

Localization of network performance problems with multi-level discrete tomography

Sajjad Zarifzadeh, Constantine Dovrolis

Abstract—The focus of network monitoring and problem diagnosis gradually moves towards “soft failures” and performance problems, such as noticeable jitter or loss rate. A major, and still unresolved, problem in this space is to localize a performance problem at the granularity of individual IP-layer links.

We propose a network tomography framework that aims to localize performance problems in the set of paths that interconnect a full-mesh topology formed by N sensors. As opposed to Boolean tomography, which models links as “good” versus “bad”, we consider a multi-level discrete tomography framework that can capture the large diversity of performance levels seen at the Internet. We evaluate two tomography methods: the first assumes that the performance of a path is determined by the lowest-performance link in the path; the second also takes into account the number of such links. We propose an efficient algorithm for each method and then evaluate their accuracy and also show their superiority over two traditional approaches (Boolean and Analogue) on real single-provider networks (ESnet and Internet2) as well as in an interdomain context (PlanetLab).

I. INTRODUCTION

In the not so distant past, the main goal of network monitoring was to detect broken links or routers that cause reachability problems. Today, network operators and customers are increasingly concerned about performance. Even a small loss rate, say 0.1%, will be enough to limit the throughput of a bulk TCP transfer to less than few Mbps. Jitter of few tens of milliseconds can cause problems for VoIP and gaming applications. Additionally, enterprise customers demand performance assurances and so network operators have every reason to monitor the performance of their networks closely.

A major open problem in network monitoring is to automatically locate the source of a performance issue. In the intradomain context, it may be possible to instrument and passively monitor every single link and router so that the operator can detect performance problems in her network using SNMP or through proprietary network management solutions. This approach can be expensive however and, more importantly, it is applicable only in the intradomain context.

When network paths cross multiple administrative domains, an operator would not be willing typically to expose performance data about her network to other operators or customers. It is not uncommon, actually, that operators argue about performance problems claiming that “the problem is not in my network - it is in your network.” Besides, Internet users or enterprise customers would not have access, in general, to the required performance data so that they can localize where the problem is themselves.

Another approach to localize performance problems is to rely on *end-to-end active measurements and network tomog-*

raphy methods. It is simpler to describe this approach in the context of an enterprise network. This enterprise may be present at several locations, connected to the Internet through different providers. Each location can deploy a measurement host, or *sensor*, that periodically performs active measurements (such as one-way delay, loss rate, TCP throughput tests, VoIP streams, etc) to all other sensors. The collected performance measurements can be analyzed in real-time to detect end-to-end paths that experience similar performance problems simultaneously. At the same time, sensors can run traceroute towards every other sensor, mapping the common links between different paths. Armed with this spatial information it is often possible to identify the links that most likely cause a performance problem.

In this paper, we propose a new approach to network tomography, referred to as *Multi-level Discrete Tomography (MDT)*. In MDT, links and paths are classified with respect to a given performance metric in few discrete Performance Levels (P-Levels). For instance, the P-Levels may be (in decreasing order of performance) Green (i.e., good), Yellow, Orange and Red. In this framework, we consider two possible formulations (*Max* and *Max-Count*) that differ in the way we classify paths based on the P-Level of the underlying links. We propose efficient algorithms to solve these two MDT problems, and evaluate their accuracy and diagnostic resolution in three real topologies (Internet2, ESnet, and a subset of the PlanetLab topology).

The rest of the paper is structured as follows. In Section II we motivate, describe and formally state the MDT problem. In Sections III and IV, we consider the *Max* and *Max-Count* formulations, respectively, and describe the proposed algorithms. Section V focuses on the selection of boundaries between successive P-Levels. In Section VI, we evaluate the proposed methods using computational experiments as well as Internet measurements. Section VII discusses related work in network tomography. We conclude the paper in Section VIII.

II. MULTI-LEVEL DISCRETE TOMOGRAPHY

In this section, we motivate, state and explain the Multi-level Discrete Tomography problem.

A. Motivation

Boolean tomography is a practical approach to diagnose reachability problems and localize likely link failures [1], [2], [3]. It assumes that the state of links and paths can be represented with binary variables; namely a link or path is either “good” or “bad”. A binary state can represent well

the failure (or very bad performance) of a link or path compared to a working link or path, respectively. On the other hand, the diagnosis and localization of *performance problems* requires more information about the magnitude of continuous metrics, such as queuing delay, jitter, loss rate or available bandwidth. The representation of such metrics with just two states is clearly insufficient. At the same time it is difficult or impossible to accurately estimate these metrics at every link of a network as continuous variables (as discussed in Section VII). A different approach, that we refer to as *Multi-level Discrete Tomography* or *MDT*, is to represent the state of each link and path with a discrete variable that has more than just two levels. To keep it practical, we consider a small number of levels, typically 3-5.

To illustrate the wide variation in path performance levels, we have conducted simple path loss rate measurements between 30 PlanetLab hosts around the world. Even though PlanetLab hosts and their connectivity are certainly not representative of Internet performance in general, they give some insight about the abundant variability in path performance. Each host estimates the long-term loss rate q to every other host with active measurements (using OWAMP), sending ten UDP 100-byte packets per second. In 75% of the paths q is less than 0.1%. Among the remaining paths, in 40% of them q is between 0.1% and 1%, in 50% of them q is between 1% and 10%, and in 10% of them q is higher than 10%. The representation of paths as “good” versus “bad” would be problematic in this case. On the other hand, a classification of paths as “almost-lossless”, “low-loss”, “medium-loss” and “high-loss” (corresponding to the previous four loss rate ranges) would provide much more information and it would remain practical.

B. Problem statement

We are given a set of N measurement points or hosts, referred to as *sensors* $S = \{s_1, \dots, s_N\}$. 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” using traceroute-like tools (such as Paris traceroute [20]). 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.

We consider $Q + 1$ discrete performance levels, or *P-Levels*, for links and paths. Specifically, the P-Level of a link $e \in E$ is denoted by $l(e)$ and it is assigned from a set $L = \{l_0, \dots, l_Q\}$, where Q is a small integer. The lowest P-Level (l_0) is special because it is used to represent links or paths that are viewed as “good”, i.e., without performance issues. The remaining P-Levels represent different degrees of “bad” performance with increasingly worse performance as the P-Level increases, i.e., l_i is worse than l_{i-1} (we write $l_{i-1} < l_i$).

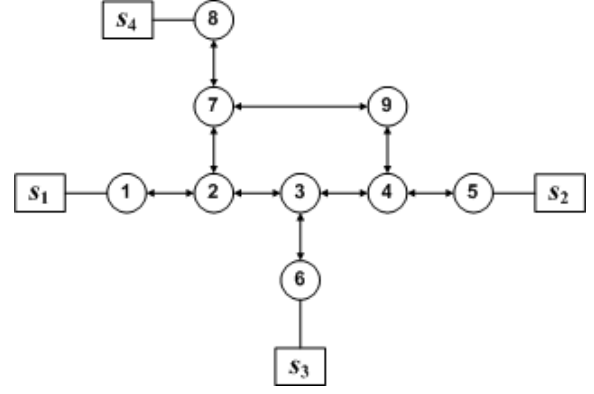


Fig. 1: A simple example network with 4 sensors. The paths between sensors follow shortest-path routing.

To classify paths into different P-Levels, we are given a set B of boundaries between P-Levels, $B = \{b_1, \dots, b_{Q-1}\}$. For instance, if the performance metric is loss rate, a path will be classified as l_i if its loss rate is between b_i and b_{i+1} , with $b_0=0$ and $b_Q=1$. The same classification boundaries apply for links. The selection of classification boundaries is discussed in Section V.

Suppose that $p_{i,j}$ denotes the path from sensor s_i to sensor s_j , and $m_{i,j}$ is the P-Level of $p_{i,j}$ for a given performance metric. We assume that there is a function f that expresses the P-Level of each path based on the P-Level of its constituent links,

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

We refer to f as the *P-Level function*. For instance, in the case of available bandwidth, the available bandwidth of the end-to-end path is determined by the link with the minimum available bandwidth. Thus, the P-Level function for that metric would be based on the *MIN* operator. In MDT, we attempt to invert function f and estimate the P-Level of all links in a network given the P-Levels of a set of paths in that network. 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 [9].

At this point we have to make three important assumptions:

- (1) Regardless of f , the P-Level of a path is always larger or equal than the P-Level of any link in that path (i.e., a path cannot be better than its worst link).
- (2) The P-Levels of different links are independent (i.e., there are no spatial performance correlations between links).
- (3) The P-Level of links and the routing paths between sensors remain constant during the measurement process (this should be feasible if we can measure the performance of all paths within few tens of seconds).

Let us consider the simple topology of Figure 1, which interconnects four sensors. $p_{i,j}$ is the shortest path from s_i to s_j . Suppose we have three P-Levels representing three ranges

of loss rates. The end-to-end measurements indicate that paths $p_{1,2}, p_{3,2}, p_{4,2}$ are lossy; the first two paths are classified as l_2 , while the third is classified as l_1 . All remaining paths are good (l_0). Boolean tomography would choose the link from node-4 to node-5, represented by (4,5), as the only bad link because it can cover all lossy paths. However, choosing a single lossy link cannot justify the different P-Level of these paths. We need at least another lossy link, e.g. (3,4), to justify all measured P-Levels. One plausible result is that link (4,5) is l_1 while link (3,4) is l_2 .

The *Multi-level Discrete Tomography (MDT)* problem can be stated as follows:

MDT Problem: *Infer the P-Level of all links in G so that the measured P-Levels of all paths in Π satisfy the P-Level function constraints:*

$$\forall p_{i,j} \in \Pi : m_{i,j} = f\{l(e) | \forall e \in p_{i,j}\} \quad (2)$$

and the number of links with non-zero P-Level is minimized.

Note that without the objective to minimize the number of bad links, a solution of the MDT problem could result to many false positives. For instance, for certain P-Level functions we can simply assign to each link the worst P-Level among all paths that traverse that link. Hence, as in Boolean tomography, it is essential to include an optimization objective so that the number of bad links (non-zero P-Level) is minimized.

The complexity and practicality of the MDT problem highly depend on the P-Level function f . In the following two sections, we consider two specific instances of the function f that can be used to capture various performance metrics. In each case, we analyze the complexity of the corresponding MDT problem and present a heuristic to solve it.

We should mention that the Boolean tomography can be considered a special case of MDT with two P-Levels and a P-Level function that determines that a path is bad if there is at least one bad link in that path. The Boolean tomography problem can be reduced to the Min-Set-Cover problem [4], which is NP-Hard (by mapping links into sets and paths into elements of sets). There is a greedy heuristic for the Min-Set-Cover problem with an approximation ratio of $\log|S|$, where S is the set of elements from which the hypothesis set can be chosen ($|S|$ denotes the size of S) [5]. The *Tomo* algorithm, presented in [3], uses that heuristic to compute a solution with approximation ratio $\log|\Pi|$ for the Boolean tomography problem.

III. MAX P-LEVEL FUNCTION

We first consider a function that determines the P-Level of a path $p_{i,j}$ as the maximum (worst) P-Level among all links of that path, i.e.,

$$m_{i,j} = \max_{e \in p_{i,j}} \{l(e)\} \quad (3)$$

The *Max* function captures well performance metrics at which the performance of the entire path is mostly determined by a bottleneck link. This is true for available bandwidth or capacity, but it can also be true for loss rate, jitter or other metrics when there is a single bottleneck link that is much worse than any other link in the path.

A. Solution and analysis for Max

In the case of the *Max* P-Level function, the MDT problem can be thought of as multiple instances of the Boolean tomography problem: in the i^{th} instance we only consider the set of paths with P-Level l_i and compute the smallest set of links that satisfy the P-Level constraints of those paths.

Consider again the topology of Figure 1. Suppose we have 7 bad paths, with $p_{1,3}, p_{2,3}, p_{4,3}, p_{1,2}$ classified as l_2 , while $p_{1,4}, p_{3,2}, p_{4,2}$ classified as l_1 . The remaining paths are good (l_0). First, all links on good paths (e.g., links on $p_{3,4}$) are marked as good. In the first instance we focus on l_1 paths. The link (4,5) can be first marked as l_1 because that assignment covers (i.e., satisfies the P-Level constraint of) the highest number of l_1 paths. Then, link (1,2) is detected as l_1 because it is the only remaining link that can cover the P-Level of $p_{1,4}$. After covering all l_1 paths, we can classify all remaining links on those paths as good (e.g., link (3,4) on $p_{3,2}$). We then consider the second instance of the problem, which focuses only on l_2 paths. Link (3,6) is determined as l_2 because it covers the highest number of l_2 paths. Finally, we also classify link (2,3) as l_2 because it is the only remaining link that can cover $p_{1,2}$. Any remaining links are classified as l_0 .

Note that the multiple-destination Boolean tomography problem [3] is a special case of the MDT problem with only two P-levels. The former is NP-hard though, and so the MDT problem is NP-hard too. So, we propose a heuristic, referred to as *Max-Tomo*, to approximate the solution of the MDT problem with the *Max* P-Level function. Algorithm 1 describes *Max-Tomo* in detail. It first removes all links of good paths from the topology. Then it starts with l_1 paths (as the current P-Level). The algorithm works iteratively and in each iteration it greedily chooses the link that covers the largest number of paths at the current P-level. The process is repeated for the next higher P-Levels in order until there is no remaining bad path in the network.

The run-time complexity of *Max-Tomo* is $O(|\Pi| |E|)$. The actual running time for a realistic topology (based on PlanetLab measurements) with about 870 paths and 1500 links (where about 10% of paths were bad) is less than 1.5msec on a workstation with a 2.6GHz Intel i5 processor. Using the approximation ratio for the Min-Set-Cover problem [5], we conclude that the approximation ratio for the solution to the i^{th} instance is $\log|\Pi_i|$, where Π_i is the set of l_i paths. Since $\sum_{\forall l_i \in L} |\Pi_i| = |\Pi|$, the approximation ratio over all instances is maximized when the number of paths at different P-Levels is the same, i.e. $|\Pi_i| = |\Pi|/Q$. Therefore, the approximation ratio of *Max-Tomo* is $Q \log(|\Pi|/Q)$.

IV. MAX-COUNT P-LEVEL FUNCTION

The *Max* P-Level function does not capture that the presence of more than one bad links in a path can make the performance of that path worse than the performance of its worst link. We should also consider that in practice it is unlikely that an Internet path has more than few bad links [15], and so the P-Level of a path would probably not be much higher than the P-Level of its worst link. Along these lines, we propose a

Algorithm 1 Max-Tomo

Require: Set of P-Levels $L = \{l_0, \dots, l_Q\}$
Require: Set of all links E
Require: Set of all paths Π
Require: The measured P-Level of every path in Π , $m_{i,j}, \forall p_{i,j} \in \Pi$

- 1: Initialize $U = E$ {set of unexplained links}
- 2: Remove from U every link e on a good (l_0) path
- 3: **for** $t = 1$ **to** Q **do** {Go over all P-Levels in L , from l_1 to l_Q }
- 4: Initialize $\Pi_t = \{p_{i,j} | m_{i,j} = l_t\}$ {set of unexplained l_t paths}
- 5: Initialize $H_t = \emptyset$ {set of links detected as l_t link}
- 6: **while** $\Pi_t \neq \emptyset$ **do**
- 7: **for** each link e in U **do**
- 8: $C(e) =$ set of paths in Π_t containing e
- 9: $score(e) = |C(e)|$ {number of paths in $C(e)$ }
- 10: **end for**
- 11: $e_m = \operatorname{argmax}_{e \in U} score(e)$ {a link with the maximum score}
- 12: $H_t = H_t \cup \{e_m\}$ {Add e_m to the set of l_t links}
- 13: $\Pi_t = \Pi_t - C(e_m)$ {Now, all paths in $C(e_m)$ are explained}
- 14: $U = U - \{e_m\}$
- 15: **end while**
- 16: Remove from U every link e on a l_t path
- 17: **end for**
- 18: **return** $H = \{H_1, \dots, H_Q\}$

second P-Level function that considers the maximum P-Level l_m among all links in the path. If the number of links at P-Level l_m (the “max-count” of the path) is less than a parameter k (the “max-count threshold”), the P-Level of the path is also l_m . Otherwise, if the max-count is larger or equal than k , the P-Level of the path increases to the next higher layer l_{m+1} (as long as $m < Q$). We refer to this P-Level function as *Max-Count*(k). More precisely, if $W_{i,j}$ is the set of links at P-Level l_m in path $p_{i,j}$, the *Max-Count*(k) function classifies the path at P-Level $m_{i,j}$, as follows:

$$m_{i,j} = \begin{cases} l_m & \text{if } |W_{i,j}| < k \text{ or } l_m = l_Q, \\ l_{m+1} & \text{if } |W_{i,j}| \geq k \end{cases} \quad (4)$$

In Section V, we discuss how to compute the set of P-Level boundaries B in the case of loss rate measurements so that *Max-Count* function classifies paths properly.

To understand the difference between the *Max* and *Max-Count* functions, consider the example of the previous section and suppose that $k = 2$. As in the previous example, links (4,5) and (1,2) are first classified as l_1 . The classification of these two links as l_1 however, also explains why path $p_{1,2}$ is l_2 . Hence, the only link that should be classified as l_2 is (3,6). This example illustrates that the *Max-Count* function can cover the same set of bad paths with fewer bad links than the *Max* function. The *Max-Count* function is the same with the *Max*

function when the max-count threshold k is sufficiently large (larger than the link-count of the path we consider).

The *Max-Count MDT problem* is also NP-hard because the Exact-Set-Cover problem [4] is a particular instance of the former. Given a collection S of subsets of an element set X , the Exact-Set-Cover problem asks if there is a sub-collection $S' \subseteq S$ such that each element in X is contained in exactly one subset in S' . If we model subsets in S as links and elements in X as paths¹, we can represent any instance of the Exact-Set-Cover problem as an instance of the *Max-Count MDT problem*.

A. Solution and analysis for Max-Count

The proposed *Max-Count-Tomo* heuristic is shown in Algorithm 2. The input arguments are the same with *Max-Tomo* in Algorithm 1 and they are omitted. *Max-Count-Tomo* first removes all links of good paths from the topology, classifying them as l_0 . Then, it repeats the following process considering paths at the next higher P-Level in each iteration (starting at l_1).

There are two main differences with *Max-Tomo*:

- (1) In line-16, we ensure that the P-Level assignment for link e_m does not exceed the P-Level of any path traversing e_m . To do so, *Max-Count-Tomo* does not remove links of l_t paths at the end of the t^{th} iteration because it is possible that some of them will be classified as l_t in the $(t + 1)^{\text{th}}$ instance.
- (2) After having computed the P-Level of link e_m , we check in lines 21-24 whether any remaining paths at P-Levels l_t and l_{t+1} can be now covered based on the P-Level of e_m ; if so, we remove them.

The *Max-Count-Tomo* algorithm approximates the solution of the *Max-Count MDT problem* in two ways. First, as in the case of *Max-Tomo*, it may not return the minimum number of bad links. Second, it can only approximate the constraint of the *Max-Count* function so that a path is at P-Level l_t if it includes at least one l_t link, or at least k l_{t-1} links.

The run-time complexity of *Max-Count-Tomo* is also $O(|\Pi||E|)$. In practice, the *Max-Count-Tomo* algorithm runs almost equally fast with *Max-Tomo*. We were not able to compute a bound for the approximation ratio of *Max-Count-Tomo*. Because the *Max-Count* function captures the relationship between successive P-Levels, the number of bad links it returns is always smaller or equal to the number of bad links returned by *Max-Tomo*.

V. P-LEVEL BOUNDARIES

As described in Section II, the classification of links and paths to Q different P-Levels is based on a set of boundaries $B = \{b_1, \dots, b_{Q-1}\}$. In this section, we discuss the selection of these boundaries for the *Max* and *Max-Count* P-Level functions. In the latter, we also describe an approach to choose the max-count threshold k .

¹A link e is included in path p if the element that corresponds to p is included in the subset that corresponds to e .

Algorithm 2 Max-Count-Tomo

```

1: Initialize  $U = E$  {set of unexplained links}
2: Remove from  $U$  every link  $e$  on a good ( $l_0$ ) path
3: for each link  $e$  in  $U$  do
4:    $best(e) = \min\{m_{i,j} | l \in p_{i,j}\}$  {the best level among
     the P-Levels of all paths traversing  $e$ }.
5: end for
6: for  $t = 1$  to  $Q$  do {Examine all P-Levels in  $L$ , from  $l_1$ 
  to  $l_Q$ }
7:   Initialize  $\Pi_t = \{p_{i,j} | \forall p_{i,j} \in P, m_{i,j} = l_t\}$  {set of
  unexplained  $l_t$  paths}
8:   Initialize  $H_t = \emptyset$  {set of links detected as  $l_t$ }
9:   while  $\Pi_t \neq \emptyset$  do
10:    for each link  $e$  in  $U$  do
11:       $C(e) =$  set of paths in  $\Pi_t$  including  $e$ 
12:       $score(e) = |C(e)|$  {number of paths in  $C(e)$ }
13:    end for
14:     $e_m = \operatorname{argmax}_{e \in L} score(e)$  {a link with the maxi-
    mum score}
15:     $l_r = best(e_m)$ 
16:    if  $l_r < l_t$  then {P-Level of  $e_m$  is the minimum
    between  $l_t$  and  $best(e_m)$ }
17:       $H_r = H_r \cup \{e_m\}$ 
18:    else
19:       $H_t = H_t \cup \{e_m\}$ 
20:    end if
21:     $S_t(e_m) =$  set of paths in  $\Pi_t$  that include  $e_m$  and that
    are covered by P-Level of  $e_m$ .
22:     $S_{t+1}(e_m) =$  set of paths in  $\Pi_{t+1}$  that include  $e_m$  and
    that are covered by P-Level of  $e_m$ .
23:     $\Pi_t = \Pi_t - S_t(e_m)$ 
24:     $\Pi_{t+1} = \Pi_{t+1} - S_{t+1}(e_m)$ 
25:     $U = U - \{e_m\}$ 
26:  end while
27: end for
28: return  $H = \{H_1, \dots, H_Q\}$ 

```

Let $x(e)$ and $y(p)$ represent the continuous performance metric (e.g. delay or loss rate) of link e and path p , respectively. There is a metric function g that expresses a given performance metric of a path based on the corresponding metric of its links, i.e., $y(p) = g(\{x(e) | \forall e \in p\})$. For example, if the performance metric is available bandwidth, $y(p) = \min_{e \in p} x(e)$, while if it is queueing delay $y(p) = \sum_{e \in p} x(e)$. The path loss rate can be also approximated by the sum of link loss rates as long as there are only few lossy links in the path and their loss rate is low.²

The P-Level function f and the metric function g are not necessarily the same. In the special case that f and g are both the *Max* function, the classification of a path based on f will be identical to its classification based on g independent of the boundary set B ; in both cases the path's P-Level is determined by the worst link in the path.

²We use this approximation in the Evaluation Section.

In the general case however, the selection of the boundaries B matters. It is important to choose them so that the classification of a path using the P-Level function f is the same with the classification that would result using the metric function g and the boundaries B . To illustrate this point, suppose that f is the *Max* function, g is the *Sum* function, $Q = 2$ and $b_1 = 10$. There are two bad links in the path with metric values 5 and 8, respectively. The P-Level of the path would be mis-classified as l_0 using f , even though it should be l_1 . We refer to such errors as *classification errors*.

In the following we focus on a specific instance of this problem. Specifically, we only consider the sum function for g (i.e. $y(p) = \sum_{e \in p} x(e)$), and the *Max* and *Max-Count* functions for f . The extension of the following methods for other functions g can be done in a similar manner. Additionally, to simplify the presentation we set the number of P-Levels to $Q=3$ and normalize the two boundaries so that $0 < b_1 < b_2 < 1$. It is straightforward to consider more P-Levels or arbitrary boundary ranges.

A. Trivial boundaries

Both *Max-Tomo* and *Max-Count-Tomo* perform better, in terms of the number of inferred bad links, if paths are uniformly distributed among all P-Levels. So, the simplest way to select the boundaries in B is to ignore classification errors, and select the boundaries so that the probability that a path falls in any of the three P-Levels is roughly the same. This approach requires some prior information about the statistical distribution of the path metric y , which can be estimated from path measurements.

Let f_p denote the probability density function (pdf) for the path metric. Among all possible boundary values, we select b_1 and b_2 to satisfy the following constraints:

$$\int_0^{b_1} f_p(y) dy \approx \int_{b_1}^{b_2} f_p(y) dy \approx \int_{b_2}^1 f_p(y) dy \quad (5)$$

Note that these trivial boundaries are the same for both *Max* and *Max-Count* P-Level functions.

B. Proper boundaries

If we also have some prior information about the statistical distribution of the link metric, we can select P-Level boundaries more properly, so that the probability of classification errors is bounded. We first show how to do this in the case of the *Max* P-Level function, and then in the case of *Max-Count*.

1) *Max P-Level function*: Suppose that f_l is the probability density function (pdf) for the link-level metric. Also, F_l and F_p are the cumulative density functions (cdf) for the link-level and path-level metric, respectively. Y is a continuous random variable representing the path metric while X is a continuous random variable representing the highest link metric among all links in that path. Suppose we first calculate a boundary b to partition the range $[0,1]$ to two P-Levels; any metric below b is mapped to l_1 ; otherwise it is mapped to l_2 . If the path metric Y is larger than b , the *Max* P-Level function assumes

that X is also larger than b . Therefore, a proper value of b should keep the following *error probability* small:

$$P(X \leq b | Y > b) \quad (6)$$

We can estimate this probability using Bayes' rule:

$$P(X \leq b | Y > b) = \frac{P(Y > b | X \leq b)P(X \leq b)}{P(Y > b)} \quad (7)$$

where $P(X \leq b) = F_l(b)$ and $P(Y > b) = 1 - F_p(b)$.

To compute $P(Y > b | X \leq b)$ we rely on the convolution of the corresponding per-link pdf's, assuming that the metrics of different links are independent and identically distributed. Without loss of generality consider only two bad links, i.e., $Y \simeq X_1 + X_2$ where X_1 and X_2 are the metric random variables associated with two bad links. Then,

$$P(Y = y | X \leq b) = \int_{x \leq b, y-x \leq b} f_l(x) f_l(a-x) dx \quad (8)$$

and, finally:

$$P(Y > b | X \leq b) = \int_b^{+\infty} P(Y = y | X \leq b) dy \quad (9)$$

So, the algorithm to compute the P-Levels boundaries for the Max function when the metric function g is given by a sum of link metrics is as follows:

a) Calculate the error probability in Equation (6) for every possible boundary (between 0 and 1) and reject any value of b that results in a significant error probability (say more than 10%).

b) From the remaining values of b , choose b_1 and b_2 to satisfy the constraints of Equation (5) to the largest possible degree.

2) *Max-Count P-Level function*: Suppose we have already computed a P-Level boundary b' and we now want to compute the next higher boundary b ($b > b'$). The initial value of b' can be set to the highest metric value that a good link or path can have.

Let $N_{b',b}$ be a random variable representing the number of links in a path whose link metric is between b' and b . The *Max-Count* P-Level function assumes that any combination of k (or more) links whose metric is between b' and b would make the path metric greater than b . On the other hand, if the number of such links is less than k and there is no any other link whose metric is more than b , the path metric should be no larger than b . Therefore, a proper value of b should keep the following two error probabilities small:

$$P(Y \leq b | X \leq b, N_{b',b} \geq k) \quad (10)$$

$$P(Y > b | X \leq b, N_{b',b} < k) \quad (11)$$

These two probabilities can be computed similarly with the *Max* function case. For instance, if $Y = X_1 + X_2$ and $k=2$, the probability in (10) is $\int_0^b P(Y = y | X \leq b) dy$ and the probability in (11) is zero.

So, the algorithm to compute the P-Levels boundaries for the Max-Count function when the metric function g is given by a sum of link metrics is as follows:

a) Set b' to the highest metric value that a good link or path can have.

b) Calculate the error probabilities in (10) and (11) for values of b between b' and 1. Reject any value of b that results in a significant error probability (say more than 10%).

c) From the remaining values, select the value of b that makes the probability that the path metric is in $[b', b]$ roughly $1/Q$.

d) Set $b' = b$ and repeat the previous process to find the next higher boundary value.

If the P-Level threshold k is not known in advance, we can calculate the previous error probabilities for few different values of the integer k (say between 1 and 5), and choose the value of k that gives the lowest sum of the two error probabilities (10) and (11).

VI. EVALUATION

This section presents an evaluation study of the proposed algorithms using computational experiments on real topologies. Here, we only focus on the loss rate metric.

A. Evaluation model

We have developed a packet-level event-driven simulator to evaluate tomography algorithms under various network conditions. In each experiment, we first fix the number of links that are congested to c . We vary c over several experiments to generate different levels of congestion on the network. We then select a given number of network links as lossy links and assign an initial loss rate to each of them. The link loss rates are drawn from a log-normal distribution with mean 0.04 and standard deviation 0.1. With these parameters, the log-normal distribution gives us roughly the same path loss rates we measured between several PlanetLab hosts. We set the maximum loss rate to 0.2, so all values above 0.2 are set to 0.2. The loss rate of all remaining links are assumed to be zero. Once each link is assigned a loss rate, we utilize a Bernoulli process with the corresponding loss probability to decide the loss of each packet³. So, even though we do not simulate a particular measurement method, we capture that the *actual* loss rate at a link during a short measurement period may be different than the assigned (long-term) loss rate.

We simulate 2000 probing packets in each path, sending 10 packets per second, and measure the actual loss rate in a path based on the fraction of lost probing packets. We assume that a path with a loss rate lower than 0.001 is good (i.e. $b_0 = 0.001$); otherwise it is "bad" or "lossy". We consider 4 P-Levels (i.e. $L = \{l_0, l_1, l_2, l_3\}$). We classify each path based on its loss rate into a P-Level using two sets of level boundaries: (1) the trivial set and (2) the proper set (see Section V). For the *Max-Count* function, the P-Level threshold k is set to 2 (based on the method described in Section V).

We consider three real router-level topologies. Two of them (Internet2 and ESnet) have been provided to us by

³We also run simulations with a Gilbert loss process where the link fluctuates between good and bad states and only drops packets when in bad state (the transition probabilities between good and bad states were set according to [22]). The difference between the results of Bernoulli and Gilbert losses is insignificant. Hence, we only report results of Bernoulli losses.

the corresponding operators. The third was obtained running traceroute between 30 PlanetLab hosts. Because some routers do not respond to TTL-EXCEEDED packets, some paths in the PlanetLab topology include one or more *star-hops*, i.e., links with unknown IP address (represented by the star symbol (*) in traceroute results).

The three topologies are:

- 1) The “PlanetLab topology”, resulting from full-mesh traceroute runs between 30 PlanetLab hosts that are used as sensors. This topology includes 1553 links and 869 paths. The average path length (including star-hops) in this topology is around 15 hops.
- 2) The “Internet2 topology” interconnects 9 Internet2 sensors (located at the major hubs of Internet2). It includes 44 links (all with known IP addresses) and 72 paths. The average path length in this topology is around 4 hops.
- 3) The “ESnet topology” interconnects 22 ESnet sensors (located at the major hubs of ESnet). It includes 117 links (all with known IP addresses) and 382 paths. The average path length in this topology is around 6 hops.

A closer look at above topologies shows that in each topology there are some links that have the same *path cover set* (i.e., the same set of paths traversing them). When two or more links have the same path cover set, even an optimal tomography algorithm would not be able to differentiate between them and detect which link among them (if any) is bad. Hence, like previous works [15], [22], we group links with the same path cover set together⁴ and localize *bad groups* instead of bad links. Clearly, the size of groups is a topological property of the underlying network. For instance, the average size of groups is around 1.8 in our PlanetLab topology, while it is near 1 in ESNet and Internet2.

Table I shows the trivial and proper sets of the P-Level boundaries (b_1, b_2) that we use in the rest of this study for each of the above topologies. These boundaries have been computed using the previously mentioned log-normal loss rate distribution.

TABLE I: Trivial and proper P-Level boundaries b_1, b_2 .

	Trivial	Proper Max	Proper Max-Count
Internet2	0.01, 0.04	0.01, 0.05	0.01, 0.03
ESnet	0.01, 0.04	0.005, 0.05	0.005, 0.02
PlanetLab	0.008, 0.03	0.005, 0.05	0.01, 0.03

The evaluation is based on the following metrics. Let I be the set of actual bad links and O be the set of lossy links inferred by a tomography algorithm. Also, A is the set of lossy links whose P-Level is correctly estimated. The *precision* is defined as the ratio of links that are correctly identified as lossy to the total number of inferred lossy links, i.e. $precision = \frac{|I \cap O|}{|O|}$. The *recall* is the ratio of links that are detected correctly as lossy to the actual number of

⁴The groups of links over a path correspond to the Minimal Identifiable Link Sequence (MILS) defined in [23] to refer to the smallest segment of the path whose loss rate can be uniquely identified through end-to-end measurements.

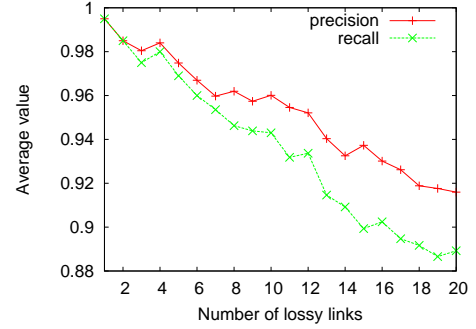


Fig. 2: Precision and recall results obtained by *Max-Tomo* in the PlanetLab topology.

lossy links, i.e. $recall = \frac{|I \cap O|}{|I|}$. The link precision and recall parameters capture the *false positive* and *false negative* probabilities, respectively. In general, reducing the frequency of false negatives in network tomography is more important. Another evaluation metric is the *accuracy* which is defined as the ratio of inferred bad links whose P-Level is correctly classified, i.e. $accuracy = \frac{|A|}{|I \cap O|}$.

We compare our algorithms with two other tomography techniques: Boolean *Tomo* [3] and the analogue *Norm* method (L1-norm minimization with non-negativity constraints) [21]. *Norm* starts with equations imposed by the metric function g (i.e. $y(p) = g(\{x(e)|\forall e \in p\})$) and solves them using the the norm minimization approach, which estimates link loss rates that may not exactly satisfy these equations but minimize the corresponding error, favoring solutions that involve fewer lossy links. There are some recent tomography algorithms, such as Netscope [15] and LIA [22], which basically apply the same norm minimization approach, but additionally they also exploit the variance of link loss rates over multiple measurement snapshots, assuming that more lossy links remain lossy during more measurement snapshots. We do not rely on such variance measurements because they require multiple measurement snapshots (with 1,000-10,000 probing packets in each snapshot), and so they are more sensitive to non-stationarities. We intend to localize short-term congestion events, and so we only compare the proposed methods with the original *Norm* algorithm.

The following experiments are repeated 200 times. The standard error for all results in this section is negligible (mostly between 0.005 and 0.01) and so we simply show the average across all runs.

B. Evaluation results

We start with the *Max-Tomo* algorithm and the PlanetLab topology. Figure 2 shows the precision and recall metrics with *Max-Tomo* (using the “proper boundaries”) when the number of lossy links increases from 1 to 20. We observe a gradual degradation in both metrics to about 90% when the network has 20 bad links. The accuracy results (not shown here) indicate that *Max-Tomo* can also detect the P-Level of the detected lossy links. Specifically, if a link is detected correctly

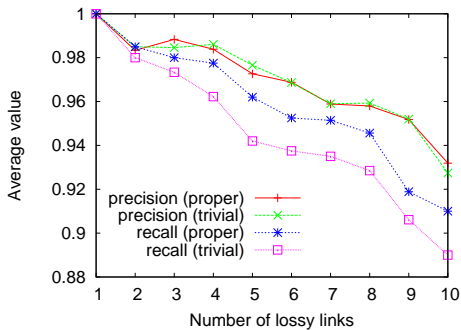


Fig. 3: Precision and recall results using *Max-Count-Tomo* with trivial and proper boundaries in the Internet2 topology.

by *Max-Tomo* as bad, then with a very high probability (above 95%) its P-Level is determined correctly as well.

Figure 3 shows the precision and recall achieved by the *Max-Count-Tomo* method with two sets of level boundaries, “trivial” versus “proper”, in the Internet2 topology. Since the number of links in the Internet2 topology is much smaller than in the PlanetLab topology, we have also considered a smaller number of lossy links (up to 10 lossy links, which is about 20% of the total links). *Max-Count-Tomo*, using the proper boundaries, can detect more than 91% of lossy links, while at most 7% of the detected bad links are actually good. In addition, the accuracy (not shown here) is always higher than 95%. The precision curves achieved with trivial and proper boundaries are very close, but the recall obtained with the proper boundaries is on average 2% higher than that with trivial boundaries. With the former, the ratio of paths that the *Max-Count-Tomo* method classifies at a wrong P-Level reduces from 8% to about 4% (with 10 bad links). Compared to *Max-Count-Tomo*, the selection of proper boundaries in *Max-Tomo* has a smaller impact on the ratio of paths that are classified at a wrong P-Level (it is improved by about 2%). We observe roughly the same differences between trivial and proper boundaries in the two other topologies.

Figure 4 compares the results of the two proposed algorithms (using proper boundaries) with the results of two previously published algorithms: the Boolean *Tomo* method [3] and the analogue *Norm* method [21]. These results refer to the ESnet topology, but the two other topologies produce similar results. The precision and recall are shown separately in Figure 4a and Figure 4b, respectively.

First, if we compare the results of *Max-Tomo* and *Max-Count-Tomo*, we see that the recall with both algorithms is very similar. The precision achieved with *Max-Count-Tomo*, however, is much better than with *Max-Tomo* (by about 4%). As discussed earlier, *Max-Count-Tomo* is able to explain bad paths with fewer links. For example, it returns about 5% fewer links than *Max-Tomo* in ESnet. Generally, *Max-Count-Tomo* has a higher potential than *Max-Tomo* in the detection of bad links, especially when there are many lossy links in the network.

Next, we compare the two proposed algorithms (*Max-Tomo* and *Max-Count-Tomo*) with *Tomo* and *Norm*. The *Tomo* algorithm can achieve a good precision (it is only 2% worse than *Max-Count-Tomo*), while the *Norm* method gives better recall results (it is up to 4% better than *Max-Count-Tomo*). However, the proposed algorithms (especially, the *Max-Count-Tomo* method) outperform the two earlier methods when we consider both recall and precision. For instance, the recall achieved by *Max-Count-Tomo* in ESnet with 20 bad links is about 10% better than that with *Tomo*, and its precision is about 17% better than that with *Norm*. This is because *Tomo* justifies bad paths with fewer links (about 10% less than the actual number of bad links), while *Norm* “explains” the loss rate of bad paths with more lossy links (about 25% more than the actual number of bad links). The difference between the precision of *Max-Count-Tomo* and *Norm* becomes even larger in the PlanetLab topology (by about 30%). Moreover, the recall of *Max-Count-Tomo* is up to 20% better than that with *Tomo* in the Internet2 topology.

The previous results confirm the potential of multi-level tomography compared to earlier Boolean and analogue approaches, and corroborate the idea that using coarse information about the performance level of paths helps significantly to localize bad links more accurately. In addition to better localization performance, the multi-level tomography methods give accurate information about the actual performance level of links. For instance, the accuracy of our algorithms is more than 95% when we consider up to 20 bad links in ESNet.

VII. RELATED WORK

Network tomography refers to the inference of internal network properties, such as layer-2 topology or link-level loss rate and delay variation, from end-to-end measurements [6], [7], [8]. We refer the reader to a thorough review for a more complete coverage of the prior work in this area [9].

In general, there are two classes of network tomography approaches in the literature: Analogue (or continuous) and Boolean (or binary) [19]. In continuous variable tomography, a typically under-constrained linear system of equations is used to model the relation between all 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. For instance, Shavitt et al. [10] estimate the delay of links using a least-squares method, while Bu et. al [11] use expectation-maximization to infer link loss rates. Chen et al. [12] use Fourier domain analysis to infer link delays in a computationally efficient manner. NetScope [15] is a recent tomographic method that estimates link loss rates also considering the observed variance of loss rates.

Analogue tomography methods would obviously be ideal. Their application in practice, however, has been quite limited for several reasons. First, they require accurate end-to-end path measurements (instead of only estimating a discrete P-level for each path). Second, some link-level parameters may not

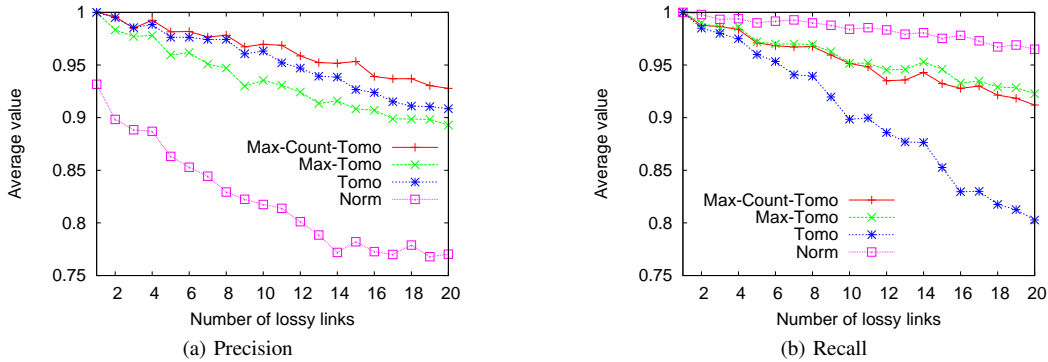


Fig. 4: The results obtained by *Max-Count-Tomo*, *Max-Tomo*, *Tomo*, and *Norm* in the ESnet topology.

be statistically identifiable, meaning that different sets of link-level parameters produce the same statistical distribution of path measurements [1], [2], [18]. Third, Analogue tomography methods can be computationally intensive and prone to numerical instabilities [1].

To simplify network tomography and make it more practical, Duffield [1], [18] introduced the Boolean tomography framework in 2003. In follow-up work, NetDiagnoser [3] extends binary tomography to the case of multiple sources and multiple destinations. Kompella et al. [13] consider a similar approach to detect “silent failures” in MPLS/IP networks, by using active measurements between edge routers. Nguyen and Thiran consider the problem of localizing lossy links using passive measurements of end-to-end traffic [14]. The same authors introduced a Boolean method to infer link state probabilities from multiple measurements over time, using those probabilities to then identify congested links [2]. Another approach for inferring lossy links is to use Bayesian technique (with Gibbs sampling) to determine a posteriori distribution about link loss rates in the network [16]. The authors in [17] use prior link state probabilities for diagnosing the underlying cause behind the faulty state of links.

VIII. CONCLUSIONS

We proposed a multi-level discrete tomography framework to localize performance problems, capturing the wide diversity in the performance of Internet paths. We introduced two tomography methods: *Max-Tomo* that considers only with the lowest-performance link, and *Max-Count-Tomo* that also considers the number of such links in a path. The evaluation results show that the proposed multi-level tomography algorithms perform better than earlier Boolean and analogue methods when we consider both precision and recall. The *Max-Count-Tomo* generates up to 5% lower false positive error rate than *Max-Tomo*, especially when the network has many bad links. Both algorithms can classify correctly the actual performance level of more than 95% of the inferred bad links.

REFERENCES

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

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

[3] 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.

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

[5] D. Johnson. Approximation Algorithms For Combinatorial Problems. In *Jnl. of Comp. and System Sciences*, 1974.

[6] R. Caceres, N. G. Duffield, J. Horowitz, D. F. Towsley, and T. Bu. Multicast-Based Inference of Network-Internal Characteristics: Accuracy of Packet Loss Estimation. In *Proc. IEEE INFOCOM*, 1999.

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

[8] M. Coates, R. Castro, R. Nowak, M. Gadhio, R. King, and Y. Tsang. Maximum Likelihood Network Topology Identification from Edge-based Unicast Measurements. In *Proc. ACM SIGMETRICS*, 2002.

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

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

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

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

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

[14] H. Nguyen, and P. Thiran. Using end-to-end data to infer lossy links in sensor networks. In *Proc. IEEE INFOCOM*, 2006.

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

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

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

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

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

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

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

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

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