two independent samples [15]. Our null hypothesis is that the A and P flows sample the same loss process, while the alternate hypothesis is that the two flows sample a different loss process (we do not assume that the P flow will necessarily see a lower loss rate in the alternate hypothesis, as it may be that the ISP is discriminating in favor of the A flow). The two-proportion test uses the z-score statistic:

$$z = \frac{\ell_A - \ell_P}{\sqrt{l\,(1-l)\left[\frac{1}{s_A} + \frac{1}{s_P}\right]}}, l = \frac{\ell_A s_A + \ell_P s_P}{s_A + s_P}$$

The z-score has an asymptotic standard Normal ($\mathcal{N}(0,1)$) distribution when null hypothesis is true. The p-value is computed as:

$$p = Prob\left[|z| < \mathcal{N}(0,1)\right]$$

The test rejects null hypothesis with this p-value if $p < 0.05$.

A rule of thumb for Normal approximation of z statistic in the two-proportion test is that s_A and s_P both include at least 10 dropped packets. We diagnose loss discrimination as *unidentifiable* if this is not the case. To help with the identifiability objective, we make the LIP duration sufficiently large (always maintaining $\lambda_A(t) + g(t)\lambda_P(t) \approx C_a\,(1-\epsilon)$), so that we can observe a sufficiently large number of packet losses in this period.

We determine the LIP duration as follows. We observe the loss rate of the A flow during the BLP; call it ℓ_A^{BLP}. Suppose that the rate of A packets sent during BLP is λ_A^{BLP} (packets per unit time). The minimum LIP duration is then chosen so that the expected number of A losses is at least 10. The LIP duration Δ_{LIP} is thus:

$$\Delta_{LIP} \geq \frac{10}{\ell_A^{BLP}\lambda_A^{BLP}}$$

For example, using a G.711 (voice) A flow that sends about 33 packets per second, and with a 1% loss rate during the BLP period, we would determine that the LIP should be at least 30s long.

VI. Implementation and Test Runs

We have implemented *DiffProbe* in a completely automated tool. The current version is about 7,500 lines of C code and it has been tested on Linux platforms so far. In this section we describe the tool, and we also show some test runs at large access ISPs.

DiffProbe consists of two endpoints, the client ($\mathcal{CLI}$), and the server ($\mathcal{SRV}$). $\mathcal{CLI}$ is run by a user connected to the target ISP. DiffProbe operates in two phases. In the first phase, $\mathcal{CLI}$ sends timestamped probing streams to $\mathcal{SRV}$. For each probing structure, $\mathcal{SRV}$ collects one-way delay timeseries of A and P flows. In the second phase, the roles of $\mathcal{CLI}$ and $\mathcal{SRV}$ are reversed.

Capacity Estimation: Before probing, we make a rough estimate of the upstream and downstream capacities at the end-to-end path using packet trains of back-to-back packets over UDP. We use this estimate to decide the LIP probing rate. More precisely, we send K packet trains of length L packets, each of size S. At the receiver, and for each train, we measure the dispersion Δ and estimate the path capacity as: $C_a = \frac{(L-1)S}{\Delta}$. Finally, we take the median of the K trains. C_a is an estimate of the capacity of the *narrow link* between $\mathcal{CLI}$ and $\mathcal{SRV}$. For residential ISPs, this link is most likely the home access link in both upstream and downstream directions. In the current implementation, we set $K = 10$, $L = 50$, and $S = 1450B$, and send the trains over a port which is not likely to be classified low-priority by an ISP.

Probing: Each probing session consists of the BLP and LIP probing periods. After we probe the upstream direction, we repeat that sequence in the downstream direction. Each probing packet of the A flow is replayed according to a pre-recorded application flow trace. We maintain the same port number(s), transport protocol, packet sizes, inter-packet gaps and payload as in the trace file while replaying the A flow. We overwrite the last four bytes of the payload with the sender timestamp for one-way delay measurement.

We create the P flow using the last sent A packet size. The payload is randomized (excluding the sender timestamp) and we use port numbers that are not likely to be classified as low priority. The user can choose between two UDP Skype voice (taken from [6]) and two UDP Vonage voice traces (each 10min. long) to test for discrimination. In our implementation, we use a single probing session. Unless otherwise mentioned, the tool uses the following parameters: LIP and BLP durations 30s, LIP probing rate is estimated with $\epsilon = 0.1$.

A. Test runs

We have run DiffProbe at some large residential ISPs[4]. Note that it is not possible to know the ground truth in such experiments. We can, however, say that there is no discrimination between the A and P flows if no significant delay differences have been *perceived* by the user, or if the KL-test reports a high p-value. All experiments were done in April and July of 2009. The LIP duration in these experiments is 10s while the BLP duration is 5s.

ISP	Upstream	Downstream
ISP-1 (US)	0.01-0.04	0.0-1.0
ISP-2 (Switzerland)	1.0	0.28-1.0
ISP-3 (US)	0.54-1.0	1.0
ISP-4 (US)	0.87-1.0	0.17-1.0
ISP-5 (Belgium)	1.0	-
ISP-6 (Norway)	0.25-0.98	-
ISP-7 (US)	0.82	0.98

TABLE I: Access ISP test runs: p-values across Skype and Vonage tests. Some runs on ISP-1 showed routing differences between the two flows, which explains the low p-values.

Table I shows access ISP locations and the p-values from our KL-test for delay discrimination. We test for discrimination against the four Skype and Vonage traces once for each ISP. The table shows that we do not detect payload and/or port based discrimination in these ISPs (all tests were identified as "detectable")[5]. An exception is the case for ISP-1: 1 out of 4 downstream trials and all upstream trials showed discrimination. Upon visual inspection, we noticed that the two flows follow different paths in the ISP-1 network, while one of the paths introduces higher queueing delays than the other. We plan to include an automated way to detect such routing differences between the A and P flows in the future.

VII. Simulation/Emulation Evaluation

In this section we first evaluate the accuracy of Differential Probing using simulations, and then show some realistic emulation experiments.

We evaluate the accuracy of the discrimination detection methods using NS2 simulations. The simulation topology is as follows. The discriminating link capacity is 100Mbps. The A and P flows are generated from a server and they are sent to a residential client. All servers and residential users have access links of 1Gbps and 10Mbps, respectively. The capacity of the discriminatory link is 100Mbps. We simulate 200 residential clients generating closed-loop ("interactive") TCP sessions by downloading Pareto-sized heavy-tailed content from 200 randomly chosen servers. These well-provisioned servers are connected to the discriminating link through links of different propagation delays. We provision all link buffers according to the bandwidth-delay product. The setup for reverse-direction cross traffic is similar. We perform at least 96 trials for each utilization point of the discriminating link, so that we have an error margin of 2% at 95% confidence assuming a prior proportion of 0.9. A utilization of $U\%$ encompasses all trials in the interval $(U-5, U+5]\%$.

Unless otherwise mentioned, we use the following parameters. Cross-traffic is classified at the access links on the basis of the generating source as low or high priority with probability 0.5. We use a Skype iSAC packet trace as the A flow. We use $\epsilon = 0.1$ to adjust the LIP rate; the LIP and BLP

[4]We repeat that our focus in this paper is on the detection methods - not on a large-scale study.

[5]We were not able to collect data for two downstream cases due to NAT issues.

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE INFOCOM 2010 proceedings
This paper was presented as part of the main Technical Program at IEEE INFOCOM 2010.

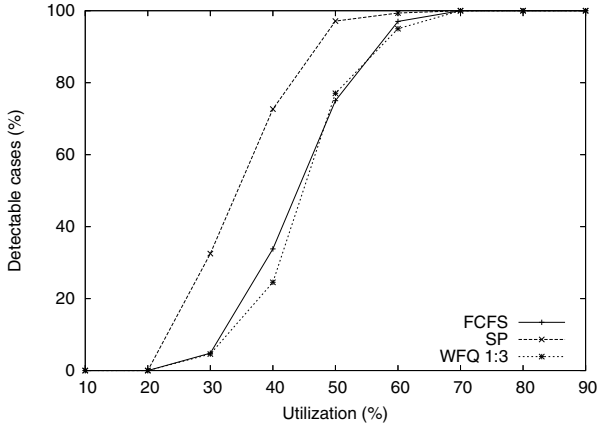

Fig. 3: Fraction of detectable trials: delay discrimination.

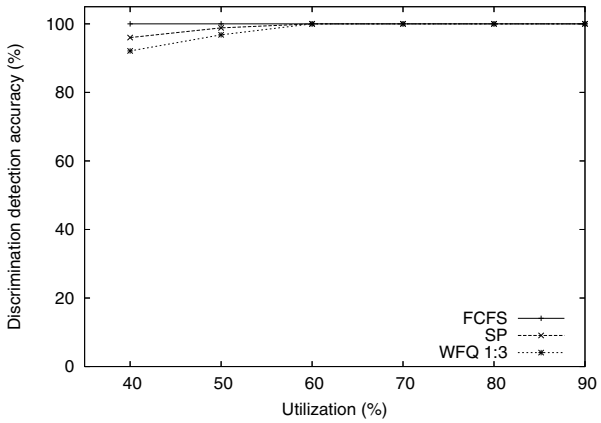

Fig. 4: Delay discrimination detection accuracy.

durations are 30s long. We consider three weight ratios of WFQ, 1:1.5, 1:3 and 1:10. We will see that the first weight ratio is small and performs similar to FCFS-DT, while the third case performs similar to SP; the second ratio is realistic, given that half of cross-traffic utilization comes from high priority flows. We start with an evaluation of delay and loss discrimination detection accuracy.

A. Delay Discrimination

In this subsection, we evaluate detection accuracy of FCFS, SP, and WFQ schedulers.

Detectability: A detection threshold factor (the ratio $\mathcal{D}_{0.9}(P)/\mathcal{D}_{0.5}^{BLP}(P)$ in eq. (1) of 1.3 is sufficient to get detection accuracy higher than 90%. Using this threshold, we show the fraction of detectable trials at each utilization range in Figure 3. Note that at low utilization ($\leq 40\%$) we are not able to detect the majority of trials. This also implies that there is no *user-perceived delay discrimination* at low utilization in the discriminating link, because it is then unlikely that the user traffic will observe any queueing at the discriminating link. We also found that without this detectability condition, we only get 90%+ detection accuracy when the utilization exceeds 50%.

Detection accuracy: Figure 4 shows discrimination and no-discrimination detection accuracy with utilization for FCFS, SP, and WFQ (weight ratio 1:3). We get high detection accuracy at all utilizations of the discriminating link. Note that false positives would correspond to inaccurate detection of FCFS - but we see that there are no such cases.

WFQ weight ratio: The effect of the WFQ weight ratio on delay discrimination detection accuracy is shown in Figure 5. The weight ratio 1:10 performs similar to SP and leads to high detection accuracy, while the ratio 1:1.5 performs close to FCFS (no significant delay discrimination) and hence it leads to low detection accuracy.

WFQ and SP: Figure 6 shows the accuracy of distinguishing the SP and WFQ schedulers for a threshold $\kappa = 0.7$ms. We see that low utilization leads to low accuracy; this is expected, since, although the difference in the delay distributions of A and P flows is large enough to show the presence of discrimination, WFQ and SP service P packets quite similarly. A large WFQ weight ratio (1:10) makes this scheduler similar to SP, and so the detection accuracy is lower than for lower weight ratios. We also found that for more reasonable weight ratios (e.g. 1:3), the detection of SP and WFQ can be done accurately with a wide selection of κ values.

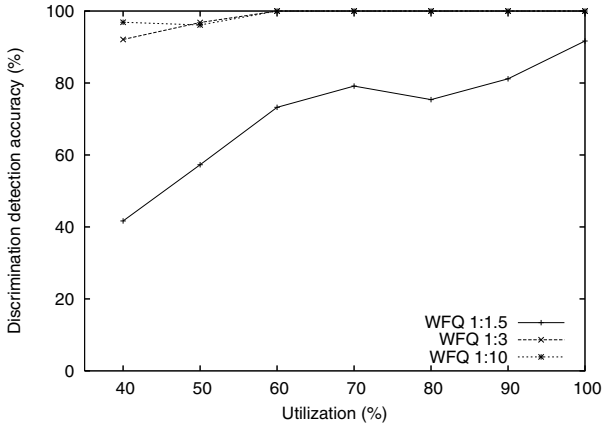

Fig. 5: Effect of WFQ weight ratio on delay discrimination detection accuracy.

B. Loss Discrimination

In this subsection, we evaluate detection accuracy of Drop-Tail, WRED, and drop from longest queue (Drop-Longest-Queue) buffer managers. We use the DT buffer manager in FCFS and SP, while for our WFQ implementation, we use a discriminatory buffer manager that drops packets from the longest queue. Note that we get false positives when the accuracy is less than 100% in the case of DropTail (none in FCFS-DT and less than 2% in SP-DT).

DT and Drop-Longest-Queue: Table II shows the accuracy of DropTail and Drop-Longest-Queue buffer managers for detectable trials. We see a high detection accuracy for both discriminatory and non-discriminatory buffer managers. The table shows Drop-Longest-Queue detection for three different WFQ weight ratios. A WFQ ratio of 1:1.5 is close to DropTail

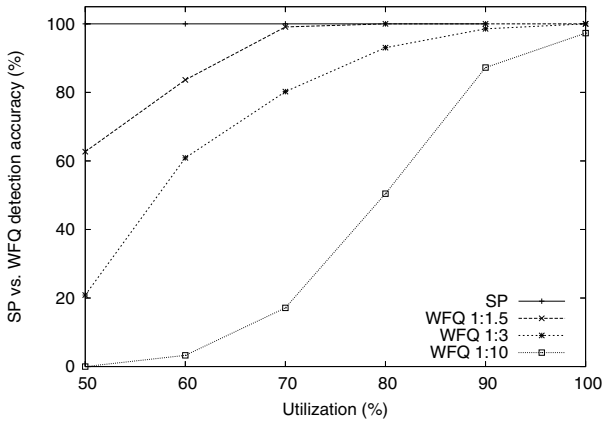

Fig. 6: Distinguish SP from WFQ: effect of utilization and WFQ weight ratio (κ=0.7ms).

Buffer mgr.	DropTail		Drop from longest (WFQ)		
	FCFS	SP	1:1.5	1:3	1:10
Accuracy	100%	98.57%	33.75%	98.75%	100%

TABLE II: Loss discrimination detection accuracy.

in terms of loss discrimination, and yields low accuracy.

WRED accuracy: We choose the following WRED parameters for L and H traffic: $\hat{q}_{\max}(H) = \hat{q}_{\max}(L) = 500$ (buffer size of discriminating link in packets; avg. packet size is 1000B); $\hat{q}_{\min}(L) = \hat{q}_{\max}(L)/2$; $\hat{q}_{\min}(H) = \hat{q}_{\min}(L)\,[1+f]$; $p_H = 0.15$; $p_L = 0.20$. We vary the parameter f, which quantifies the difference between the H and L classes. Figure 7 shows the effect of f on the detection accuracy. At low values of f (≤ 0.3), the detection accuracy is low since the loss discrimination between the two priorities is not significant.

C. Discrimination Emulations

In this subsection, we evaluate the tool in a realistic emulation setup. Our emulated discrimination scenario is as follows. Our testbed is connected to a residential cable ISP in Atlanta, GA (US). The DiffProbe client runs inside the residence, and the server is hosted in the Georgia Tech campus. We emulate the discriminating link on a multihomed Linux router that connects to the cable modem, and serves the client machine connected through a Fast Ethernet interface. Note that the narrow link in this case is between the cable modem and CMTS, which at the time of experiments was a 10Mbps upstream and 17Mbps downstream DOCSIS link. Cross traffic is generated using two Pareto sources (mean gap 100ms and shape 1.5), with a packet size of 600B, over ICMP. Our DiffProbe tests on this ISP (without emulated discrimination) show that the ISP does not discriminate against the A flow. We show results for at least 10 trials for each experiment.

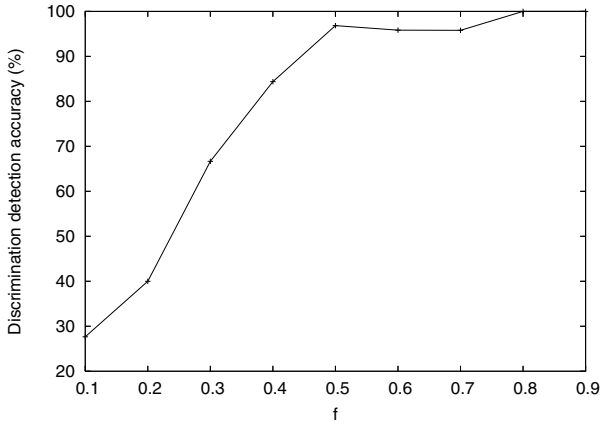

Fig. 7: WRED: effect of f on detection accuracy.

FCFS: We start with the no-discrimination case. We pair A and P samples when they are in 1ms send-time proximity. DiffProbe does not reject the null hypothesis of equal A and P distributions, with p-values in $[0.86, 0.92]$. Note that we reject the null hypothesis if $p < 0.05$.

SP and WRR: We use the the Linux Advanced Routing and Traffic Control framework [3] to implement scheduling and buffer management on a 2.6.22 kernel. We classify traffic using protocol, port numbers and destination IP address. The classifier is built out of *tc* filter rules. One of the cross traffic sources and the P flow are classified as high priority, while the other source and the A flow are classified as low priority.

We implement SP using LARTC's *prio* scheduler. We also limit the service rate to 1Mbps using a token bucket of small depth. The queue size for each class was 50KB. DiffProbe rejects the null hypothesis of equal distributions with $p = 0$ for all trials.

We also implement Weighted Round Robin (WRR) scheduling using LARTC's CBQ scheduler with a weight ratio of 1:3 (link capacity of 1Mbps). The queue size for each class was 50KB. Note that although WRR is implemented in most network devices, it does not account for packet sizes during scheduling and hence it is not as fair in weighted rate allocation as WFQ or DRR. DiffProbe rejects the null hypothesis of equal distributions with $p = 0$ for all trials.

Loss discrimination: The SP and WRR implementations use separate physical queues for each priority, and incoming packets are dropped using the Drop-Longest-Queue policy. For illustration, we consider one WRR 1:3 trial, in which the estimated loss rates were 1.68% for the A flow and 0.15% for the P flow. DiffProbe rejects the null hypothesis of equal loss rates with $p = 0$.

VIII. Related Work

There has been some recent interest in active and passive methods for detecting traffic discrimination. Perhaps the closest in spirit to our work is Zhang et al.'s NetPolice [21], an active probing methodology that replays application traces with limited TTL values to solicit TTL-expired messages from intermediate routers. They compare loss rates with an HTTP flow as a baseline, and show discrimination in backbone ISPs. We are concerned about the validity of this conclusion mostly for two reasons. First, two simultaneous flows (say HTTP and BitTorrent) can observe very different loss rates if they do not

"sample" a lossy queue with packets of the same size and at about the same time. We dealt with this issue using paired statistics, considering packets of the same size that were also sent at about the same time. Second, NetPolice relies on router-generated ICMP responses; the generation of such packets is subject to rate-limiting and vendor-specific lower-priority processing. We believe that further work is needed to validate the conclusions in [21].

Bin Tariq et al. propose a passive detection methodology, NANO [18], which uses throughput observations from many end-hosts to detect discrimination. They use causal inference in the client data, and information about confounding variables to group clients according to performance. BTTest [8] focuses on detecting BitTorrent traffic blocking by ISPs using forged TCP RST packets. It emulates BitTorrent flows, and correlates client and server traces to detect RST messages. The authors show that ISPs mostly block BitTorrent in the upstream direction and classify based on payloads. Siganos et al. [16] use the BitTorrent protocol to measure download throughput across ISPs, and show that the measurements are correlated with performance reports from Akamai.

Lu et al. propose POPI [13] to detect priority based forwarding. They use high-rate active probing to induce losses, and observe loss rates to analyze forwarding. Kuzmanovic et al. [12] use passive monitoring at a single-hop path to observe service rates at different timescales and infer the parameters of a WFQ scheduler. Biczók et al. [5] propose a method to detect discrimination with prior information about the classifier type. They send a single flow and observe performance difference between the sender and receiver sides. Mahajan et al. use ICMP probes and associate performance with geography to measure differences in backbone ISPs in NetDiff [14].

Automated traffic classification often relies on machine learning and statistical techniques. A recent comparison of traffic classification methods is presented in [11]. Commercial classification products include (but not limited to) Sandvine PTS8210 and Cisco NBAR. Traffic Morphing [20] shows that it is possible to avoid certain kinds of classifiers by altering flow characteristics.

IX. Discussion and conclusions

We have presented Differential Probing, a general method for the detection of delay and loss discrimination. This paper focused on the accurate detection of two packet scheduling mechanisms, SP and WFQ, as well as on the detection of loss discrimination, in the case of two classes of service. The simulation and emulation experiments showed that the detection methods are accurate, as long as our probing traffic can create some queueing at the discriminatory link and when the delay and loss differentiation is non-negligible (in other words, discrimination is *perceivable* by user traffic). DiffProbe can also distinguish between SP and WFQ, as long as the weight ratio in the latter is not so high that would make WFQ behave similarly to SP. Our test runs at some major access ISPs show that, at least so far, there is no delay and loss discrimination against Skype and Vonage traffic at those ISPs. A large-scale measurement study of ISP discrimination practices is part of our ongoing work, deploying DiffProbe at the Measurement Lab (M-Lab) [4].

We are also looking at extending the DiffProbe framework to detect more than two classes of service, additional packet scheduling and buffer management mechanisms, and also to quantitatively characterize the parameters of some mechanisms (e.g., to infer the WRED parameters or the WFQ weights). We are also going to generate additional pairs of A and P flows for applications such as BitTorrent and for IPTV applications such as Veoh and Hulu.

The DiffProbe source code and binaries are available at http://www.cc.gatech.edu/~partha/diffprobe/, and we will also provide DiffProbe as a public service hosted at M-Lab.

Acknowledgements

This research was supported by funding from Google and the National Science Foundation (award: 0347374). We would like to thank C. Gkantsidis, T. Karagiannis, E. Athanasopoulos, and E. Markatos for early discussions.

References

[1] *Cisco Systems: Configuring Weighted Random Early Detection*. http://www.cisco.com/en/US/docs/ios/12_2/qos/configuration/guide/qcfwred_ps1835_TSD_Products_Configuration_Guide_Chapter.html.
[2] *Cisco Systems: Congestion Avoidance*. http://www.cisco.com/en/US/docs/ios/12_2/qos/configuration/guide/qcfconav.html.
[3] *Linux Advanced Routing and Traffic Control*. http://lartc.org/.
[4] *Measurement Lab (M-Lab)*. http://www.measurementlab.net/.
[5] G. Biczók, W. Young, and A. Kuzmanovic. Monitoring network bias. In *ACM SIGCOMM 2008 (poster)*.
[6] D. Bonfiglio, M. Mellia, M. Meo, D. Rossi, and P. Tofanelli. Revealing Skype traffic: when randomness plays with you. In *ACM SIGCOMM*, 2007.
[7] R. Castro, M. Coates, G. Liang, R. Nowak, and B. Yu. Network tomography: Recent developments. *Statistical Science*, 2004.
[8] M. Dischinger, A. Mislove, A. Haeberlen, and K. Gummadi. Detecting Bittorrent blocking. In *ACM SIGCOMM IMC*, 2008.
[9] M. Honan. *Inside Net Neutrality: Is your ISP filtering content?* http://www.macworld.com/article/132075/2008/02/netneutrality1.html.
[10] T. Karagiannis, K. Papagiannaki, and M. Faloutsos. BLINC: multilevel traffic classification in the dark. In *ACM SIGCOMM*, 2005.
[11] H. Kim, K. Claffy, M. Fomenkov, D. Barman, M. Faloutsos, and K. Lee. Internet traffic classification demystified: myths, caveats, and the best practices. In *ACM CoNEXT*, 2008.
[12] A. Kuzmanovic and E. Knightly. Measuring service in multi-class networks. In *IEEE INFOCOM*, 2001.
[13] G. Lu, Y. Chen, S. Birrer, F. Bustamante, C. Cheung, and X. Li. End-to-end inference of router packet forwarding priority. In *IEEE INFOCOM*, 2007.
[14] R. Mahajan, M. Zhang, L. Poole, and V. Pai. Uncovering performance differences among backbone ISPs with Netdiff. In *USENIX NSDI 2008*.
[15] D. Sheskin. *Handbook of Parametric and Nonparametric Statistical Procedures*. CRC Press, 2004.
[16] G. Siganos, J. Pujol, and P. Rodriguez. Monitoring the Bittorrent monitors: A bird's eye view. In *PAM 2009*. Springer.
[17] M. Tariq, A. Dhamdhere, C. Dovrolis, and M. Ammar. Poisson versus periodic path probing (or, does PASTA matter?). In *ACM SIGCOMM IMC*, 2005.
[18] M. B. Tariq, M. Motiwala, N. Feamster, and M. Ammar. Detecting network neutrality violations with causal inference. In *ACM CoNEXT*, 2009.
[19] G. Varghese. *Network Algorithmics: an interdisciplinary approach to designing fast networked devices*. Morgan Kaufmann, 2005.
[20] C. Wright, S. Coull, and F. Monrose. Traffic Morphing: An efficient defense against statistical traffic analysis. In *IEEE NDSS*, 2009.
[21] Y. Zhang, Z. M. Mao, and M. Zhang. Detecting traffic differentiation in backbone ISPs with NetPolice. In *ACM SIGCOMM IMC*, 2009.