Network Loss Inference with Second Order Statistics of End-to-End Flows

Hung X. Nguyen
School of Computer and Communication Sciences, EPFL
CH-1015 Lausanne, Switzerland
hung.nguyen@epfl.ch

Patrick Thiran
School of Computer and Communication Sciences, EPFL
CH-1015 Lausanne, Switzerland
patrick.thiran@epfl.ch

ABSTRACT

We address the problem of calculating link loss rates from end-to-end measurements. Contrary to existing works that use only the average end-to-end loss rates or strict temporal correlations between probes, we exploit second-order moments of end-to-end flows. We first prove that the variances of link loss rates can be uniquely calculated from the covariances of the measured end-to-end loss rates in any realistic topology. After calculating the link variances, we remove the un-congested links with small variances from the first-order moment equations to obtain a full rank linear system of equations, from which we can calculate precisely the loss rates of the remaining congested links. This operation is possible because losses due to congestion occur in bursts and hence the loss rates of congested links have high variances. On the contrary, most links on the Internet are un-congested, and hence the averages and variances of their loss rates are virtually zero. Our proposed solution uses only regular unicast probes and thus is applicable in today's Internet. It is accurate and scalable, as shown in our simulations and experiments on PlanetLab.

Categories and Subject Descriptors

C.2.3 [**Computer-Communication Networks**]: Network Operations—*Management, Monitoring*

General Terms

Management, Measurement, Performance

Keywords

Network Tomography, Identifiability, Inference

1. INTRODUCTION

Many IP network inference problems are ill-posed: the number of measurements are not sufficient to determine uniquely their solution. For example, the traffic matrix estimation problem is finding the Origin-Destination (OD) pairs of traffic flows from the link counts. As the number of OD pairs far exceeds the number of links, the resulting system of equations is under-determined. Various heuristics, such as the gravity model, can then be used to reduce the set of possible solutions.

Another tomography problem, which we address in this paper, is to compute the loss rates of IP links from end-to-end path measurements. This problem, unfortunately, is also under-determined. Different methods to overcome the identifiability issue have been proposed in the literature, which we review in Section 2. For several practical reasons these approaches have not been widely used in large scale applications. For those methods that use multicast, the multicast routers are not widely deployed. Methods based on active unicast packet trains present development and administrative costs associated with deploying probing and data collection. Other approaches that achieve more pragmatic targets, such as detecting shared congestion, locating congested links or calculating loss rates of groups of links, cannot provide fine grain information required by many applications.

We are interested in inferring IP link loss rates using only the readily available end-to-end measurements. We argue in this paper that the regular unicast probes provide enough information to do so. More precisely, instead of artificially introducing temporal correlations among the probe packets with multicast, we seek to exploit the spatial correlations that already exist among the regular unicast probes. Our approach is based on two key properties of network losses. First, losses due to congestion occur in bursts and thus loss rates of congested links have high variances. Second, loss rates of most un-congested links in the Internet have virtually zero first- and second-order moments.

After having described the performance model and assumptions in detail in Section 3, we first prove our main result in Section 4: the link variances are statistically identifiable from end-to-end measurements of regular unicast probes in general mesh topologies. This is in sharp contrast with what can be achieved with the mean loss rates of network links, which are not statistically identifiable from end-to-end measurements. This result shows that these variances can be correctly learnt from a sufficient large set of end-to-end measurements if we have a proper learning method. We provide such an algorithm in Section 5.1. It combines the end-to-end loss rates to provide a linear system of equations relating the link variances with the covariances of the end-to-end loss rates. This system of equations can then be

IMC'07, October 24-26, 2007, San Diego, California, USA.

solved in an accurate and scalable way using standard linear algebra methods.

At the end of this learning phase, we can use the link variances as additional information together with the most recent measurement to calculate the link loss rates. The method boils down to exploiting a monotonic dependence between the mean and variance of a link loss rate. Specifically, we first sort the links according to their congestion levels using their variances. By eliminating the least congested links (with smallest variances) from the first-order moment equations we can then obtain a reduced system of full column rank and can solve it in order to find the loss rates of network links as showed in Section 5.2. Because most links on the Internet are un-congested, this operation is very accurate.

Finally, the simulations in Section 6 and Internet experiments in Section 7 show that our algorithm is both accurate and scalable.

2. RELATED WORK

The inference of internal link properties from end-to-end measurements is called *Network Performance Tomography* [14]. In this paper we are interested in a particular network performance tomography branch that infers link loss rates from end-to-end measurements. This problem, unfortunately, is under-determined when we use only the average end-to-end loss rates (i.e., the first moment) as shown in Figure 1.

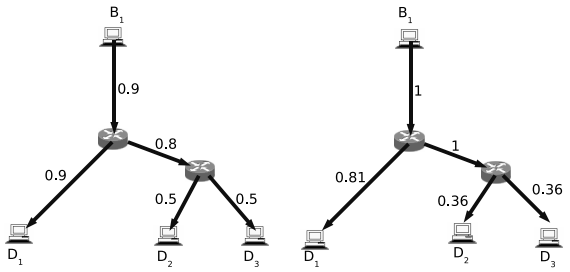

Figure 1: A network with three end-to-end paths from the beacon B_1 to each destination D_1, D_2 and D_3. The number next to each link denotes the transmission rate of the link. Both sets of link transmission rates give the same end-to-end transmission rates. Link transmission rates therefore cannot be uniquely calculated from the average end-to-end transmission rates.

To overcome the un-identifiability problem, we need to either bring in additional constraints or to look for more pragmatic goals. Most existing end-to-end tomography methods fall into one of two classes: methods that use strong temporal correlation between probing packets in a multicast-like environment [1, 6, 7, 9, 12, 13] (see [11] for a review of these methods), and methods that exploit the distribution of congested links in the Internet [14, 22, 24] for a more pragmatic goal of identifying the congested links.

The initial methods in the first class [1, 6, 7] infer the loss rate of network links using multicast probing packets. Most recently, [9] shows that all moments of link delays, except the means, are statistically identifiable in a *multicast tree*. However, statistical identifiability in general topologies remains to be a conjecture in [9]. As multicast is not widely deployed in the Internet, subsequent methods [12,13,29] emulate this approach using clusters of unicast packets. These methods are less accurate than their multicast counterparts and also require substantial development and administrative costs. Furthermore, the iterative algorithms used to compute link loss rates in these approaches are expensive for real-time applications in large networks.

Methods in the second class [14,22,24,36] use only unicast end-to-end flows for a simpler goal of identifying the congested links. As different loss rate assignments are possible, these methods use additional information or assumptions. For example, the methods in [14, 24] identify the congested links by finding the smallest set of links whose congestion can explain the observed measurements. These methods essentially use two assumptions: (i) network links are equally likely to be congested, and (ii) the number of congested links is small. In [22], we propose to use multiple network measurements in order to learn the probabilities of network links being congested. Using these probabilities, instead of the assumptions in [14, 24], the CLINK algorithm of [22] can identify the congested links with higher accuracy. Using again unicast probes, [36] by passes the un-identifiability of first order moment equations by finding the smallest set of consecutive links whose loss rates can be determined. This method, however, cannot be used to calculate link loss rates at the granularity of each individual link, even if some of these links belong to different end-to-end probing paths. In contrast, we will see in Theorem 1 that the loss variances of these links are identifiable.

A large body of research has also been devoted to detecting shared congestion of network flows. All of these studies use the correlations between different flows to identify the shared bottlenecks. In [26], cross-correlation between two flows is compared against the autocorrelation of each flow. If the former is greater than the latter, then a shared congestion is concluded, otherwise there is none. Harfoush [17] et al. use back-to-back packet pairs to calculate the correlations from which they infer shared congested links. Kim et al. [19] use a wavelet denoising technique to improve upon the work of [26]. Most recently, Arifler [4] used the spatial correlations of TCP flow throughput to infer shared congestion with a factor analysis model.

We summarize the existing end-to-end tomographic techniques in Table 1. We classify them according to their inference methodologies and objectives.

Our solution in this paper is also based on end-to-end tomography. We use unicast probes in a totally different way than [12, 13, 29]: we exploit the spatial correlations between flows rather than the temporal correlations between packets as in [12, 13, 29]. The flow-level approach offers two advantages over its packet counterpart. First, the spatial correlations exist even among the flows of regular and weakly correlated unicast packets. A flow level approach can therefore be used with existing application traffic from a single end-host as it does not require access to both ends of a flow as shown in [24]. Second, our inference technique does not rely on the strong assumption that back-to-back unicast packets are strongly correlated. Hence it is more tolerant to cross traffic, contrary to the the approach of [29] that also works with passive unicast probes. Compared to the approach in [22], our solution in this paper also uses multiple measurements of unicast flows but can provide more information about network links, i.e., link loss rates rather than just their congestion statuses.

Table 1: A Summary of Techniques in Network Loss Tomography.

	Temporal Correlations		First Order Moments		Higher Order Moments	
	Multicast [6, 9]	Packet Trains [12, 13]	Prior Knowledge [14, 24]	Link Groups [36]	One Snapshot [4, 26]	Multiple Snapshots [22]
Shared Congestion	*Yes*	*Yes*	*Yes*	*Yes*	*Yes*	*Yes*
Congested Links	*Yes*	*Yes*	*Yes*	*Yes*	*No*	*Yes*
Link Loss Rates	*Yes*	*Yes*	*No*	*Yes*	*No*	*No*

There are also non-tomographic techniques for calculating link loss rates such as [3, 20]. Instead of using end-to-end measurements, these approaches use router supports to calculate link loss rates and therefore depend heavily on the cooperation of routers. For example, Tulip [20] requires routers to support continuous IP-ID generated ICMP packets. Unfortunately for security and performance issues, many routers do not respond or limit the response rates to ICMP requests. Tulip is also dependent on the cross traffic and consistently underestimates link loss rates [20]. Furthermore, these approaches require that each end-to-end path is measured separately and therefore do not scale well.

Techniques that use higher order moments for traffic matrix estimations are proposed in [8, 30]. In both of these works, the unknown variables are assumed to follow a certain parametric distribution. Our approach differs in two aspects. First, we are interested in the "dual" problem of inferring link properties given the end-to-end measurements, whereas in [8, 30] link measurements are known, but end-to-end traffic counts need to be inferred. The structures of the linear equations relating measurements with unknowns in the two problems are therefore different. Second, we do not assume any parametric distribution for the inferred variables. A more detailed comparison between the two approaches will be given in Section 5.

3. THE NETWORK MODEL

We consider an overlay inference system that consists of regular users who can send and receive UDP packets and perform traceroute [18]. End-hosts periodically measure the network topology and end-to-end loss rates by sending probes to a set of destinations $\mathcal{D}$. They also periodically report the measurement results to a central server. We call these end-hosts that send probes *beacons*, their set is denoted by $\mathcal{V}_B$.

In this section, we describe the input data that can be gathered from direct measurements: network topology and end-to-end loss rates. We also describe the performance model used by the central server to infer link loss rates. Even though the framework we present in this paper can be used to infer link delays and with passive measurements, we only consider loss rates with active probes in this paper.

3.1 Network Topology

The network topology is measured using tools such as traceroute [18]. There are several practical issues concerning the use of traceroute to obtain network topologies, which we will elaborate in the experiments of Section 7.

We model the network as a *directed* graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, where the set $\mathcal{V}$ of nodes denotes the network routers/hosts and the set $\mathcal{E}$ of edges represents the communication links connecting them. The numbers of nodes and edges are denoted by $n_v = |\mathcal{V}|$, and $n_e = |\mathcal{E}|$, respectively. Furthermore, we use $P_{s,d}$ to denote the path traversed by an IP packet from a source node s to a destination node d. Let $\mathcal{P}$ be the set of all paths between the beacons and the destinations. Any sequence of consecutive links without a branching point cannot be distinguished from each other using end-to-end measurements: These links, which we call "alias" links, are therefore grouped into a single *virtual link*. We call this operation the *alias reduction* of the topology. In the rest of this paper, we use the term "link" to refer to both physical and virtual links.

For a known topology $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and a set of paths $\mathcal{P}$, we compute the reduced routing matrix R of dimension $n_p \times n_e$, where $n_p = |\mathcal{P}|$, as follows. The entry $R_{i,j} = 1$ if the path $P_{s,d} \equiv P_i$, with $i = (s, d)$, contains the link e_j and $R_{i,j} = 0$ otherwise. A row of R therefore corresponds to a path, whereas a column corresponds to a link. Clearly, if a column contains only zero entries, the quality of the corresponding link cannot be inferred from the paths in $\mathcal{P}$. Hence we drop these columns from the routing matrix. After the alias reduction step and this removal step, the columns of the resulting *reduced routing matrix* R are therefore all distinct and nonzero. The dimensions of R are $n_p \times n_c$, where $n_c \leq n_e$ is the number of links that are covered by at least one path in $\mathcal{P}$. We denote this set of covered links by $\mathcal{E}_c$, $|\mathcal{E}_c| = n_c$.

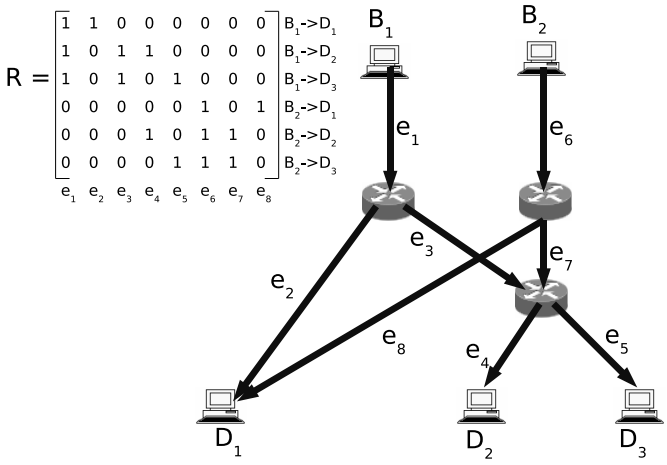

Figure 2: Two end-hosts B_1 and B_2 are beacons, D_1, D_2 and D_3 are the probing destinations.

In the example of Figure 2, each of the two beacons B_1 and B_2 sends probes to three destinations D_1, D_2 and D_3. The aggregated routing topology contains 6 end-to-end paths and 8 directed links with the reduced routing matrix R as shown in the figure.

We make the following assumptions on the reduced rout-

ing matrix R:

- **T.1 Time-invariant routing:** R remains unchanged throughout the measurement period.
- **T.2 No route fluttering:** There is no pair of paths P_i and $P_{i'}$ that share two links e_j and $e_{j'}$ without also sharing all the links located in between e_j and $e_{j'}$. That is, the two paths never meet at one link, diverge, and meet again further away at another link.

These two assumptions are needed to guarantee the identifiability of the variances of link loss rates from end-to-end measurements as we will show later.

The first assumption (**T.1**) can be violated in the Internet where routing changes can happen at any timescale. As a consequence, we will have some noise in our estimation of the routing matrix. To overcome routing changes, we could remeasure the network topologies frequently. However, we avoid this approach as it could require a significant amount of repeated traceroute measurements, which is prohibitive in many networks. We show in our experiments in Section 7 that despite the potential errors in network topology, our algorithm is still very accurate.

Assumption **T.2** can be violated if multiple paths are used to route packets from one host to another. These paths do not occur frequently and this phenomenon is called *route fluttering* [32]. We call these paths *fluttering paths*. Fluttering paths can occur for several reasons. The most common cause is load balancing at routers. However, most routers perform load balancing deterministically using flow identifications (source IP, source port, destination IP, and destination ports) [5]. Therefore, an end-to-end flow between two end-hosts, with fixed source and destination ports, does not observe these route flutterings. Another cause of route flutterings is routing failure. Significant route flutterings can occur during the convergent period when a new route is searched. These flutterings are usually transient and come with high packet loss rates. To handle these flutterings, measurements with high frequency are needed and we do not consider them in this paper. In the rare cases that long-term flutterings indeed appear, we keep only the measurements on one path and ignore the measurements on the others. To infer the performances of the links on the different fluttering paths, we repeat the inference multiple times, each time including one of the alternative paths in the routing matrix.

Under assumption **T.2**, the paths from a beacon B form a tree rooted at B. For example, the paths from beacon B_1 in Figure 2 form a routing tree rooted at B_1.

3.2 End-to-End Probes

Another type of data that can be obtained from direct measurements are the end-to-end loss rates. In this work, we use periodic probes that are sent with constant inter-arrival times. From the probes, we can calculate the end-to-end loss rates and their covariances.

To estimate the end-to-end transmission rates, each beacon in $\mathcal{V}_B$ sends S unicast probes to every destination in $\mathcal{D}$. For each path P_i, let $\widehat{\phi}_i$ be the random variable giving the fraction of these S probes that arrive correctly at the destination, and let $\widehat{\phi}_{i,e_k}$ be the fraction of these probes that have traversed link e_k successfully. Let $\widehat{\phi}_{e_k}$ be the fraction of probes from all paths passing through link e_k that have not been dropped by that link. The loss rate of path P_i is the complement to 1 of its transmission rate, defined as $\phi_i = \mathbb{E}[\widehat{\phi}_i]$. Similarly, the transmission rate of link e_k is $\phi_{e_k} = \mathbb{E}[\widehat{\phi}_{e_k}]$

Let $Y_i = \log \widehat{\phi}_i$ and $X_k = \log \widehat{\phi}_{e_k}$, which we group in vectors. In this paper, we use a bold letter to represent a vector, for example $\mathbf{Y} = [Y_1\ Y_2 \dots\ Y_{n_p}]^T$ and $\mathbf{X} = [X_1\ X_2 \dots\ X_{n_c}]^T$, where T denotes transposition.

We make the following assumptions that are necessary to establish the linear relatioship between end-to-end loss rates and link loss rates:

- **S.1 Identical sampled rates:** $\widehat{\phi}_{i,e_k} = \widehat{\phi}_{e_k}$ almost surely (a.s.) for all paths P_i that traverse e_k.
- **S.2 Link independence:** The random variables X_k are independent.

Assumption **S.1** means that the fraction of packets lost at link e_k is the same for all paths P_i traversing the link, which holds by the law of large numbers provided S is large enough. However we do not want to take S too large, as we want to exploit the stochastic variability of these variables in different snapshots, as we will see later. Hence this assumption is actually an approximation, which is valid when

$$\mathbb{E}[(\widehat{\phi}_{e_k} - \widehat{\phi}_{i,e_k})^2] \ll \mathbb{E}[(\widehat{\phi}_{e_k} - \phi_{e_k})^2]. \quad (1)$$

Such an approximation is reasonable because a congested link e_k will experience different congestion levels at different times (hence a large value of the right hand side of (1), which is the variance of $\widehat{\phi}_{e_k}$); whereas all paths P_i traversing e_k in the same period of time will have very similar sample means of lost packets (hence a small value of the left hand side of (1)).

Assumption **S.2** may also not apply to all links, however previous works [14, 24] show that the correlation of network links is weak and does not significantly affect the accuracy of their diagnosis. Under this assumption, the covariance matrix of $\mathbf{X}$ is diagonal and reads

$$\Gamma_{\mathbf{X}} = \mathrm{diag}(\mathbf{v}) = \mathrm{diag}([v_1\ v_2\ \dots v_{n_c}]), \quad (2)$$

where $v_k = \mathrm{VAR}(X_k)$.

Using these two assumptions, given the network topology and end-to-end path loss rates, we can then easily establish linear relationships between the link loss rates and the path loss rates as (a.s.)

$$\mathbf{Y} = R\mathbf{X}. \quad (3)$$

To determine the link loss rates, we need to solve the system of linear equations (3). Recall here that it does not have a unique solution because in most networks the matrix R is rank deficient, that is, $\mathrm{rank}(R) < \min(n_p, n_c)$. In the example of Figure 2, $\mathrm{rank}(R) = 5 < \min(6, 8)$. In this paper, we overcome the rank deficient problem of (3) by exploiting spatial correlations between different path measurements.

3.3 Performance Model

To study the correlation between the path measurements, we divide the measurement time into slots, each of duration S probes. The collections of all measurements on all end-to-end paths in a given time slot, obtained by sending S probes from each beacon in $\mathcal{V}_B$ to each probing destinations in $\mathcal{D}$, is called a *snapshot* of the network. We use m snapshots $\mathcal{Y} = \{\mathbf{y}^1, \mathbf{y}^2, ..., \mathbf{y}^m\}$ to calculate the covariances between

different path loss rates. Let Σ be the covariance matrix of $\mathbf{Y}$, i.e.,

$$\Sigma = \begin{bmatrix} \sigma^2_{Y_1} & \text{COV}[Y_1, Y_2] & \dots & \text{COV}[Y_1, Y_{n_p}] \\ \text{COV}[Y_2, Y_1] & \sigma^2_{Y_2} & \dots & \text{COV}[Y_2, Y_{n_p}] \\ \vdots & \dots & \ddots & \vdots \\ \text{COV}[Y_{n_p}, Y_1] & \text{COV}[Y_{n_p}, Y_2] & \dots & \sigma^2_{Y_{n_p}} \end{bmatrix}.$$

We make the following assumption about the link variances $\mathbf{v}$

- **S.3 Monotonicity of variance:** The variance v_k of X_k is a non-decreasing function of $1 - \phi_{e_k}$.

This assumption holds in real networks as shown in all previous Internet measurements [31, 35] where high average loss-rates are always associated with high loss-rate variations. The assumption definitely holds in our own measurements of the PlanetLab network over one day as shown in Figure 3.

The idea of dividing measurements into multiple time slots was first suggested in [14] and is used in [22] to study binary network performances. By dividing measurements into small time slots, we can exploit the stochastic variations of link loss rates, which will prove essential to classifying links as congested or not - thanks to Assumption **S.3** (in the process explained in Section 5.2.) Averaging the estimated loss rates over the whole measurement duration would miss these short term variations in network links and would eventually end up in an under-determined system as in (3). But we need to choose S large enough, so that Assumption **S.1** holds. Choosing the optimal duration S requires a complete knowledge about link loss processes at small time-scales and the effect of active probes on the network. These questions have been partially addressed in the literature [25, 27] and are outside the scope of this paper. Instead, we use a heuristic that chooses $S = 1000$ in our simulations and experiments. This heuristic may not be optimal, but it is reasonable for the dynamics of the Internet losses as shown in [35]. We will see in Section 6.3 that our scheme is robust to different values of S.

4. IDENTIFIABILITY OF THE LINK VARIANCES

In this section, we show that the variances $\mathbf{v}$ of the link performances $\mathbf{X}$ can be uniquely identified from end-to-end measurements in any network topology that satisfies assumptions **T.1-2**. We call this property of the network link loss-rates the statistical *identifiability* property. This property is important because it guarantees that by using a correct inference algorithm we can accurately calculate the link variances from the measured data.

Let us formulate this property mathematically. Denote by $\Sigma_{\mathbf{v}}$ the covariance matrix of the measurements when the link variances are $\mathbf{v}$. The link variances are statistically identifiable if and only if the condition

$$\Sigma_{\mathbf{v}} = \Sigma_{\tilde{\mathbf{v}}} \text{ for any sequence of snapshots } \mathbf{y}$$

always implies that $\mathbf{v} = \tilde{\mathbf{v}}$.

Our interest in this section, is to find $\mathbf{v}$, given the m snapshots and Assumptions **T.1-2** and **S.1-3**. As we will show later, all of the Assumptions **T.1-2** and **S.1-3** are needed for the identification of $\mathbf{v}$ and the link loss rates.

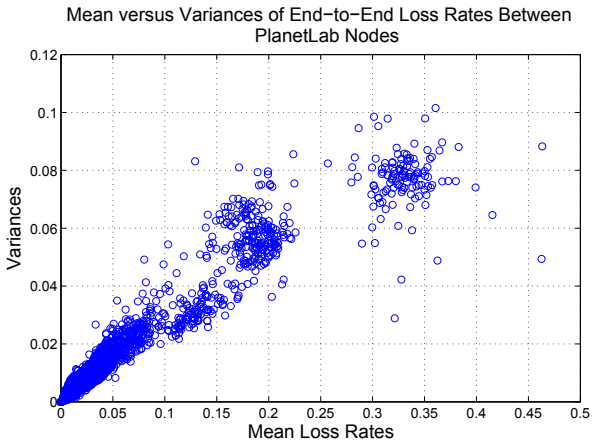

Figure 3: Relationship between the mean and variance of the loss rates on 17200 PlanetLab paths over one day period on April 20th, 2007. Each path loss rate is measured repeatedly on average every five minutes. To measure a path loss rate, we send 1000 UDP probing packets over a period of 10 seconds between the end-hosts. The mean and variance of each path loss rate are calculated using 250 loss rate samples.

From (2) and (3),

$$\Sigma = R\Gamma_{\mathbf{X}}R^T = R\text{diag}(\mathbf{v})R^T. \quad (4)$$

For a matrix R, we write the ith row of R as $\mathbf{R}_{i*}$ and the jth column of R as $\mathbf{R}_{*j}$. That is

$$R = \begin{bmatrix} \mathbf{R}_{1*} \\ \mathbf{R}_{2*} \\ \vdots \\ \mathbf{R}_{n_p*} \end{bmatrix},$$

and

$$R = [\mathbf{R}_{*1}\mathbf{R}_{*2} \dots \mathbf{R}_{*n_c}].$$

We use the notation $\otimes$ to denote the element-wise product of two vectors, i.e.,

$$\mathbf{R}_{i*} \otimes \mathbf{R}_{j*} = [R_{i,1}R_{j,1} \;\; R_{i,2}R_{j,2} \;\; \dots \;\; R_{i,n_c}R_{j,n_c}].$$

DEFINITION 1. *Let A be the augmented matrix of dimension $n_p(n_p+1)/2 \times n_c$ whose rows consist of the rows of R and the component wise products of each pair of different rows from R. The rows of A are arranged as follows: $\mathbf{A}_{((i-1)\times n_p+(j-i)+1)*} = \mathbf{R}_{i*} \otimes \mathbf{R}_{j*}$ for all $1 \le i \le j \le n_p$.*

In the example of Figure 2, if we take only the measurements from beacon B_1, i.e., the network in Figure 1, then

$$R = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}, \quad A = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}.$$

To prove the identifiability of $\mathbf{v}$, we use the same strategy of considering the algebraic properties of the augmented matrix A (similar to [8]) to estimate traffic matrices of Gaussian

traffic flows. However, the two results are totally different and require different analyses. In [8], the measurements are made on the links and the unknowns are the end-to-end flows, whereas in this work the measurements are the end-to-end flows and the unknowns are the link performances. The proof in [8] is much simpler as the traffic counts on the first and last links of an OD-path share only the traffic of this OD pair. The covariance between the first and the last link of a path is therefore sufficient to retrieve the variance of the traffic count in that path. In contrast, we do not have this simple relation here. Hence we need to exploit the more complex recursive relations between path covariances and link variances to identify the latter. Our identifiability result of $\mathbf{v}$ can be viewed as the "dual" of the identifiability result in Corollary 1 of [8].

Given Definition 1, we now state and prove the following lemma.

LEMMA 1. *The equations $\Sigma = R\,diag(\mathbf{v})R^T$ are equivalent to the equations $\mathbf{\Sigma}^* = A\mathbf{v}$, where $\mathbf{\Sigma}^*$ is a vector of length $n_p(n_p+1)/2$ and $\Sigma^*_{(i-1)n_p+j-i+1} = \Sigma_{i,j}$ for all $1 \le i \le j \le n_p$.*

PROOF. The result follows from expanding $R\text{diag}(\mathbf{v})R^T$. □

LEMMA 2. *$\mathbf{v}$ is identifiable if and only if A has full column rank.*

PROOF. Assume that two variance vectors $\mathbf{v}$ and $\tilde{\mathbf{v}}$ give the same end-to-end covariance matrix Σ: $R\text{diag}(\mathbf{v})R^T = R\text{diag}(\tilde{\mathbf{v}})R^T$. This would imply that $R(\text{diag}(\mathbf{v}) - \text{diag}(\tilde{\mathbf{v}}))R^T = 0$, or equivalently from Lemma 1 that $A(\mathbf{v} - \tilde{\mathbf{v}}) = \mathbf{0}$. Hence, $\mathbf{v} - \tilde{\mathbf{v}} = \mathbf{0}$ if and only if A has full column rank. □

Let us first consider a monitoring system with a single beacon B. Let R be the (reduced) routing matrix for this system(i.e., after grouping alias IP links together and removing all-zero columns). Observe that a node in such a tree is either a leaf (i.e., a destination in $\mathcal{D}$) or has at least two children, because otherwise the edge before and the edge after the node would be alias links and would be grouped together in a virtual link.

LEMMA 3. *Let R be the routing matrix of a single beacon network. The variances $\mathbf{v}$ of the links represented by columns of R are identifiable from end-to-end measurements if Assumptions **T.1-2** hold.*

Lemma 3 is stated indirectly without a complete proof in [9]. Here, we provide the proof of this lemma as it will be used later in our proof of Theorem 1.

PROOF. Let us pick any branch of this tree, let n be its number of nodes, which we index in increasing distance from $B = v_0$, and which we denote by $v_1, v_2, \ldots, v_n$. Without loss of generality, let us place the columns corresponding to the n links $e_k = (v_{k-1}, v_k)$, $1 \le k \le n$, of this branch in the first n columns of matrices R and A. Remember that because of the reduction step described above, the $(n-1)$ nodes $v_1, v_2, \ldots, v_{n-1}$ must have at least two children, whereas v_n is a leaf.

Let us consider the first link $e_1 = (B, v_1)$. As v_1 has at least two children, we can always find two paths P_i and $P_{i'}$, with $i < i'$, which traverse e_1 and diverge at v_1, so that P_i passes though one of these two children and $P_{i'}$ through the other. Therefore $R_{i,1} = R_{i',1} = 1$ but $R_{i,j} \neq R_{i',j}$ for all $j \ge 2$, hence $\mathbf{A}_{((i-1)n_p+(i'-i)+1)*} = \mathbf{R}_{i*} \otimes \mathbf{R}_{i'*} = [1\ 0\ \ldots\ 0]$. As a result, the first column $\mathbf{A}_{*1}$ is linearly independent from all other columns of A.

We now proceed by induction for all other links. Let $2 \le k \le n-2$, and suppose that each of the k first columns of A is linearly independent from all other columns of A. Let us then consider the link $e_{k+1} = (v_k, v_{k+1})$. As the internal node v_{k+1} has at least two children, we can always find a pair of paths P_i and $P_{i'}$, with $i < i'$, which traverse $e_1, \ldots, e_{k+1}$ and diverge at v_{k+1}, so that P_i passes though one child of v_{k+1} and $P_{i'}$ through another. Therefore $A_{(i-1)n_p+(i'-i)+1,j}$ is 1 if $1 \le j \le k+1$, and 0 if $k+2 \le j \le n_c$. This implies that column $\mathbf{A}_{*(k+1)}$ can only be a linear combination of a set of columns that contains at least one of the k first columns $\mathbf{A}_{*1}, \ldots, \mathbf{A}_{*k}$. From our induction assumption, these k columns are linearly independent from any other column of A, and thus in particular from $\mathbf{A}_{*(k+1)}$. Therefore $\mathbf{A}_{*(k+1)}$ is linearly independent from all other columns of A.

Finally, the last link of the branch $e_n = (v_{n-1}, v_n)$ ends up on the leaf node v_n. This node appears therefore in only one path P_i from B to v_n, and therefore $A_{(i-1)n_p+1,j} = R^2_{i,j} = 1$ if $1 \le j \le n$, and is equal to 0 if $n+1 \le j \le n_c$. This implies again that $\mathbf{A}_{*n}$ can only be a linear combination of a set of columns that contains at least one of the n first columns $\mathbf{A}_{*j}$, $1 \le j \le n$, which is impossible. Therefore all the n columns corresponding to the n links of the branch are linearly independent from each other and from every other column of A.

Repeating this reasoning for every branch of the tree, we find that all columns of A are linearly independent. Because of Lemma 2, $\mathbf{v}$ is identifiable. This completes the proof. □

We now consider a system with multiple beacons, where R is the reduced routing matrix after grouping alias links. Let R^B be the sub-matrix of R that contains only the rows representing the paths originating from a beacon B, and let A^B be the reduced matrix of A with only the rows representing paths from B and their element-wise products. As A^B is a sub-matrix of A with the same set of columns but with fewer rows, any set of linearly dependent columns in A is also linearly dependent in A^B. We can now state and prove the following theorem.

THEOREM 1. *Let R be the (reduced) routing matrix of any multiple beacons monitoring network. The variances $\mathbf{v}$ of the links represented by columns of R are identifiable from end-to-end measurements if Assumptions **T.1-2** hold.*

PROOF. We proceed by contradiction. Assume that there exists a set $\mathcal{F} \neq \emptyset$ of links whose columns in A are linearly dependent. That is,

$$\sum_{e_k \in \mathcal{F}} \alpha_k \mathbf{A}_{*k} = \mathbf{0} \tag{5}$$

with $\alpha_k \neq 0$ for all $e_k \in \mathcal{F}$. Note that in (5) we only consider links that are actually linearly dependent (those with coefficient $\alpha_k \neq 0$) and ignore those that are not (with coefficient $\alpha_k = 0$).

Pick any beacon B. Let $\mathcal{P}_B$ be the set of all the paths originating from B, and let $\mathcal{L}_B$ be the set of all links traversed by at least one path in $\mathcal{P}_B$, and let $\mathcal{B}_B = \mathcal{F} \cap \mathcal{L}_B$. Our assumption (5) yields that

$$\sum_{e_k \in \mathcal{B}_B} \alpha_k \mathbf{A}^B_{*k} = \mathbf{0} \tag{6}$$

because any link $e_k \in \mathcal{F} \setminus \mathcal{B}_B$ is not traversed by any path in $\mathcal{P}_B$, which implies that $A^B_{*k} = \mathbf{0}$. The columns of A^B representing links in $\mathcal{B}_B$ are thus linearly dependent.

We now consider three possible cases: (A) $|\mathcal{B}_B| = 1$, (B) $|\mathcal{B}_B| \geq 2$ and (C) $|\mathcal{B}_B| = 0$.

(A) Suppose first that $|\mathcal{B}_B| = 1$. This would yield that $\mathbf{A}^B_{*j} = \mathbf{0}$ for the link $e_j \in \mathcal{B}_B$, which in turn would imply that e_j is not traversed by any path in $\mathcal{P}_B$, an impossibility because $e_j \in \mathcal{B}_B \subseteq \mathcal{L}_B$. Hence $|\mathcal{B}_B| \neq 1$.

(B) Suppose next that $|\mathcal{B}_B| \geq 2$. We will show that this case is impossible, by proceeding as follows: (i) We show that there is a path $P_i \in \mathcal{P}_B$ that traverses all links in $\mathcal{B}_B$, which we index in increasing order of their distances to the beacon B. (ii) We show that any path $P_{i'}$ that traverses a link $e_{b_k} \in \mathcal{B}_B$ must also traverse at least one of the two links of $\mathcal{B}_B$ that are consecutive with e_{b_k} on that path. (iii) Using (ii), we prove that there is a path P_{i_1} that traverses the second and third links of $\mathcal{B}_B$. (iv) We next prove that there is a path P_{i_2} that traverses the third and fourth links of $\mathcal{B}_B$. (v) Finally, we prove by recursion that $|\mathcal{B}| < 2$.

(i) From (6) the columns representing links in $\mathcal{B}_B$ are linearly dependent. However, from Lemmas 2 and 3, any subset of columns representing links in the tree rooted at B are linearly independent. Therefore, (6) can occur only if all links in $\mathcal{B}_B$ are alias links in the tree rooted at B. As a result, there is a path $P_i \in \mathcal{P}_B$ that traverses all links in $\mathcal{B}_B$.

Let $e_{b_1}, e_{b_2}, \ldots, e_{b_n}$ be all the links of $\mathcal{B}_B$, with $n = |\mathcal{B}_B| \geq 2$. As they are all on the same path $P_i \in \mathcal{P}_B$, we can order them in increasing distance from the beacon B, with e_{b_1} being the closest link to B and e_{b_n} the farthest away.

(ii) Any path $P_{i'} \notin \mathcal{P}_B$ that traverses a link $e_{b_k} \in \mathcal{B}_B$ must also traverse at least one of the two links $e_{b_{k-1}}, e_{b_{k+1}} \in \mathcal{B}_B$. Indeed, as P_i and $P_{i'}$ both traverse e_{b_k}, $R_{i,b_k} = R_{i',b_k} = 1$ and thus $A_{((i-1)n_p+(i-i')+1),b_k} = R_{i,b_k} R_{i',b_k} = 1$. Moreover, $R_{i,j} = 0$ for all $e_j \in \mathcal{F} \setminus \mathcal{B}_B$. If $P_{i'}$ did not traverse any other link of $\mathcal{B}_B$ but e_{b_k}, then $R_{i',b_l} = 0$ for all $e_{b_l} \in \mathcal{B}_B \setminus \{e_{b_k}\}$, hence $A_{((i-1)n_p+(i-i')+1),b_j} = R_{i,j} R_{i',j} = 0$ for all links $e_j \in \mathcal{F} \setminus \{e_{b_k}\}$. But then (5) implies that $\alpha_{b_k} = 0$, a contradiction. Therefore $P_{i'}$ must traverse another link of $\mathcal{B}_B \setminus \{e_{b_k}\}$ located just before or after link e_{b_k} (see Figure 4), because otherwise P_i and $P_{i'}$ would form a route "fluttering" as they would meet at e_{b_k}, diverge at $e_{b_{k+1}}$ (respectively, $e_{b_{k-1}}$) and meet again at some link e_{b_j} with $k+2 \leq j \leq n$ (resp., $1 \leq j \leq k-2$). This is contradiction to Assumption **T.2**. Consequently, any path $P_{i'}$ that traverses a link $e_{b_k} \in \mathcal{B}_B$ must also traverse at least one of the two links $e_{b_{k-1}}, e_{b_{k+1}} \in \mathcal{B}_B$.

(iii) Now, all columns of the routing matrix R are distinct, hence the columns $\mathbf{R}_{*b_1}$ and $\mathbf{R}_{*b_2}$ corresponding respectively to e_{b_1} and e_{b_2} are distinct. Therefore, there is a path P_{i_1} that traverses only one of the links e_{b_1} and e_{b_2}, and not the other (Clearly, $P_{i_1} \notin \mathcal{P}_B$). Because of the argument in (ii), P_{i_1} must traverse e_{b_2} and e_{b_3}, and must not traverse e_{b_1}.

(iv) Columns $\mathbf{R}_{*b_2}$ and $\mathbf{R}_{*b_3}$ corresponding respectively to e_{b_2} and e_{b_3} are also distinct. Again, there is a path P_{i_2} that traverses only one of the links e_{b_2} and e_{b_3}, and not the other. If it traverses e_{b_2} but not e_{b_3}, then it must also traverse e_{b_1}. However in this case P_{i_1} and P_{i_2} share link e_{b_2}, but cannot share any other link of $\mathcal{B}_B$. Therefore $R_{i_1,b_2} = R_{i_2,b_2} = 1$ and thus $A_{((i_1-1)n_p+(i_1-i_2)+1),b_2} = R_{i_1,b_2} R_{i_2,b_2} = 1$, and $A_{((i_1-1)n_p+(i_1-i_2)+1),b_j} = R_{i_1,b_j} R_{i_2,b_j} = 0$ for all other link

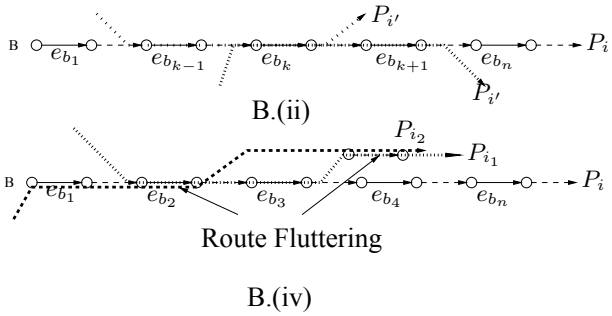

Figure 4: Illustration for the proof of the theorem. Solid links are links in $\mathcal{B}_B$. Dashed links are links not in $\mathcal{B}_B$.

$e_{b_j} \in \mathcal{B}_B \setminus \{e_{b_2}\}$. Because of (5), as $\alpha_{b_2} \neq 0$, there must be a link in $\mathcal{F} \setminus \mathcal{B}_B$ that is traversed by both P_{i_1} and P_{i_2}. However such a link would necessarily cause again route fluttering, because P_{i_1} and P_{i_2} would meet at this link and at link e_{b_2}, but they would need to separate between themselves because they do not share e_{b_1} nor e_{b_3} (see Figure 4). Consequently, P_{i_2} traverses e_{b_3}, as well as e_{b_4}, because of (ii).

(v) We apply the same argument in (iv) recursively for every pair of consecutive links $e_{b_k}, e_{b_{k+1}} \in \mathcal{B}_B$, starting with $k = 3$ until $k = n-1$. However, for $k = n$, we reach a contradiction, as there must be a path $P_{i_{n-1}}$ traversing e_{b_n} but not $e_{b_{n-1}}$. This, in turn, would imply that there is another link $e_{b_{n+1}} \in \mathcal{B}_B$ traversed by $P_{i_{n-1}}$ because of (ii), but this link does not exist. This contradiction shows that $|\mathcal{B}_B| < 2$.

(C) Consequently, points (A) and (B) above imply that for any beacon B, $\mathcal{B}_B = \emptyset$. Hence $\mathcal{F} = \cup_{B \in \mathcal{V}_B} \mathcal{B}_B = \emptyset$, a contradiction with our initial assumption, which proves the theorem. □

Theorem 1 says that if we have enough snapshots to estimate the covariances of all pairs of paths, then there is only one set of link variances that can generate this sequence of snapshots. It holds for all networks where Assumptions **T.1-2** are true.

5. INFERRING LINK LOSS RATES FROM END-TO-END FLOWS

In this section, we present our algorithm to infer link loss rates in any snapshot. The algorithm consists of two phases. In the first phase, we take multiple snapshots of the network to learn the variances of network links, using Theorem 1. In the second phase, we sort the links in increasing order of their variances. The higher the link variance, the more congested the link. We gradually remove the least congested links from (3) until we obtain a new system of full column rank. We then solve this system of equations to obtain the loss rates for the remaining links. After this step, we still do not know the loss rates of the removed links. However, these links are the least congested links and we can approximate their loss rates by 0.

5.1 Estimating the Link Variances Using Sample Moments

In Section 4, we have shown that the link variances can be learnt from a sufficiently large number of snapshots. In

practice, even though it is easy to obtain many snapshots if we wait long enough, it is unlikely that the statistical properties of the links or that the routing matrix R remain the same in all of these snapshots. It is therefore important to have an algorithm that can quickly estimate the link variances from only a small number of snapshots. The proof of Theorem 1 provides us with a hint for such an algorithm.

From Lemma 1, given the sample moments

$$\widehat{\Sigma}_{ii'} = \frac{1}{m-1}\sum_{l=1}^{m} Y_i^{(l)} Y_{i'}^{(l)} - \overline{Y}_i^{(l)}\overline{Y}_{i'}^{(l)}, 1 \le i \le i' \le n_p, \quad (7)$$

we can establish a system of linear equations relating $\widehat{\Sigma}$ and $\mathbf{v}$ as

$$\widehat{\Sigma}^* = A\mathbf{v}. \quad (8)$$

As A has full column rank (Theorem 1), we can then solve (8) to find $\mathbf{v}$ using a standard technique from linear algebra (i.e., using Householder reflection to compute an orthogonal-triangular factorization of A) [15]. The time complexity of this operation is $O(n_p^2 * n_c^2 - n_c^3/3)$ [15]. This inference method (a special case of the *generalized method of moments* [16]) is a consistent estimator and has two advantages over a Maximum Likelihood Estimator (MLE). First, it does not require an underlying assumption on the distribution of the link loss rates. Second, it is much less computationally expensive than an iterative MLE. For example, we show later that (7) can be solved within seconds for networks with thousands of nodes; whereas [8] reports that the proposed EM method to estimate the means and variances of the O-D traffic counts under the Gaussian assumption cannot scale to networks with hundreds of nodes.

In practice, when m is small, sampling variability makes $\widehat{\Sigma} \neq \Sigma$, hence (8) may be an algebraic inconsistent system of equations (with high probability). Furthermore, due to sampling variabilities, some entries of $\widehat{\Sigma}$ may be negative. These algebraic inconsistencies are expected in real systems. In our simulations and experiments in Sections 6 and 7, we ignore equations with $\widehat{\Sigma}_{ii'} < 0$. As (8) contains many redundant covariance equations, we can safely remove those with $\widehat{\Sigma}_{ii'} < 0$.

Note that calculating the matrix A can take a significant amount of time. Fortunately, we only need to do this once for the whole network. When there are changes in the routing matrix R due to arrivals of new beacons, removals of existing beacons or routing changes, we can rapidly modify A without recalculating the whole matrix, because only the rows corresponding to the changes need to be updated.

5.2 Eliminating Good Links To Obtain a System of Full Rank

Knowing the link variances, we can then identify the link loss rates as follows. First, we sort and relabel the links in increasing order of their variances $v_1 \le v_2 \le \ldots \le v_{n_c}$. From Assumption **S.3**, the sorted variances also imply $1 - \phi_{e_1} \le 1 - \phi_{e_2} \le \ldots \le 1 - \phi_{e_{n_c}}$. In real networks, the percentage p of congested links is small. Moreover, the loss rates of non-congested links are close to 0, i.e., $1 - \phi_{e_k} \approx 0$ for all $1 \le k \le (1-p)n_c$. We can therefore shrink R by removing the columns representing the links with the lowest variances until we reach a matrix R^* of full column rank. Note that if we need to remove more than $(1-p)n_c$ columns, it means that some of the congested links can form a linearly dependent set. We show in Sections 6 and 7 that this case rarely occurs.

Once R^* is obtained, we can then rewrite (3) as:

$$\mathbf{Y} = R^*\mathbf{X}^*, \quad (9)$$

where $\mathbf{X}^*$ is the reduced vector representing the log of the transmission rates of the remaining links. We then solve (9) by using a standard linear algebra method [15]. Even though we do not know exactly the loss rates of the links we remove from R to obtain R^*, these links are the best performing links in the network. It is therefore reasonable to approximate their loss rates by 0.

5.3 The Loss Inference Algorithm

Our general algorithm to infer link loss rates in a snapshot is as follows. After calculating the link variances using the m previous snapshots, it takes as input the reduced routing matrix R and the $(m+1)$th snapshot and outputs the link loss rates on this snapshot. We call our algorithm the *Loss Inference Algorithm (LIA)*. LIA has a time complexity of $O(n_p^2 \times n_c^2)$.

Input: The reduced routing matrix R and $m+1$ snapshots: $\mathcal{Y} = \{\mathbf{y}^1, \mathbf{y}^2, ..., \mathbf{y}^m, \mathbf{y}^{m+1}\}$.

Phase 1 (Learning the link variances):
Solve (8) with the first m snapshots $\{\mathbf{y}^1, \mathbf{y}^2, ..., \mathbf{y}^m\}$ to find v_k for all links $e_k \in \mathcal{E}_c$.

Phase 2 (Inferring link loss rates):
Step 1. Sort v_k in increasing order.
Step 2. Initialize $R^* = R$.
(Loop) While R^* is not of full column rank remove R^*_{1*} from R^*.
Step 3. Solve (9) for snapshot $(m+1)$th.
Approximate $\phi_{e_k} \approx 1$ for all links e_k whose columns were removed from R.

Output: Link transmission rates $\phi = [\phi_1\ \phi_2\ \ldots\ \phi_{n_c}]^T$ of the $(m+1)$th snapshot.

Loss Inference Algorithm (LIA)

6. SIMULATION

We first evaluate our algorithm by simulations in different network topologies. We fix the proportion of links that are congested to p. p is varied in our simulations to evaluate the algorithm under different congestion levels of the network. In each snapshot, each link is then randomly selected to be congested with probability p. We use the loss rate model LLRD1 of [24] where congested links have loss rates uniformly distributed in $[0.05, 0.2]$ and good links have loss rates in $[0, 0.002]$. We also evaluate our method with the loss rate model LLRD2 of [24], where loss rates of congested links vary over a wider range of $[0.002, 1]$. In both models, there is a *loss rate threshold* $t_l = 0.002$ that separates good and congested links. We found very little difference between the two models, hence we only report the results for LLRD1 in this paper. Once each link has been assigned a loss rate, the actual losses on each link follow a Gilbert process, where the link fluctuates between good and congested states. When in a good state, the link does not drop any packet, when in a congested state the link drops all packets. According to [32], we set the probability of remaining in a bad state to 0.35 (similarly to the simulations in [24, 36]).

The other transition probabilities are chosen so that the average loss rate matches the loss rate assigned to the link. When a packet on path P_i arrives at link e_k the link state is decided according to the state transition probabilities independently of other links and paths. We also run simulations with Bernoulli losses, where packets are dropped on a link with a fixed probability, but the differences are also insignificant. Therefore, we only report results with Gilbert losses in this section.

For each network, unless otherwise stated, we take $m = 50$ snapshots and $S = 1000$ probe packets in each snapshot. We apply our inference algorithm (LIA) to first calculate the link variances $\mathbf{v}$ using m snapshots in Phase 1 and then proceed to Phase 2 to find the link loss rates in a new snapshot.

The accuracy of the LIA algorithm depends on two main factors: (i) sampling errors in estimating the end-to-end loss rates and their covariances, and (ii) the approximation we apply in Phase 2 of the algorithm by assigning 0 loss rates to all links whose columns are removed from R. To assess these two sources of errors separately, we evaluate the accuracy of the LIA algorithm using two criteria. We first evaluate its capability to locate the congested links using two metrics: the detection rate (DR), which is the percentage of links that are correctly diagnosed as congested, and the false positive rate (FPR), which is the percentage of links that are good but are diagnosed as congested. With $\mathcal{F}$ denoting the set of the actual congested links, and $\mathcal{X}$ the set of links identified as congested by a location algorithm, these two rates are given by:

$$\text{DR} = \frac{|\mathcal{F} \cap \mathcal{X}|}{|\mathcal{F}|} \; ; \; \text{FPR} = \frac{|\mathcal{X} \backslash \mathcal{F}|}{|\mathcal{X}|}.$$

To calculate DR and FPR for the LIA algorithm, we compare the inferred rates against the threshold t_l (given by the loss model) to determine whether the links are good or congested. In the case of the loss models LLRD1 and LLRD2, $t_l = 0.002$.

We then evaluate the accuracies of LIA in estimating the link loss rates using the method proposed in [6]. Suppose we want to compare two loss probabilities q and q^*, where q is the real loss rate on a link and q^* is the inferred probability. For some error margin, $\delta > 0$, the *error factor* [6] is then defined to be

$$f_\delta(q, q^*) = \max\left\{\frac{q(\delta)}{q^*(\delta)}, \frac{q^*(\delta)}{q(\delta)}\right\}, \qquad (10)$$

where $q(\delta) = \max\{\delta, q\}$ and $q^*(\delta) = \max\{\delta, q^*\}$. Thus, q and q^* are treated as being not less than δ, and having done this, the error factor is the maximum ratio, upwards or downwards, by which they differ. Unless otherwise stated, we used the default value $\delta = 10^{-3}$ in this paper. This choice of metric is used to estimate the relative magnitude of loss ratios on different links in order to distinguish those that suffer higher losses.

6.1 Results on Tree Topologies

We first perform our simulations on tree topologies of 1000 unique nodes, with the maximum branching ratio of 10. The beacon is located at the root and the probing destinations $\mathcal{D}$ are the leaves. We fix the percentage of congested links $p = 10\%$. We repeat each simulation configuration 10 times. The rank of the augmented routing matrix A is always equal the number of links n_c as we have seen in Section 5, hence we can estimate the link variances $\mathbf{v}$ very accurately in all of our simulations.

Figure 5 shows the accuracy of the LIA in locating the congested links. The errors in our results come from sampling errors and they have two sources: (i) the sampling errors in estimating end-to-end loss rates and (ii) the sampling errors in estimating their covariances. Even under these errors, our method is highly accurate with large DR and small FPR. We also compare our results with the Smallest Consistent Failure Set (SCFS) algorithm of [14] that uses only a single snapshot (i.e., the current snapshot) of network measurements. As shown in the figure, our algorithm is much more accurate even with small numbers of snapshots m. The reason for the better performance of LIA is obvious: we extract more information from the measurements, namely the second order statistics. Note also that the LIA algorithm is more accurate when m is large, as in this case the errors in estimating covariances of end-to-end loss rates are small.

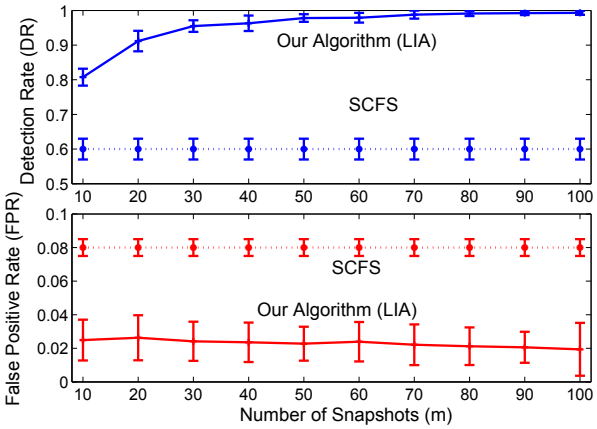

Figure 5: Accuracy of the LIA algorithm in locating the congested links.

The real power of the method is in inferring not only the locations of the congested links but also the link loss rates as shown in Figure 6. In the figure, we plot the cumulative distribution functions (CDFs) of the absolute errors (the absolute values of the differences between the inferred loss rates and their true values) and of the error factors (10) with $m = 50$ snapshots. As we can see in the figure, both errors are extremely small: the inferred values match almost exactly the true values. These small errors are not only caused by the sampling errors of end-to-end loss rates but also by our approximation of the loss rates of links removed from R (to obtain R^*) by 0. This type of error could be fatal if some congested links (with non-negligible loss rates) are removed because they affect all equations of (3) in which they appear. Fortunately, in all simulations R^* always contains the columns of all the congested links. Thus in our simulation results, the errors introduced by this approximation are small. We investigate the effect of these approximation errors further in Section 6.3.

6.2 Results for Mesh Topologies

We further evaluate our LIA algorithm on mesh topologies. We use different *mesh* topologies generated by BRITE [21]

Table 2: Simulations with BRITE, PlanetLab and DIMES Topologies.

Topology	Congested Link Location		Error Factors			Absolute Errors		
	DR	FPR	MAX	MEDIAN	MIN	MAX	MEDIAN	MIN
Barabasi-Albert	91.27%	3.78%	1.27	1.00	1.00	0.0018	0.0010	0.00
Waxman	92.67%	2.84%	1.42	1.00	1.00	0.0020	0.0009	0.00
Hierarchical (Top-Down)	87.81%	6.13%	1.55	1.00	1.00	0.0026	0.0008	0.00
Hierarchical (Bottom-Up)	90.00%	3.78%	1.44	1.00	1.00	0.0014	0.0009	0.00
PlanetLab	96.40%	2.71%	1.16	1.00	1.00	0.001	0.0008	0.00
DIMES	86.75%	6.05%	1.56	1.00	1.00	0.0017	0.0010	0.00

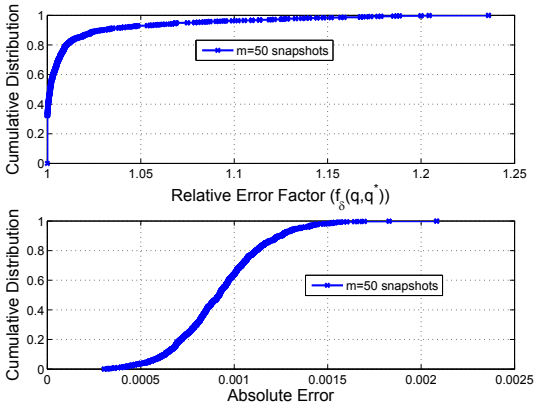

Figure 6: Accuracy of the LIA algorithm in determining the link loss rates. The number of snapshots $m = 50$.

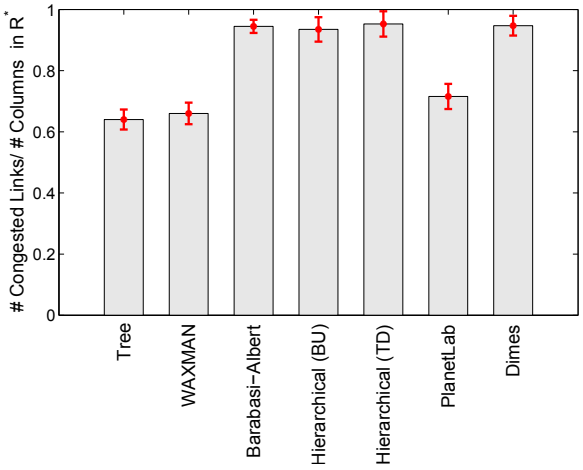

Figure 7: Ratio between the number of congested links (pn_c) and the number of columns of R remaining in R^*. In all topologies, this ratio is always below 1.

(WAXMAN, Barabasi-Albert and Hierarchical) with 1000 nodes, the PlanetLab topology [34] with 500 beacons, 14922 distinct links collected on the 15th of July, 2006, and the DIMES [33] topology with 801 beacons and 35067 distinct links collected from May to July, 2006. In the simulated topology, end-hosts are nodes with the least out-degree. In PlanetLab and DIMES topologies all end-hosts are given. Both DIMES and PlanetLab are real Internet topologies. Most of the DIMES hosts are located in the commercial Internet, contrary to most of the PlanetLab hosts that reside in Universities and research organizations. In all simulations, the end-hosts are both beacons and probing destinations. The link loss model is LLRD1 with $p = 10\%$ congested links. We use the default values $m = 50$ and $S = 1000$. The results are reported in Table 2. Each entry in the table is an average of 10 runs. We omit the confident intervals for clarity reasons.

The results confirm that the LIA algorithm achieves high accuracies in all topologies. This is understandable as the algorithm provably can work with all topologies (Theorem 1). On some topologies such as the BRITE hierarchical topologies or the DIMES topology, LIA performs less well than on other topologies. These inaccuracies are the results of the approximation in (9). Note however that in all topologies the link variances are learnt exactly.

The ratio between the number of congested links and the number of columns of R^* is shown in Figure 7. We could see clearly that the number of congested links is always smaller than the number of columns in R^*. As we remove columns of R until we reach a full column rank matrix R^*, this means that we reach a full rank matrix R^* by only removing columns of un-congested links. Hence, our approach that first calculates the links variances and then solves (9) is appropriate as it does not assign zero loss rates to any congested link.

6.3 Effect of p and S

In this section, we explore the influences of the percentage of congested links p and the number of probes S on the accuracy of the LIA algorithm. We run simulations on the PlanetLab network with $m = 50$ snapshots. In Figure 8.a, we plot the DR and FPR of the LIA algorithm in locating congested links when varying p from 5% to 25% with $S = 1000$. In Figure 8.b we plot the DR and FPR of the LIA when varying S from 200 to 1000 with $p = 10\%$.

As p increases, the link variances $\mathbf{v}$ can still be learnt precisely, as proved in Section 4. However, when p is large, some of the congested links may need to be removed from R to obtain R^* in (9). As a consequence, the accuracies of our LIA algorithm in calculating link loss rates, hence in locating congested links, degrade as p grows. When S is small, sampling errors of end-to-end loss rates ϕ also affect the accuracy of LIA. The impact of S, however, is less severe.

6.4 Running Time

In all of our simulations, the time needed to solve (3) is in the order of milliseconds, whereas the time required to solve (9) is about 10 times longer. These running times are

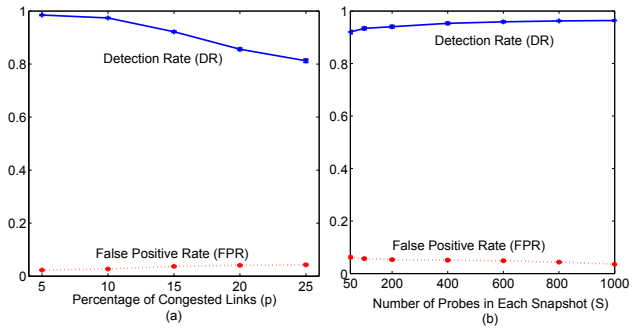

Figure 8: Accuracy of the LIA algorithm under different values of p and S.

for Matlab programs on a Pentium 4 with a 2Ghz processor and 2GB of memory. The computation of A, for our simulated networks could take up to an hour on the same computer. However, as explained earlier, we need to calculate A only once. After calculating A (only once), the time needed to run the inference algorithm is in the order of less than a second.

7. INTERNET EXPERIMENTS

We test our LIA algorithm on the PlanetLab network with 716 end-hosts. All end-hosts are beacons ($\mathcal{V}_B$) and probing destinations ($\mathcal{D}$).

7.1 Methodology

We first use traceroute to measure the network topology. Traceroute is performed in parallel from each beacon to all other beacons. The collected routes will then be combined to form a complete network topology. Using traceroute to build network topology can lead to errors because of several factors. First, for security and performance reasons, many routers in the Internet do not respond or limit the rate of responses to ICMP queries. The paths to nodes behind these routers cannot be ascertained. According to our own traceroute measurements between PlanetLab hosts, 5 to 10% of routers do not respond to ICMP requests. Second, many routers have multiple interfaces and return different IP addresses to different traceroute requests. We employ the sr-ally tool [28] to disambiguate multiple interfaces at a single node. In our measurements on the PlanetLab, about 16% of routers on PlanetLab have multiple interfaces. Unfortunately, the sr-ally tool does not guarantee complete identification of routers with multiple interfaces [28]. As a result, there could be errors in the measured topology of the network. Despite these errors, the experiment results show that the LIA algorithm is accurate.

After constructing the routing topology with traceroute, we remove fluttering paths by examining all pairs of paths. We found very few of them in our data set. To simplify the analysis we take one of the fluttering paths to include in the topology and completely ignore the others. As a result, we remove 52 out of 48151 paths from the routing matrix. By removing these paths, we lose information about the links that only appear in them but they account for less than 0.01% of the total number of links and thus are negligible.

Nodes then send probes between each other to determine the end-to-end loss rate. We use periodic probes with an inter-arrival time of 10ms. Each probe is a UDP packet of 40 bytes. The probing packets consist of a 20-byte IP header, an 8-byte UDP header, and a payload of 12 bytes that contains the probing packet sequence number. End-to-end loss rates are calculated at the receivers based on the number of received and sent packets over a period of 10 seconds (which corresponds to 1000 probe packets). To avoid overloading any host or creating additional congestion, we limit the probing rate from each host to 100KB/sec, i.e., each beacon probes 150 paths in 1 minute. Previous experiments on Planetlab [10] show that this probing rate does not cause additional congestion on the PlanetLab network. We also randomize the order in which a host sends probes to other hosts. Every five minutes all hosts send their measurement data to our central server at EPFL. Note that the probing rate, the impact of active probes on the network and the stationarity of network loss processes intermingle and make the problem of finding the optimal probing strategy very difficult. We do not address this question in the current paper.

We launched the experiments on all PlanetLab hosts (716 hosts) on April 20, 2007. For various reasons, such as nodes being down and failed ssh connections, only 381 hosts successfully ran both the traceroute and loss measurements. The experiment lasted a total of 24 hours with approximately 2 GB of loss and topology data. In total we have 250 snapshots of the network. We then run our LIA algorithm on this data set. To infer the link loss rates on any given snapshot, we first use the previous m snapshots to learn the link variances. After that we use these variances to infer the link loss rates. We report our analysis of the data in the next section.

7.2 Results

In the Internet, we do not know the real loss rates of the network links as in the simulations of Section 6. Therefore we cannot validate our inference results by comparing them against the true values. We adopt the indirect validation method of [24], where the set of end-to-end measured paths in any time slot is divided into two sets of equal size: the inference set and the validation set. The partition is done randomly. We run first our LIA algorithm on the inference set to calculate the link loss rates of the links $\mathcal{E}_{\text{inf}}$ that appear in the inference topology. We then use the measurements in the second set to validate the inferred values. Let ϕ_i be the measured transmission rate of path P_i in the validation set. We compare ϕ_i with the product of the estimated link rate $\widehat{\phi}_{e_k}$ on the link $e_k \in P_i \cap \mathcal{E}_{\text{inf}}$ to check the consistency of the inference. More precisely, we declare the estimate correct if:

$$\left| \widehat{\phi}_i - \prod_{e_k \in P_i \cap \mathcal{E}_{\text{inf}}} \phi_{e_k} \right| \leq \epsilon, \tag{11}$$

where ϵ is the *tolerable error* that is used to account for sampling errors in estimating the end-to-end loss rates with probes. In our experiment, we choose $\epsilon = 0.005$.

7.2.1 Accuracy of LIA

In Figure 9 we plot the percentage of paths that are consistent with the test in (11) as a function of the number of snapshots m used in the variance learning phase. Each point in the curve is an average of 10 runs. The result shows that

Table 3: Location of Congested Links.

t_l	inter-AS	intra-AS
0.04	53.6%	46.4%
0.02	56.9%	43.1%
0.01	57.8%	42.2%

more than 95% of the paths in the validation set are consistent with the inferred loss rates. Predictably, the accuracy of the LIA algorithm increases as we increase m. However, the accuracy flattens out when m is significantly large ($m > 80$).

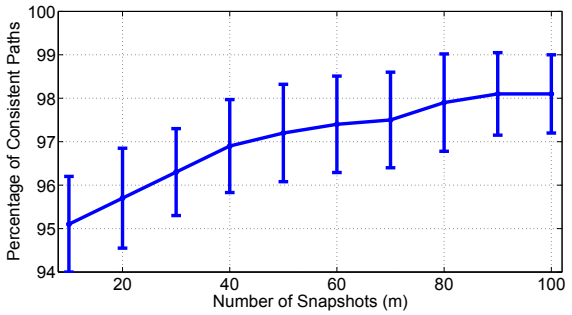

Figure 9: Cross validation of the LIA algorithm on the PlanetLab network.

In all of our experiments, the time needed to solve (3) is in the order of milliseconds, whereas the time required to solve (9) is about a second. Overall, the running time of LIA is in the order of seconds for the PlanetLab network.

7.2.2 Statistics of Congested Links on PlanetLab

Having the link loss rates, we can study statistical properties of the congested links. These studies allow for a deeper understanding of the Internet performance. Namely, we can answer two questions: (i) are the congested links inter- or intra-AS links? and (ii) How long does a link remain congested?

To answer the ASes question, we need to map IP addresses into ASes. We use the BGP table available at Route-Views [23] to construct the mapping. In our analysis, we use the mapping that was obtained on April 20th, 2007. Table 3 shows the locations of lossy links for different link loss threshold t_l (Recall that t_l is used to classify links as good or congested: a link is (not) congested if its loss rate is (less) more than t_l).

We observe that congested links are more likely to be inter-AS than intra-AS, especially for small t_l. However, compared to the study in [36], the percentage of intra-AS lossy links in our study is larger. There are two explanations for the differences between our findings and those in [36]. First, we do not apply the probe optimization technique in [36]. Hence, in our experiments, each beacon needs to send more active probes than the experiments in [36]. These active probes may create additional packet losses at the access links in our experiments compared to [36]. Second, in [36], only the loss rates of groups of links (MILSes) can be computed. Therefore, even though a lossy MILS spans several ASes (30% of them span more than 3 ASes), it is not clear that the actual lossy links in this group are inter-AS or intra-AS. Indeed, in [36], only 27.5% of lossy links are confirmed to be inter-AS whereas 15% are intra-AS. The other links could be either inter- or intra-AS. Furthermore, our observation is consistent with the findings of [2] where it was observed that non-access bottleneck links are equally likely to be intra- and inter-AS. This is especially true for measurements performed from Planet-Lab hosts with high access bandwidth.

To study the duration of the congested links, we apply the LIA algorithm on 100 consecutive snapshots and compare the set of inferred congested links (with $t_l = 0.01$, $m = 50$). We find that 99% of congested links remain congested for only one snapshot (5 minutes). The other 1% span two snapshots. Note that this analysis is sensitive to the duration S of each snapshot. A complete study of these properties would require a thorough understanding of the impact of S, the stationarity of link loss processes, etc. and is a topic of our future research.

8. CONCLUSION

First-order moments of end-to-end loss rates are in general not sufficient to uniquely identify average link loss rates. More information is needed. In contrast, we have shown in this paper that second-order spatial statistics are sufficient to uniquely identify the variances of loss rates, which in turn uniquely determine the average loss rates of the most congested links, under the assumptions that their variance is a non-decreasing function of their mean, and thus that the loss rate of non-congested links is virtually zero. We show that this method is both accurate and scalable in our simulations and experiments.

We expect that the sufficient information brought by second-order statistics of end-to-end paths, without multicast support, to identify problematic links in a network with a general topology can be exploited for other problems of network inference.

A first immediate extension is to compute link delays. Congested links usually have high delay variations. In this direction, we first need to take multiple snapshots of the network to learn about the delay variances. Based on the inferred variances, we could then reduce the first order moment equations by removing links with small congestion delays and then solve for the delays of the remaining congested links.

A second extension is the detection of anomalies in the network, from a few vantage points. The inference method is fast and so could have potential for such problems.

An important question to address is the choice of the optimal value for S. As stated earlier, the answer requires understanding the statistical properties of link performances on small time-scales. We intend to work on this problem in our future research.

Acknowledgments

This work is financially supported by grant ManCom 2110 of the Hasler Foundation, Bern, Switzerland. We would like to thank the anonymous reviewers for their useful discussions and suggestions. We are particularly grateful to Sridhar Machiraju for his helpful feedback during the shepherding phase of the paper.

9. REFERENCES

[1] A. Adams, T. Bu, T. Friedman, J. Horowitz, D. Towstey, R. Caceres, N. Duffield, F. L. Presti, S. B. Moon, and V. Paxson. The use of end-to-end multicast measurements for characterizing internal network behavior. *IEEE Communications Magazine*, May 2000.

[2] A. Akella, S. Seshan, and A. Shaikh. An empirical evaluation of wide-area internet bottlenecks. In *Proc. IMC 03*, 2003.

[3] K. Anagnostakis, M. Greenwald, and R. Ryger. Cing: Measuring network internal delays using only existing infrastructure. In *Proc. IEEE Infocom*, 2003.

[4] D. Arifler, G. de Veciana, and B. L. Evans. A factor analysis approach to inferring congestion sharing based on flow level measurements. *IEEE/ACM Transactions on Networking*, 2007.

[5] B. Augustin, X. Cuvellier, B. Orgogozo, F. Viger, T. Friedman, M. Latapy, C. Magnien, and R. Teixeira. Avoiding traceroute anomalies with paris traceroute. In *Proc. of the Internet Measurement Conference*, October 2006.

[6] T. Bu, N. Duffield, F. L. Presti, and D. Towsley. Network tomography on general topologies. In *Proceedings ACM Sigmetrics 2002*, Marina Del Rey, CA, 2002.

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

[8] J. Cao, D. Davis, S. V. Wiel, and B. Yu. Time-varying network tomography: Router link data. *Journal of the American Statistical Association*, 95(452):1063–1075, Dec. 2000.

[9] A. Chen, J. Cao, and T. Bu. Network tomography: Identifiability and fourier domain estimation. In *Proceedings of the IEEE Infocom*, Alaska, May 2007.

[10] Y. Chen, D. Bindel, H. Song, and R. H. Katz. An algebraic approach to practical and scalable overlay network monitoring. In *Proceedings of the ACM SIGCOMM*, Portland, August-September 2004.

[11] M. Coates, A. Hero, R.Nowak, and B. Yu. Internet tomography. *IEEE Signal Processing Magazine*, 19, May 2002.

[12] M. Coates and R. Nowak. Network loss inference using unicast end-to-end measurement. In *Proceedings of the ITC Seminar on IP Traffic, Measurements and Modelling*, Monterey, September 2000.

[13] N. Duffield, F. L. Presti, V. Paxson, and D. Towsley. Inferring link loss using striped unicast probes. In *Proc. of the IEEE Infocom 2001*, Alaska, April 2001.

[14] N. G. Duffield. Network tomography of binary network performance characteristics. *IEEE Transactions on Information Theory*, 52(12):5373–5388, Dec. 2006.

[15] G. H. Golub and C. F. V. Loan. *Matrix Computations*. The Johns Hopkins University Press, 1996.

[16] L. P. Hansen. Large sample properties of generalized method of moments estimators. *Econometrica*, 50:1029–1054, 1982.

[17] K. Harfoush, A. Bestavros, and J. Byers. Robust identification of shared losses using end-to-end unicast probes. In *Proc. of ICNP'00*, 2000.

[18] V. Jacobson. traceroute, ftp://ftp.ee.lbl.gov/traceroute.tar.z, 1989.

[19] M. S. Kim, T. Kim, Y. S. Hin, S. S. Lam, and E. J. Powers. A wavelet-based approach to detect shared congestion. In *Proceeding of the ACM SIGCOMM'04*, 2004.

[20] R. Mahajan, N. Spring, D. Wetherall, and T. Anderson. User-level internet path diagnosis. In *Proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP'03)*, pages 106–119, 2003.

[21] A. Medina, I. Matta, and J. Byers. On the origin of power-laws in internet topologies. *ACM Computer Communication Review*, pages 160–163, 2000.

[22] H. X. Nguyen and P. Thiran. The boolean solution to the congested IP link location problem: Theory and practice. In *Proceedings of IEEE INFOCOM 2007*, May 2007.

[23] U. of Oregon Route Views Archive Project. http://www.routeviews.org/.

[24] V. N. Padmanabhan, L. Qiu, and H. J. Wang. Server-based inference of internet performance. In *Proceedings of the IEEE INFOCOM'03*, San Francisco, CA, April 2003.

[25] M. Roughan. Fundamental bounds on the accuracy of network performance measurements. In *Proceedings of ACM SIGMETRICS*, June 2005.

[26] D. Rubenstein, J. Kurose, and D. Towsley. Detecing shared congestion of flows via end-to-end measurement. *IEEE/ACM Transactions on Networking*, 10(3), June 2002.

[27] J. Sommers, P. Barford, N. Duffield, and A. Ron. Improving accuracy in end-to-end packet loss measurement. In *Proceedings of ACM SIGCOMM*, August 2005.

[28] N. Spring, D. Wetherall, and T. Anderson. Scriptroute: A public internet measurement facility. In *USENIX Symposium on Internet Technologies and Systems (USITS)*, 2003.

[29] Y. Tsang, M. Coates, and R. Nowak. Passive network tomography using the EM algorithms. In *Proc. IEEE ICASSP.*, 2001.

[30] Y. Vardi. Network tomography: Estimating source-destination traffic intensities. *Journal of the American Statistical Association*, 91:365–377, 1996.

[31] V.Paxson. End-to-end routing behaviour in the internet. In *Proceedings of ACM SIGCOMM*, Aug 1996.

[32] V.Paxson. End-to-end internet packet dynamics. In *Proceedings of the ACM SIGCOMM*, Sep 1997.

[33] www.netdimes.org.

[34] www.planet-lab.org.

[35] Y. Zhang, N. Duffield, V. Paxson, and S. Shenker. On the constancy of internet path properties. In *Proceedings of ACM SIGCOMM Internet Measurement Workshop*, San Francisco, 2001.

[36] Y. Zhao, Y. Chen, and D. Bindel. Toward unbiased end-to-end network diagnosis. In *Proceedings of ACM SIGCOMM*, Pisa, Italy, 2006.