On the Feasibility of Network Delay Tomography using only Existing Infrastructure

Kostas G. Anagnostakis Michael B. Greenwald

Distributed Systems Laboratory
CIS Department, University of Pennsylvania
{anagnost,mbgreen}@dsl.cis.upenn.edu

Abstract

We propose a novel measurement technique for network delay tomography that infers network-internal queueing delay using network routers as measurement nodes. In contrast to existing tomography techniques, the proposed technique does not require any additional infrastructure support, and can thus be directly applied to the analysis of network performance and the study of the Internet. We study the implications of using network routers as measurement nodes, showing the feasibility and constraints of our approach using extensive network measurements. We also report on the use of our method for the study of Internet paths with respect to the existence of single or multiple bottlenecks. Our preliminary results show the existence of such paths may not be negligible. This result has potential implications on the evaluation and fairness of congestion control mechanisms.

1 Introduction

Network measurement techniques are crucial for obtaining insight into network performance, as well as for understanding the behavior of network control mechanisms such as TCP congestion control [12]. In this context, there is a growing interest in techniques for inferring internal network performance from end-to-end (*e.g.* active) measurements. These inference techniques, often referred to as *network tomography* techniques [21], derive network-internal statistics by injecting probe packets from one source to multiple destinations and correlating the observed packet behavior. Early work in this direction [7, 9] has studied the use of multicast to infer link properties; a multicast tree is set-up so that delay and loss to the leaves of the multicast tree can be correlated to reveal per-link properties. Practical concerns about the limited deployment of multicast have recently shifted focus to end-to-end unicast techniques [10, 8]. The unicast techniques are based on the idea that the behavior of a nominal multicast packet can be approximated with multiple back-to-back unicast packets. Due to being closely spaced, back-to-back packets are expected to observe similar queueing and loss effects on the common part of the paths traversed. Based on the dynamics of back-to-back packets, also discussed in the development of TCP congestion control [12] and packet-pair flow control [13], the work and results of multicast inference techniques could be naturally extended to the case of back-to-back unicast packets.

Despite the extensive work and interest in this area, a major practical constraint remains in that current network tomography techniques require the deployment of an appropriate measurement infrastructure. Thus, it is not currently possible to use these techniques for large-scale studies of Internet performance. Additionally, the deployment of a partial or sparse measurement infrastructure may be insufficient, as it affects the technique's accuracy and does not allow for wider, large-scale operational or research use.

Our work explores the possibility of performing network tomography without additional infrastructure support. In this paper we restrict our focus to network *delay* tomography, leaving inference of other metrics (such as loss and throughput) outside the scope of this paper. We design and analyze a novel technique that uses existing infrastructure, particularly network-internal routers, provided they support the Timestamp service of the ICMP protocol [17]. This technique removes the need for a widely deployed measurement infrastructure, and greatly simplifies algorithms for network delay tomography in a significant number of cases. However, our technique introduces a number of new questions and complications, which we attempt to address in this paper.

First, we studied a large number of Internet paths and determined that ICMP Timestamp, which is not a required function by Internet standards [5] and is rarely used (and often misused, as we will discuss in Section 3.4), is supported widely enough to support our technique. Second, we performed extensive network measurements, to investigate issues arising from routing instability and path structure. These issues have a significantly effect on the design, scope and feasibility of our measurement technique. In summary, our results show that network delay tomography is indeed feasible, to a certain level, without infrastructure support. While it is not yet possible to state whether this technique is appropriate for wide-spread end-user measurement of network performance, we can assert that it is sufficiently useful for the particular needs of larger-scale research studies.

The main contribution of this paper is a new technique that is both simpler and more accurate than existing network delay tomography techniques. We demonstrate that it is applicable for a significant fraction of network paths. For paths over which our new technique is inapplicable because of routing anomalies, we are investigating simple optimizations to existing techniques that improve both accuracy and efficiency.

Using the improved tomography techniques, we performed network delay tomography on actual Internet paths with the purpose of obtaining information on spatial and temporal characteristics of network congestion. Our initial measurements searched for instances of paths with multiple congested segments, and measured the distribution of queuing delay on individual links. The single-bottleneck assumption dominates the congestion control literature, and motivates the ubiquitous "bar-bell topology" in simulation studies. Although we used a set of random paths sourced at a single measurement site, which may introduce bias in our results, the results indicate that multiple-bottleneck paths are not too uncommon. This indicates the need for a more detailed investigation as well as further consideration of implications and alternatives to the single-bottleneck model. The existence of multiple congested segments, coupled with our per-link measurements of delay distribution impact the behavior of algorithms that make assumptions about the behavior of network delay (c.f. TCP Vegas[6], or TCP-BFA [4]).

1.1 Paper organization

The rest of this paper is organized as follows. In Section 2 we describe the measurement methodology and in Section 3 we analyze the constraints and potential sources of error. In Section 4, we report on the use of the technique for studying Internet path properties. We briefly discuss a number of issues that remain unanswered and directions for further study in Section 5 and conclude in Section 6.

2 Methodology

The basic concepts of our methodology are similar to network delay tomography with end-to-end unicast measurements [8]. The goal of network delay tomography is to provide insight into which links or path segments contribute to delay variations on a network path. The technique is based on the principle that when back-to-back packets are sent to different receivers, they observe similar performance on the common part of the paths they traverse. Thus, the delays observed at the receivers can be statistically correlated to identify which segments or links are responsible and contribute to end-to-end delay.

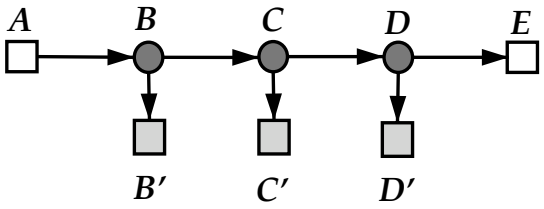


Figure 1: **Topology for network tomography**

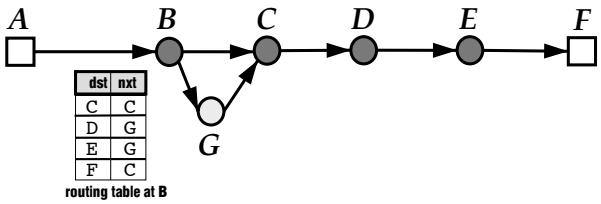


Figure 2: **Example path structure complication**

The main difference in our technique, compared to previous efforts, is that we use routers on the path under study as measurement nodes, in order to remove the need for deploying a measurement infrastructure. Two advantages accrue because routers are more widely available than nodes in a special purpose infrastructure. First, we can use two nodes on the same path, serially, and use simple subtraction to compute per-link delays, rather than using two parallel paths and computing correlations to compute delays. Second, we can use tomography techniques for large-scale measurement studies of the Internet.

In the following paragraphs, we first introduce the basic concepts of network delay tomography and briefly describe the available router mechanisms we can exploit. Thereafter, we focus on the questions raised and constraints introduced by involving routers in the measurement function, and their implications on the design and feasibility of our approach.

2.1 Basic network delay tomography

We model a network as an arbitrary graph G, with nodes N and links L. Each link, $l \in L$, is an ordered tuple $\langle n_i, n_j \rangle$ in $N \times N$. A link $\langle n_i, n_j \rangle$ implies that nodes n_i and n_j are neighbors, and that n_i can send a packet directly to n_j with no intermediate hops. Let $n_i \xrightarrow{*} n_j$ denote a path taken by a packet, in this case from n_i to n_j. $n_i \xrightarrow{*} n_j \xrightarrow{*} n_k$ denotes a path from n_i to n_k that passes through n_j. When relevant, we denote a direct one-hop path from a node n_i to its neighbor n_j by $n_i \xrightarrow{1} n_j$. We use the notation $\vec{P}$ to represent path variables.

We define the notion of **source**, **head**, **tail**, prefix, suffix, and **joint** path in the obvious ways. For a node n in a path $\vec{P}$, let n partition $\vec{P}$ as follows. $\vec{P} = \vec{P_1} \xrightarrow{1} n \xrightarrow{1} \vec{P_2}$. Then $\vec{P_1} = \textbf{head}(\vec{P}, n)$ and $\vec{P_2} = \textbf{tail}(\vec{P}, n)$. Let $\textbf{source}(\vec{P})$ denote the unique n such that $\textbf{head}(\vec{P}, n) = \epsilon$. $\vec{P_1}$ is a *suffix* of $\vec{P_2}$ if there exists some node n in $\vec{P_2}$, s.t. $\vec{P_1} = \textbf{tail}(\vec{P_2}, n)$. Analogously, $\vec{P_1}$ is a *prefix* of $\vec{P_2}$ if there exists some node n in $\vec{P_2}$, s.t. $\vec{P_1} = \textbf{head}(\vec{P_2}, n)$. Let $\textbf{joint}(\vec{P_1}, \vec{P_2})$ denote the largest common prefix of $\vec{P_1}$ and $\vec{P_2}$. More formally, if $\textbf{source}(\vec{P_1}) = \textbf{source}(\vec{P_2})$, and node n is in both $\vec{P_1}$ and $\vec{P_2}$ and $\textbf{tail}(\vec{P_1}, n)$ and $\textbf{tail}(\vec{P_2}, n)$ are totally disjoint, and $\textbf{head}(\vec{P_1}, n) = \textbf{head}(\vec{P_2}, n)$ then $\textbf{joint}(\vec{P_1}, \vec{P_2}) = \textbf{head}(\vec{P_1}, n) \xrightarrow{1} n$.

Each node n contains a routing map $\mathcal{R}_n(d) : N \longrightarrow N$. $\mathcal{R}_n$ takes a destination node $d \in N$ and returns the next hop $h \in N$. $h = \mathcal{R}_n(d)$ implies that $\langle n, h \rangle$ is in L, and that packets with destination d passing through n travel on the path $n \xrightarrow{1} h$. For all $\langle n, n' \rangle \in L, \mathcal{R}_n(n') = n'$.

Let $s \xRightarrow{*} d$ represent the path (possibly multi-step) induced by $\mathcal{R}$ on packets travelling from node s to node d. Packets travel on paths based on purely local next-hop routing decisions. That is, $s \xRightarrow{*} d = s \xrightarrow{1} \mathcal{R}_s(d) \xRightarrow{*} d$.

A map $\mathcal{R}$ is *regular* over a graph G if $\forall n \in s \xRightarrow{*} d, \forall m \in \textbf{tail}(s \xRightarrow{*} d, n)$, then $\mathcal{R}_n(m) = \mathcal{R}_n(d)$. $\mathcal{R}$ in the Internet is irregular. For example, let $h = \mathcal{R}_n(d)$. $\mathcal{R}_n(\mathcal{R}_h(d))$ does not necessarily equal h. Further, $\mathcal{R}_n$ is not stable in the Internet — routing maps change over time in response to routing updates. The former irregularity has significant impact over which links we can directly measure. In contrast, measurements suggest that for our purposes we need not model the latter instability [15].

The queueing delay on a path segment is extracted as follows. At times a_i, for $1 \leq i \leq m$, a pair of

back-to-back packets is sent from A to B' and C'. The value of a_i is given by the local clock at node A which is relative to global time *e.g.* $a_i = a'_i + a_G$. For simplicity, we assume that a_G is constant over time. If packets arrive on nodes B' and C' at times b_i and c_i, then $\Delta_i(A \rightarrow B) = b_i - a_i$ and $\Delta_i(A \rightarrow C) = c_i - a_i$ provide the one-way delay from A to B and A to C respectively, relative to the offsets of the two clocks from global time. These offsets may not be known but as we are interested in queueing delay, the absolute one-way delay is not required.

Note that if for some pair $i = k$, there is no queueing delay on links $A \rightarrow B$ and $B \rightarrow C$, then $\Delta_{min}(A \rightarrow C) = c_k - a_k$ is the sum of transmission plus propagation delay between A and C, relative to the clock difference $c_G - a_G$. Naturally, for sufficiently large m, $\Delta_{min}(A \rightarrow C) = min(\Delta_i(A \rightarrow C))$, *e.g.* we can expect that over a sufficiently large number of samples, we will discover the minimum (relative) delay which only contains transmission and propagation components but no queueing. The one-way queueing delay between A and C is then computed as $\Delta^*_i(A \rightarrow C) = \Delta_i(A \rightarrow C) - \Delta_{min}(A \rightarrow C)$.

Given the above, the one-way queueing delay between B and C can then be inferred as $\Delta^*_i(B \rightarrow C) = \Delta^*_i(A \rightarrow C) - \Delta^*_i(A \rightarrow B) + E_i$, where E_i is the difference in delay observed between the two packets, and is assumed to be negligible. This technique can be extended to any pair of nodes to perform tomography on different segments or links of an end-to-end path, or any part of the network. In the rest of this paper, we restrict the scope of our design and analysis to the use of network delay tomography on segments or links of specific end-to-end paths, although most of the observations and results are relevant to some degree and can be extended to measurements on more complex topologies.

2.2 Router mechanisms for network delay tomography

There are two ways of probing routers on an end-to-end path. Generally speaking, the first technique is to send packets that will provoke the router into sending an error packet to the source. The second is to send a packet, in a protocol supported by the router, that legitimately demands a response to the source.

An example of the first technique is to appropriately set the Time-to-Live (TTL) field in the IP header of the probe packet. As the packet travels through the network, the TTL value is decremented at each hop, and when it becomes zero, the router is expected to send an ICMP `UNREACHABLE/TIMXCEED` message to the packet source. While the original purpose of this mechanism is to prevent routing loops, this approach is exploited by tools such as `traceroute` to discover the path between two hosts. The problem with this approach is that the delay between sending the probe and receiving the `TIMXCEED` message is the round-trip delay between the sending host and the router. If there is delay variation on the return path, it is not clear how one could isolate the one-way delay on the path between sending host and router. As different routers on the path have different return paths, and back-to-back probes are spaced at least by the one-way delay between the two measurement points, the correlation assumed by end-to-end unicast techniques does not hold.

An example of the second technique is to use the Timestamp primitive of the ICMP protocol. The ICMP Timestamp option records the time a packet is received at a router. If clocks on the sending host and Timestamp destination are synchronized, the difference between the transmit time and the timestamp in the ICMP reply would be the one-way delay. If the clocks are not synchronized to global time, the difference does not provide one-way delay, but repeated measurements will reveal delay variation, assuming negligible difference in clock rate. The accuracy can be further improved by incorporating mechanisms such as those described in [16] that compensate for clock jumps (due to being synchronized to NTP) and differences in clock rate. Note that ICMP Timestamp is required by the ICMP specification to provide timing in millisecond accuracy, which is sufficient for inferring congestion, assuming typical router buffer sizes.

A rather obvious problem with using ICMP Timestamp is that an ICMP packet sent to a specific router on an end-to-end path may follow a different path from a packet sent to the actual destination. An example

is shown in Figure 2. This is natural, as a probe packet to the target node and a packet to a network-internal node have different destination IP addresses and may thus be routed differently by intermediate routers (obviously, routing in the Internet is not strictly shortest-paths). Additional complications may rise from the processing burden placed on intermediate routers, which could introduce error, but, for the purposes of this study, this error is assumed to be negligible.

We considered several alternatives that are unfortunately less practical or not currently implemented. For instance, using the loose source-routing option of the Internet Protocol could solve part of this problem but network operators unfortunately tend to filter out source-routed packets due to security concerns (or rumors thereof). As our intention is to use existing infrastructure, we cannot assume or expect new mechanisms (e.g. IPMP [14]), protocol amendments (e.g. `TIMXCEED` furnished with timestamps), or a permanent architectural fix like smart packets [19] to provide us with the needed measurement capabilities.

Consequently, the use of ICMP Timestamp appears as the best candidate measurement primitive for our purposes. Nevertheless, we first need to understand how the limitations outlined above would influence the feasibility and accuracy of our technique. To our knowledge, these questions have not been previously studied and thus, Section 3 is devoted to the analysis of these problems.

2.3 Choosing which routers to use as measurement nodes

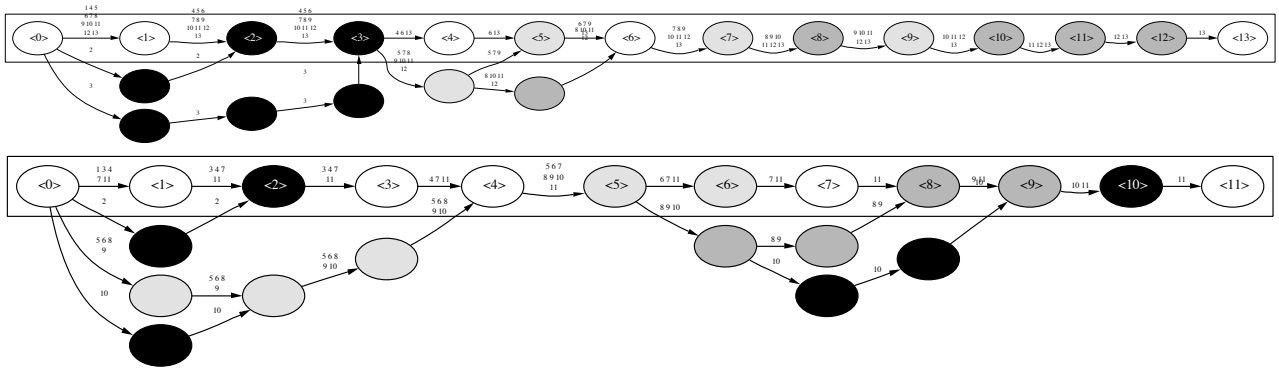


Figure 3: **Path structure and resulting tomography group example: `zakynthos.cis.upenn.edu` to `www.research.att.com` and `www.sprintlabs.com`. The end-to-end path is shown in the box, black nodes are orphans, nodes with the same color are members of the same tomography group, edges are labeled with the destination hops that take that particular link.**

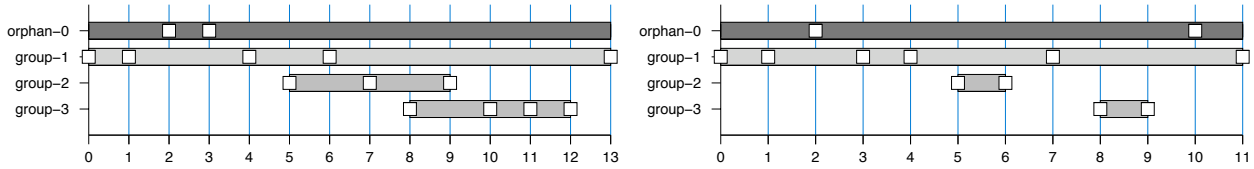


Figure 4: **Orphans and tomography groups for path to `www.research.att.com` and `www.sprintlabs.com`**

The routing issues that result from the use of ICMP Timestamp do not allow just any node on an end-to-end path to be used for network tomography. For choosing which combinations of routers are suitable as measurement nodes. we define the notion of a *tomography group* as a maximum set of nodes that satisfies two basic conditions. First, all nodes in a tomography group must follow the same path up to the first node in the group. Second, the path from the head of the group (the closest node to the source) to each of

the group's nodes lies on the end-to-end path. All pairs of nodes in a tomography group can be used for measuring delay in the path segments connecting them. The constraint that packets to the nodes in a group follow the same path up to the first node in the group is the basic assumption in network tomography and packet-pair techniques in general. Note that the path up to the first node in the group does not necessarily have to be on the end-to-end path. Also, there is always at least one group in a path, containing at least the source and destination nodes. A node is called *orphan* if its tomography group has only one member. It is not generally possible to use an orphan in any combination with another node for measurement.

A particularly interesting situation arises from overlapping tomography groups. Such cases may , in certain circumstances, improve the accuracy of inferring which segments contribute to end-to-end delay, but may also cause ambiguity in attributing queuing to a particular path segment. For example, consider the path of Figure 3 (top) and its tomography groups, as shown in Figure 4 (left). Consider the measurement configuration of nodes 4 and 6 (from group 1), and nodes 5 and 7 (on group 2). If there is queueing between 4 and 6, but not between 5 and 7, queueing is occurring between nodes 4 and 5. If queueing is observed in both segments, then it is not immediately clear which of the links could be congested: it could be the link between 5 and 6, or some combination that includes at least one link in each of the two segments. One can try to tackle this ambiguity by considering the corelation of queueing delays between the two segments: if there is strong corelation, then queueing is most likely occurring on the link where the two segments overlap (*e.g.* between nodes 5 and 6) and there is little or no queueing on the other two links. However, if the corelation is weak, it remains unclear which of the links contribute to the delay, although it becomes evident that there are at least two links where there is queueing occurring. We observe, however, that using node 13 (in fact, the destination in the particular case) could help in showing whether there is queueing in the segment between 6 and 13. If not, we can infer that the link between nodes 6 and 7 is queueing-free, and thus, queueing is most likely to be occurring on the links between 4 and 5, and between 5 and 6. If queueing delays are measured on the segment between 6 and 13, the ambiguity persists, and in the case where there is no additional overlapping group available, it would not be possible to disambiguate the source of queuing with this procedure. It becomes clear from this example that consideration of these issues can significantly help in both data analysis, and also affects the choice of measurement nodes. However, for simplicity, we do not expand into this direction further in this paper.

The choice of nodes clearly depends on the measurement scope and goals. For instance, it is possible to design an algorithm for the case where there is a specific path segment of interest, given the path structure and the ambiguity issues discussed in the previous paragraph. Similar algorithms are discussed in [1] for computing an appropriate tree layout for multicast-based inference techniques. A different problem, that is also closer to our motivation of identifying multiple bottlenecks, is to pick a set of nodes that provide a well-balanced split of the end-to-end path. We provide a simple algorithm, that, given a target number of measurement nodes to employ, computes out of all possible configurations the one that optimizes a certain quality metric. The metric is calculated as follows, given a path structure graph and a set of measurement nodes: for each link, we compute the distance between the link and the two closest nodes that are on the same tomography group. The sum of squared distances provides a reasonable metric of how balanced a split is. Our algorithm calculates the set of nodes that minimizes this metric. Note that, as it does not account for the overlapping segments case discussed above, the algorithm is rather conservative in evaluating the actual quality of a measurement configuration.

2.4 Implementation

The technique described in the previous paragraphs is implemented as a stand-alone user-level tool called `cing` that has two parts: the networking code, written in C, which is used to collect information on path structure, send probe packets and collect timestamp replies, and a set of `awk` scripts that are used for analysis and producing appropriate input to visualization tools (we use `gnuplot` and `ploticus` for most

dataset	Source	Destinations	Measurement period	Paths
PENN-R1	`upenn.edu`	random	Dec 16, 2001 - Dec 18, 2001	18462
PENN-W1	`upenn.edu`	Web log IPs	Dec 15, 2001 - Dec 18, 2001	8502
AUTH-R1	`auth.gr`	random	Dec 16, 2001 - Dec 17, 2001	6742
NJISP-W1	DSL provider, NJ	Web log IPs	Dec 16, 2001	8609

Table 1: **Description of data-sets for path structure analysis**

of the graphs in this paper). As the technique matures further, we intend to integrate the analysis mechanisms with the networking code for compact distribution.

The tool was developed and tested on BSD-based operating systems and networking stacks. Superuser privileges are needed to create a raw socket requiring the tool to be installed as setuid-root for use by non-privileged users. As privileges are only needed during the initialization phase, there are no security threats of using the tool to obtain root access on the executing host. The threat of using `cing` comes primarily from the amount of traffic that can be generated, which could potentially be used for DoS attacks. Note that such attacks can be mounted anyway, hence, `cing` does not provide any advantage to malicious users.

3 Feasibility analysis

In this section we investigate the various issues that affect the feasibility and effectiveness of our technique. Most of the results are based on four data-sets containing measurements of path characteristics (see Table 1). Measurements were taken from three different sites: the University of Pennsylvania campus network, a host behind a DSL service provider in NJ, USA and a university site in Europe. Two of these data-sets contain paths to random destinations (PENN-R1 and AUTH-R1) while the other two contain paths to specific destinations, taken from the University of Pennsylvania SEAS Web server logs (PENN-W1 and NJISP-W1). These data-sets were used to determine whether routers and end-points support ICMP Timestamp, to examine path characteristics with respect to stability and structure, and to investigate the implications of these characteristics on the proposed measurement technique.

3.1 ICMP Timestamp support

The results of measuring how widely ICMP Timestamp is supported in the Internet are summarized in Table 2. We observe that Timestamp is supported on 95.31% of the routers probed. The cases where Timestamp is disabled (while ECHO and TIMXCEED are enabled) is small, at about 2.84%. The case where at least one router in the path does not support Timestamp is however high, accounting for 35.72% of the paths. This means that performing per-link tomography and accurately isolating the sources of delay variation may not always be possible. However, for 48.63% of the paths, all nodes in the path fully support the needed measurement primitives. A specific problem observed, that is not shown in the results of Table 2, is a bug in the ICMP Timestamp implementation that provides the Timestamp reply in the wrong byte order, which was traced to end-points running Microsoft operating systems. However, this problem is detectable on the second probe and thus is easily solved by flagging these nodes and reversing the byte-order before further processing. Note that the paths in the data-sets contain only destinations that reply to ICMP ECHO, hence, the figures presented here do not account for networks that block ICMP ECHO traffic. Also, the results are conservative due to packet loss and given that we limited the number of retries (four) and the associated timeout (2 seconds) for probing each host or router for each of the ICMP-based services.

	routers		paths	
(total probed)	147169		8502	
TIMXCEED, ECHO, TIMESTAMP	140269	(95.31%)	4135	(48.63%)
TIMXCEED, ECHO	4194	(2.84%)	3037	(35.72%)
TIMXCEED, TIMESTAMP	43	(1.39%)	43	(0.05%)
TIMXCEED	1242	(0.84%)	1040	(12.23%)
-no reply-	1421	(0.90%)	1004	(11.80%)

Table 2: **ICMP Timestamp support for the paths in PENN-W1 data-set**

3.2 Path stability

A common assumption in end-to-end measurement techniques is that the paths from the source to the different measurement nodes are stable over the time-scales of measurement. For the described technique, if the end-to-end path changes, it may or may not contain the nodes picked for measurement, and depending on the measurement goals, may require picking a different set. Additionally, the paths to different nodes on the end-to-end path may also change, independently of whether the end-to-end path has changed or not.

A routing study by Paxson [15] reports that 91% of end-to-end paths appear to be stable over periods of hours. However, this study is restricted to end-to-end paths and does not cover paths towards network-internal nodes, although we are not aware of any major reason why the same should not apply to network-internal nodes. However, our technique robustness depends on *the probability that all paths involved in measurement are stable* and not just the end-to-end path, making our technique more sensitive to path instability. We have thus performed a limited measurement study to validate this assumption.

The experiment was setup as follows. We took 40 repeated path structure samples on each of 1263 paths from the PENN-W1 data-set. The median measurement period for taking all 40 samples for each path was 15.06 minutes. We observe that 991 end-to-end paths (78%) are stable during the experiment, a figure that is significantly lower than the 91% reported by Paxson. However, 155 of the 272 non-stable end-to-end paths changed within the UPENN network (hops 2, 3 and 4) while 66 of these 155 paths changed only on hop 2. This indicates that local multi-path routing within the UPENN network is the primary cause for the observed difference. Discounting for this bias, 90.7% of the paths can be considered stable. Analysis of the paths to each hop forming the 991 end-to-end paths that are stable shows further instability in the path structure: only 423 out of 991 end-to-end paths had stable paths to each hop. For the 568 remaining paths, 254 exhibited similar effects with the end-to-end paths described above *e.g.* localized route instability in the UPENN network, 130 had similar single-hop changes in other parts of the network, while 184 had other changes, that would cause problem to network delay tomography. The results are summarized graphically in Figure 5.

Further study is needed to quantify the effect of routing instability on network delay tomography. The conclusion that we are allowed to make here is that stability analysis may be needed for robust network delay tomography. Developing efficient algorithms for detecting routing changes instead of periodically re-analyzing every path is a possibility, but we do not address this aspect further in this paper.

3.3 Path structure

We analyze the structure of network paths and the effect this structure has on our technique, given the requirements and constraints discussed in Section 2.3. In Figure 6 we see that most paths generally have between 1 and 4 tomography groups while no paths have an excessive number of groups. We observe differences in the distribution for each data-set indicating that some aspects of path structure depend on

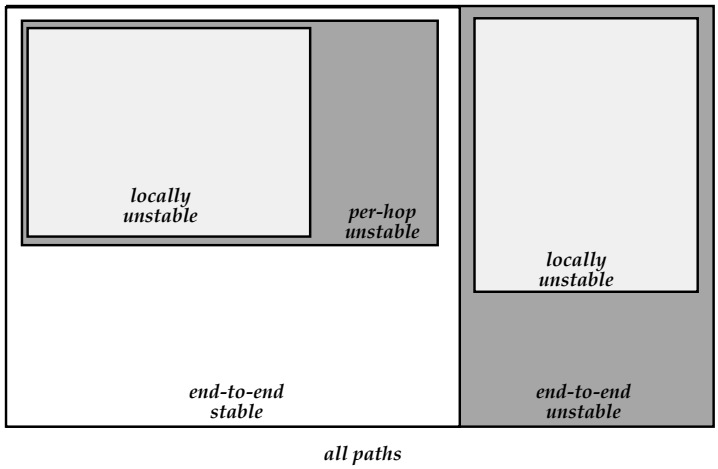


Figure 5: **Graphical representation of path stability results.**

the measurement source. Also, there is a notable difference between PENN-R1 and PENN-W1, shifting the distribution to smaller groups due to the high number of more local IP destinations in the PENN-W1 data-set.

The diameter of a group, *e.g.* the distance between the first and last node, tends to be either relatively small or relatively large, as shown in Figure 7. Note that these statistics do not consider the tomography group containing source and destination, whose diameter is equal to the path length. This also indicates that that overlapping of groups *e.g.* very small groups within larger groups, as well as segments which can be analyzed in-detail down to the level of 2 or 3 hops, are likely to be common.

A key metric for the "coverage" of a path in terms of potential measurement nodes is the number of non-orphan nodes. In Figure 9 we show the number of non-orphan nodes per path, for paths with different numbers of groups. We see that more groups implies more non-orphan nodes, but the improvement diminishes as the number of groups per path increases. We also observe that the median number of non-orphan nodes is around 53% to 73%, indicating that paths are typically well covered.

The relationship between path length and the number of groups is is illustrated in Figure 10, verifying our intuition that the number of groups per path is likely to be higher in longer paths. Furthermore, the ratio of non-orphan nodes decreases with path length, with a significant drop between small (possibly local) paths and medium-length paths , but with slower than linear decrease as we move into longer paths (Figure 11).

In Figure 12 we illustrate the percentage of non-orphan nodes at a given normalized distance from the source node. We observe that the chance of getting a useful node decreases significantly as we move deeper into the path and closer to the destination. Our technique is hereby more likely to be useful and accurate on links and segments closer to the measurement source. In the proximity of the destination, nodes tend to be non-orphans with higher probability than in the core. There is significant difference in the figures for the four data-sets depending on the measurement source, while the difference between random and real destinations does not affect the shape of the curve. Note also that in the PENN-R1 and PENN-W1 datasets, the rate of decrease early in the path is higher than in the other datasets, due to the local routing phenomena also observed in the stability study.

Having analyzed the impact of the routing structure on the feasibility of our technique, it remains to understand the origin of the problem. First, it is clear that network-internal nodes may have an address on different Autonomous Systems (ASes) from the destination node, and AS numbers are a central parameter in inter-domain (*e.g.* BGP [18]-based) routing. Additionally, parallel paths and intra-domain routing may further affect the routing structure, especially considering adaptive intra-domain routing and traffic engineer-

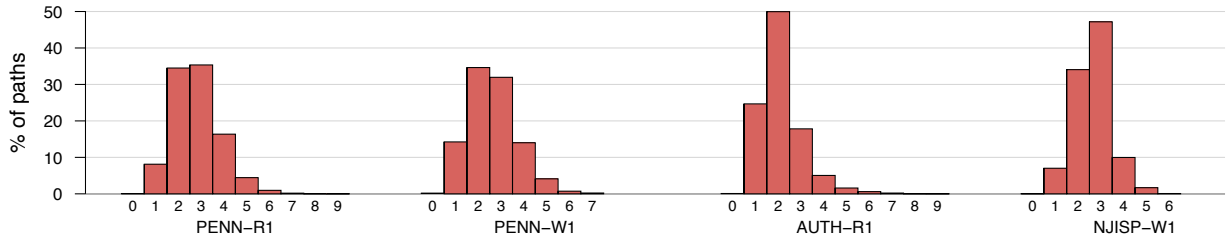


Figure 6: **Number of tomography groups per path**

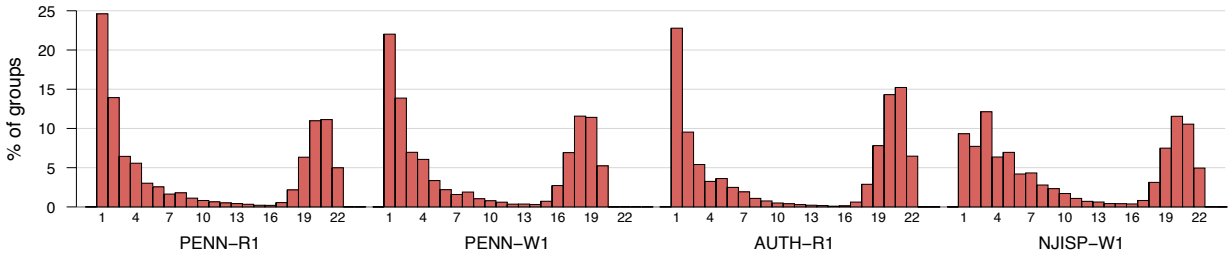


Figure 7: **Tomography group diameter (distance between first and last node in a group)**

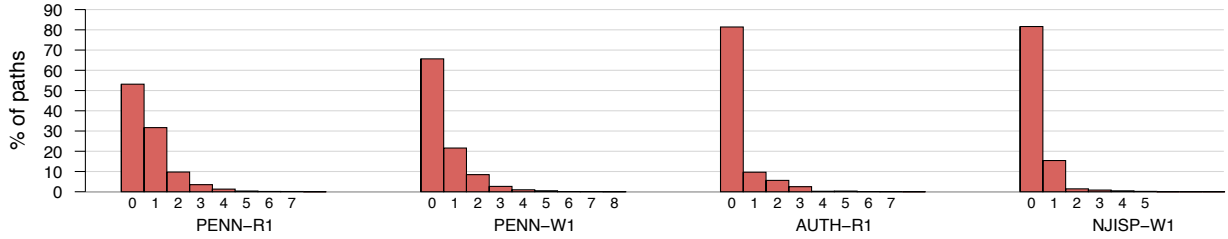
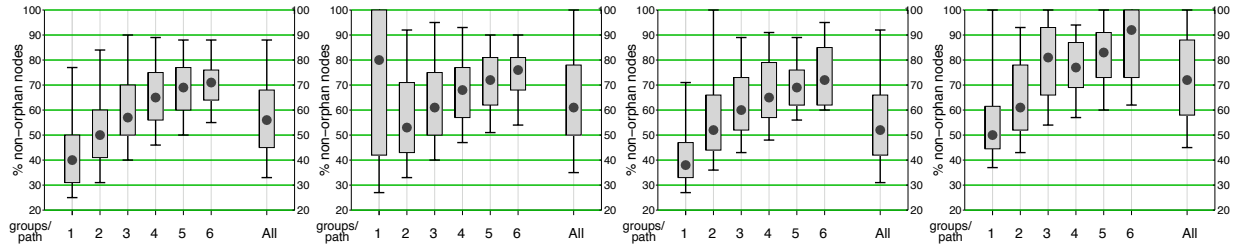


Figure 8: **Number of ASes in non-end-to-end paths that are do not appear on the end-to-end path**

ing techniques [20]. A simple way of gaining insight into this matter is to see if paths to internal network nodes tend to include ASes that are not on the original end-to-end path. In Figure 8 we show the number of ASes crossed by the non-end-to-end paths that do not appear in the end-to-end paths. We observe that in 53% (PENN-R1 data-set) to 80% (NJISP-W1) of the measured paths, all the paths remain within the ASes traversed by the end-to-end path, but the percentage of paths where paths to internal network nodes include other ASes is significant. However, the number of additional ASes appearing in non-end-to-end paths rarely exceeds one or two. From these results we can conclude that both intra-domain as well as inter-domain routing are responsible for the observed routing structure and the problems it causes to measurement techniques such as the one described in this paper.

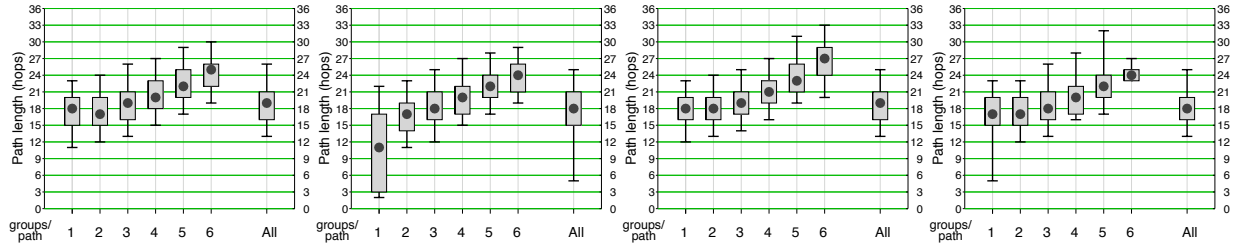
3.4 Network load and security issues

For experimental purposes, the network and router processing resources required is not expected to be a concern. Resource-conscious choice of the number of network nodes participating in the measurement



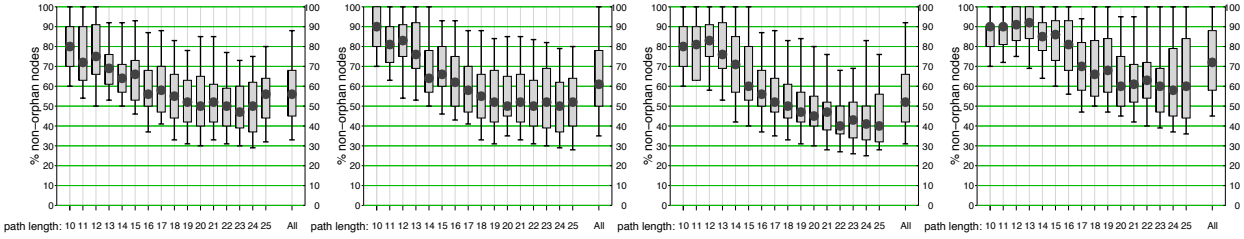
PENN-R1 PENN-W1 AUTH-R1 NJISP-W1

Figure 9: **Relationship between number of groups per path and number of non-orphan nodes**



PENN-R1 PENN-W1 AUTH-R1 NJISP-W1

Figure 10: **Relationship between number of groups per path and path length**



PENN-R1 PENN-W1 AUTH-R1 NJISP-W1

Figure 11: **Relationship between path length and number non-orphan nodes**

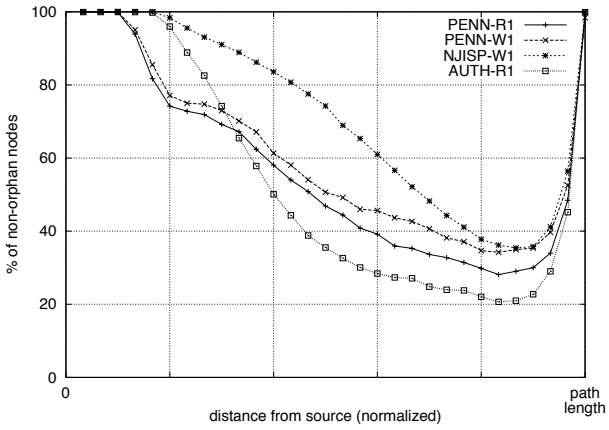


Figure 12: **Percentage of non-orphan nodes vs. distance from source**

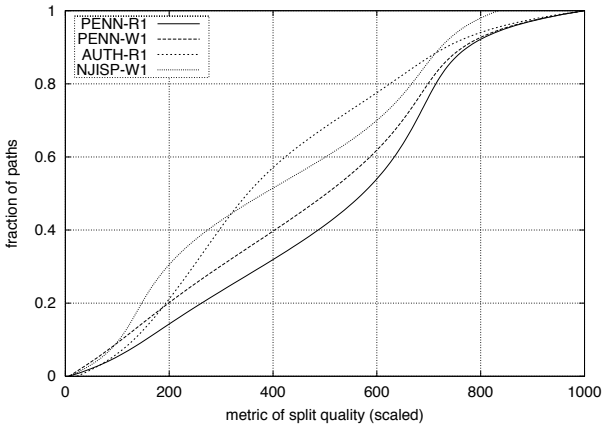


Figure 13: **CDF of the quality metric for the best 3-split for paths in the four data-sets**

is, however, important, as in many cases it could bias measurement results. However, wider use of this technique on network paths is likely to consume significant processing resources at intermediate routers and therefore needs caution.

A particular security-related issue arises from the fact that ICMP Timestamp messages are often used for malicious purposes, such as scanning and fingerprinting of remote systems for mounting appropriate attacks. The technique is hereby very likely to cause alarms when observed by firewalls or intrusion detection systems. The intensity of our experiments, as well as the need for picking random paths, which involves probing a large number of IP addresses, further adds towards a suspiciously looking traffic signature. We therefore initially chose to embed a URL in our probe packets, pointing to a Web page explaining the nature of these probes. The Web page was visited several times, and there was one routine complaint made through email to `abuse@seas.upenn.edu`. For ensuring uninterrupted experiments and to make the innocent nature of our probes more obvious, we were later advised to advertise appropriate host-names in DNS (e.g. `netmap-X-contact-telnumber-email-at.cis.upenn.edu`).

4 Internet path study

We have used the technique for investigating whether there is one or more bottlenecks in a typical network path, using a subset of 9,117 paths from the PENN-R1 data-set. This study demonstrates the value of network delay tomography without infrastructure support, while also providing first *indications* on the possible existence of paths with multiple bottlenecks.

The measurements are taken after first analyzing the structure of the path and find a balanced split in 4 slices, using the algorithm described in Section 2.3. Note that we did not incorporate the more complex analysis of overlapping slices in finding the path split or inferring which slice is congested. Once the path is appropriately segmented, we send 200 probes to each measurement node, including the destination in intervals of 1 second. The measured delays on one example path are shown graphically in Figure 14.

4.1 Congestion measure

Although the exact definition of congestion varies depending on context (*e.g.* application or user needs), the objective symptoms of congestion are clearly understood: increasing queueing delays, packet loss and throughput decrease. As transient queueing, *e.g.* light congestion is expected to be common due to the

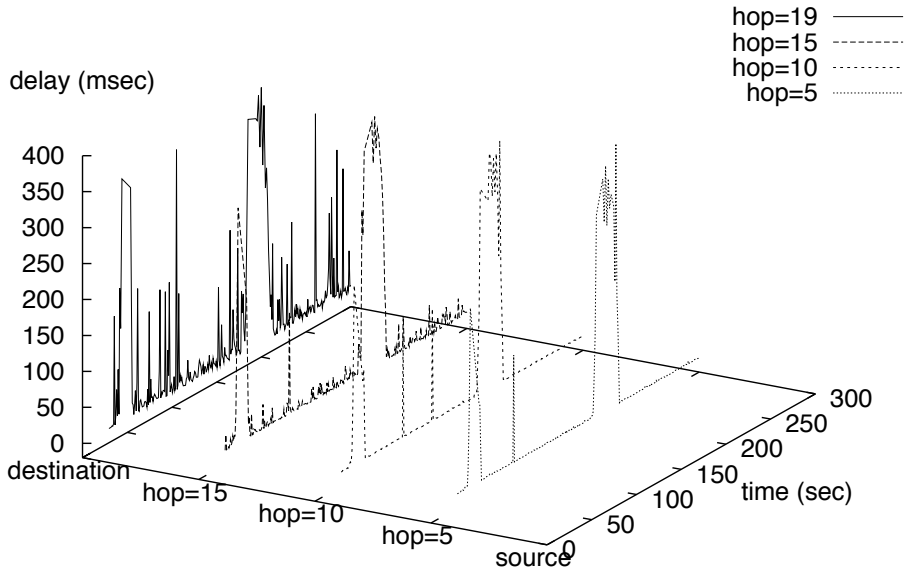


Figure 14: **Example four-slice tomography of a network path**

---

bursty nature of Internet traffic, it does not appear to be a good indication of congestion. More persistent queueing phenomena, possibly combined with observation of loss events, is more likely to be acceptable as a congestion metric. Thus, we can define a congestion metric based on queueing delay statistics, which can be obtained using our network delay tomography technique. Probe packet loss events are questionable due to the possibility of congestion in the return-path, but can still be useful as a secondary measure, as we will see below.

A simple way for characterizing a path segment as congested, based on queueing delay statistics, is to take a x-quantile statistic on the measured queueing delay for each segment. This information is obtained using the efficient self-scaling histogram algorithm described in [11]. A path segment is characterized as congested if the x-quantile is more than a threshold th. It is thus important to use appropriate values of x and th.

In this direction we use loss statistics to observe the corelation between queueing delay and loss: good values of x and th are those that capture paths with higher loss rates. The scatter-plots in Figure 15, despite being biased by return-path packet loss, allow several observations to be made. First, the higher values of x (for example, 0.9 and 0.93) tend to increase the fraction of paths that while having no loss (and thus less congestion) have a large delay quantile, which makes it harder to define an appropriate threshold th. Lower values of x (for example, 0.75 and 0.78), attract the x-quantile to lower values, making it equally hard to distinguish between transient queueing and more persistent congestion involving packet loss. For x between 0.83 and 0.88 one can observe that the areas around 100 ms and also below 20 ms tend to attract most of the paths, whereas the area in between is more sparsely covered. For the paths under 20 ms with non-zero loss, this can be attributed to loss on the return path. While one-way loss measurements could aid in providing a more accurate picture, we can relatively safely expect that x values between 0.83 and 0.88 and th values between 30 and 70 provide a reasonably good measure of congestion. We will, however, display other

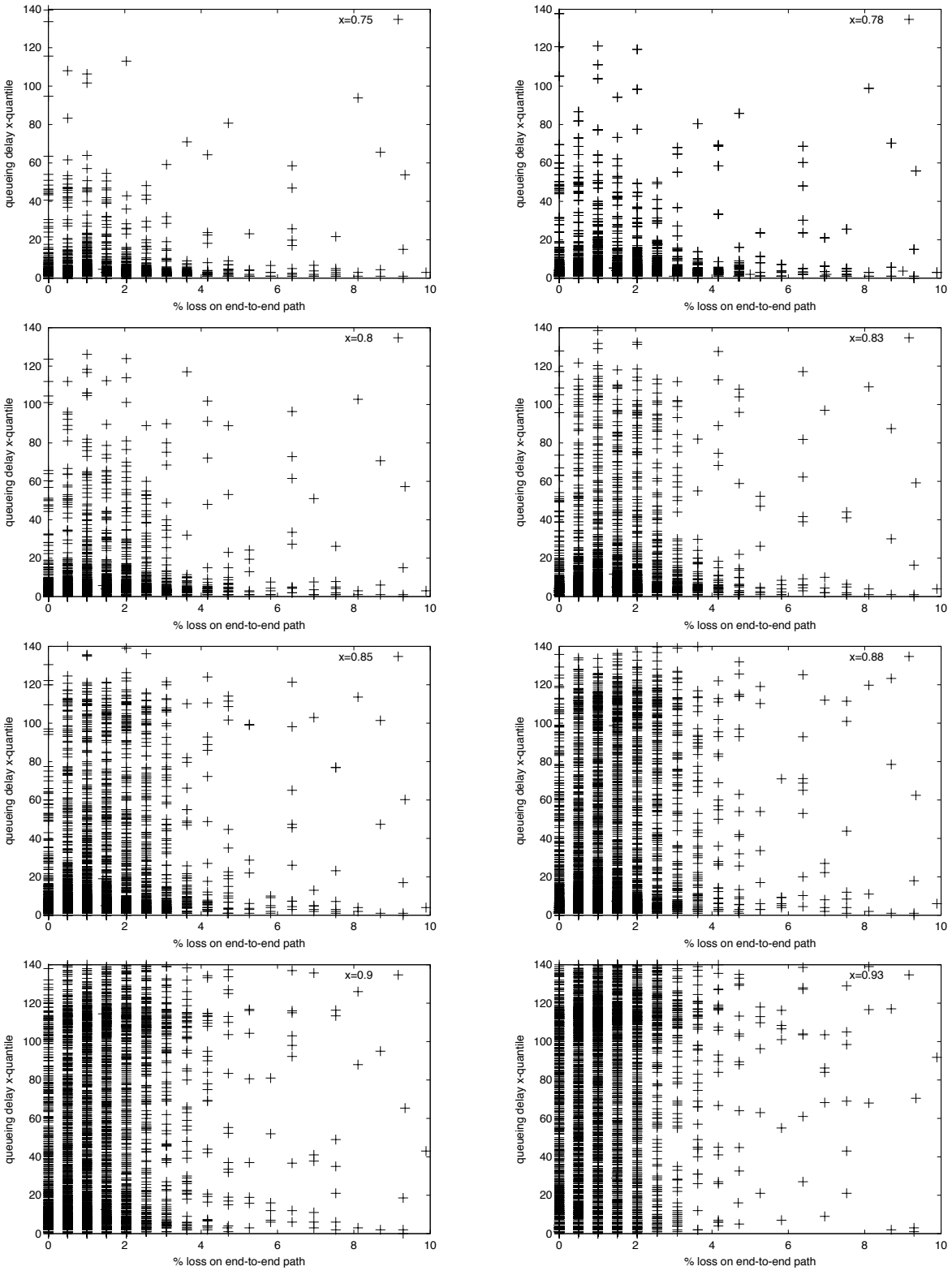


Figure 15: **Scatter-plot of path loss vs. queueing x-quantile, for different values of x (top row: 0.75, 0.78, 0.8, 0.83, bottom: 0.85, 0.88 , 0.9, 0.93)**

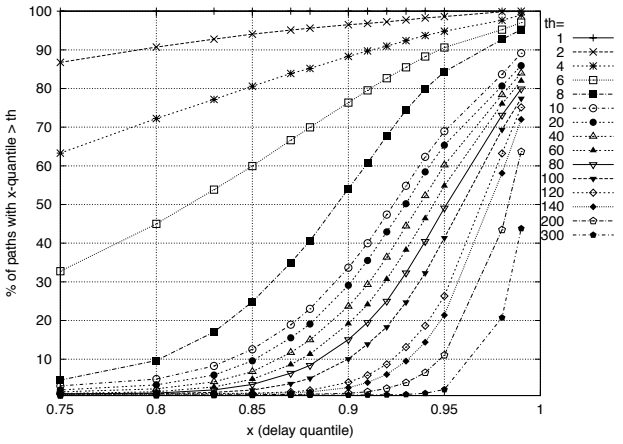


Figure 16: **Percentage of paths with x-quantile larger than th, for different values of x and th**

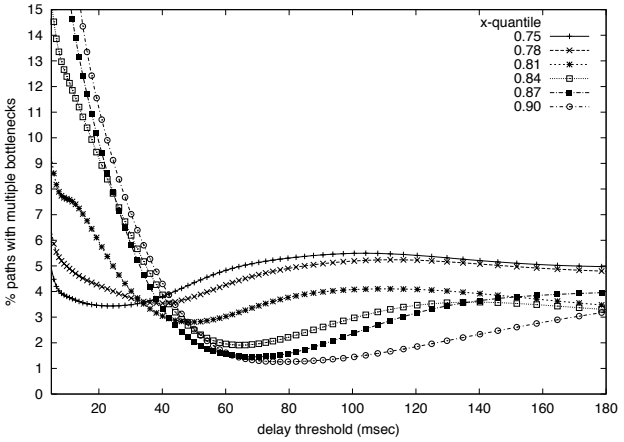


Figure 17: **Percentage of multiple-bottleneck paths for different values of x and th**

values of these parameters throughout the rest of this study, for comparison and illustration. In Figure 16 we present the percentage of paths characterized as congested, for different values of x and th. We observe that, for the suggested ranges for x and th, the percentage of paths characterized as congested is between $2-4\%$ for $x = 0.8$ and between $8-15\%$ for $x = 0.88$, for th between 20 and 80 ms.

4.2 The existence of multiple bottlenecks

The fraction of paths with multiple bottlenecks vs. the total number of paths characterized as congested, for different values of x and th are presented in Figure 17. We observe that the fraction of multiple-bottleneck paths is between 2 and 6% for most of the suggested values of x and th, as discussed in the previous section. However, it is important to note that these results are conservative in the estimated fraction of multiple-bottleneck paths and the representativeness is limited, for the following reasons. First, the paths measured were picked randomly; a mix of "real" paths might provide different results. Second, resources on each of the measured paths are not necessarily "actively" used at the time of measurement. The cases where congestion would occur at the edge of the network (e.g. dial-up or DSL links) is hereby not reflected in our results. Third, the granularity of detecting the point of congestion is limited as the experiment detects congestion per path slice and not per link, the number of slices is kept small and static to avoid interfering with network dynamics, and the technique itself is limited due to the routing structure. Therefore, the results should be interpreted as providing a qualitative *indication*, asserting the existence of multiple bottleneck paths. Based on these results we believe that multiple-bottleneck paths cannot be ignored or treated as an exception, but deserve appropriate consideration in the design and evaluation of future network control mechanisms.

5 Future work

The development of the technique and our experience with it so far have highlighted a number of issues that are interesting and deserve further work. For improving the robustness, accuracy, efficiency and value of the measurement technique, the following issues need to be addressed. First, the technique needs to address the case of analyzing the performance of overlapping path segments, exploiting the correlation of performance observed on the different segments. An appropriate path segmentation procedure is needed, to decide how to best break down the path to isolate the segments or links of interest. Secondly, a thorough

investigation on the accuracy of the ICMP Timestamp service is needed, as there may be cases where delayed ICMP processing at network routers (for instance, because a router CPU is busy updating routing tables) may introduce error in Timestamp-based measurements. Additionally, the design of an adaptive algorithm for dynamically picking appropriate measurement nodes in order to accurately isolate congested segments or links appears as a useful direction and an interesting challenge. Finally, the technique can be extended to perform combined measurement on complete network topologies instead of exclusively end-to-end paths, possibly using multiple measurement points.

With regard to the specific question of detecting multiple-bottleneck paths and quantifying the extent of this phenomenon in the Internet, work could proceed in the following directions. First, one needs to collect more data for each path, possibly with a smaller sampling interval to capture further transient phenomena. Repeating measurements over longer periods of time (hours, days and weeks) could aid in determining whether multiple bottleneck paths are persistent over time. This could mean that some users are constantly "handicapped" due to sitting behind multiple bottlenecks, which adds a significant negative element in the TCP fairness equation. Secondly, using different measurement sites could provide additional and maybe different results, as the particular case of the UPENN campus network, well connected through the high-speed academic Internet, may may provide a conservative estimate on the existence of multiple-bottleneck paths.

Additionally, as discussed in Section 4.2, the study could benefit from measuring network paths that are actively used at the time of measurement, instead of only random paths. This can be done by considering the clients of a Web server and measuring the paths from a machine on the same network as the Web server to the clients. This should probably be done with caution to avoid interfering with the performance on these paths, and using only those connections that persist over time e.g. large file transfers. One particular complication that makes this difficult is that Web servers usually write a transaction entry into the access logs after the transfer is complete, hence, watching for new entries in a web server log becomes a less attractive solution. The alternatives require effort as well as cooperation with the Web server or network owner. The Web Server software could be altered to provide a special log at the time the HTTP reply header is available. A passive monitoring system such as OC3MON [3] or FLAME [2] could be instrumented to provide a "live" stream of IP addresses (as well as transfer sizes if snooping into the HTTP reply is allowed) for considering as measurement targets. Except for Web connections, it may be possible and interesting to probe paths of other applications, such as peer-to-peer systems, remote login, or FTP. Finally, different congestion measures, such as throughput or one-way loss statistics could also aid in better analyzing path properties with respect to congestion.

6 Summary and concluding remarks

We have proposed a novel technique for network delay tomography using end-to-end measurements. The main advantage of our technique is that it uses routers as measurement nodes, instead of requiring the deployment of a new measurement infrastructure, as is the case in existing approaches. We have presented the technique, the differences from existing approaches and the complications arising from the use of routers as measurement nodes. We have investigated the effect of these issues on the design and feasibility of our technique, by studying a large number of Internet paths. Our results show that despite these complications, the technique is indeed applicable to the analysis of network paths without significant loss in effectiveness or accuracy. We then applied the new technique for a first large-scale study of Internet paths. The purpose of this experiment was to test whether multiple-bottleneck paths are an exceptional, rare or common case in the Internet. The results of this preliminary study are mostly qualitative and indicative. However, the results show that, while neither dominant nor exceptional, multiple bottleneck paths do exist to a level that merits further investigation as well as careful consideration in the design and evaluation of congestion control

mechanisms.

Both the development of practical, easy-to-use measurement techniques as well as the multiple-bottleneck issue are important questions in networking research. As the Internet architecture was not designed with any of the existing measurement techniques in mind, it is surprising how much useful information one can extract from the existing primitives, reflecting on the elegance of the Internet architecture. On the other hand, the difficulties encountered in developing these techniques and the limitations inherited by the measurement techniques, besides pointing to architectural problems, have not yet pointed to a specific set of primitives that the network infrastructure should provide. Thus, the development of constrained yet instantly usable techniques, such as the one described in this paper, appears as the most likely way forward in the design of measurement techniques and the evaluation of network control mechanisms for the Internet.

Acknowledgments

This work was supported by the DoD University Research Initiative (URI) program administered by the Office of Naval Research under Grant N00014-01-1-0795. We thank John C. Parker, Dave Millar and Charles Buchholtz for providing us with real IP addresses from the UPENN SEAS Web Server, and C. Ricudis for providing access to the European site for one of the datasets. The authors would like to thank Jonathan M. Smith and the members of the Distributed Systems Laboratory for useful comments and suggestions.

References

[1] M. Adler, T. Bu, R. K. Sitaraman, and D. Towsley. Tree layout for internal network characterizations in multicast networks. In *3rd International Workshop on Networked Group Communication (NGC 2001)*, November 2001.

[2] K. G. Anagnostakis, S. Ioannidis, S. Miltchev, J. Ioannidis, M. B. Greenwald, and J. M. Smith. Efficient packet monitoring for network management. In *Proceedings of the 8th IEEE/IFIP Network Operations and Management Symposium (NOMS)*, April 2002.

[3] J. Apisdorf, k claffy, K. Thompson, and R. Wilder. OC3MON: Flexible, Affordable, High Performance Statistics Collection. In *Proceedings of the 1996 LISA X Conference*, 1996.

[4] A. A. Awadallah and C. Rai. TCP-BFA: Buffer fill avoidance. In *IFIP High Performance Networking Conference*, September 1998.

[5] R. Braden. Requirements for internet hosts – communication layers. RFC 1122/STD 3, http://www.rfc-editor.org/, October 1989.

[6] L. S. Brakmo, S. W. O'Malley, and L. L. Peterson. TCP Vegas: New Techniques for Congestion Detection and Avoidance. In *Proc. 1994 ACM SIGCOMM Conference*, August 1994.

[7] R. Caceres, N. G. Duffield, J. Horowitz, and D. Towsley. Multicast-based inference of network-internal loss characteristics. *IEEE Transactions on Information Theory*, 45:2462–2480, 1999.

[8] N. G. Duffield, J. Horowitz, F. L. Presti, and D. Towsley. Network delay tomography from end-to-end unicast measurements. In *Proc. of the 2001 International Workshop on Digital Communications 2001 - Evolutionary Trends of the Internet*, September 2001.

[9] N. G. Duffield and F. L. Presti. Multicast inference of packet delay variance at interior network links. In *Proceedings of IEEE INFOCOM 2000*, April 2000.

[10] N. G. Duffield, F. L. Presti, V. Paxson, and D. Towsley. Inferring link loss using striped unicast probes. In *Proceedings of IEEE INFOCOM 2001*, April 2001.

[11] M. Greenwald. Practical algorithms for self scaling histograms or better than average data collection. *Performance Evaluation*, 27&28:19–40, 1996.

[12] V. Jacobson. Congestion Avoidance and Control. In *SIGCOMM Symposium on Communications Architectures and Protocols*. 1988.

[13] S. Keshav. *Congestion Control in Computer Networks* . PhD thesis, University of California Berkeley, September 1991.

[14] M. J. Luckie, A. J. McGregor, and H.-W. Braun. Towards improving packet probing techniques. In *ACM SIGCOMM Internet Measurement Workshop 2001*, November 2001.

[15] V. Paxson. End-to-end routing behavior in the internet. In *Proc. 1996 ACM SIGCOMM Conference*, August 1996.

[16] V. Paxson. On calibrating measurements of packet transit times. In *Proc. 1998 ACM SIGMETRICS Conference*, 1998.

[17] J. Postel. Internet control message protocol. RFC 792, http://www.rfc-editor.org/, September 1981.

[18] Y. Rekhter and e. T. Li. A Border Gateway Protocol 4 (BGP-4). RFC1771, http://www.rfc-editor.org/, March 1995.

[19] B. Schwartz, A. Jackson, T. Strayer, W. Zhou, R. Rockwell, and C. Partridge. Smart packets: Applying active networks to network management. *ACM Transactions on Computer Systems*, 18(1):67–88, February 2000.

[20] A. Shaikh, J. Rexford, and K. G. Shin. Load-sensitive routing of long-lived ip flows. In *Proc. 1999 ACM SIGCOMM Conference*, September 1999.

[21] Y. Vardi. Network tomography: estimating source-destination traffic intensities from link data. *Journal of the American Stat. Association*, pages 365–377, 1996.