Range Tomography: Combining the Practicality of Boolean Tomography with the Resolution of Analog Tomography

Sajjad Zarifzadeh
szarifzadeh@gmail.com
Georgia Tech

Madhwaraj Gowdagere
madhwarajgk@gmail.com
Georgia Tech

Constantine Dovrolis
constantine@gatech.edu
Georgia Tech

ABSTRACT

The objective of early network tomography approaches was to produce a point estimate for the performance of each network link (Analog tomography). When it became clear that the previous approach is error-prone in practice, research shifted to Boolean tomography where each link is estimated as either "good" or "bad". The Boolean approach is more practical but its resolution is too coarse. We propose a new tomography framework that combines the best of both worlds: we still distinguish between good and bad links (for practicality reasons) but we also infer a range estimate for the performance of each bad link. We apply the *Range tomography* framework in two path performance metric functions (Min and Sum) and propose an efficient algorithm for each problem. Together with simulations, we have also applied Range tomography in three operational networks allowing us to identify the location of bad links and to estimate their performance during congestion episodes. We also compare the proposed method with existing Analog and Boolean tomography algorithms.

Categories and Subject Descriptors

C.2.3 [**Computer-communication Networks**]: Network Operations—*Network management, Network monitoring*; C.4 [**Performance of Systems**]: Measurement techniques

Keywords

Network tomography, localization, performance metric

1. INTRODUCTION

The objective of network tomography is to infer the internal characteristics of a network (e.g., link delay and loss rate, topology) from end-to-end path measurements. As a research area, network tomography is both interesting and significant. It is interesting because the underlying estimation problem is typically under-constrained (more unknowns than equations) and so its solution requires creative and domain-specific additional constraints and objectives. Further, network tomography combines the art of active probing with the science of statistical inference. Network tomography is also significant in practice because it allows users and network administrators to monitor and troubleshoot a set of network paths of interest without having direct access to every router and switch in those paths. This is particularly important in the context of interdomain paths, i.e., paths that traverse more than one network provider. Additionally, as IP-based networks move towards performance-based Service Level Agreements (SLAs) tomographic methods will become even more valuable. In fact, the motivation for this work resulted from an actual system that is used to monitor in real-time the performance of network paths that interconnect a large number of scientific research labs around the world using end-to-end measurements.

The first efforts in network tomography, about 15 years ago, focused on the most challenging instance of the problem: estimate the actual performance (e.g., loss rate or delay variation) of every link in the monitored network [1, 2]. We refer to those tomographic methods as *Analog* because they provide a real-valued performance estimate for each link. Analog tomography is a very challenging problem for several reasons, some of which are related to the underlying path measurements. For instance, those early methods assume that if two paths go through a single and shared lossy link, their measured path loss rates will be equal. This may be true if the measurements last for a very long period, assuming that the network load is stationary and the routes do not change. In practice, however, network conditions change significantly with time, and so the path measurements must be performed in short time intervals (e.g., one minute) before they become stale [3]. To make things worse, the probing frequency cannot be too high because the path measurements can then affect the underlying network performance [4]. So, Analog tomography methods need to work with noisy path estimates that result from short-term active measurements.

In 2003, Duffield proposed a paradigm shift in network tomography [5]. Instead of trying to estimate the actual performance of a link, it is more practical to only infer whether the link's performance is "good" versus "bad". This approach is referred to as *Boolean tomography* and it has also been followed by other researchers [6, 8]. Here, path measurements become much simpler (just detect congested paths) and so they can take place in shorter time periods and with fewer probes. The problem, of course, is that the result of the tomographic process has low resolution: just a single bit for

IMC'12, November 14–16, 2012, Boston, Massachusetts, USA.

each link. This may be sufficient if the objective is to simply identify severe congestion. In practice, however, the congestion level of network links varies widely and even though an application or operator may be satisfied with a loss rate of 0.1% or a queueing delay of 50msec, another may find these congestion levels unacceptable.

In this work, we propose a new approach in network tomography that attempts to combine the higher resolution of Analog tomography with the practicality of Boolean tomography. The path measurements are still classified as good or bad, to exploit the fact that most network paths typically operate without congestion [15]. This allows us to quickly remove a large number of links from the problem, as long as they only appear in good paths. For the remaining paths, we aim not only to identify bad links, but also to estimate a *range* (interval) for their actual performance. The width of the resulting range depends on the variability in the underlying path measurements: more accurate path measurements will result in narrower range estimates. We refer to the proposed framework as *Range Tomography*.

We apply the Range tomography framework in two path performance formulations: a *Min function* in which a path's performance is determined by the link with the minimum performance, and a *Sum function* in which a path's performance is determined by the sum of its links' performance values. Min is more appropriate for metrics such as available bandwidth, while Sum can be used for delay and (as an approximation) for loss rate. For each formulation we propose an efficient algorithm to identify bad links and estimate their performance range. The algorithms are first evaluated with simulations in three real topologies. Simulations are not ideal, but they allow us to examine the false positives, false negatives and accuracy of Range tomography under controllable and repeatable experiments. We performed these simulations under both random and bursty loss processes, showing the significant impact of the latter in network tomography methods. Further, we compare the proposed algorithms with "canonical" methods from Analog and Boolean tomography.

We have also applied the proposed methods in practice to detect and localize congestion events in three operational large networks, where the number of monitored paths varies between 70-3600. Even though we cannot use those real-world experiments for validation (we do not have access to routers and switches in those networks), they show some interesting points. First, in practice, there is rarely more than one bad link in the monitored paths and so the false positive/negative probability is close to zero. Second, the actual congestion level of those bad links can vary widely, and so it is important to also estimate a congestion range instead of a single bit.

The rest of the paper is structured as follows. In Section 2, we describe the Range Tomography problem. In Sections 3 and 4, we consider the Min and Sum formulations, respectively. In Section 5, we evaluate tomography methods using simulations. We present a short summary of the experimental results in Section 6. Section 7 discusses the related work and we conclude in Section 8.

2. RANGE TOMOGRAPHY

In this section, we motivate, state and explain the Range Tomography problem.

2.1 Problem statement

We are given a set of N measurement points or hosts, referred to as *sensors* $S = \{s_1, ..., s_N\}$. Each sensor measures the end-to-end path performance, with respect to a certain metric such as loss rate, one-way delay or available bandwidth, to all other sensors. We are interested to localize performance problems, with respect to the corresponding metric, in the set Π of $N \times (N-1)$ paths between sensors. The forwarding paths between sensors can be "mapped" at the IP-layer using traceroute-like tools (such as Paris traceroute [22]). Each path is represented by a sequence of links and each link is represented by the IP address of the corresponding router or host interface. Let $G = (V, E)$ be the directed graph constructed from the union of all paths between sensors.

Suppose that $p_{i,j}$ denotes the path from sensor s_i to sensor s_j, and $m_{i,j}$ is the *actual* performance of $p_{i,j}$ for a given performance metric in the time interval of interest. We assume that there is a function f that expresses the performance of a path based on the performance of its constituent links,

$$m_{i,j} = f(\{x_l | \forall l \in p_{i,j}\}) \quad (1)$$

where x_l is the *actual* performance of link l. We refer to f as the *path metric function*. For instance, in the case of available bandwidth, the performance of the end-to-end path is determined by the link with the minimum available bandwidth. Thus, the function f for available bandwidth is based on the *MIN* operator. In the case of delay, jitter and (as an approximation) loss rate, the path performance can be estimated by the sum of link performances. Hence, the function f would be the *SUM* operator for those metrics. In network tomography, we attempt to invert function f and estimate the performance of some links given the measured performance of a set of paths that traverse those links. As in other instances of tomography, this "inversion problem" is typically ill-defined and it can be solved only if we impose additional constraints, optimization objectives and assumptions [10].

In practice, the performance of a network path must be measured with low frequency probing (i.e., the probing traffic load should be low compared to the capacity of the path), and in short time intervals (due to dynamic network conditions). Hence, the measured path performance can be significantly different than the actual path performance. In other words, the measured performance of two paths sharing the same bottleneck link can be different even though their actual performance is the same. Let $\tilde{m}_{i,j}$ be the *measured* performance of path $p_{i,j}$ in the time interval of interest. Because of the previous limitations of the measurement process, we typically have that $m_{i,j} \neq \tilde{m}_{i,j}$, while the difference between the two metrics can vary significantly over different paths. To illustrate this issue, consider the following simple example. Assume that the packet drops at a link l follow a Bernoulli random process with probability $x_l = 0.01$, while the measurement process sends $n = 1000$ probing packets on that link. The variance of the measured packet losses is $n\,x_l\,(1 - x_l)$ and so the standard deviation ($\approx \sqrt{10}$) is significant compared to the expected number of packet losses at link l (10 packets).

In the following, we introduce the Range Tomography framework as a way to deal with the previous uncertainty in path measurements. First, we inherit from Boolean tomography the notion that paths and links can be distinguished

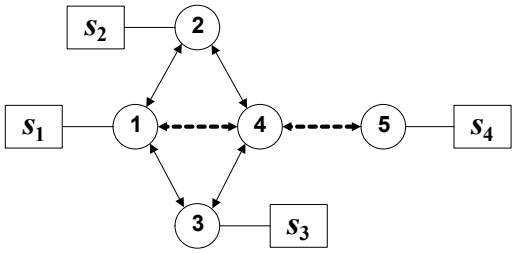


Figure 1: A simple network example with four sensors. The dashed lines indicate lossy links.

as "good" versus "bad". Formally, a path $p_{i,j}$ is called *bad* if its measured performance $\tilde{m}_{i,j}$ is worse than a threshold δ (δ may be zero, as in the case of loss rate for example). Similarly, a link is *bad* if its performance metric is worse than δ, otherwise it is *good*. After we infer the bad links from path measurements, we also estimate the performance of every bad link with a "range estimate" for its metric. In other words, we start with path point measurements and end up with link range estimates. To this end, we first identify and group paths that are likely affected by the same bad link and then use the measured performance of these paths to determine a narrow *performance range* for that link.

Definition 1. Two paths $p_{i,j}$ and $p_{k,l}$ are *α-similar* (represented by $p_{i,j} \parallel p_{k,l}$), if their relative performance difference is within a parameter α,

$$\frac{|\tilde{m}_{i,j} - \tilde{m}_{k,l}|}{min(\tilde{m}_{i,j}, \tilde{m}_{k,l})} \leq \alpha \tag{2}$$

where α is a positive parameter that quantifies the accuracy of the measurement process (we also say that the measurements $\tilde{m}_{i,j}$ and $\tilde{m}_{k,l}$ are *α-similar*). α is related to the measurement process (e.g., size of probing packets, probing frequency, probing method) and to the characteristics of the estimated path performance process. The parameter α can be estimated experimentally, measuring paths that are known to be limited by the same bottleneck link. The relative difference of the resulting path measurements should be less than the chosen value of α. In the rest of this section, we assume that α is given as an input to the network tomography process. In Section 5, we discuss how we estimated α in our experiments.

Let us consider the example of Figure 1. $p_{i,j}$ is the shortest path from s_i to s_j. End-to-end measurements indicate that the paths $p_{1,4}, p_{2,4}, p_{3,4}$ are lossy; their measured loss rates are 15%, 5% and 7%, respectively. All remaining paths are good. Boolean tomography would choose the link from node-4 to node-5, represented by (4,5), as the only bad link because it appears on all lossy paths. However, choosing a single lossy link cannot explain why these paths have so different loss rates. If $\alpha = 0.5$, paths $p_{2,4}$ and $p_{3,4}$ are α-similar. Then, one plausible solution is that link (4,5) (shared by the two α-similar paths) has a loss rate between 5% and 7% while link (1,4) on $p_{2,3}$ has a loss rate between 8% and 10%. In general, the notion of α-similar paths improves the network tomography process in two ways: 1) As opposed to Analog tomography, we can avoid the incorrect detection of multiple bad links in paths that show different performance even though they are affected by the same bad link. 2) Unlike Boolean tomography, when the paths sharing the same bad link are not α-similar, we can conclude that there are additional bad links (as in the previous example).

Let $\tilde{x}_l = [s_l..e_l]$ denote the performance range assigned to link l (i.e., the actual performance of l is estimated to be between s_l and e_l). $|\tilde{x}_l|$ denotes the width of range $\tilde{x}_l$. We define next how to examine whether the assigned link performance ranges are consistent with the path performance measurements.

Definition 2. The performance ranges assigned to bad links on a path $p_{i,j}$ are *consistent* with the measured performance of $p_{i,j}$ if there is at least one value in each of these ranges that would satisfy the path performance constraint in (1). We denote the consistency relation as:

$$\begin{aligned} &\tilde{m}_{i,j} \simeq f(\{\tilde{x}_l | \forall l \in p_{i,j}\}) \text{ if:} \\ &\forall l \in p_{i,j}, \exists t_l \in \tilde{x}_l : \tilde{m}_{i,j} = f(\{t_l | \forall l \in p_{i,j}\}) \end{aligned} \tag{3}$$

To illustrate the previous definition, consider the following example. Suppose that the measured loss rate of a path $p_{i,j}$ is 1% and there are two bad links l_1 and l_2 in that path. The performance range of both links is estimated as [1%-2%]. Obviously, these range estimates are not consistent with the path measurement. On the contrary, if the performance ranges are both [0.4%-0.6%], then they are consistent with the measured path loss rate. In this case, we also say that the performance ranges assigned to l_1 and l_2 *justify* the bad path $p_{i,j}$ ($p_{i,j}$ is then called *justified* path).

Before stating the Range Tomography problem more formally, we need to make the following assumptions:

(1) There are no spatial correlations between different bad links.[1]

(2) The routing paths between sensors remain constant during the measurement process.[2]

(3) A path is bad, if and only if, there is at least one bad link in that path.

Given the directed graph G and a measurement for each path in Π, the Range Tomography problem can be stated as follows:

Range Tomography (RT) problem: *Infer all bad links in G and assign a performance range to each bad link so that:*

1. *The number of bad links is minimized.*
2. *The assigned performance ranges to bad links are consistent with the path performance measurements, $\forall p_{i,j} \in \Pi : \tilde{m}_{i,j} \simeq f(\{\tilde{x}_l | \forall l \in p_{i,j}\})$.*

In practice, it is not common to have multiple bad links at the same time (this is also observed in our experimental results in Section 6). So, as it is common in Boolean tomography, we consider an optimization objective that aims to minimize the number of detected bad links. Without

[1]To the extent of our knowledge, there is no prior evidence that such correlations are common or significant in practice. We make this assumption so that we can calculate the performance of a path from the performance of its constituent links through a simple function.

[2]If the network deploys multipath routing, we need to make sure that both the Paris-traceroute packets and the probing packets sent from one sensor to another use the same port numbers so that they follow the same forwarding path.

this objective, the solution of the RT problem can result in several false positives. The complexity of the RT problem depends on the metric function f. In the following two sections, we consider two specific instances of the function f that can be used to capture different performance metrics.

3. MIN METRIC FUNCTION

We first consider a function that determines the performance of path $p_{i,j}$ as the minimum performance among all links in that path,

$$m_{i,j} = min_{\forall l \in p_{i,j}}\{x_l\} \quad (4)$$

We refer to this instance of the RT problem as *RT-Min*. The previous function can be used for performance metrics such as capacity or available bandwidth, because these metrics are mostly determined by a single "bottleneck" link. In the rest of this section, we will refer to available bandwidth as the performance metric of interest (even though the proposed algorithm is applicable to any metric that satisfies the previous path performance constraint). We say a path (or a link) is *bad* if its available bandwidth is less than a threshold δ.

3.1 Solution and analysis for RT-Min

Definition 3. A bad link l is *detectable* if there is at least one path $p_{i,j}$ traversing l such that (I) there is no other bad link in $p_{i,j}$ **OR** (II) other bad links in $p_{i,j}$ have higher available bandwidth than link l.

The above condition implies that for every bad link l, there should be at least one path $p_{i,j}$ such that l is the bottleneck link of that path.

Definition 4. A good link l is *removable*, if there is at least one good path traversing l.

Lemma 1. Assume that there is no measurement error, i.e. $\tilde{m}_{i,j} = m_{i,j}, \forall p_{i,j} \in \Pi$. In that ideal case, the available bandwidth of every bad link can be determined if (I) all good links are removable, **AND** (II) all bad links are detectable.

The proof is straightforward and it is omitted. If both conditions of Lemma-1 hold, we can use the following method (referred to as *Min-Ideal*) to estimate the available bandwidth of the bottleneck link for every bad path. First, "mark" all links in good paths as good; the rest of the links are "unmarked". Then, select the bad path p_b with the highest available bandwidth among the remaining paths and assign the available bandwidth of p_b to all unmarked links on p_b (p_b is then labeled as *justified* and all links in p_b are marked). We repeat the previous step until there is no unjustified path in Π.

However, the two conditions of Lemma-1 (good links are removable and bad links are detectable) may not be true in practice. In addition, the measurement process is always error-prone. Therefore, we need to consider an algorithm that can *infer* bad links, even when the assumptions of Lemma-1 are not met.

Based on the complexity analysis of the Boolean tomography problem [8], it is easy to see that the RT-Min problem is also NP-hard[3]. So, we need to approximate solution of RT-Min with heuristics. Our solution to this problem, referred to as *Min-Tomo*, is based on the following two principles:
I) As in the case of *Min-Ideal*, the *Min-Tomo* algorithm runs iteratively, and in each iteration it considers the bad path p_b that has the highest available bandwidth among all remaining paths. Paths that are α-similar with p_b are then grouped into the set Ω.
II) Since good links are not necessarily removable, it is possible that we consider a link as bad even though it is good, and vice versa. To reduce this classification error, *Min-Tomo* greedily selects the link l that is traversed by the largest number of bad paths in Ω and marks it as bad. The intuition is that it is unlikely that a large number of bad paths traverse a good link, without *any* good path traversing that link.[4] Additionally, this greedy heuristic aims to minimize the number of links identified as bad.

The pseudo-code of *Min-Tomo* is presented in Algorithm 1. We first mark all links on every good path as good (line 3). Then, we consider the path with the highest available bandwidth among all bad paths. Let p_b be such a path. Instead of considering only path p_b, we consider the set of paths that are α-similar with p_b (this set is referred to by Ω in line 7). We then select link l_m as the link that is shared most between paths in Ω. In the case of a tie, we select the link which is traversed by most unjustified paths (lines 17 and 18). We also check in line 11 that there is at least one path that is α-similar with p_b and traverses l_m. Taking $\bar{r}$ as the average available bandwidth of paths going over l_m (line 19), the performance range of l_m is determined in line 20 so that every value in the range $\tilde{x}_{l_m}$ is α-similar with $\bar{r}$. We label all paths traversing l_m as *justified* (line 21), if their available bandwidth falls in the performance range $\tilde{x}_{l_m}$. We repeat this process until there is no unjustified path.

The estimated ranges produced by *Min-Tomo* satisfy the consistency requirement for the following reason. Let $h(l_m)$ be the highest available bandwidth path traversing l_m. Suppose there are several paths that are α-similar with $h(l_m)$ and traverse l_m. Let $\Phi(l_m)$ be the set of these paths. In the RT-Min problem, the range $\tilde{x}_{l_m}$ assigned to a link l_m is consistent with the measured performance of path $p_{i,j} \in \Phi(l_m)$ if the measurement $\tilde{m}_{i,j}$ belongs in the range $\tilde{x}_{l_m}$. It can be shown that this is true for every path in $\Phi(l_m)$ if we select the range $\tilde{x}_{l_m}$ as shown in line 20.

The run-time complexity of *Min-Tomo* is $O(|\Pi|^2 + |\Pi|\,|E|)$. The running time for a network with about 1000 bad paths and 4600 links is less than 5.5msec on a workstation with a 2.6GHz Intel i5 processor.

4. SUM METRIC FUNCTION

In this section we consider the following path metric function,

$$m_{i,j} = \sum_{\forall l \in p_{i,j}} x_l \quad (5)$$

The corresponding range tomography problem is referred to as *RT-Sum*. The Sum metric function can capture performance metrics such as delay and jitter. Under the previous spatial independence assumption, the path loss rate is given by

$$m_{i,j} = 1 - \prod_{\forall l \in p_{i,j}} (1 - x_l) \quad (6)$$

[3]To prove this we have to consider the simplified case where the available bandwidth of all bad links is the same. In that case, the RT-Min problem is the same with the Boolean tomography problem, which is NP-hard [8].

[4]The analysis of three real topologies, described in Section 5, confirms this intuition.

Algorithm 1 Min-Tomo

Require: Set of all links E
Require: Set of all paths Π
Require: The measured available bandwidth of every path in Π, $\tilde{m}_{i,j}, \forall p_{i,j} \in \Pi$
1: Initialize $R = \Pi$ {set of unjustified paths}
2: Initialize $U = E$ {set of unmarked links}
3: Remove from R and U all good paths and all links on good paths, respectively.
4: $h(l) = argmax_{\forall p_{i,j} \in R}\{\tilde{m}_{i,j}\}, \forall l \in U$ {path with highest available bandwidth going over l}
5: **while** $R \neq \varnothing$ **and** $U \neq \varnothing$ **do**
6: $p_b = argmax_{\forall p_{i,j} \in R}\{\tilde{m}_{i,j}\}$ {path with highest available bandwidth in R}
7: $\Omega = \{p_{i,j} | p_{i,j} \in R, p_{i,j} \parallel p_b\}$ {set of paths which are α-similar with p_b}
8: $num(l) = |\{p_{i,j} | l \in p_{i,j}, p_{i,j} \in R\}|$ {number of unjustified paths going over l}
9: **for** each link l in U **do**
10: $C(l)$ = set of paths in Ω containing l
11: **if** $h(l) \parallel p_b$ **then**
12: $score(l) = |C(l)|$ {number of paths in $C(l)$}
13: **else**
14: $score(l) = 0$
15: **end if**
16: **end for**
17: $L_m = argmax_{l \in U} score(l)$ {set of links with the maximum score}
18: $l_m = argmax_{l \in L_m} num(l)$ {link with the maximum score and maximum number of unjustified paths going over it}
19: $\bar{r} = avg\{\tilde{m}_{i,j} | p_{i,j} \in \Omega, l_m \in p_{i,j}\}$ {average available bandwidth of paths in Ω going over l_m }
20: $\tilde{x}_{l_m} = [\bar{r}(\frac{1}{1+\alpha}), \bar{r}(1+\alpha)]$ {performance range of link l_m }
21: $R = R - \{p_{i,j} | l_m \in p_{i,j}, p_{i,j} \in R, \tilde{m}_{i,j} \in x_{l_m}\}$ {Now, some paths going over l_m are justified}
22: $U = U - \{l_m\}$
23: **end while**
24: **return** $\tilde{x}_l, \forall l \in E$

However, if the link loss rates are low and there are few lossy links in each path, the two formulas give similar results (e.g., in the case of three lossy links each with 1% loss rate, the actual loss rate is about 2.97% even though the Sum approximation gives 3%). In the rest of this section, we consider loss rate as the performance metric of interest, and we refer to bad links as "lossy" (i.e., δ=0).

4.1 Solution and analysis for RT-Sum

We first present a necessary condition for the identification of lossy links assuming error-free measurements.

Definition 5. A lossy link l is *detectable-1* if there is at least one path $p_{i,j}$ traversing l such that there is no other lossy link in $p_{i,j}$. A lossy link l is *detectable-n* ($n > 1$) if it is either *detectable-(n-1)* **OR** there is at least one path $p_{i,j}$ traversing l such that all other lossy links on $p_{i,j}$ are *detectable-(n-1)*.

Lemma 2. Assume there is no measurement error, i.e., $\tilde{m}_{i,j} = m_{i,j}, \forall p_{i,j} \in \Pi$. The loss rate of every lossy link can be determined if (I) all good links are removable **AND** (II) all lossy links are *detectable-n* ($n \geq 1$).

Proof: Under the previous conditions, we can use the following method (referred to as *Sum-Ideal*) to estimate the loss rate of lossy links. Let U be the set of links identified as lossy. Based on the recursive definition of *detectable-n* links, there should exist at least one path that traverses only one link in U. Suppose $p_{i,j}$ is such a path, traversing link $l \in U$. Hence, the loss rate of l is the same as the loss rate of $p_{i,j}$ ($x_l = m_{i,j}$). We remove l from U and then subtract the loss rate of l from the loss rate of every path traversing l. In the next iteration, there should again exist at least one path that traverses only one link of U; otherwise, the remaining links in U would not be *detectable-n* ($n \geq 1$). We repeat this process until the set U is empty. In each iteration, one link is removed from U, and so this algorithm is guaranteed to determine the loss rate of every bad link.

However, the two conditions of Lemma-2 may not be true in practice. We conducted a simple experiment to evaluate how often those two conditions are violated in three real topologies (described in Section 5). We randomly select a number of lossy links and assign a loss rate to them from a uniform distribution. The percentage of lossy links that are *not detectable-n* is quite low in all three topologies (less than 2% on average when up to 10% of links in the ESNet topology are lossy). But, the percentage of good links that are *not* removable is higher (around 7% when up to 10% of links in ESNet are lossy). In other words, it is possible that all paths traversing a good link are bad. Therefore, we need to consider an algorithm that can *infer* lossy links, even when the conditions of Lemma-2 are not met.

The RT-Sum problem is also NP-hard. To prove this, it is enough to consider the case where all lossy paths have the same loss rate. The solution in that case is the same with the solution of the Boolean tomography problem. We approximate the solution of the RT-Sum problem with a heuristic, referred to as *Sum-Tomo*, which is based on the following three principles:
I) In each iteration, we only consider the set of paths that are α-similar with the path that has the lowest loss rate. This set is referred to as Ω.
II) Similar to *Min-Tomo*, we greedily choose the link l that is traversed by the maximum number of lossy paths in Ω.
III) We subtract the assigned loss rate of a detected bad link from all paths traversing that link.

The pseudo-code of *Sum-Tomo* is presented in Algorithm 2. The input arguments are the same with *Min-Tomo* in Algorithm 1 and they are omitted. After removing good paths and their links (in line 3), we consider the path p_b with the lowest loss rate; let Ω be the set of paths that are α-similar with p_b (line 6 and 7). In line 14, we greedily choose link l_m as the link that is traversed by most paths in Ω (if there are multiple such links, the link that is traversed by most unjustified paths is chosen). We calculate $\bar{r}$ as the average loss rate of paths in Ω that traverse l_m (line 15), and so the performance range of l_m (line 16) is such that every point in $\tilde{x}_{l_m}$ is α-similar with $\bar{r}$. We then mark all paths (traversing l_m) as *justified* (line 17), if their loss rate is within the performance range of l_m. In that case, we also subtract the average loss rate $\bar{r}$ from the loss rate of all paths that traverse l_m (line 18).

The run-time complexity of *Sum-Tomo* is also $O(|\Pi|^2 + |\Pi|\,|E|)$. The *Sum-Tomo* algorithm can only approximate

Algorithm 2 Sum-Tomo

Initialize $R = \Pi$ {set of unjustified paths}
Initialize $U = E$ {set of unmarked links}
Remove from R and U all good paths and all links on good paths, respectively.
Initialize $r_{i,j} = \tilde{m}_{i,j}, \forall p_{i,j} \in R$ {the loss rate of $p_{i,j}$ which has not been justified yet}
while $R \neq \varnothing$ **and** $U \neq \varnothing$ **do**
  $p_b = argmin_{\forall p_{i,j} \in R}\{r_{i,j}\}$ {path with lowest loss rate in R}
  $\Omega = \{p_{i,j} | p_{i,j} \in R, p_{i,j} \parallel p_b\}$ {set of paths which are α-similar with p_b, using $r_{i,j}$ instead of $m_{i,j}$}
  $num(l) = |\{p_{i,j} | l \in p_{i,j}, p_{i,j} \in R\}|$ {number of lossy paths going over l}
  for each link l in U **do**
    $C(l) =$ set of paths in Ω containing l
    $score(l) = |C(l)|$ {number of paths in $C(l)$}
  end for
  $L_m = argmax_{l \in U} score(l)$ {set of links with the maximum score}
  $l_m = argmax_{l \in L_m} num(l)$ {link with the maximum score and maximum number of lossy paths going over it}
  $\bar{r} = avg\{r_{i,j} | l_m \in p_{i,j}, p_{i,j} \in \Omega\}$ {average loss rate of paths in Ω going over l_m }
  $\tilde{x}_{l_m} = [\bar{r}(\frac{1}{1+\alpha}), \bar{r}(1+\alpha)]$ {performance range of link l_m }
  $R = R - \{p_{i,j} | p_{i,j} \in R, r_{i,j} \in \tilde{x}_{l_m}\}$ {Some paths going over l_m are now justified}
  $r_{i,j} = max\{0, r_{i,j} - \bar{r}\}, \forall p_{i,j} \in R, l_m \in p_{i,j}$ {Update loss rate of paths going over l_m}
  $U = U - \{l_m\}$
end while
return $\tilde{x}_l, \forall l \in E$

the optimal solution to the RT-Sum problem for two reasons. First, as in the case of *Min-Tomo*, it may not return the minimum number of bad links. Second, as opposed to *Min-Tomo*, *Sum-Tomo* may violate the consistency constraint. To understand how this can happen consider the following example. Assume we have three links l_1, l_2 and l_3, and three lossy paths $p_1 =< l_1 >$, $p_2 =< l_1, l_2 >$ and $p_3 =< l_2, l_3 >$. The measured path loss rates are 3%, 4% and 2%, respectively. Suppose that $\alpha = 0.1$. The *Sum-Tomo* algorithm would only detect l_1 and l_2 as lossy links, with the same loss rate range [1.8%-2.2%] for both links. Note that these ranges are *not* consistent with the measurement for path p_1. A consistent solution in this example would be that all three links are lossy and their loss rate ranges are $x_{l_1} = [2.7\% - 3.3\%]$, $x_{l_2} = [0.9\% - 1.1\%]$ and $x_{l_3} = [0.9\% - 1.1\%]$. Notice however that this solution results in more lossy links that the solution provided by *Sum-Tomo*. In other words, there can be a trade-off between minimizing the number of inferred lossy links and satisfying the consistency requirement. In fact, in the *RT-Sum* problem, we can show that finding a solution that satisfies the consistency requirement is NP-hard[5] (assuming that the performance ranges cannot contain zero loss rate).

[5]The proof uses a reduction to the *Exact-Set-Cover* problem [9].

5. EVALUATION

This section presents an evaluation study of different tomography algorithms using simulations on real topologies. For brevity, we only consider the *RT-Sum* problem and the loss rate metric.

5.1 Simulation model

We have developed a packet-level event-driven simulator to evaluate tomography algorithms under various network conditions. In each simulation run, we first set the number of lossy links to c. We vary c to examine how different tomography algorithms perform as the likelihood of lossy links increases. We then select the lossy links in the underlying network and assign an average loss rate to each of them. In the following results, the link loss rates are drawn from a Lognormal distribution with mean 0.04 and standard deviation 0.1[6] (these parameters were estimated from a large set of loss rate measurements in about 3600 Planetlab paths). The loss rate of the remaining links is set to zero.

We consider two loss processes:
(I) A Bernoulli random process where each arriving packet at a link is dropped with a given probability (equal to that link's loss rate).
(II) A Gilbert random process where the link's state varies between good and congested. In the good state, the link does not drop packets. In the congested state, the link drops arriving packets with a certain probability. The duration of the good and congested states follows an exponential distribution (with average duration T_{good} and T_{cong}, respectively). The loss probability in the congested state is calculated so that the long-term loss rate, across both good and congested time periods, is equal to the assigned average loss rate for that link.

The Gilbert process can better capture the bursty congestion events that are commonly observed in practice, especially when T_{cong} is much shorter than T_{good}. In our simulations, we have examined two cases for T_{good} and T_{cong}: 1) they are both set to 10sec and, 2) T_{good} is set to 100sec and $T_{cong} = 10sec$. The results with the first pair of values are similar to the Bernoulli case, and so we only report the results with the second pair of values.

To measure path loss rates, we simulate sending 4000 probing packets in each path at a rate of ten packets per second (i.e., the measurement duration is 400 seconds). If the measured path loss rate is less than 0.1%, the path is good (i.e. $\delta = 0.1\%$). The actual loss rate of a lossy link l is the ratio of probing packets (across all paths traversing l) that have been dropped at l.

We consider three IP-layer network topologies. Two of them (Internet2 and ESNet) have been provided to us by the corresponding operators. The third was obtained running Paris-traceroute [22] between 100 PlanetLab hosts. In more detail, the three topologies are:

1. The Internet2 topology, interconnecting 9 Internet2 sensors (located at the major PoPs of Internet2). It includes 44 links (all with known IP addresses) and 72 paths. The average path length is around 4 hops.

2. The ESNet topology, interconnecting 22 ESNet sensors (located at the major PoPs of ESNet). It includes 117

[6]But, we limit the maximum loss rate to 20%.

links (all with known IP addresses) and 382 paths. The average path length is around 6 hops.

3. The "PlanetLab topology", resulting from full-mesh Paris-traceroute measurements between 100 PlanetLab hosts at different sites. This topology includes 4672 links and 5917 paths. The average path length is about 15 hops.

In each of the previous topologies, there are some links that have the same *path cover set* (the path cover set of a link is the set of paths traversing that link). When two or more links have the same path cover set, it is impossible to distinguish those links and detect which of them (if any) is bad. Hence, as in previous work [15, 24], we group links with the same path cover set together[7] and localize *bad link groups* instead of bad links. The size of such groups is a topological property of the underlying network. The average group size is around 1.8 in the PlanetLab topology, while it is almost 1.0 in ESNet and Internet2.

In each simulation run, we select a topology and then select lossy links either randomly or based on a distribution that favors links close to the edge of the network (about 80% of the lossy links are at most three hops away from at least one sensor). This second approach allows us to simulate scenarios where congestion takes place mostly at the periphery of the network rather than the core.

We consider the following metrics. Let I be the set of lossy links and O be the set of lossy links inferred by a tomography algorithm. The *precision* is the ratio of links that are correctly identified as lossy to the total number of inferred lossy links, $precision = \frac{|I \bigcap O|}{|O|}$. The *recall* is the ratio of links that are detected correctly as lossy to the actual number of lossy links, $recall = \frac{|I \bigcap O|}{|I|}$. The precision and recall parameters capture the frequency of *false positives* and *false negatives*, respectively.[8] Let Q be the set of lossy links whose performance range is accurately estimated, i.e., the performance range assigned to a link in Q includes the actual loss rate of that link. The *accuracy* is defined as the ratio of inferred lossy links whose loss rate range is accurately estimated, $accuracy = \frac{|Q|}{|I \bigcap O|}$.

We consider two methods to set the α parameter. The first method (*Method-1*) relies on prior measurements in which the ground truth about the identity of bad links is known (e.g., using SNMP data from routers). The second method (*Method-2*) does not require any prior knowledge. Specifically, in Method-1 we calculate the variation of the measured end-to-end loss rates in all paths that traverse a single lossy link, and set α to the minimum value with which all these paths are α-similar. In the simulation experiments, Method-1 gives $\alpha \approx 0.3$ for Bernoulli losses, and about 0.5 for Gilbert losses. In Method-2, we first remove links that appear in good paths, as well as links that are traversed by only a small number of bad paths (say 3). For each remaining link, we compute the average loss rate $\bar{r}$ of all paths traversing that link, and determine the minimum value of α with which the loss rate of every such path is α-similar with $\bar{r}$. We finally use the median of the previous α values. The rationale in Method-2 is that if a link is traversed by multiple bad paths it is probably a bad link, and so we set α so that most such links are traversed by α-similar paths. There are two cases in which this assumption is incorrect: when a good link is identified as bad (false positive) and when a path traverses more than one bad link. However, as shown in the simulation results of this section and the experimental results of the next section, the likelihood of these two events is low in practice. In addition, by taking the median of α values across all links traversed by multiple bad paths, we reduce the impact of these events.

We compare *Sum-Tomo* with two other tomography algorithms: the Boolean *Tomo* method [8] and the Analog *Norm* method ("L1-norm minimization with non-negativity constraints") [23]. *Norm* starts with similar constraints as in (1) and solves them heuristically using a certain error minimization approach (the estimated link loss rates may not satisfy the path equations exactly, but they minimize the corresponding inference error favoring solutions that involve fewer lossy links). Some recent tomography algorithms, such as *Netscope* [15] and *LIA* [24], apply the previous norm minimization method but additionally they rely on the variance of link loss rates over multiple measurement snapshots. We do not consider such methods because they require multiple measurement snapshots (with 1,000-10,000 probing packets in each snapshot), and so they are less robust to load variations and routing changes. Our focus is on short-term measurements, and so we only compare our method with the original *Norm* algorithm. The precision and recall of *Norm* are computed by comparing the returned point estimates with the threshold δ (so that we can determine the set of bad links according to that method). To calculate the accuracy of *Norm*, we first convert the reported point estimates to range estimate using the relation shown in line-16 of Algorithm 2.

The results of each simulation are repeated 200 times. The standard error for all results in this section is negligible (mostly between 0.005 and 0.02) and so we simply show the average across all runs.

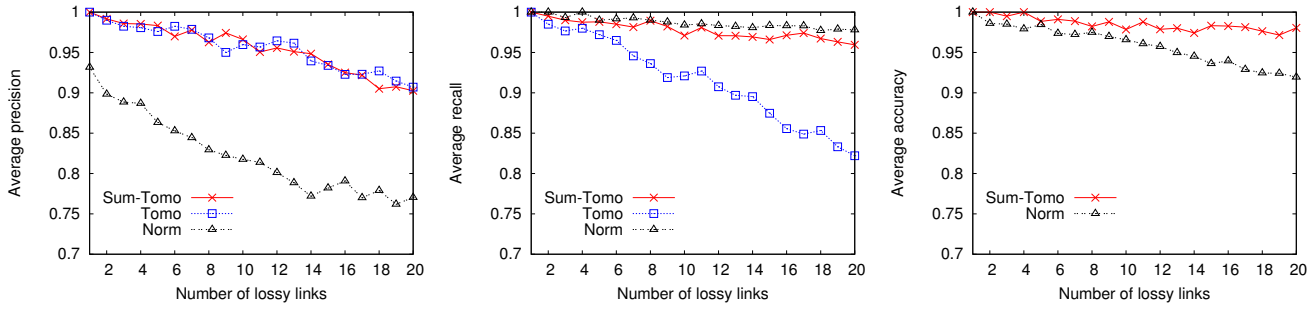
5.2 Results

We start with the ESNet topology. Figure 2 shows the results with three tomography algorithms in ESNet, when the number of lossy links varies from 1 to 20 (i.e., up to about 20% of the total links in that network). Lossy links are selected randomly and the loss process follows the Bernoulli model. The value of α is set based on Method-1; a comparison with Method-2 is given later in this section (Figure 5). Since *Tomo* is not able to determine the loss rate of links, Figure 2c only compares the results of *Sum-Tomo* and *Norm*. As the number of lossy links increases, we see a degradation in the performance of all tomography algorithms. This is because more lossy links create more lossy paths, and hence fewer links can be accounted as good at the initial step of the algorithms. Therefore, the tomography methods have to opt their candidate lossy links from a larger set, which increases the localization errors.

As illustrated in Figure 2a, the precision of *Tomo* and *Sum-Tomo* is very close. However, the recall of *Sum-Tomo* is higher than of *Tomo* by up to 13% (Figure 2b). The reason is that *Tomo* does not consider the loss rate of lossy paths and it simply aims to find the minimum set of links shared between such paths. This approach can be clearly wrong, however, because a link can justify different paths

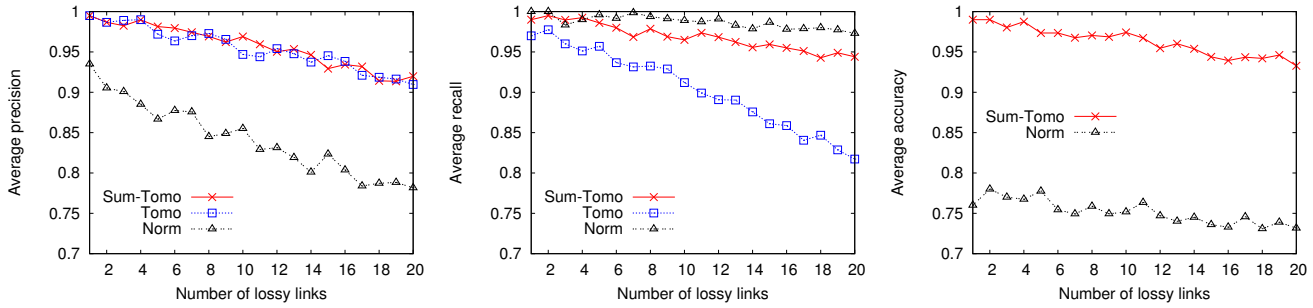
[7]These groups of links over a path correspond to the Minimal Identifiable Link Sequence (MILS), defined in [25].

[8]In tomography, a false positive refers to detecting a link as bad while it is good.



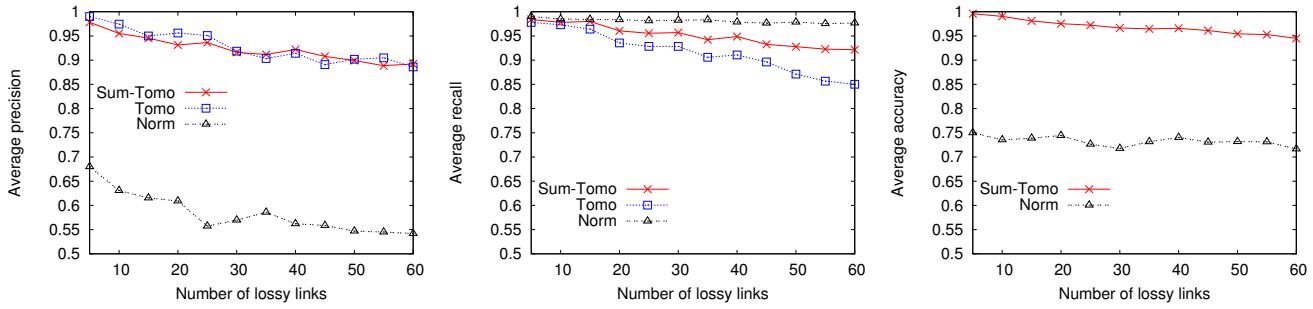
(a) Precision (b) Recall (c) Accuracy

Figure 2: Results of different algorithms obtained in ESNet when actual link loss rates follow Bernoulli process.



(a) Precision (b) Recall (c) Accuracy

Figure 3: Results of different algorithms obtained in ESNet when actual link loss rates follow Gilbert process.



(a) Precision (b) Recall (c) Accuracy

Figure 4: Results of different algorithms obtained in PlanetLab when actual link loss rates follow Gilbert process.

even though their loss rates are very different. Hence, *Tomo* often identifies fewer lossy links. On the contrary, *Norm* usually selects more lossy links to justify the loss rate of every lossy path (the number of links reported by *Norm* as lossy is 30% more than *Sum-Tomo*). This is the reason that *Norm* gives low precision and high recall. Moreover, the accuracy results in Figure 2c show that *Sum-Tomo* can estimate a loss rate range for lossy links quite well. Specifically, if a link is correctly detected by *Sum-Tomo* as lossy, then with a high probability (above 95%) its loss rate range includes the actual loss rate of that link.

The results of the previous algorithms in the Internet2 topology are very similar with the results in the ESNet topology (when considering the same fraction of lossy links), and so they are omitted.

Figure 3 shows the results of the three tomography algorithms in ESNet, when lossy links are selected randomly and their loss process follows the Gilbert model. Comparing the results of Figures 2 and 3, note that the precision of all algorithms does not change much. In addition, the recall of *Tomo* remains the same because the binary state of a path remains the same under these two loss processes. However, we observe a small decrease (by about 2%) in the recall of *Sum-Tomo* and *Norm*, compared to Figure 2b. With the Gilbert process, the measured loss rate of a path can be significantly different from the actual loss rate of its constituent links, and so the value of α should be higher. This means that each link that is detected by *Sum-Tomo* as lossy can justify more lossy paths, and so *Sum-Tomo* may fail to identify some lossy links. Nonetheless, the recall of *Sum-*

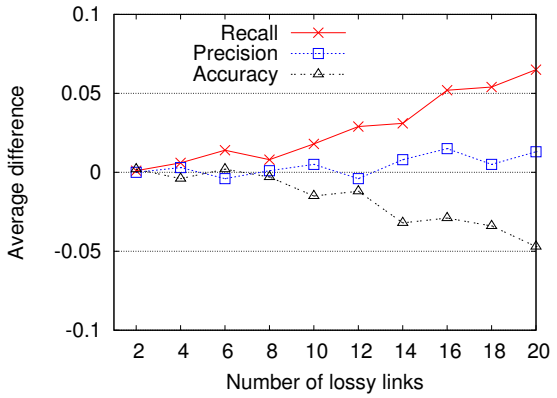


Figure 5: The difference in the results of *Sum-Tomo* with Method-1 versus Method-2 for the selection of α.

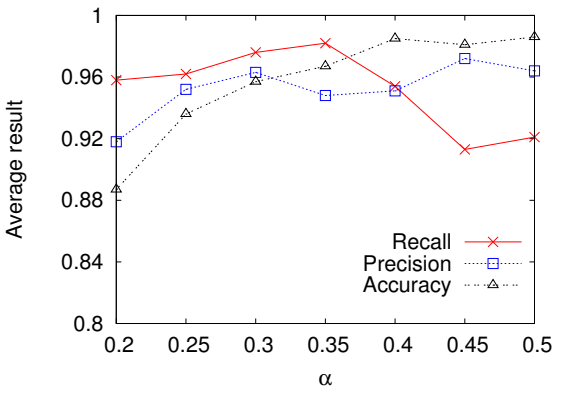


Figure 6: The sensitivity of various parameters to the value of α.

Tomo is still considerably higher than that of its Boolean counterpart (by up to 10%), while it remains close to the recall of *Norm*. Because the Gilbert process introduces more error in the path measurements, the accuracy of both *Sum-Tomo* and *Norm* algorithms declines (Figure 2c). However, because the Analog approach is more sensitive to measurement errors, we see a larger accuracy decrease in *Norm* (by about 25%).

Similarly, Figure 4 shows results obtained with the PlanetLab topology. Here, the loss process follows the Gilbert model while the selection of lossy links has the previously mentioned edge-network bias (α is the same as before). Because the PlanetLab topology is much larger, we consider up to 60 lossy links (about 5% of the total number of links in that network). Figure 4a shows that the precision of *Tomo* and *Sum-Tomo* are close. However, as shown in Figure 4b, the recall of *Sum-Tomo* is higher than that of *Tomo*. This is again because *Tomo* selects fewer links to justify lossy paths (e.g. about 10% fewer links than *Sum-Tomo*). On the other hand, the precision of *Norm* is quite low, as it does not capture that paths sharing the same lossy link may measure different loss rates. So, it reports about 40% more lossy links than *Sum-Tomo*. The accuracy of the loss rate ranges assigned to lossy links by *Sum-Tomo* is always higher than 94%.

The width of the loss rate ranges assigned to bad links depends on the parameter α. When we use the Bernoulli loss model (i.e. $\alpha \approx 0.3$), the width of the resulting loss rate range is about 50% of the center value. For example, when the center of the loss rate range is 0.04, the reported range is from 0.03 to 0.05. With the Gilbert loss model (i.e. $\alpha \approx 0.5$), the width of the loss rate range is about 80% of the center value.

In comparison to a random selection of lossy links, the edge-network bias results in a small decrease (between 2-3%) in precision and recall for *Tomo* and *Sum-Tomo*. This is because fewer paths traverse edge links than core links, and so in some cases these two greedy tomography algorithms fail to select the right lossy link because they prefer to select the most shared link (which is often in the network core).

We now focus on the selection of α, comparing Method-1 with Method-2. Here, we use the ESNet topology, while lossy links are selected randomly and the link loss rates follow the Gilbert process. Figure 5 shows the differences in precision, recall and accuracy between Method-1 and Method-2. Method-1 provides better estimates for α, as it relies on prior measurements in which the ground truth is known. The difference between the two methods is negligible when the number of lossy links is not high (e.g., less than 10% of all network links). With more lossy links, the recall obtained with Method-2 is worse (by up to 6%), while the accuracy improves (by at most 5%). The precision remains roughly the same. The reason is that, when there are many lossy links, Method-2 over-estimates the value of α since each bad path is more likely to traverse more than one lossy link. Then, *Sum-Tomo* becomes more similar to the Boolean *Tomo* algorithm, which yields lower recall, as noted earlier. On the other hand, Method-2 gives higher accuracy because the performance ranges become wider as a result of a larger α.

Finally, we investigate the sensitivity of the precision, recall and accuracy metrics to the value of α. Here, we use the ESNet topology, while losses follow the Bernoulli process. We fix the number of lossy links to 10 (i.e. about 10% of the network's links are lossy). The value of α resulting from Method-1 and Method-2 is around 0.3 and 0.4, respectively. We vary α around these two values, from 0.2 to 0.5. As Figure 6 shows, the accuracy is always improved with increasing α because the estimated loss rate range becomes wider. The precision has an increasing trend with α because each link that is detected as lossy can potentially justify more bad paths, and so *Sum-Tomo* detects fewer links as lossy. For the same reason, the recall drops significantly as we increase α above the value recommended by Method-1 or Method-2. In fact, when α is sufficiently high the *Sum-Tomo* algorithm behaves similar with the Boolean tomography algorithm *Tomo*.

In summary, the errors with *Sum-Tomo* increase as the number of lossy links increases, their locations are moved closer to the edge of the network, and their loss rate varies rapidly over time.

6. EXPERIMENTAL RESULTS

We have conducted extensive experiments on real networks to understand the practical issues involved in network tomography and to examine the behavior of Range Tomography algorithms in practice. In fact, the main purpose of

this section is to show the characteristics of actual congestion events in Internet paths and to illustrate how the *Sum-Tomo* algorithm localizes such events. We first present our method to detect congestion events, and analyze the results of that detection phase. We then use the *Sum-Tomo* algorithm to localize those congestion events.

6.1 Detection of congestion events

We performed our experiments over three real networks: Internet2, ESNet, and PlanetLab (the same topologies that were described in Section 5, except that the PlanetLab topology here consists of 60 sensors). The measurement data for Internet2 and ESNet have been provided to us by the corresponding network operators, while the measurement data for PlanetLab were obtained by us running the same measurement tools that are used in Internet2 and ESNet.

The forwarding paths between sensors were measured using Paris-traceroute [22] every 30 minutes. Because multipath forwarding is common in practice (as reported by [26]), it is necessary to use tools such as Paris-traceroute to avoid false inference of paths that do not actually exist in the network. We measured the one-way delay variations and the loss rate in each path with active measurements (using OWAMP), sending ten UDP 60-byte packets per second. In the following results, we analyze a 3-day PlanetLab data set, a 35-day Internet2 data set, and a 14-day ESNet data set.

Generally speaking, we define a *congestion event* as a significant increase in the path one-way delays and/or as the appearance of packet losses. Specifically, if we observe a significant increase in the path one-way delay for at least k consecutive packets, relative to the path's base delay, we detect a congestion event in that path. This approach falsely detects congestion events in two cases: I) when the forwarding path between the two sensors changes (e.g. due to a routing change or because of an NTP clock adjustment), or II) when there is a CPU context-switch event in either sensor, causing a short-term disruption in the measurement process. In the former, we observe a level-shift (rising or falling) in the one-way packet delays. In the latter, we observe a sudden substantial increase in the one-way packet delays, followed by a linear decrease towards the base delay. Specifically, if we observe an increase of at least 80ms in the one-way delay of two consecutive packets, we detect the start of a context-switch event; the event lasts until the delay returns back to its base level[9]. We preprocess the data set to remove all detected context-switches and level-shifts before applying the tomography algorithm.

6.2 Detection results

Table 1 shows the congestion event detection results for the three data sets. The frequency of level-shifts is quite low in all three networks (especially, in ESNet), suggesting that the paths in these networks are stable over several hours. On the other hand, the frequency of context-switches is relatively high in ESNet and PlanetLab paths, indicating that sensors are often highly-loaded (Internet2 sensors are much better in this aspect). The frequency of congestion events is much higher in Internet paths between PlanetLab hosts, while this frequency is quite small in ESNet.

[9]The base delay of a path at time t is set to the median of the one-way delays among the last 100 packets received in that path before t.

Table 1: The results of the detection process on three networks. The value in each cell represents the average occurrence frequency per day and per path.

	Level-shift	Context-switch	Congestion
Internet2	0.01	0.011	0.002
ESNet	0.0002	0.48	0.0002
PlanetLab	0.007	0.61	0.005

Figure 7a shows the distribution of congestion event duration in the PlanetLab network. About 95% of the congestion events last less than 30 seconds. These durations become even shorter in Internet2 and ESNet (the maximum duration is 27sec and only 5% of congestion events last more than 20 seconds in those two networks). This result clearly contradicts the assumption made by several tomography methods that the network conditions (e.g., congestion level at different links) remain constant for long periods of time. It becomes clear that *tomography methods should be able to produce results based on short-term measurements.*

Table 2 summarizes statistics about lossy congestion events (congestion events with at least one lost packet)[10]. The frequency of lossy congestion events is negligible in Internet2 and ESNet, but it is large in PlanetLab paths. More than 50% of the lossy congestion events include just one lost packet, while there are at most 4 packet losses in about 80% of such events (the loss rate is below 1% in about 80% of lossy congestion events).

It is important to note that several tomography methods rely on loss rate to infer and localize congestion events. The previous results imply that most congestion events do not introduce any packet losses, and if there are losses the loss rate is often very low (making it difficult to estimate the loss rate in practice with a limited frequency of probing packets). For these reasons, we define congestion events based on one-way delay variations in the rest of this section.

Table 2: The number and frequency of lossy congestion events (over all congestion events) in the three data sets.

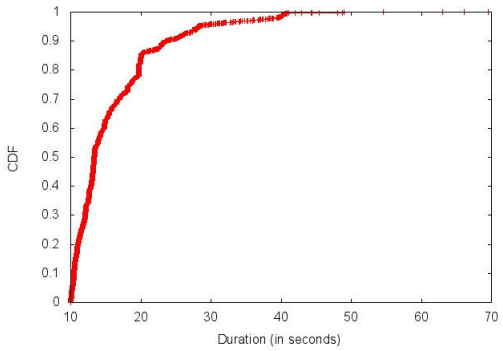
	# lossy events	Frequency of lossy events	# lost packets
Internet2	6	0.2%	1
ESNet	0	0%	0
PlanetLab	535	16.5%	[1-320]

Assume we have detected a congestion event in a path between the k^{th} and l^{th} probing packets. We use the average one-way delay increase during that period to quantify the *congestion magnitude* M,

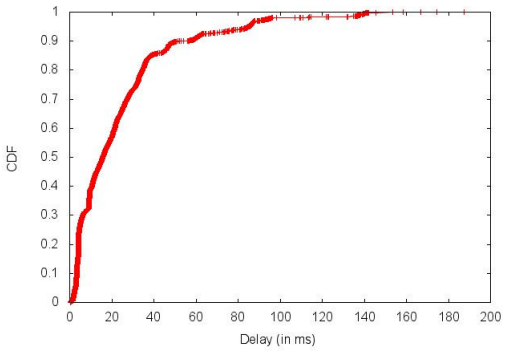
$$M = \frac{\sum_{i=k}^{l} (d_i - d_{base})}{l - k} \quad (7)$$

where d_{base} is the base one-way delay (estimated as the median one-way delay of the last 1000 probing packets before the k^{th} packet). A higher value of M indicates heavier congestion in that path. Figure 7b shows the congestion magnitude distribution in the PlanetLab network. The average delay increase during half of the congestion events is less

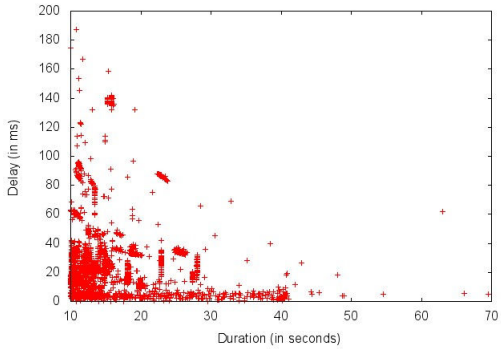
[10]Note that we have ignored cases where all packets sent (or received) over every path to (or from) a specific host are dropped because the problem then can be easily associated with the corresponding sensor and its local-area network.



(a) CDF of congestion duration



(b) CDF of congestion magnitude



(c) Congestion magnitude vs. congestion duration

Figure 7: Results of the detection process in PlanetLab.

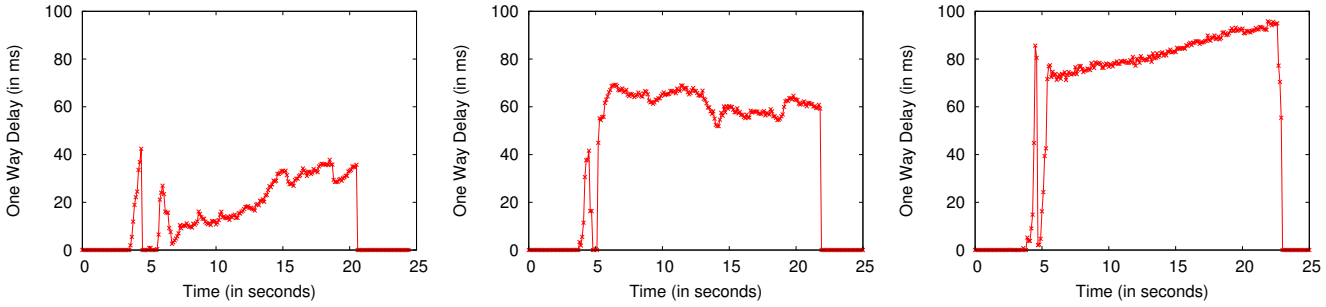
than $20ms$, and it does not exceed $50ms$ in 90% of those events. The distribution of M in the other two networks has the same trend, but the maximum value of M is at most $60ms$ there. Finally, Figure 7c shows the magnitude of congestion events versus their duration. It is interesting to see that heavier congestion events usually last for shorter durations, emphasizing again that a practical tomography method should be able to work with short-term measurements. It should be noted here that network tomography methods cannot localize congestion events that are very short in duration because it is hard to reliably detect such events, and further to estimate their magnitude, with active measurements. For instance, even if the probing frequency is 10 packets per second, a congestion event that lasts for three seconds introducing a loss rate of 1% would probably not be observed by the probing stream. However, because the Range Tomography methods do not require a long measurement history or precise estimates of path performance, they are able to localize congestion events that last for at least few tens of seconds, as shown in the following results.

6.3 Localization results

We run the *Sum-Tomo* algorithm on the detected congestion events to localize congested links. Ideally, we expect that a congested link causes the same congestion event (in terms of duration and congestion magnitude) to paths that traverse that link. However, these congestion periods do not exactly overlap in practice due to lack of perfect clock synchronization in the measurement hosts. Therefore, we consider the time interval that covers all overlapping congestion periods as the designated congestion period that the tomography method analyzes.

We then compute the congestion magnitude M in every path during that designated period. To distinguish good paths from bad paths, we set the threshold δ as the minimum congestion magnitude over all detected congestion events in that network (the value of δ is around $2ms$ in practice). The α parameter is determined based on Method-2, introduced in Section 5.1. This process results in α=2 in Internet2 and ESNet, and α=1.2 in PlanetLab. In our experiments, we ignore paths containing links with unknown IP addresses (only happens in the PlanetLab data set).

It is not possible to directly validate the results of the *Sum-Tomo* algorithm because we do not have access to the routers and switches along each path. Instead, we use the indirect validation method introduced in [19]. In this method, the measured paths in each period are divided in two groups, *inference paths* and *validation paths*. Each link should be included in both sets. The inference paths are then used to identify congested links and to estimate their congestion magnitude range using the *Sum-Tomo* algorithm. Then, we use the validation paths to examine if the ranges that were computed based on the inference paths are consistent with the measured congestion magnitude of the validation paths. Finally, we measure the *validation error* as the fraction of bad links in which the estimated congestion magnitude range



(a) Delay timeseries of a path traversing a congested link l_1.

(b) Delay timeseries of a path traversing a congested link l_2.

(c) Delay timeseries of a path traversing both congested links l_1 and l_2.

Figure 8: The timeseries of one-way packet delays measured in three paths during a congestion event in Internet2. The delays are relative to the base delay in that path (i.e., they represent one-way delay increase due to queueing). There are two congested links during this episode. Paths in (a) and (b) traverse two different congested links, but the path in (c) traverses both links. Note that the sum of the delay increase in (a) and (b) results to roughly the same delay increase we observe in (c).

for a link is *not* consistent with the measured congestion magnitude in the corresponding validation path.

Overall, we analyzed 254 congestion events in Internet2, 336 events in ESNet, and 159 events in PlanetLab. In almost all events, *only one congested link* was identified as the root cause of the congestion event. Specifically, more than 97% of congested paths in all three networks include just one congested link, and only 0.5% of paths contain more than two congested links (we did not find any path with four congested links in our data sets).

The validation error in all three data sets was zero. In addition, the time-series of one-way delay variations in the inference and validation paths are highly correlated in the time domain when the corresponding congested paths traverse the same congested link. For instance, the time-series in Figure 8 show an interesting example of a congestion event in Internet2 that involves two congested links. The congestion magnitude range assigned to links is mostly below 70ms in PlanetLab, and mostly less than 40ms in Internet2 and ESNet.

7. RELATED WORK

We refer the reader to a thorough review [10] for a coverage of the prior work until 2004.

In Analog tomography, a typically under-constrained linear system of equations models the relation between path and link parameters (assuming that the topology is known). Techniques from parametric or non-parametric statistical inference, jointly with additional constraints, assumptions and optimization objectives, are then used to infer the most likely values of the link parameters from the measured path parameters [10, 11]. For instance, Shavitt et al. [12] estimate link delays using a least-squares method, Bu et al. [13] use expectation-maximization to infer link loss rates, while Chen et al. [14] use Fourier domain analysis to infer link delays. NetScope [15] is a recent method that estimates link loss rates also considering the observed loss rate variances. Ghita et al. study the case that different links can be correlated (all other related work assumes independent links) [16]. The application of Analog tomography in practice has been rather limited for several reasons. First, they require accurate end-to-end path measurements, which are hard to get in short timescales and without intrusive probing. Second, some link-level parameters may not be statistically identifiable, meaning that different assignments of link-level metrics produce the same statistical distribution of path measurements [5, 6, 7]. Third, Analog methods can be computationally intensive [7].

Duffield introduced the Boolean tomography framework in 2003 [5, 7], while Nguyen and Thiran compared Analog with Boolean tomography [17]. NetDiagnoser [8] extends Boolean tomography to multiple sources and destinations. Kompella et al. [18] consider a similar approach to detect "silent failures" in MPLS/IP networks using active measurements between edge routers. Nguyen and Thiran introduce a Boolean method to infer link state probabilities from multiple measurements over time, and then using those probabilities to identify congested links [6]. Bayesian approaches to infer lossy links have also been proposed [19]. The authors in [20] use prior link state probabilities to diagnose the underlying cause behind the faulty state of links. Barford et al. have proposed a framework to detect and localize performance anomalies using active probe-based measurements [21]. To reduce probing overhead, they give an algorithm to select the paths that should be probed at any point in time.

8. CONCLUSIONS

We proposed a new tomography framework that combines the best features of Analog and Boolean tomography. Range tomography estimates a range estimate for each bad link, instead of aiming to infer a point estimate or a binary estimate for every link. We applied the range tomography framework in two path performance metric functions (Min and Sum) and presented an efficient heuristic for each function. The *Min-Tomo* algorithm considers only the lowest-performance link over a path, while the *Sum-Tomo* algorithm considers the sum of the performance metrics for all bad links in that path.

Simulation results show that the proposed tomography method performs much better than earlier Boolean and Analog techniques in terms of precision, recall and accuracy. For instance, *Sum-Tomo* generates up to 35% less false positives

than the Analog Norm-minimization method, while its false negative error is less than the Boolean Tomo method by up to 15%. Moreover, the accuracy of the resulting range estimates is always high (more than 93%).

We have also applied the *Sum-Tomo* method in three operational networks to detect and localize congested links and to estimate their congestion magnitude. According to an indirect validation method, the resulting link range estimate is consistent with the measured path congestion magnitude in every congestion event we have analyzed. The experimental results also emphasize that congestion events in Internet paths are often short-lived, and so any practical tomography methods should be accurate even if the path measurements result from few probes and they are error-prone. Finally, at least for the networks we measured in this study, we often see only one bad link during any congestion event, and rarely more than 2-3 bad links.

Acknowledgements

We are grateful to Partha Kanuparthy for his help with the detection logic and data management. We also thank Jason Zurawski from Internet2, and Joe Metzger, Brian Tierney and Andrew Lake from ESnet for providing us with the OWAMP data sets. We are also grateful to the anonymous reviewers and our "shepherd", Matthias Grossglauser, for their constructive comments.

This research was supported by the U.S. Department of Energy under grant number DE-FG02-10ER26021.

9. REFERENCES

[1] R. Caceres, N.G. Duffield, J. Horowitz, D. Towsley. Multicast-Based Inference of Network Internal Loss Characteristics. *IEEE Trans. on Information Theory*, 45(7), 2462-2480, 1999.

[2] M. Coates, R. Nowak. Network loss inference using unicast end-to-end measurement, In *Proc. ITC Conf. IP Traffic, Modeling and Management*, 2000.

[3] Y. Zhang, N. Duffield, V. Paxson, S. Shenker. On the Constancy of Internet Path Properties. In *ACM SIGCOMM Workshop on Internet Measurement*, 2001.

[4] M. Roughan. Fundamental Bounds on the Accuracy of Network Performance Measurements. In *ACM SIGMETRICS*, 2005.

[5] N. Duffield. Simple network performance tomography. In *Proc. ACM IMC*, 2003.

[6] H. Nguyen, and P. Thiran. The Boolean Solution to the Congested IP Link Location Problem: Theory and Practice. In *Proc. IEEE INFOCOM*, 2007.

[7] N. Duffield. Network Tomography of Binary Network Performance Characteristics. In *IEEE Trans. Information Theory*, 52, 2006.

[8] A. Dhamdhere, R. Teixeira, C. Dovrolis, C. Diot. NetDiagnoser: Troubleshooting network unreachabilities using end-to-end probes and routing data. In *Proc. ACM CoNEXT*, 2007.

[9] M. R. Garey, and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Co., 1979.

[10] R. Castro, M. Coates, G. Liang, R. Nowak, B. Yu. Network Tomography: Recent Developments. *Statistical Science*, 19(3):499-517, 2004.

[11] R. Caceres, N. Duffield, S. Moon, D. Towsley. Inference of Internal Loss Rates in the MBone. In *Proc. IEEE Global Internet*, 1999.

[12] Y. Shavitt, X. Sun, A. Wool, B. Yener. Computing the unmeasured: An algebraic approach to internet mapping. In *Proc. IEEE INFOCOM*, 2001.

[13] T. Bu, N. Duffield, F. L. Presti, D. Towsley. Network tomography on general topologies. In *Proc. ACM SIGMETRICS*, 2002.

[14] A. Chen, J. Cao, T. Bu. Network tomography: Identifiability and fourier domain estimation. In *Proc. IEEE INFOCOM*, 2007.

[15] D. Ghita, H. Nguyen, M. Kurant, K. Argyraki, P. Thiran. Netscope: Practical Network Loss Tomography. In *Proc. IEEE INFOCOM*, 2010.

[16] D. Ghita, K. Argyraki, P Thiran. Network Tomography on Correlated Links. In *Proc. ACM IMC*, 2010.

[17] H. X. Nguyen, and P. Thiran. Binary versus analogue path monitoring in IP networks. In *LNCS*, Vol. 3431, Jan. 2005, p. 97.

[18] R. R. Kompella, J. Yates, A. Greenberg, A. C. Snoeren. Detection and Localization of Network Blackholes. In *Proc. IEEE INFOCOM*, 2007.

[19] V. Padmanabhan, L. Qiu, H. Wang. Server-based Inference of Internet Link lossiness. In *Proc. IEEE INFOCOM*, 2003.

[20] S. Kandula, D. Katabi, J.-P. Vasseur. Shrink: A Tool for Failure Diagnosis in IP Networks. In *Proc. ACM SIGCOMM MineNet Workshop*, 2005.

[21] P. Barford, N. Duffield, A. Ron, and J. Sommers. Network Performance Anomaly Detection and Localization. In *Proc. IEEE INFOCOM*, 2009.

[22] B. Augustin et al. Avoiding traceroute anomalies with Paris traceroute, In *Proc. ACM IMC*, 2006.

[23] H. H. Song, L. Qiu, Y. Zhang. NetQuest: A Flexible Framework for Large-Scale Network Measurement. In *Proc. ACM SIGMETRICS*, 2006.

[24] H. X. Nguyen, and P. Thiran. Network Loss Inference with Second Order Statistics of End-to-End Flows. In *Proc. ACM IMC*, 2007.

[25] Y. Zhao, Y. Chen, D. Bindel, Towards Unbiased End-to-End Network Diagnosis, In *Proc. ACM SIGCOMM*, 2006.

[26] M. Luckie, A. Dhamdhere, K. Claffy, D. Murrell. Measured impact of crooked traceroute. In *ACM SIGCOMM CCR*, Vol. 41, Issue 1, 2011.