# Detecting Network Neutrality Violations with Causal Inference

Mukarram Bin Tariq, Murtaza Motiwala, Nick Feamster, Mostafa Ammar
School of Computer Science, College of Computing.
Georgia Institute of Technology, Atlanta, GA
{mtariq, murtaza, feamster, ammar}@cc.gatech.edu

## ABSTRACT

We present NANO, a system that detects when ISPs apply policies that discriminate against specific classes of applications, users, or destinations. Existing systems for detecting discrimination are typically specific to an application or to a particular discrimination mechanism and rely on active measurement tests. Unfortunately, ISPs can change discrimination policies and mechanisms, and they can evade these tests by giving probe traffic higher priority. NANO detects ISP discrimination by passively collecting performance data from clients. To distinguish discrimination from other causes of degradation (*e.g.*, overload, misconfiguration, failure), NANO establishes a causal relationship between an ISP and observed performance by adjusting for confounding factors. NANO agents deployed at participating clients across the Internet collect performance data for selected services and report this information to centralized servers, which analyze the measurements to establish causal relationship between an ISP and performance degradations. We have implemented NANO and deployed clients in a controlled environment on Emulab. We run a combination of controlled experiments on Emulab and wide-area experiments on PlanetLab that show that NANO can determine the extent and criteria for discrimination for a variety of discrimination policies and applications.

**Categories and Subject Descriptors:.** C.2.3 [Computer Communication Networks]: Network Operations, Network Management

**General Terms:.** Management, Measurement

**Keywords:.** Network Neutrality, Causal Inference

## 1. INTRODUCTION

*Network neutrality* states that ISPs remain neutral to how they forward user traffic, irrespective of content, application, or sender [9]; ISPs may *discriminate* against certain subsets of users or services. Rather than taking a stance in this debate, we aim to make the policies of Internet service providers more *transparent* to end users, so that they can detect when ISPs degrade performance or connectivity for some subset of users or applications.


*CoNEXT'09,* December 1–4, 2009, Rome, Italy.


Because discrimination can take many forms, detecting it is difficult. ISPs have been interfering with TCP connections for BitTorrent and other peer-to-peer applications [8]; recently, British Telecom throttled video content [4] after previously demanding compensation from content providers such as BBC for increased traffic due to their content [3]; Cox Communications also said that it planned to begin throttling peer-to-peer traffic [1]. Other types of discrimination may include blocking specific ports, shaping traffic for specific services, or enforcing traffic quotas.

Existing mechanisms for detecting ISP discrimination actively probe ISPs to test for specific cases: Glasnost detects spurious TCP reset packets of BitTorrent connections [8], Beverly *et al.* study port-blocking [24], and NVLens detects prioritization by observing the type-of-service field in ICMP time-exceeded messages [29]. These tools detect specific classes of discrimination, but they have several drawbacks. First, they are *specific* to either the application (*e.g.*, BitTorrent) or the mechanism that the ISP is using to discriminate (*e.g.*, resetting TCP connections, or setting TOS bits). Second, they rely primarily on *active probes*, which are typically detectable, making it possible for an ISP to either block or prioritize them. Because discrimination may vary depending on the application or the mechanism, and ISPs can evade detection mechanisms that rely on active probes, users need detection tools that rely primarily on observations of *in situ* network traffic.

We present the design, implementation, and controlled evaluation of Network Access Neutrality Observatory (NANO), a system that infers the extent to which an ISP's policy causes performance degradations for a particular service. Instead of trying to determine whether an ISP is discriminating using a particular mechanism, NANO tries to infer whether there are differences in performance achieved through a particular ISP when compared to other ISP(s) for a given service. NANO tries to establish a causal relationship between an ISP's policy and the *observed* degradation of performance for a service using only passively collected data. Because NANO directly uses the observed performance of the service, it is difficult for ISPs to evade NANO inference, while at the same time discriminate against a service to degrade its performance. NANO's techniques apply to general performance metrics and can thus apply to many services and applications. For example, throughput can be used to characterize the performance for both Web traffic (including pages, embedded content, video, etc.) and non-Web traffic (*e.g.*, FTP, BitTorrent). Similarly, jitter and loss rate can characterize the performance of many real-time services, such as interactive voice, video, or gaming traffic.

NANO's design draws inspiration from statistical epidemiology: Just as epidemiologists seek to determine whether a particular drug might be responsible for the improved health of a patient, we seek to determine whether a particular ISP affects performance degradation. The challenge in establishing causality is that many *confound-*

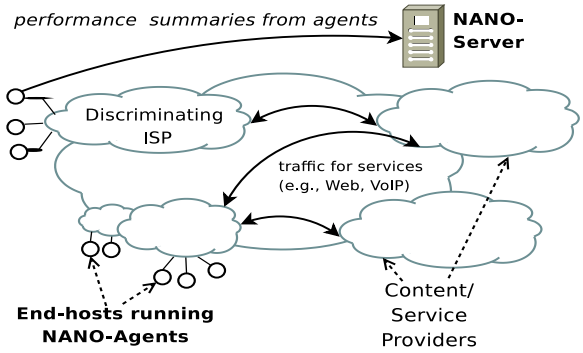



**Figure 1.** NANO Architecture.

*ing* factors may be the underlying cause for the observed outcome. Many factors other than the ISP may affect the performance of a particular service or application. For example, a service may be slow (*e.g.*, due to overload at a particular time of day). A service might be poorly located relative to the customers of the ISP. Similarly, a service may be fundamentally unsuitable for a particular network (*e.g.*, Internet connectivity is not suitable for VoIP applications in many parts of the world). Similarly, the performance might depend on software or hardware, or other network peculiarities.

NANO identifies when service performance differs across ISPs but confounding factors are equal. A big challenge in designing NANO is to identify the confounding factors and create an environment where all confounding factors are equal or independent of the ISP or service performance. Although these goals are difficult to achieve, NANO can infer causal relationships by adjusting for confounding variables on passively observed data. Applying this approach has two main requirements: (1) enumerating the confounding factors and collecting data for the possible values of these variables, and (2) establishing a "baseline" level of service performance for a given set of values for the confounding variables that serves as a point of comparison. NANO's client-side software, NANO-Agent, collects and reports performance data to NANO-Servers regarding their traffic, as well as various meta-data (*e.g.*, the CPU load on the machine at the time, the operating system, the type of connection, etc.) as shown in Figure 1. NANO then analyzes this performance data to quantify the causal relationship between an ISP's policy and the observed service degradation.

We have implemented the NANO-Agent and Server and made the NANO-Agent available for download [20]. We have evaluated NANO in a controlled environment; we emulate access network ISPs on Emulab, where clients perform HTTP and BitTorrent downloads from hundreds of PlanetLab nodes across the Internet; some ISPs in our setup discriminate while others remain neutral. We demonstrate that, even when the distribution of performance from the discriminating ISPs may look similar to the distribution of performance from the neutral ISPs, NANO detects discrimination, estimates the total causal effect on the performance of the services, and determines the discrimination criteria. Our goal in this paper is to describe the NANO techniques and describe the implementation and controlled evaluation as a proof-of-concept. We do not yet have a sufficient deployment to infer ISP discrimination in real networks, but the NANO project Web site [20] provides the participating clients with other useful performance statistics. With a more extensive deployment, we hope to ultimately report on general discrimination practices across ISPs.

This paper is organized as follows. Section 2 defines and motivates the problem, provides definitions, and articulates the challenges. Section 3 provides background on causal inference and formulates ISP discrimination detection as a statistical causal inference problem. Section 4 describes the design of NANO and Section 5 describes the implementation. Section 5.2 evaluates the accuracy, sensitivity, and scalability of NANO. Section 6 lists various open issues with NANO. Section 7 discusses related work, and Section 8 concludes.

## 2. PROBLEM AND MOTIVATION

**Problem statement.** We aim to detect whether an ISP causes performance degradation for a service when compared to performance for the same service through other ISPs.

**Definitions.** A *service* is an "atomic unit" of discrimination (*e.g.*, a group of users or a network based application). *Discrimination* is an ISP policy to treat traffic for some subset of services differently such that it causes degradation in performance for the service. Metrics for performance may be service-specific. We say that an ISP *causes* degradation in performance for some service (*i.e.*, that it discriminates against some service) if we can establish a causal relation between the ISP and the observed degradation. For example, an ISP may discriminate traffic for a particular application (*e.g.*, Web search), traffic for a particular domain, or traffic carrying particular type of media, such as video or audio, such that performance for these services degrades.

**Challenges.** Detecting discrimination is challenging for the following reasons.

1. *The mechanism for discrimination may not be known.* Although existing tools for detecting network neutrality all assume that either the mechanism for discriminating against traffic or the application being discriminated against is known, this is generally not the case. Users often do not even know whether an ISP might be discriminating certain subsets of traffic. These users need methods for detecting discrimination that do not rely on testing for specific discrimination types.
2. *The baseline performance for a service in an ISP is not known.* Users do not know what the "baseline" performance is for a given service through their ISP, so detecting when the performance is degraded, potentially as a result of discrimination is difficult. We propose one approach to establish baseline performance in Section 4.2.
3. *Many factors can cause performance degradation.* Any tool that detects discrimination must identify the ISP—as opposed to any other possible factor—as the underlying cause of discrimination. An industry source recently expressed skepticism about the effectiveness of existing tools: "However, one ISP industry source, who asked not to be identified, questioned whether the tools would accurately point to the cause of broadband problems. 'Spyware or malware on computers can affect browser performance, and problems with the wider Internet can cause slowdowns, the source said.'" [11]. It is precisely this problem—adjusting for such external causes—that we tackle.

We believe that NANO is the first technique that can isolate such discrimination from other confounding factors, without *a priori* knowledge of an ISP's discrimination policy. NANO relies on knowledge of confounding variables, but these are not difficult to enumerate using domain knowledge. NANO uses monitoring agents to collect values for the confounding variables.

# 3. BACKGROUND

In this section, we give a brief overview of causal inference and how it can be used to quantify causal effect.

## 3.1 Causal Effect and Confounders

Statistical causal inference is applied in many observational and experimental studies [12, 22]. We review causal inference and describe how it relates to inferring ISP discrimination.

**Causal effect.** *"X causes Y"* means that a change in the value of $X$ (the "treatment variable") should cause a change in value of $Y$ (the "outcome variable"). Accessing a particular service through an ISP is our treatment variable ($X$), and the observed performance of a service ($Y$) is our outcome variable: $X = 1$ when a user accesses a service through some ISP, and $X = 0$ when the user does not access the same service through that same ISP (*e.g.*, it accesses it through an alternate ISP). The value of the outcome variable $Y$ depends on the service; it might be a direct measure of quality, such as Mean Opinion Score (MOS) for VoIP applications, or another variable that is highly indicative of the application performance, such as, throughput, loss-rate, or jitter.

Causal inference estimates the effect of the treatment variable (the ISP) on the outcome variable (the service performance). Let's define a *ground-truth* value for the outcome random variable as $G_X$, so that $G_1$ is the outcome value for a client when $X = 1$, and $G_0$ is the outcome value when $X = 0$. We refer to the outcome when not using the ISP ($X = 0$) as the *baseline*.

We can quantify the *average causal effect* of using an ISP as the expected difference of the ground truth of service performance between using the ISP and the baseline.

$$\theta = \mathbb{E}(G_1) - \mathbb{E}(G_0) \tag{1}$$

To compute the causal effect, $\theta$, we must observe the outcome both under the treatment and without the treatment.

**Association vs. causal effect.** In a typical *in situ* dataset, such as network traffic, each sample presents only the value of the outcome variable either under the treatment, lacking the treatment, but not both. Because the ground-truth values $(G_0, G_1)$ are not simultaneously observable, we cannot estimate the true causal effect from an *in situ* dataset alone (Eq. 1). Instead, we can compute association. Let's define *association* as the difference of performance with or without the ISP:

$$\alpha = \mathbb{E}(Y|X = 1) - \mathbb{E}(Y|X = 0) \tag{2}$$

Of course, association is not a sufficient metric for causal effect, and in general $\alpha \neq \theta$, mainly due to the effects of confounding variables.

**Confounding variables.** A *confounding variable* (or simply "confounder") is one that correlates *both* with the treatment variable in question (*i.e.*, the ISP) and the outcome variable (*i.e.*, the performance). Confounding variables make it difficult to assess the true extent of causal relationship between the treatment and outcome variable because if we observe a correlation between treatment and the outcome variables, we cannot be certain whether that correlation is because of a causal relationship between the treatment and outcome, or whether it is because of the confounding variable. For example, if the location of the client correlates with the ISP and the service, then we cannot blindly attribute any observed association between ISP and the service performance to the causal relationship between the two, because the difference in performance might be due to the differences in location of the ISP or services. The next section describes techniques for computing causal effect in the presence of confounding variables.

## 3.2 Dealing with Confounders

This section presents two techniques for estimating causal effect: random treatment and stratification. Stratification essentially emulates random treatment, albeit passively; we explain why stratification is more appropriate for network data.

**Strawman: Random treatment.** If we assign clients to the treatment randomly, then under certain conditions, association is an unbiased estimator of causal effect.[1] All other variables that have an association with the outcome variable must remain fixed when the treatment changes. With *in situ* network traffic data, we must find a way to change the treatment variable—the user's ISP—while keeping other factors fixed. While random treatment is ideal for lab experiments, it is difficult to emulate on the Internet because it is difficult to make a user switch to an arbitrary ISP.

**NANO approach: Stratification.** NANO uses a technique called stratification to adjust for confounding variables [12]; stratification places measurements into strata so that all samples in each stratum have "similar" values for the confounding variables, creating conditions that resemble random treatment: treatment and outcome variables become independent of confounding variables. Informally, one might think of this as placing all measurements where everything that could possibly be attributed to performance is equal, except for the ISP.

Stratification requires enumerating the confounding variables. Unfortunately, there is no automated way to enumerate all the variables that might affect an outcome variable; the problem is similar to any machine learning problem where accurate prediction depends on enumerating all of the features. Instead, we must rely on domain knowledge to enumerate the confounding variables and heuristics to identify when some confounding variables may be missing. Although our evaluation (Section 5.4.3) shows that we have enumerated these confounders in our controlled environment, as clients become more diverse in a wide-area deployment, we may need to revisit and expand this list.

**Formalization.** Let the *causal effect*, $\theta_{ij}$, quantify how the performance of a service $j$, denoted by $Y_j$, changes when it is accessed through ISP $i$, versus when it is not accessed through ISP $i$. Let $Z$ denote the set of confounding variables, and let $s$ be a stratum. Then causal effect within a stratum, $\theta_{ij}(s)$ is:

$$\theta_{ij}(s;x) = \mathbb{E}(Y_j|X_i = x, Z \in \mathbb{B}(s)) \tag{3}$$

$$\theta_{ij}(s) = \theta_{ij}(s;1) - \theta_{ij}(s;0) \tag{4}$$

where $\mathbb{B}(s)$ is the range of values of confounding variables in the stratum $s$. The variable $\theta_{ij}(s;0)$ in Equation 4 represents the *baseline* service performance. We can estimate the variance of $\theta_{ij}(s)$ (estimated as the variance of mean differences), as:

$$\sigma^2(\theta_{ij}(s)) = \frac{(n_{s1}-1)\sigma^2_{ijs1} + (n_{s0}-1)\sigma^2_{ijs0}}{n_{s0}+n_{s1}-2}\left(\frac{1}{n_{s0}} + \frac{1}{n_{s1}}\right) \tag{5}$$

where $\sigma^2_{ijsx}$ is shorthand for $\sigma^2(\theta_{ij}(s;x))$ and $n_{s1}$ and $n_{s0}$ are the number of performance measurements that we observe with and

[1]This property holds because when $X$ is independent of $G_X$, then $\mathbb{E}(G_X) = \mathbb{E}(G_X|X) = \mathbb{E}(Y|X)$; see [28, pp. 254–255] for a proof.

without the ISP, respectively. Assuming that $\theta_{ij}(s)$ follows a Normal distribution, the (1-$\alpha$) confidence interval is:

$$\hat{\theta}_{ij}(s) \pm z_\alpha \sigma(\theta_{ij}(s)) \quad (6)$$

Equations 4 and 6 estimate the causal effect for *each* stratum $s$. We would like to summarize the overall causal relationship between ISP $i$ and a service $j$ across *all* the strata. Unfortunately, all strata may not be equally likely and causal effect might vary across strata. The techniques used in epidemiology for this purpose (*e.g.*, the Mantel-Haenszel statistic [12]) only work for binary outcome variables and also require estimates of the prevalence of each outcome, so they are not appropriate in this context. Instead, we simply average across the strata as:

$$\hat{\theta}_{ij} = |s|^{-1} \sum_s \theta_{ij}(s) \quad (7)$$

If we assume that causal effect is independent across strata, we can estimate the variance and the corresponding (1-$\alpha$) confidence intervals as:

$$\sigma^2(\theta_{ij}) = \quad |s|^{-2} \textstyle\sum_s \sigma^2(\theta_{ij}(s)) \quad (8)$$

$$(1-\alpha) \text{ CI for } \theta_{ij} \quad = \hat{\theta}_{ij} \pm z_\alpha \sigma(\theta_{ij}) \quad (9)$$

The units for causal effect are same as for service performance, so the values are straightforward to interpret: essentially, for metrics such as, throughput, a negative value indicates performance degradation, and for metrics such as loss or jitter, a positive value indicates performance degradation.

# 4. NANO APPROACH

This section enumerates the confounding variables required for this inference and explains how NANO performs causal inference.

## 4.1 Confounders for Network Performance

In this section, we enumerate three categories of confounding variables and explain why they might correlate with both the treatment variable (the ISP) *and* the outcome (the performance). NANO-Agents collect statistics for variables that help determine the level of various confounding factors. Table 1 shows these variables.

**Client-based confounders.** The application that a client uses for a service can affect service performance. Certain Web sites may be optimized for a particular Web browser, or differences in Web browsers may affect performance; for example, Opera, Firefox, and Internet Explorer use a different number of simultaneous TCP connections, and only Opera uses HTTP pipelining by default. Other features that may affect performance include the operating system and the configuration of the client's computer and local network, as well as a client's service contract. These variables clearly affect performance, but they may also correlate with the ISP. For example, we expect that Microsoft Windows may be more popular among home users that other operating systems, while Unix variants may be more common in academic environments. Similarly, certain browsers may be more popular among certain demographics and localities. If the ISPs cater different demographic groups, then these variables may correlate with the ISP brand.

**Network-based confounders.** Various properties of the Internet path, such as the location of the client or ISP relative to the location of the servers, can degrade service. A path segment to a particular service provider might not be sufficiently provisioned. If we wish to disregard these effects, we must adjust for the path properties. We do not treat congestion as a confounder because we believe that there is value in determining whether the performance degradation related to congestion are specific to an ISP. NANO cannot differentiate between network-wide congestion that affects all the services and a congestion that is targeted at a particular service because NANO does not compare performance across services.

| **Client-Based Features** |
|---|
| IP address |
| Process for each flow |
| CPU usage |
| Memory usage |
| Operating System |
| Uptime |
| Network devices and counters |
| **Network-Based Features** |
| *From network traffic* |
| IP address, port number, and protocol |
| Timestamps for first, last packet for each flow |
| Cumulative, periodic bytes, packets per flow |
| SYN/SYN-ACK SYN-ACK/ACK RTTs |
| TCP state for active connections |
| TCP retransmits |
| TCP duplicate ACKs |
| *Provided by user* |
| ISP contract level/SLA |
| Geographic Location |
| User Location (home/work/etc.) |
| Type of link (wireless/Ethernet) |
| Whether using a home router |
| Type of router |

**Table 1.** Data collected by NANO.

**Time-based confounders.** Service performance varies widely with time of day, due to changes in utilization, and ISPs may also experience different performance at different times of day.

## 4.2 Establishing Causal Effect

NANO uses the five steps in Figure 2 to estimate causal effect of an ISP for service degradation. First, NANO stratifies the service performance data reported from the NANO-Agents. Next, NANO estimates the extent of possible causal effect of ISP on performance within each stratum by comparing the performance within the stratum with the baseline performance from other ISPs. Then NANO summarizes the causal effect by aggregating on all the strata and finally tests whether the aggregate causal effect is statistically significant. Optionally, NANO can also infer the criteria that the ISP is using for discrimination.

**Step 1: Stratifying the data.** To stratify the data, NANO creates "bins" (*i.e.*, ranges of values) for each of the confounding variables. The main criteria for stratification is to create bins on the value of the confounding variable such that the value of the treatment and outcome variables can be considered *independent* of the confounding variable for the span of the bin. As a result, the bin size depends on the nature of the confounding variable.

For discrete variables, creating bins is simple: we create strata such that there is a bin for every unique value of the variable. For example, all the clients using a particular version of the browser

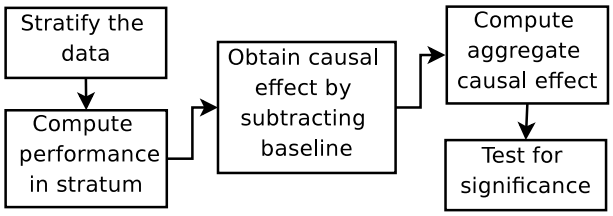



**Figure 2.** Steps for computing causal effect.

are in one stratum. For continuous variables, the bins must be sufficiently small such that the variable has essentially a constant value within the stratum. For example, many network performance metrics are correlated with the time-of-day variable due to diurnal cycles, but these metrics are typically independent of the time-of-day across days for a given hour [27], or time-of-day and day-of-the-week combined [15]. This characteristic makes one-hour bin a reasonable choice for time-of-day variable. If the performance cycles repeat only on weekly basis, then we may additionally stratify on a variable representing the day-of-the-week.

We apply standard correlation tests to determine whether the treatment variable and the outcome variable are independent of the confounding variable within a stratum. To reduce the number of strata and the overall number of samples needed to establish confidence intervals, we combine adjacent strata if the distribution of the outcome variable conditioned on the treatment variable is identical in each stratum.

**Step 2: Computing causal effect.** We compute causal effect by plugging in the performance estimates for each stratum in Eq. 4. One challenge with using Eq. 4 is establishing the baseline performance (term $\theta_{i,j}(s;0)$ in Eq. 4). Intuitively, this value reflects the performance a user would see *without* treatment (*i.e.*, not using an ISP $i$ for some service). Simply using a different ISP is insufficient if that ISP is *also* discriminating against service $j$.

To address this problem, NANO computes the baseline, $\theta_{i,j}(s;0)$, as the average service performance when not using ISP $i$; *i.e.*, the average over all other ISPs that have users in that stratum:

$$\theta_{i,j}(s;0) = \sum_{k \neq i}^{n_p} \theta_{k,j}(s;1)/(n_p - 1)$$

where $n_p > 2$ is the number of ISPs for which we have clients in stratum $s$. One limitation of this definition is that if many ISPs are discriminating against a service, the baseline performance will reflect discrimination (*i.e.*, discrimination becomes the norm). In such cases, we could derive the baseline in other ways, such as by comparing against the best performance instead of the average, or by using a performance model of the service from laboratory experiments or mathematical analysis.

**Step 3: Inferring the discrimination criteria.** Although NANO's causal inference makes no assumptions about discrimination criteria used by the ISP, NANO can determine discrimination criteria as a side effect of stratification. NANO infers the discrimination criteria that an ISP uses by using simple decision-tree based classification methods. For each stratum and service where NANO detects discrimination, NANO assigns a negative label, and for each stratum and service where it does not detect discrimination, it assigns a positive label. NANO then uses the values of the confounding variables and the service identifier as the feature set and the discrimination label as the target variable, and a decision-tree algorithm to train the classifier. The rules that the decision tree generates indicate the discrimination criteria that the ISP uses because the rules indicate the boundaries of maximum information distance between discrimination and normal behavior.

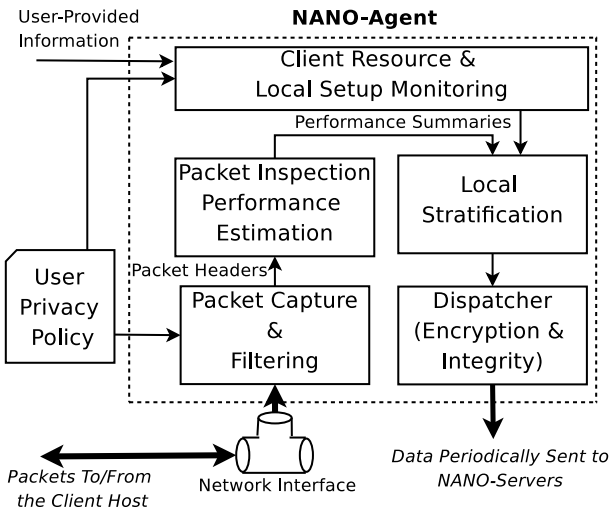



**Figure 3.** Design of the NANO-Agent, which runs on a client machine.

# 5. DESIGN AND IMPLEMENTATION

This section describes the implementation of NANO, which has two parts: *NANO-Agents* and a *NANO-Server*. NANO-Agents reside on participating clients and continuously monitor and collect data for traffic from that client host to various destinations, and send aggregate traffic statistics to the centralized NANO-Server. The NANO-Server collects these statistics and performs the inference described in Section 3 to quantify the ISP's effect on performance. The primary source of data for NANO are client-side agents installed on computers of voluntarily participating clients (NANO-Agents). We describe these components in detail below.

## 5.1 Agents

We have developed NANO-Agent as a packet-level sniffer, running at clients, that can access fine-grained information from the client machines including the various system resource utilization and client machine setup information.

Figure 3 shows the architecture for the NANO-Agent. After a packet is captured from the network interface (via pcap), the modules shown in Figure 3 strip the protocol headers and compute performance summaries for the traffic. The agent only computes performance summaries for traffic that is allowed by a user-specified privacy policy. The agent periodically dispatches the summaries to NANO-Server. The NANO-Agent collects three types of features for the confounding factors, corresponding to the three classes of confounding variables described in Section 4.1 and Table 1.

**Data collection.** The NANO-Agent analyzes the network and transport protocol headers to identify the service and assess performance. For the experiments that we describe in Section 5.2, we focus on features from the TCP/IP headers of the packets and associated timing information. We try to estimate the throughput and latency that the packets experience for a TCP flow. To estimate the throughput, the agent continuously measures the bytes uploaded

and downloaded in a specified (configurable) interval. To estimate the latency on a TCP flow, the agent measures the latency between the SYN and SYN-ACK packets for the flows that originate at the client, and the latency between the SYN-ACK and the subsequent ACK, for the incoming connections that the client receives. The agent keeps track of connection duration and events such as losses, timeouts (by tracking the TCP duplicate acknowledgments) or unexpected connection terminations, such as, with a TCP reset flag. Our implementation also infers the application associated with the active flows from the `proc` file system. In addition, the agent also continuously monitors the average CPU and memory utilization on the client host.

In addition to runtime statistics, the agent also collects information about the client setup which includes the client host specification, including the platform, CPU and memory specification, and the active operating system on the host. NANO relies on the user to provide the agent with information that it cannot infer. This information includes the type of network interface (wired or wireless), the type of contract the client has with the ISP, and the user's location (city and country). In the future, we plan to use an IP-to-geographic location database in lieu of client-provided information. Table 1 describes the variables that the NANO-Agent collects.

**Protecting user privacy.** Information that NANO-Agents collect may contain sensitive information (*e.g.*, destinations a client has visited or the amount of content it has downloaded). Protecting user privacy is paramount. Unfortunately, standard anonymization techniques, such as anonymizing IP addresses and various other features (*e.g.*, browser type, operating system) obfuscate the very features used to stratify the data, thus preventing causal inference. We must apply techniques that mask client identities but preserve the features that NANO uses to stratify the data. NANO employs three measures that protect user's private data from eavesdropping, allows user's some control over the type of traffic that NANO-Agents monitor, and reduce the granularity of information that is sent to the NANO-Server. These measures mitigate privacy concerns that arise due to passive monitoring, although these concerns are not completely alleviated.

*1. Local stratification*. The NANO-Agent performs local stratification to reduce the granularity of information that the client sends to the server. The NANO-Agent can optionally report only the /24 prefix of the source and destination IP addresses, allowing some obfuscation of client identity, and similarly round off the round-trip time measurements to (configurable) nearby values.

*2. User-specified filters.* The NANO-Agent allows the users to specify a set of fully qualified or wild-card domain names, IP addresses, or port numbers (or ranges) that they wish to have excluded from the monitoring. NANO-Agent does not collect information about the flows matching the filters. We are working on allowing users to specify filters based on application names as well, so that the user can, for example, specify whether to prevent BitTorrent traffic from being monitored. Users can also temporarily disable NANO-Agent; during this period, NANO-Agent does not monitor any traffic but continues to send beacon packets to the server indicating that it is alive but not monitoring.

*3. Secure communication*. NANO-Agents transmit data to the servers over SSL. The SSL certificate is included with the NANO-Agent distribution.

**Implementation.** We have implemented the NANO-Agent using C++ for Linux-based systems. The agent promiscuously captures the packets that flow from the client host's network interface using the `pcap` library.

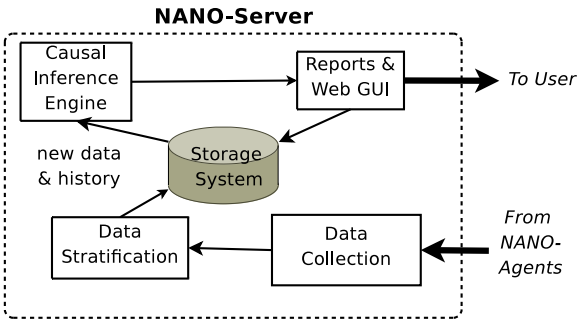



**Figure 4.** NANO-Server design.

The NANO-Agent implements user privacy filters as follows. If the user specifies a source or destination IP address or port ranges, the filtering is straightforward: NANO ignores packets for corresponding flows. For the filters using domain names, the NANO-Agent inspects the DNS traffic at the client and learns the IP addresses for the domain names that match the filters. The NANO-Agent then ignores the traffic with the matching IP addresses for the duration of TTL specified in the DNS. The NANO-Agent keeps track of CNAME records and ignores traffic for IP addresses mapped to canonical names of the domain names in the filters. For example, if the user specifies `mail.google.com` as one of the filters, then the NANO-Agent would also ignore packets with IP addresses for `googlemail.l.google.com`, as it is (sometimes) returned as canonical name for `mail.google.com`.

The NANO-Agent uses protocol buffers [10] to maintain the information that it collects. Protocol buffers offer high speed and compact serialization that allows us to minimize the computational and communication overhead for running the agent at the client. The agent periodically serializes the data that it has collected and sends it to a NANO data collection server. The NANO-Agent implementation is open-source and freely available [20].

## 5.2 Server

The NANO-Server periodically receives information from NANO-Agents running on the clients and performs the causal analysis. Although some of the causal inference logic is currently offline, the NANO-Server provides a Web-based interface [20] where participating users can observe live statistics only about the traffic summaries that their NANO-Agents send to the NANO-Server. We are working on making these statistics more useful, including providing real-time diagnostics information, and network "health" monitoring that can be inferred from the data that the NANO-Agent collects. Additionally, we allow users to selectively delete their data from NANO-Servers.

**Implementation.** We have implemented the server using a combination of C++ to implement the data collection and demarshalling and using a Python and MySQL back-end for analysis and causal inference. Our implementation can compute causal effect over 20,000 strata in about one minute using two threads on a dual 3.2 GHz processor Intel Pentium 4 machine with 4 GB of memory. In a real-world deployment, the number of strata can easily approach in millions; for this, we plan to port the NANO-Server to a Map-Reduce-based implementation [7].

sectionEvaluation

This section presents the results of our evaluation. The experimental setup is as shown in Figure 5 and a summary of experiments that we conduct is presented in Table 2. We study the following four questions in our evaluation:

- *Can NANO detect different types of discrimination?* (Section 5.4.1) To answer this question, we tested NANO's detection algorithms with three different types of discrimination and in the presence of various network and client-side confounding variables.
- *Can NANO determine discrimination criteria?* (Section 5.4.2) Our evaluation shows that NANO can determine the discrimination criteria used by the ISP using a decision-tree based classifier.
- *Can NANO determine when confounders are missing?* (Section 5.4.3) Our experiments show that NANO's heuristic for testing sufficiency of confounders is a reasonable one, although this problem in general is an open question.
- *How does NANO scale with the size of the input data?* (Section 5.4.4) We study how NANO's accuracy is affected by the amount of input data, and how memory and CPU requirements scale with the size of the input.

## 5.3 Experiment Setup

**Tested: PlanetLab and Emulab.** We use PlanetLab [2] nodes as servers. Specifically, we configured a set of geographically distributed PlanetLab nodes with two types of services. First, we configure two Web servers on each of these PlanetLab nodes to represent two different Web services. Second, we have configured the PlanetLab nodes to act as BitTorrent clients.

We use Emulab to create a set of ISPs, each with its set of clients that connect to the ISP using links of configurable characteristics. The ISP provides connectivity to the Internet to its clients. Figure 5 shows this arrangement. The clients in the Emulab environment access these services through emulated ISPs where we can introduce discrimination using Click routers. The clients can be configured with different physical configurations and run different operating systems on them. This setup allows us to control some of the confounding variables on the client side and use various discrimination criteria that might be implemented by an ISP. Each client also runs an instance of NANO-Agent which periodically reports the performance data to a NANO-Server running at Georgia Tech. We did not gather samples from all ISPs for all strata, so we only consider the strata where at least three of the five ISPs in our experiment had 20 samples or more. As a result, the baseline performance for some of the strata might comprise fewer than four ISPs. For the experiment involving discrimination against long flows, 96% of strata met the other criteria; this figure was even higher for the other experiments.

A potential problem with this setup is that if an ISP in the wide-area (outside our Emulab environment) is discriminating against traffic between the Emulab clients and the PlanetLab nodes, then NANO would not be able to detect that, since NANO only has data from the NANO-Agents which all use the ISP outside of Emulab. Indeed, we encountered paths to PlanetLab nodes that were very lossy to begin with. However, because we have no reason to believe that the losses on these paths correlate with the ISPs that we emulate on Emulab, these paths are not a confounder for our experiments, and we can afford to ignore these even if the ISPs on those paths are potentially discriminating.

**Emulating discrimination.** We emulated ISP discrimination by running Click on the Emulab node that acts as the ISP router to connect the ISP clients to the Internet (see Figure 5). We pass the client traffic through the Click router running as a kernel module on the router node. We used a combination of Click elements to perform various forms of discrimination including probabilistically dropping packets on all flows, or flows which exceed a certain length, dropping of TCP acknowledgments, dropping packets for a particular service or destination, and sending TCP RST packets back to the client (similar to the practice by Comcast).

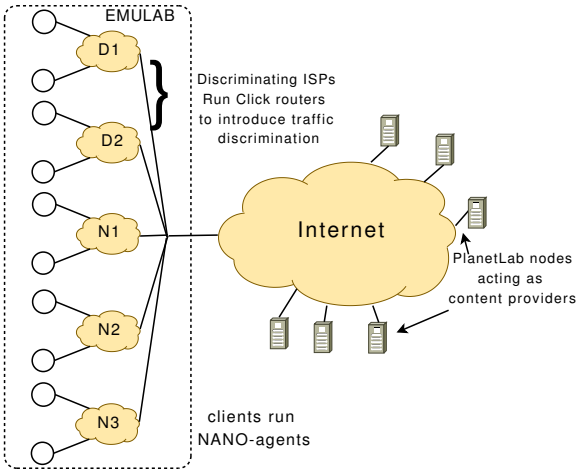



**Figure 5.** Experimental setup on Emulab. Each ISP is an Emulab node with a Click router that performs different types of discrimination, depending on the experiment. Clients run NANO-Agents and report information to a NANO-Server.

We used available Click router elements such as `IPClassifier` to classify packets based on the various IP and TCP fields, `RandomSample` for dropping packets and, `AggregateIPFlows` and a modified version of `AveragePktCounter` to classify flows that exceed a certain length. Running Click router as a kernel module was sufficient to ensure that the Emulab ISP node could sustain a reasonable amount of traffic (maximum of 20 Mbps).

Figure 6 shows an example of discrimination against long flows that we implemented using a Click router. In this case, we configured the router to probabilistically drop TCP packets of flows that exceeded 13,000 packets. For the flow shown in Figure 6, this event occurs at around 10 seconds after the start of the flow, at which point a drop in both throughput (calculated as bytes transferred per ten seconds) and the rate of cumulative packets for the flow occurs.

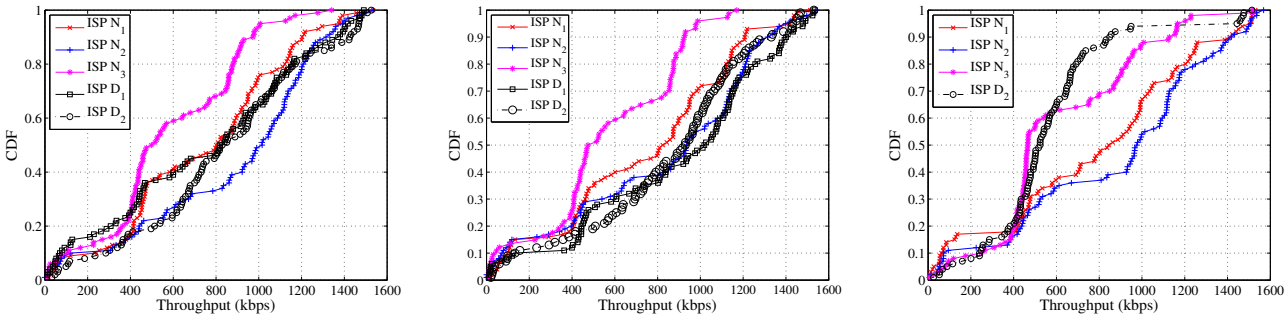
## 5.4 Results

This section presents our experimental results; we answer the questions that we posed at the beginning of Section 5.2.

### *5.4.1 Can NANO detect discrimination?*

We performed three experiments to evaluate whether NANO could detect discrimination. We create five ISPs: two of these discriminate, and we call them discriminating ISPs, ISP $D_1$ and ISP $D_2$. The remaining three ISPs use best-effort service for all the packets on their routers; we refer to these as the neutral ISPs and ISP $N_1$, ISP $N_2$, and ISP $N_3$. Table 2 summarizes these experiments. The first is a simple discrimination against HTTP traffic; the second is a discrimination against long-running flows (in practice, this might be bulk transfers like movies); the third involves discrimination against BitTorrent traffic.

**It is not possible to detect discrimination from distributions alone.** Figure 7 shows the distribution of performance for the




(a) Simple Discrimination. (b) Long Flow Discrimination. (c) BitTorrent Discrimination.

**Figure 7.** Performance distribution for the clients in all the ISPs. Although ISP $D_1$ and ISP $D_2$ are discriminating in each of the three scenarios, it is not possible to correctly identify the ISPs that are discriminating or the extent of discrimination by observing the over the performance distributions.

**Internet Service Providers (ISPs)**: ISPs $N_1$, $N_2$, and $N_3$ are neutral. ISPs $D_1$ and $D_2$ discriminate.

**Experiment 1. Simple Discrimination.** ISPs $D_1$ and $D_2$ discriminate against HTTP traffic for all clients. ISPs $D_1$ and $D_2$ drop 0.1% and 0.3% of the packets respectively. Location of the HTTP server is a confounding variable. ISPs $N_1$, $N_2$, $N_3$, $D_1$, and $D_2$ access the content from the *near* PlanetLab servers with probabilities, 0.4, 0.1, 0.7, 0.6, and 0.9, respectively, and access the *far* PlanetLab servers with the remaining probability.

**Experiment 2. Long Flow Discrimination.** ISPs $D_1$ discriminates against $S_1$, and $D_2$ discriminates the HTTP traffic for $S_2$ for all their clients *if* the flows from $S_1$ or $S_2$ exceeds certain limits. ISPs $D_1$ and $D_2$ drop 0.1% and 0.3% of the packets for flows exceeding 10,000 and 13,000 packets respectively.
Server location is a confounder, as in Experiment 1, with same probabilities for the *near* HTTP servers. All HTTP servers provide both $S_1$ and $S_2$, albeit on different ports.

**Experiment 3. BitTorrent Discrimination.** ISP $D_2$ discriminates the BitTorrent traffic for all its clients if the BitTorrent peer is not in certain subset of PlanetLab nodes. dropping 0.3% of the packets of the flows that are established with the non-preferred peers.

**Table 2.** Summary of experiments.

clients of each ISP in all three experiments. For the first two experiments, we compute the average throughput over the life of the flow and use that as a metric. Note that it is difficult to detect which ISP is discriminating solely based on the performance distribution. For example, in the first two experiments (Figures 7 (a) and (b)), the overall performance distribution for the discriminating ISPs is similar or better than the distribution of performance in the neutral ISPs because the clients in the discriminating ISPs access the *near* servers with higher probability. Because geographically closer servers are more likely to provide higher throughput, the discriminating ISPs still have a larger fraction of sessions with higher throughput. Similarly, the throughput for most BitTorrent clients in $D_2$ is similar to the throughput in the other ISPs when the ISP is discriminating against a subset of destinations.

**NANO detects discriminating ISPs and quantifies the extent of discrimination.** We present the average causal effect on throughput in kbps with a 90% confidence interval (cf. Equation 9) for each ISP, service, and experiment in Table 3. (Due to the number

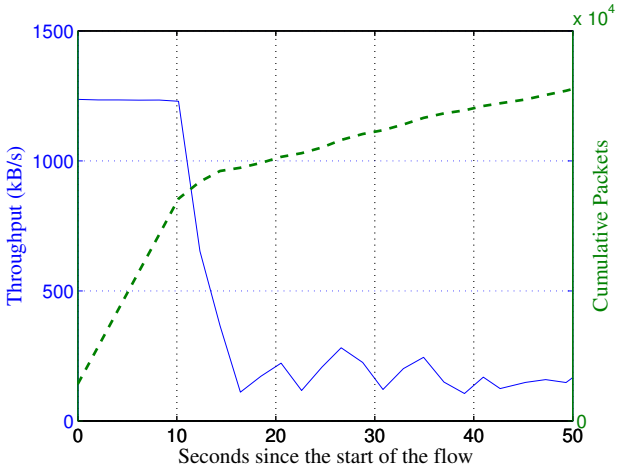



**Figure 6.** Throttling throughput for long TCP flows using Click router.

of strata, we do not have enoug space space to list the causal effect of each ISP for each strata.)

For the neutral ISPs, $N_1$, $N_2$, and $N_3$, we find that the adjusted difference in performance compared to average is slightly positive in all experiments; *i.e.*, the performance of the service is somewhat better than average for these ISPs. If we had a large number of ISPs, we would have expected the average causal effect to be exactly zero (*i.e.*, the performance using these neutral ISPs should be equal to the baseline). Because the experiment has only five ISPs (three neutral and two discriminating), the performance from the discriminatory ISPs decreases the baseline performance, which in turn causes the performance of the neutral ISPs to appear slightly above baseline. The 90% confidence interval for each of the neutral ISPs spans both sides of zero (neutral).

For the discriminating ISPs, $D_1$ and $D_2$, NANO finds negative causal effect on throughput in each of the experimental scenarios where these ISPs discriminated. For example, the confounding adjusted causal effect for ISP $D_1$ is -108 kbps for the Simple Discrimination experiment, indicating that the throughput is 108 kbps less than the baseline throughput; the entire confidence interval is in the negative range.

Table 3 also shows that NANO also avoids mischaracterizing an ISP when it is not discriminating. In the Long Flow Discrimination experiment, ISP $D_1$ does not discriminate against service $S_2$; similarly, ISP $D_2$ does not discriminate the flows for service $S_1$: NANO

correctly estimates close to neutral performance for these ISPs in these cases. On the other hand, in the cases where these ISPs discriminate, NANO correctly detects a negative effect on throughput. In the third experiment involving BitTorrent, NANO computes a negative causal effect for ISP $D_2$, which degrades performance for BitTorrent flows that are not in a preferred set of subnets. The causal effect on the throughput is about 300 kbps below the baseline throughput performance.

The confidence intervals for discriminating ISPs are wider than the confidence intervals for the neutral ISPs because when the discriminating ISPs drop packets for TCP flows, the throughput for these flows becomes more variable. The confidence intervals for the experiment involving BitTorrent are also larger than other experiments involving discrimination, which probably results from three causes. First, BitTorrent downloads smaller chunks at a time, and such transfers can be bursty and have high variance in throughput. Second, BitTorrent uses more simultaneous connections, which interfere with each other and increase variance. Finally, BitTorrent's peer selection criteria means that there may not be a steady transfer between a pair of peers. We also observe that in general the causal effect of ISP $D_2$ is more negative than ISP $D_1$, which occurs because ISP $D_2$ uses a higher packet drop rate (0.3%) than ISP $D_1$ (0.1%). As a result, more degradation occurs in ISP $D_2$ than in $D_1$.

### 5.4.2 *Can NANO determine discrimination criteria?*

We evaluate the extent to which NANO can determine the discrimination criteria for each of the three experiments.

To infer the criteria that ISP $D_1$ is using to discriminate against long flows, we labeled the stratum for service $S_1$ where we detected more than 100 kbps of causal effect as the discriminated strata and the remaining strata as undiscriminated. We then ran the J.48 decision tree on this labeled dataset, where the data columns included the /24 subnet (`location`) of servers and average number of cumulative packets (`cum_pkts`) for a session. The decision tree produces the following rules:

```
cum_pkts <= 10103 -> not_discriminated
cum_pkts > 10103 -> discriminated
```

and yields 89% accuracy with a 7% false positive rate. The decision tree ignored the `location` variable, correctly inferring that ISP $D_1$ is not discriminating based on destination but rather when the flow's duration exceeds 10,103 packets. Recall from Table 2 that $D_1$ drops packets for flows exceeding 10,000 packets.

For the BitTorrent experiment, we labeled the strata similarly, and used the J.48 algorithm. The resulting decision tree correctly identifies most of the identifiers for the /24 networks that the ISP $D_2$ discriminates. The accuracy and false positive rates are 76% and 14% respectively.

We used the same technique for ISP $D_1$ in the Simple Discrimination experiment, but the decision tree fails to produce a conclusive tree, which is expected because $D_1$ discriminates against all sessions without any criteria.

### 5.4.3 *Can NANO identify sufficiency of confounders?*

There is no automated way to enumerate all the confounding variables or determine that a passive dataset has all the confounding variables. We apply a heuristic that helps determine whether we are missing any major confounding variables.

If we are capturing all the confounding factors, then a regression function $f(X;Z)$, trained on the ISP, $X$, and the confounding factors $Z$, should accurately predict the response time $y$, and the predicted value $\hat{y}$, should be unbiased. We can test for bias by verifying that the distribution of error, $(y-\hat{y})/y$, is centered at zero, with high confidence, and that there is no correlation between the error and the outcome variable. Additionally, if the confounders are proximate or direct (other) causes for service performance, $f()$, should be able to predict the outcome variable. There may be other causal variables besides the ISP and the confounders that influence performance and are needed for high-accuracy prediction. However, as long as these variables are not correlated with the ISP, we do not need to account for them. As a result, the predictor $f()$ does not have to be high accuracy to rule out missing confounders, it only needs to be unbiased.

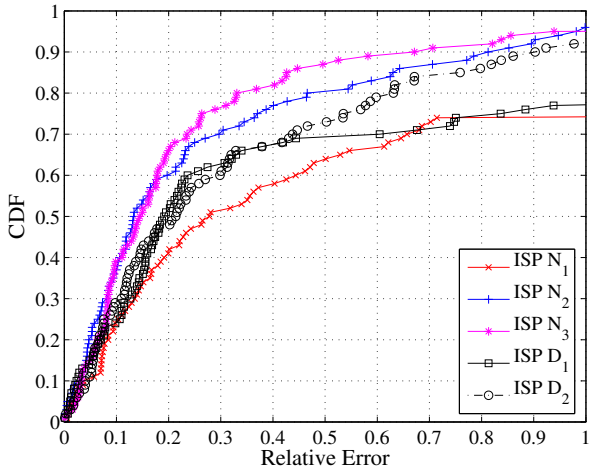



**Figure 8.** Distribution of relative error in predicting the throughput for the Long Flow Discrimination experiment with the variables that NANO collects.

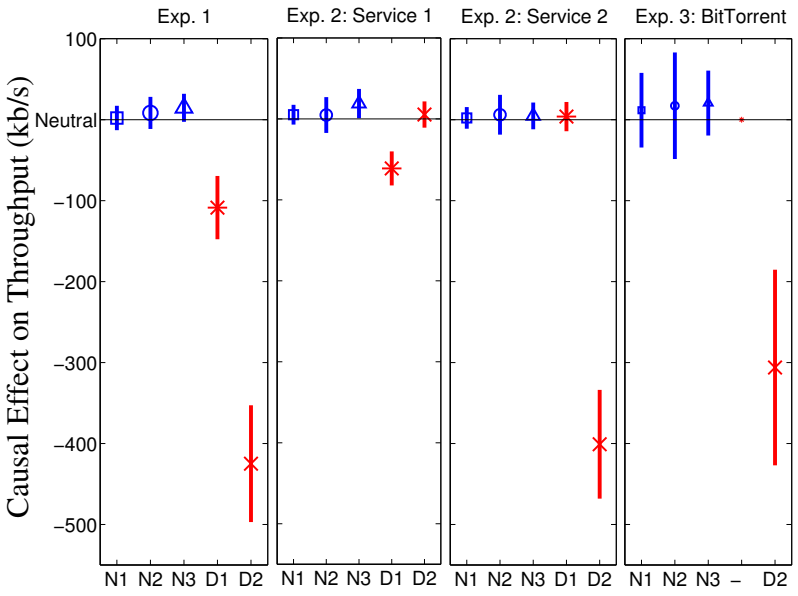
We analyze this heuristic for determining the performance for service $S_1$. We used the ISP with the network round-trip time, the relative location of clients and servers, and the cumulative bytes as the input variables for the predictor, $f()$. We use one random 2/3 of the data for this experiment as training and other 1/3 as test, and predict the performance for the test data. We found that the error was centered at zero and showed small correlation with the outcome value. The correlation values between the error and outcome variable are 0.01, 0.08, 0.05, 0.01, and 0.05 for ISPs $N_1$, $N_2$, $N_3$, $D_1$ and $D_2$, respectively, thus indicating that it is unlikely that there is a major confounding variable missing.

Figure 8 shows the modulo prediction error, $|(y-\hat{y})/y|$. The median error is less than 30% for all the ISPs. Between 10% and 30% of the measurements have 60% or more error for various ISPs, indicating that there may be other causes for performance degradation besides what we capture. Because the error does not seem to correlate with either the ISP or the performance, we believe that those causes do not correlate with ISP and performance at the same time, and are thus not confounding.

### 5.4.4 *Does NANO scale with the size of the input data?*

We discuss the effect of data volumes on accuracy and the effect of number of clients on NANO server load.

**Data vs. accuracy.** As shown in Equation 4 and Equation 9, the confidence level on the number of session measurements from the discriminating and the neutral ISPs, as well as the number of strata in which we observe data.




| ISP | Service Performance | |
|---|---|---|
| *Experiment 1. Simple Discrimination* | | |
| | HTTP Service | |
| $N_1$ | 2.10±15.1 | |
| $N_2$ | 8.39±19.8 | |
| $N_3$ | 14.65±17.3 | |
| $D_1$ | **-108.6±39.1** | |
| $D_2$ | **-424.91±72.1** | |
| *Experiment 2. Long Flow Discrimination* | | |
| | HTTP $S_1$ | HTTP $S_2$ |
| $N_1$ | 5.17±12.2 | 2.20±13.4 |
| $N_2$ | 4.80± 6.1±24.6 | |
| $N_3$ | 18.9±18.1 | 4.65±16.5 |
| $D_1$ | **-61.20±21.0** | 3.82±18.1 |
| $D_2$ | 5.36±16.2 | **-400.91±67.2** |
| *Experiment 3. BitTorrent Discrimination* | | |
| | BitTorrent | |
| $N_1$ | 11.71±46.1 | |
| $N_2$ | 17.2±65.8 | |
| $N_3$ | 20.56±40.1 | |
| $D_2$ | **-306.13±120.8** | |

**Table 3.** Causal effect (in kbps) for each ISP using Eq. 9, with 90% confidence intervals. Each point indicates the confounding-adjusted average difference in throughput for each service and ISP in various experiments; the lines extend to 90% confidence intervals. Near-zero values imply that the performance is close to *baseline* and there is no observed causal relationship between the ISP and the service. Large negative values indicate a significant causal relationship between the ISP and performance degradation. The accompanying table provides values for the causal effect and confidence intervals.

The results in Table 3 for Experiment 2 are obtained using about 1,900 strata (121 for location, and 15 for cumulative packets in a flow—other confounding factors did not have variability for this experiment) and about 100,000 periodic measurements obtained from the NANO-Agents. To assess the effect of fewer measurements, we performed two random sub-samples of 50,000 and 20,000 measurements each. We re-calculated the causal effect, and found that the mean causal effect did not change appreciably for any of the ISPs, but the confidence interval widened by 1.2–1.9 times for various ISPs using 50,000 measurements, and by 1.8–2.7 times for various ISPs using only 20,000 measurements; this conforms to our expectations of having confidence intervals expand by a factor of $\sqrt{5}$ and $\sqrt{2}$, respectively. After a participating client installs the NANO-Agent, NANO servers receive data continually, which makes it easy to collect several hundred thousand measurements for every client in a matter of few days (We collected the measurements for the experiments in roughly 2 hours). Thus, in a real deployment, NANO should be able to gather enough measurements to be highly accurate.

The other aspect of accuracy is *coverage*; the more strata that NANO-Agents cover, the better the information it can provide information about discrimination based on certain features. Certain confounders, such as time-of-day variation, are easy to cover because they simply require gathering estimates over a long time interval without requiring additional clients to participate. Increasing coverage across other confounders, such as location or application type, the type of network interface, or operating system, require participation from additional clients. The required number of clients grows roughly as the size of the cross-product of the range of the confounding variables that require separate clients for measurements. We note that NANO-Agents running on laptops may be quite valuable for helping to increase coverage, because they can cover multiple locations, ISPs, and network-interface types as the user roams. NANO tracks the changes. Unfortunately, as the user roams, some of the information that the user provides to NANO-Agent at installation time becomes invalid. We are improving NANO so that it can automatically infer these variables instead of relying on users to provide the correct values.

**CPU and storage overhead.** We present some back-of-the-envelope numbers on the scalability of NANO. Based on the data that NANO-Server is receiving from the early adopters, uncompressed reports require about 3.2 kB every ten seconds on average per user. Using an estimate of 2.5 kbps per user, to support 10,000 users, the NANO-Servers need roughly 25 Mbps bandwidth and about 12 GB of storage per day to archive the user reports. On a server with 3.2GHz CPU and 4 GB of RAM, NANO takes about 22 ms on average to demarshall a client report, stratify the data, and insert it into a database. At this rate, NANO can support about 5,500 clients in real-time. Our current implementation spawns a new process for batch processing of all the reports that the server receives every minute. Optimizing this process can improve scalability further. Since NANO-Agents perform a DNS lookup to establish a connection to the NANO-Server, we could ultimately use DNS load balancing to distribute load and storage across multiple servers.

## 6. DISCUSSION

In this section, we address limitations with NANO and our ongoing work to address these limitations.

**Sufficiency of confounding variables for real world application.** If NANO fails to adjust for certain confounding variables, it may miscalculate the causal effect: both over and underestimation are possible. Unfortunately, there is no automated way to enumerate all confounding variables for a problem, or to conclusively test that

a given set of confounding variables is sufficient. As in biostatistics and epidemiology where passive datasets are used for causal inference, enumerating confounding variables relies on domain knowledge. Fortunately, network performance is well understood among researchers, and it is relatively easy to enumerate the variables that correlate with ISPs and can also significantly affect service performance. We believe that the variables that we listed is comprehensive, and any remaining confounding variables will have only minimal effect on accuracy of causal inference. Still, if we find that there are additional confounding variables, we can collect data for them in updated releases for NANO-agents and correct the inference at the server end.

**Better privacy for users.** In addition to the techniques already implemented in NANO-Agents, NANO-Agents could further protect user privacy in three ways. First, NANO-Agents could collect data from only the top 'k' Web sites (*e.g.*, according to Alexa) and strip personally-identifiable information from the payloads of this traffic. The data collected from clients would only reveal whether they had visited popular sites (not overly sensitive, since many users visit these sites) and the performance they were experiencing to those sites. Second, NANO could use a combination of passive and active measurements to produce the corpus of data used for inference; in such cases, the NANO-Server would receive all measurements, but would not be able to distinguish which traffic was generated solely to probe the network and which traffic was actually initiated by the client. Finally, clients using NANO might send their reports through an anonymizing network (*e.g.*, Freenet [6]) that obfuscates the source of the original report. Of course, the IP addresses of the clients would still be contained in the traffic traces, but in the process of mixing, IP addresses on various traces could possibly be swapped without affecting the stratification.

**Integrity of reports from agents.** NANO-Agents could lie about the data they collected, by producing false traces or modifying the statistics about the traffic at the client. In these cases, it may be difficult to detect when a client reports false statistics about its observed network performance. We suggest two possible techniques that could help mitigate this possibility. First, NANO could collect data from NANO-Agents that have similar values for various confounding factors (*e.g.*, same upstream ISP, same portion of the network topology). In these cases, reported performance measurements yield continuous discrepancies, NANO could determine that a NANO-Agent was reporting inaccurate results.

**Defense against evasion.** NANO establishes causality by measuring the difference in expected values for response times given the use of a specific ISP and its deviation from baseline measurements. An ISP might try to conceal discrimination by treating traffic such that *mean* value of performance remains unaffected. For example, ISP may give exceptionally good performance to some clients and degrade performance for others. To defend against this type of attack, we imagine that NANO might be extended in two ways. First, we could modify NANO's causal inference algorithms to operate on *multiple points in the response-time distribution*, as opposed to simply inferring causality based on mean values. Second, presuming that an ISP's attempt to game the detection may vary over a range of time, we could run NANO's inference algorithm over different time granularity to attempt to catch more fine-grained variations in an ISP's policies across users, services, or applications.

**Augmenting with active measurements.** NANO's reliance on passive measurements is limiting because NANO can obtain measurements for specific services only when the user of the client accesses those services. On demand or active measurements from the client can be beneficial in a number of ways; (a) measurements to known good services can help mitigate chances of false inference; (b) if there are not enough samples for comparison between ISPs for certain stratum or services, these samples can be obtained actively. NANO-Agent has an active probing client that periodically contacts NANO-Server to obtain information about servers for which active measurements are desired. Currently, NANO-Agent can perform HTTP GET and POST to servers that the NANO-Server directs the client to using HTTP redirects.

**Motivating users to install NANO-Agents.** NANO users can only draw meaningful conclusions if they compare their measurements to those of other users. Thus, encouraging users to deploy NANO is critical to its success. To provide users an immediate incentive to install NANO-Agents, independently of whether other users have deployed agents, we have developed a Web interface where users can view their own performance statistics in isolation, as well as compare their performance statistics to other users. We hope that allowing users to analyze their own data through an interactive interface will give users the incentive to deploy the tool and help build a critical mass of users running NANO-agents.

# 7. RELATED WORK

We survey related work on detecting ISP discrimination.

**Measurement tools.** Glasnost [8] detects TCP connection resets of peer-to-peer applications. It simulates the BitTorrent protocol and detects spurious TCP RST packets which might be generated by the ISP to throttle BitTorrent. Glasnost is effective at detecting current discrimination technique against BitTorrent traffic, but if ISPs change the discrimination mechanism, then Glasnost will need to incorporate new tests. NVLens [29] focuses on detecting performance degradation among backbone ISPs via setting of TOS bits in the IP headers of the ICMP packets, so its analysis is specific to this mechanism.

Both Network Diagnostic Tool (NDT) [5] and Network Path and Application Diagnostics (NPAD) [18] rely on active client probing to detect network performance issues. Netalyzr [21] performs a series of tests using the client's browser to check the status of commonly used protocols, such as, POP, Bittorrent, and SSH. Diffprobe [13] performs active measurements from client machines to M-Lab [19] nodes to detect any ISP traffic discrimination based on traffic shaping mechanisms. These tools perform active measurements, which are detailed, but also evadable.

Tripwire uses a fingerprint-based technique to detect modification of in-flight packets [23]. We focus on violations that result in performance degradation, rather than modification of content. These techniques rely on active measurements and focus on specific discrimination mechanisms as opposed to *in situ* measurements.

**Comparing performance across ISPs.** NetDiff [17] detects performance differences between backbone ISPs. NetDiff uses the geographic location as a normalizing factor for fair comparison between ISPs, and in a sense adjusts for a confounding factor in the assertion that one ISP is better than another. NetDiff uses ICMP packets to probe the paths, but an ISP can evade detection by detecting such probe packets. NANO overcomes these difficulties by passively monitoring the performance of the various services. NANO builds on previous work on characterizing ISP networks [16] and monitoring ISP SLAs [25] to adjust for ISP topology differences. Keynote [14] compares performance across backbone ISPs; in ad-

dition to the above drawbacks, it also requires ISP cooperation for placement of measurement nodes, which may not be possible.

**Comparing services within an ISP.** This paper extends and evaluates the ideas presented in a preliminary paper [26]. Our approach to inferring the effect of an ISP's policy on a service's performance relies on comparing performance of a service across multiple ISPs. Another interesting point in the design space is comparison performance of similar services within an ISP and then determining whether there is a difference in performance among these services; NVLens [29], for instance, compares the latency for BitTorrent packets with the performance for HTTP packets. We believe that while interesting, this choice of comparison presents additional challenges that can be difficult to overcome. The services may differ in ways that naturally affect their performance: for example, a service that sends packets at a higher burst-rate may experience higher loss and latency for similar average transfer rate. Even if two services have similar traffic patterns, (*e.g.*, due to both using TCP), the completion time for a request may depend on additional server side variables, such as the back-end delays, caching rate, or rate-limiting at the server end. A fair direct comparison between performance would require comparing the performance of services that are *similar* in all aspects that can affect a service's performance; this can be difficult to achieve in general.

# 8. CONCLUSION

This paper presented NANO, a system that applies causal inference to passively collected data from end-hosts to detect service degradation caused by ISP discrimination. In contrast to existing approaches, which apply rules to detect specific types of discrimination and actively probe ISPs to detect discrimination, NANO observes *in situ* traffic and performs causal inference to determine whether the characteristics of the actual application traffic itself shows any variations that can be attributed to the ISP. NANO's approach enables it to detect more types of discrimination than existing approaches and makes it more difficult for ISPs to evade by treating test probes differently than data traffic.

Our evaluation showed that NANO can detect discrimination for different policies and application types, thus demonstrating that NANO's detection is general and can detect discrimination even when the ISP's discrimination policies are not known *a priori*, as long as the variables that may significantly confound the relationship between ISP policy and service performance are known. NANO can also quantify the extent of discrimination and identify the discrimination criteria. Our experiments also showed that NANO scales well with the number of clients.

We have released NANO and intend to collect data from a wide range of heterogeneous NANO-Agents across the Internet. We are working with a large content provider to deploy NANO on a large, geographically distributed measurement platform.

# Acknowledgements

We would like to thank Joseph Hellerstein at UC Berkeley for his valuable feedback that helped improve several aspects of our work. This work is supported by NSF Awards CNS-0643974, CNS-0721581, and CNS-0721559.

# References

[1] N. Andersen. Cox ready to throttle P2P, non "time sensitive" traffic. http://tinyurl.com/bcexla, Jan. 2009.

[2] A. Bavier, M. Bowman, D. Culler, B. Chun, S. Karlin, S. Muir, L. Peterson, T. Roscoe, T. Spalink, and M. Wawrzoniak. Operating System Support for Planetary-Scale Network Services. In *Proc. First Symposium on Networked Systems Design and Implementation (NSDI)*, San Francisco, CA, Mar. 2004.

[3] Its back to 'Pipes' and 'Free rides': Internet neutrality under attack (again). http://tinyurl.com/lyjo98, June 2009.

[4] BT Heavily Throttling BBC, All Video. http://http://tinyurl.com/m2v7f5, May 2009.

[5] R. Carlson. Network Diagnostic Tool. http://e2epi.internet2.edu/ndt/.

[6] I. Clarke. A distributed decentralised information storage and retrieval system. Master's thesis, University of Edinburgh, 1999.

[7] J. Dean and S. Ghemawat. Mapreduce: Simplified data processing on large clusters. In *Proc. 6th USENIX OSDI*, San Francisco, CA, Dec. 2004.

[8] M. Dischinger, A. Mislove, A. Haeberlen, and K. P. Gummadi. Detecting bittorrent blocking. In *Proc. Internet Measurement Conference*, Vouliagmeni, Greece, Oct. 2008.

[9] E. Felten. Three Flavors of Net Neutrality. http://www.freedom-to-tinker.com/blog/felten/three-flavors-net-neutrality, Dec. 2008.

[10] Protocol Buffers. http://code.google.com/apis/protocolbuffers.

[11] G. Gross. Google, partners release net neutrality tools. http://www.thestandard.com/news/2009/01/28/google-partners-release-net-neutrality-tools, Jan. 2009.

[12] N. Jewell. *Statistics for Epidemiology.* Chapman & Hall/CRC, 2004.

[13] P. Kanuparthy. Diffprobe: Detecting ISP Traffic Discrimination. http://www.cc.gatech.edu/~partha/diffprobe/.

[14] Keynote Home Page. http://www.keynote.com/, 1999.

[15] D. Lambert and C. Liu. Adaptive thresholds: Monitoring streams of network counts. In *Journal of the American Statistical Association*, volume 101, No. 473. Applications and Case Studies, Mar. 2006.

[16] H. V. Madhyastha, T. Isdal, M. Piatek, C. Dixon, T. E. Anderson, A. Krishnamurthy, and A. Venkataramani. iPlane: An information plane for distributed services. In *Proc. 7th USENIX OSDI*, Seattle, WA, Nov. 2006.

[17] R. Mahajan, M. Zhang, L. Poole, and V. Pai. Uncovering Performance Differences among Backbone ISPs with Netdiff. In *Proc. 5th USENIX NSDI*, San Francisco, CA, Apr. 2008.

[18] M. Mathis, J. Heffner, and R. Reddy. Network Path and Application Diagnosis. http://www.psc.edu/networking/projects/pathdiag/.

[19] Measurement Lab. http://measurementlab.net, Jan. 2009.

[20] NANO Website. http://www.gtnoise.net/nano.

[21] Netalyzr. http://netalyzr.icsi.berkeley.edu/.

[22] J. Pearl. *Causality: Models, Reasoning, and Inference.* Cambridge University Press, 2000.

[23] C. Reis, S. D. Gribble, T. Kohno, and N. C. Weaver. Detecting in-flight page changes with web tripwires. In *Proc. 5th USENIX NSDI*, San Francisco, CA, Apr. 2008.

[24] S. B. Robert Beverly and A. Berger. The internet's not a big truck: Toward quantifying network neutrality. In *Passive & Active Measurement (PAM)*, Louvain-la-neuve, Belgium, Apr. 2007.

[25] J. Sommers, P. Barford, N. Duffield, and A. Ron. Efficient Network-wide SLA Compliance Monitoring. In *Proc. ACM SIGCOMM*, Kyoto, Japan, Aug. 2007.

[26] M. B. Tariq, M. Motiwala, and N. Feamster. NANO: Network Access Neutrality Observatory. In *Proc. 7th ACM Workshop on Hot Topics in Networks (Hotnets-VII)*, Calgary, Alberta. Canada., Oct. 2008.

[27] M. B. Tariq, A. Zeitoun, V. Valancius, N. Feamster, and M. Ammar. Answering "What-if" Deployment and Configuration Questions with WISE. In *Proc. ACM SIGCOMM*, Seattle, WA, Aug. 2008.

[28] L. Wasserman. *All of Statistics: A Concise Course in Statistical Inference.* Springer, 2003.

[29] Y. Zhang, Z. M. Mao, and M. Zhang. Ascertaining the Reality of Network Neutrality Violation in Backbone ISPs. In *Proc. 7th ACM Workshop on Hot Topics in Networks (Hotnets-VII)*, Calgary, Alberta. Canada., Oct. 2008.